ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

----------∞ 🕮 ∞----------

Báo cáo
Mini Project
Tối ưu lập kế hoạch

Giáo viên hướng dẫn: TS. Bùi Quốc Trung

Nhóm sinh viên thực hiện: Nhóm 2

Họ và tên	MSSV
Nguyễn Quang Linh	20194092
Hồ Mạnh Thắng	20183627
Nguyễn Thịnh Vượng	20183676
Ngô Trường Giang	20180068

Hà Nội – 07/2022

Mục lục

Lời cảm ơn

Kỳ 20212 này, nhóm em đăng ký học phần Tối ưu lập kế hoạch và nhận được phân công dưới sự hướng dẫn của thầy – TS. Bùi Quốc Trung.

Sau khoảng thời gian nghiên cứu và nhận được sự chỉ bảo của thầy, nhóm em không chỉ hoàn thành Miniproject mà còn học hỏi được rất nhiều kiến thức mới.

Tuy nhiên vì kiến thức bản thân còn hạn chế, trong quá trình học tập, hoàn thiện miniproject này nhóm em không tránh khỏi những sai sót, kính mong nhận được những ý kiến đóng góp từ thầy để giúp nhóm em hoàn thiện hơn.

Nhóm em xin chân thành cảm ơn!.

1. Mô tả đề tài

Có N cánh đồng 1,2,3,...N trồng cùng một loại nông sản.
Mỗi cánh đồng i có sản lượng di và cần được thu hoạch trong khoảng thời gian từ ngày si đến ei .
Nhà máy xử lý nông sản có công suất xử lý tối đa M (tổng lượng sản phẩm có thể xử lý trong ngày). Ngoài ra tổng sản lượng cần xử lý trong một ngày phải lớn hơn hoặc bằng m thì nhà máy mới quyết định mở máy vận hành.
Hãy lập kế hoạch thu hoạch nông sản (mỗi cánh đồng thu hoạch vào ngày nào) sao cho thỏa mãn ràng buộc về công suất tối đa và tối thiểu của nhà máy đồng thời chênh lệch sản lượng nông sản cần xử lý giữa các ngày là nhỏ nhất.

Input của bài toán

Dòng 1: N, m, M

Dòng i+1 ($1 \le i \le N$): d_i, s_i, e_i

2. Phương pháp

2.1. Integer Programming

Đặt biến:

- $\begin{cases} x_{i,j} = 1 & \text{nếu cánh đồng i thu hoạch ngày j} \\ x_{i,j} = 0 & \text{ngược lại} \end{cases}$
- day_j là tổng sản lượng thu hoạch của ngày thứ j
- min_day là sản lượng thu hoạch ít nhất trong một ngày
- max_day là sản lượng thu hoạch nhiều nhất trong một ngày
- $\begin{cases} flag_j = 1 & \text{nếu ngày j có thu hoạch} \\ flag_j = 0 & \text{ngược lại} \end{cases}$

Trong đó: $1 \le i \le N$ - $1 \le j \le \max(e_i) = L$

Tập các ràng buộc:

- Cánh đồng i phải thu hoạch vào ngày $j \in [s_i, e_i]$

$$\begin{cases} x_{i,j} = 0 & \text{nếu } \forall j \notin [s_i, e_i] \\ \sum x_{i,j} = 1 & \text{nếu } \forall j \in [s_i, e_i] \end{cases}$$

- Tổng sản lượng ngày j là day_j:

$$day_j = \Sigma(x_{i,j} * d_i)$$

- $x_{i,j} \le flag_j \le \Sigma x_{i,j}$ $(1 \le i \le N, 1 \le j \le L)$

Ở đây, điều kiện này ràng buộc nếu có một ngày không thu hoạch thì các biến $x_{i,j}$ trong ngày đó phải đều bằng 0, còn nếu ngày đó có thu hoạch thì bắt buộc phải có ít nhất 1 biến bằng 1 tức là có ít nhất 1 cánh đồng thu hoạch trong ngày đó.

