Phương pháp mã hóa trong bài toán RCPSP

Ngày 25 tháng 9 năm 2025

0 Một số kí hiệu

Tham số

$V = \{0, \dots, n+1\}$	Tập hợp các công việc
$A = \{1, \dots, n\}$	Tập hợp các công việc thật sự
p_i	Thời gian thực hiện của công việc i
$G = (V, E)$	Đồ thị thứ tự
$(i, j) \in E$	Quan hệ thứ tự giữa cặp công việc i và j, cụ thể công việc i được thực hiện xong trước khi công việc j bắt đầu
$R = \{1, \dots, v\}$	Tập hợp các tài nguyên
B_k	Số lượng tài nguyên tối đa trên tài nguyên k
$b_{i,k}$	Yêu cầu tài nguyên của công việc i trên tài nguyên k
UB	Giá trị lớn nhất có thể của makespan
LB	Giá trị nhỏ nhất có thể của makespan
$H = \{0, \dots, UB - 1\}$	Không gian lập lịch (Scheduling horizon)

Giá trị suy ra được từ các tham số

$G^* = (V, E^*)$	Đồ thị thứ tự mở rộng
$(i, j, l_{i,j}) \in E^*$	Quan hệ thứ tự mở rộng với time lag $l_{i,j}$
$ES(i)$, $LS(i)$	Thời gian bắt đầu sớm nhất và muộn nhất của công việc i
$EC(i)$, $LC(i)$	Thời gian kết thúc sơm nhất và muộn nhất của công việc i
$RTW(i)$	Khoảng thời gian mà công việc i có thể được thực hiện
$STW(i)$	Khoảng thời gian mà công việc i có thể được bắt đầu
$\mathcal{P} = \{P_1, \dots, P_c\}$	Path cover

Các biến được sử dụng trong mã hóa

S_i	Thời gian bát đầu của công việc i
$s_{i,t}$	Công việc i bắt đầu tại thời điểm t
$x_{i,t}$	Công việc i đang được thực hiện tại thời điểm t
m_t	Makespan có giá trị bằng t

1 Định nghĩa bài toán

Mục tiêu của bài toán *Resource-Constrained Project Scheduling Problem (RCPSP)* là tìm thời gian bắt đầu cho mỗi công việc trong project sao cho makespan đạt giá trị nhỏ nhất.

Bài toán RCPSP được định nghĩa một cách hình thức bởi một bộ (V, p, E, R, B, b) trong đó:

- $V = \{0, 1, 2, \dots, n+1\}$ là tập các công việc (activity). Công việc 0 và $n+1$ là các công việc giả (dummy job) đại diện cho điểm bắt đầu và kết thúc của project. Tập các công việc thực sự (non-dummy job) được kí hiệu là $A = \{1, 2, 3, \dots, n\}$
- $p \in \mathbb{N}^{n+2}$ là một vector chứa các số tự nhiên, trong đó p_i là thời gian thực hiện của công việc i. Với công các công việc giả, ta có $p_0 = p_{n+1} = 0$, và $p_i > 0 \quad \forall i \in A$.

- E là tập hợp chứa các cặp công việc, thể hiện quan hệ thứ tự giữa các công việc. Cụ thể, $(i, j) \in E$ khi và chỉ khi công việc i hoàn thành trước khi công việc j bắt đầu. Giả sử, quan hệ thứ tự giữa các công việc được cho bởi đồ thị $G = (V, E)$ không có chu trình, đồng thời trong G , từ công việc 0 có thể đi đến được mọi công việc khác trong G, và mọi công việc trong G đều có thể đi đến được công việc $n + 1$.
- $R = \{1, 2, .., v\}$ là tập hợp các tài nguyên có thể tái tạo (renewable resource).
- $B \in \mathbb{N}^v$ là vector chứa các số tự nhiên, trong đó B_k là số lượng tài nguyên khả dụng (capactity) của tài nguyên k.
- $b \in \mathbb{N}^{(n+2)\times v}$ là ma trận chứa các số tự nhiên tương ứng với yêu cầu tài nguyên của mỗi công việc, trong đó $b_{i,k}$ thể hiện lượng tài nguyên mà công việc i yêu cầu trên tài nguyên k trên mỗi đơn vị thời gian trong suốt khoảng thời gian mà nó được thực hiện, $b_{0,k} = b_{n+1