- $0 \le min\_day \le M$
- $0 \le max\_day \le M$ $(1 \le i \le N, 1 \le j \le L)$
- Tổng sản lượng trong một ngày phải thỏa mãn:
  - $flag_j * m \le day_j$
  - $flag_j * min\_day \le day_j \le max\_day$

Hàm mục tiêu:

$$max\_day - min\_day \rightarrow Minimize$$

Các biến flag để kiểm soát ngày đó có thu hoạch hay không, nó giúp cho min_day không cần bằng 0 vẫn nhỏ hơn một ngày không thu hoạch

2.2. Constraint Programming

Đặt biến

- $\begin{cases} x_{i,j} = 1 & \text{nếu cánh đồng i thu hoạch ngày j} \\ x_{i,j} = 0 & \text{ngược lại} \end{cases}$
- day_j là tổng sản lượng thu hoạch của ngày thứ j
- min_day là sản lượng thu hoạch ít nhất trong một ngày
- max_day là sản lượng thu hoạch nhiều nhất trong một ngày

Tập các ràng buộc

Ràng buộc:

- Sản lượng ngày j là day_j $(1 \le i \le N, 1 \le j \le L)$

$$day_j = \Sigma\, x_{i,j} * d_i$$

- Cánh đồng i phải thu hoạch vào ngày $j \in [s_i, e_i]$

$$\begin{cases} x_{i,j} = 0 & \text{nếu } \forall j \notin [s_i, e_i] \\ \sum x_{i,j} = 1 & \text{nếu } \forall j \in [s_i, e_i] \end{cases}$$

- Nếu khác 0 thì sản lượng ngày j nhỏ nhất là m

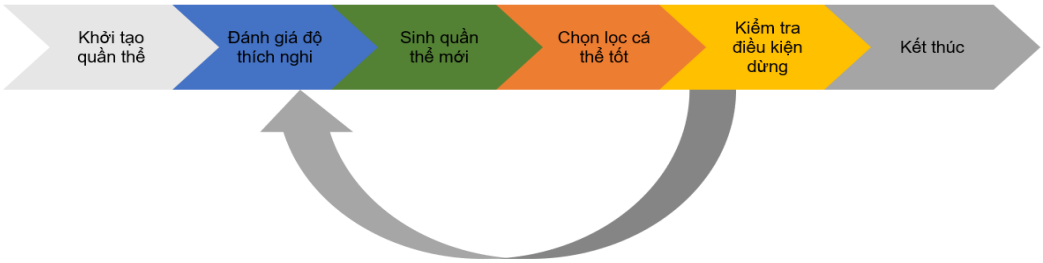
$$day_j \ne 0 \Rightarrow day_j \ge m$$

- Sản lượng ngày j lớn nhất là M: $day_j \le M$
- $min\_day = min(day_j)$
- $max\_day = max(day_j)$ $(1 \le j \le L)$

Hàm mục tiêu:

$$max\_day - min\_day \rightarrow Minimize$$

2.3. Mô hình Genetic Algorithm (GA)



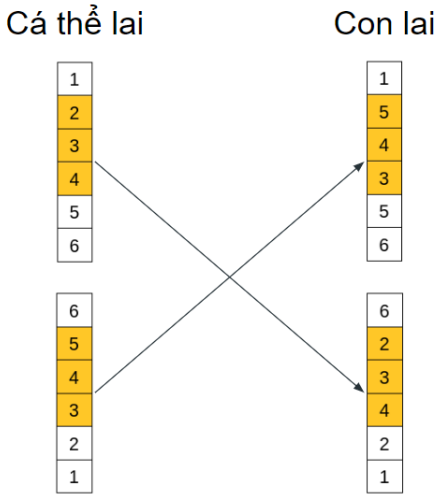
Mô hình hóa:

- Xem solution là vector x có N chiều (N - số lượng cánh đồng)
- Ngày x_i thu hoạch cánh đồng i $(s_i \leq x_i \leq e_i)_i$
- Ví dụ: x = (5, 4, 2, 1, 4, 2, 1, 2, 4, 5) (N = 10, L = 5)
  Ngày 1: thu hoạch cánh đồng 4 – 7
  Ngày 2: thu hoạch cánh đồng 3 - 6 – 8
  Ngày 3: không thu hoạch
  Ngày 4: thu hoạch cánh đồng 2 - 5 – 9
  Ngày 5: thu hoạch cánh đồng 1 - 10

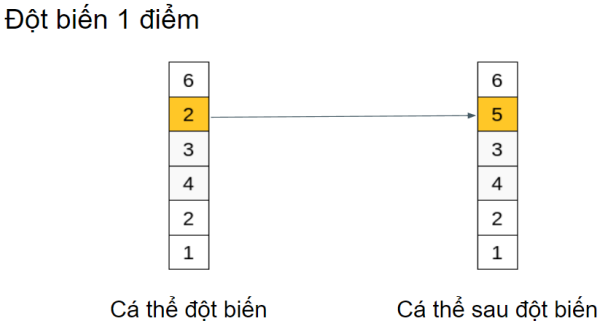
Thuật toán:

- Khởi tạo quần thể
  - Sinh ra một cá thể ngẫu nhiên trong quần thể
  - Số lượng quần thể là 300 cá thể
- Lai ghép, đột biến sinh ra thế hệ sau
  - Sử dụng lai ghép 2 điểm cắt
  - Đột biến một điểm trên gen cá thể
  - Giữ lại 10% cá thể tốt nhất của thế hệ cũ
- Loại bỏ cá thể tồi trong thế hệ cũ

Lai ghép 2 điểm cắt



Đột biến 1 điểm



- GA có độ chính xác và tối ưu sẽ bão hòa đến một kích thước quần thể nhất định

  → Kích thước vượt quá ngưỡng này thì chúng sẽ không tăng thậm chí giảm
- Với bộ dữ liệu có kích thước càng lớn trong khi kích thước 1 cá thể không đổi

  → xuất hiện nhiều cá thể trùng lặp nhau, ít sai khác khi so sánh các NST trên các cá thể. (độ đa dạng quần thể sẽ ngày càng giảm)

  → lời giải từ các thế hệ sau sẽ chỉ quanh quẩn local optimize, chất lượng lời giải kém đi
- Vì kích thước lớn hơn yêu cầu số lần chạy (số thế hệ = 300) chạy tăng theo cấp số nhân nên khi chạy trên cả 2 bộ small và big data với cùng số thế hệ, hiển nhiên kết quả của bộ big data sẽ kém hơn nhiều, chỉ tìm được tối ưu cục bộ. Ngược lại, trên small data vì nó nhỏ nên tối ưu cục bộ đó có thể chính là tối ưu toàn cục.

➞ Các quần thể khi phát triển đến một ngưỡng cố định sẽ có dấu hiệu suy thoái và bão hòa, điều này sẽ giải thích kết quả cho GA khi so sánh với các mô hình

2.4. Tabu Search

2.4.1. Mô hình hóa

- Mô hình hóa: tương tự GA
- Đánh giá sản lượng thu hoạch ngày j:

$$flag_j = \begin{cases} 1 & \text{nếu } 0 < day_j < m \\ 2 & \text{nếu } m \le day_j \le M \\ 3 & \text{nếu } M < day_j \\ 0 & \text{nếu } 0 = day_j \end{cases}$$

Vì bài toàn có ràng buộc rất chặt về sản lượng thu hoạch một ngày. Đồng thời Tabu search sẽ có việc tạo ngẫu nhiên một lời giải, sản lượng có thể bất kỳ. Nên chúng ta có biến flag để kiểm soát như ở trên.

2.4.2. Thuật toán

Hàm mục tiêu: objective = *alpha*violation + *beta*cost

Trong đó:

- violation: tổng sản lượng vi phạm
- cost = $\max(day_j) - \min(day_j)$
- alpha = 1000 (trọng số hàm mục tiêu)
  beta = 1 (để violation bằng 0 thì alpha phải lớn hơn beta rất nhiều)

Hàm tạo ngẫu nhiên lời giải

Algorithm 1: Initial Solution

Input : N, m, M, d, s, e
Output: *Random solution*

Init : $day_j \leftarrow 0, flag_j \leftarrow 0$ /* $1 \leq j \leq \max_{1 \leq i \leq N}(e_i) = L$ */
for $i \leftarrow 1$ ***to*** N **do**
  if $||T_0|| > 0$ *and* $T_0 = \{j | s_i \leq j \leq e_i, flag_j = 0\}$ **then**
    $x_i \leftarrow random(T_0)$;
  end
  else if $||T_1|| > 0$ *and* $T_1 = \{j | s_i \leq j \leq e_i, flag_j = 1\}$ **then**
    $x_i \leftarrow random(T_1)$;
  end
  else if $||T_2|| > 0$ *and* $T_2 = \{j | s_i \leq j \leq e_i, flag_j = 2, day_j + d_i \leq M\}$ **then**
    $x_i \leftarrow random(T_2)$;
  end
  else
    $x_i \leftarrow random(s_i, e_i)$;
  end
  $day_{x_i} \leftarrow \text{day}_{x_i} + \text{d}_i$;
  update $flag_{x_i}$;
end
$sol \leftarrow re\_arrange(sol)$;
return *sol*

re_arrange(sol): Từ một lời giải vừa được sinh ra, ta thực hiện chuyển cánh đồng i thu hoạch vào ngày j sang ngày k:

- $\text{flag}_j = 3$: chuyển cho ngày chưa đạt ngưỡng

  ($\text{flag}_k \leq 1$ 》 $\text{flag}_k = 2$ có $\text{day}_k + \text{d}_i \leq M$)

- $\text{flag}_j = 1$: chuyển cho ngày chưa đạt ngưỡng

  ($\text{flag}_k = 1$ 》 $\text{flag}_k = 2$ có $\text{day}_k + \text{d}_i \leq M$)

- $\text{flag}_j = 2$: chuyển cho các ngày có sản lượng ít hơn

  ($\text{flag}_k = 1$ 》 $\text{flag}_k = 2$ có $\text{day}_k + \text{d}_i \leq \text{day}_j$)

Chiến thuật tìm kiêm hàng xóm:

- Tính toán day_j - $flag_j$ - violation của solution x $(1 \le j \le L)$
- Với mỗi cánh đồng i: chuyển ngày thu hoạch sang ngày j tạo lời giải mới
  - $s_i \le j \le e_i$ và $j \ne x_i$
  - $flag_j \ne 0$ và $flag_j \ne 3$
  - $tabu\_length_j = 0$
- Tính objective_value cho lời giải mới:
  - Tính violation_1: violation với các ngày j và x_i của solution x
  - Cập nhật day_j và day_{xi}
    Tính violation_2: violation với các ngày j và x_i của solution mới
    obj = alpha*(violation + violation_2 - violation_1) + beta*newCost

Cuối cùng, thuật toán Tabu search

Algorithm 2: Tabu Search

Output: *Best solution*

```
Init     : current_solution ← init_solution() ;
for i ← 1 to NUMBER_GEN do
    if current_solution < best_solution then
        best_solution ← current_solution;
        stable ← 0;
    end
    else if stable = stable_limit then
        current_solution ← init_solution();
        stable ← 0;
        delete tabu_list
    end
    current_solution ← search_next_solution();
    update tabu_list
end
```

3. So sánh

3.1. Bộ dữ liệu

Bài toán này có các điểm cần chú ý về dữ liệu như sau:

- Chúng ta cần ước lượng được số cánh đồng thu hoạch trung bình mỗi ngày
- Khoảng thời gian thu hoạch của các cánh đồng:

$$valid_i = e_i - s_i$$

  Vì những cánh đồng này trồng cùng một loại nông sản, nên thời gian thu hoạch các cánh đồng sẽ giống nhau

- Khả năng xuất hiện ngày trống: không thể nào thu hoạch đủ sản lượng cho ngày đó

  Chúng ta sẽ có các thuộc tính tương ứng như sau:

Type	số cánh đồng / ngày	valid	Ngày trống
1	4 - 7	4	không
2	4 - 7	7	không
3	4 - 7	4	có
4	10	5	không

Trong đó, chúng ta chia mỗi type ra làm 2 loại nhỏ. Số liệu loại dữ liệu như sau:

Type	Công suất		N
Small	m	10, 20, 35, 50	10, 20, 40, 60
	M	15, 30, 50, 75	
Big	m	20, 30, 40, 50	200, 400, 600, 800
	M	30, 45, 60, 80	

Kịch bản sinh dữ liệu:

Với đầu vào là N, m, M và việc chúng ta ước lượng được số cánh đồng thu hoạch mỗi ngày, chúng ta sẽ tính được số ngày x cần thu hoạch là bao nhiêu. Sau đó, chúng ta sinh ngẫu nhiên x giá trị sản lượng cần thu hoạch mỗi ngày ($m \leq sản\ lượng \leq M$). Sau đó sẽ ngẫu nhiên chọn cánh đồng nào thu hoạch trong ngày nào. Cuối cùng, chia sản lượng ngày i cho các cánh đồng được thu hoạch vào ngày i đó. Chúng ta sẽ luôn đảm bảo có một lời giải cho bài toán này.

3.2. So sánh

3.2.1. Kết quả

Tỉ lệ lời giải hợp lệ

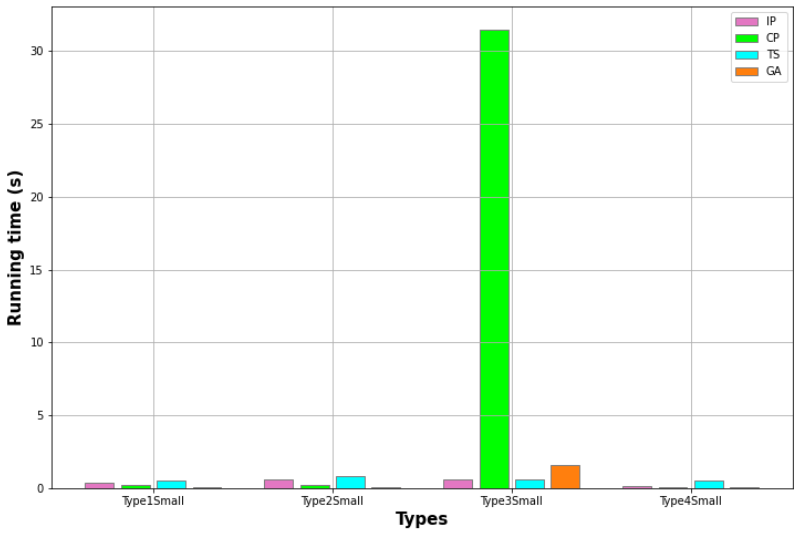
Type	IP		CP		TS		GA	
	Small	Big	Small	Big	Small	Big	Small	Big
1	100	93.75	100	100	100	36.875	100	0
2	100	100	100	100	100	70.625	100	26.25
3	100	100	100	100	90.625	60.625	88.75	0
4	100	100	100	100	100	100	100	71.875

Tỉ lệ lời giải tối ưu

Type	IP		CP		TS		GA	
	Small	Big	Small	Big	Small	Big	Small	Big
1	100	87.5	81.25	87.5	36.25	0	28.125	0
2	100	50	81.25	81.25	60.625	0	33.75	0
3	100	93.75	93.75	68.75	40	0	16.25	0
4	100	100	87.5	81.25	78.755	0	56.875	0

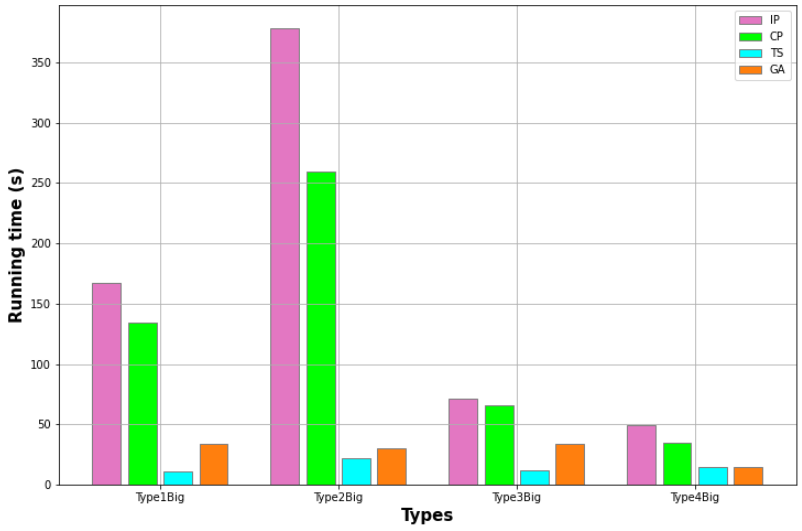
3.2.2. Thời gian

Đối với dữ liệu nhỏ



Ở đây, type3 là loại dữ liệu nhiều hơn 1 constraint so với dữ liệu khác – có ngày nghỉ giữa các ngày thu hoạch, thế nên Constraint Programming với dữ liệu nhỏ cần nhiều thời gian hơn để giải ra một solution.

Đối với dữ liệu lớn



Ở type3, Integer Programming cần nhiều thời gian để giải ra một solution với type dữ liệu lớn, nên không quá khác biệt

4. Kết luận

- Dữ liệu nhỏ, cả IP - CP - TS - GA đều có thể cho kết quả tối ưu trong thời gian cho phép
- IP và CP tỏ ra hiệu quả trên cả phương diện tìm kiếm lời giải hợp lệ và tìm ra lời giải tối ưu
- TS chạy nhanh hơn, ổn định hơn và cho ra kết quả tốt hơn hẳn GA
- Cải thiện GA: khởi tạo, tối ưu hóa tham số, tăng số thế hệ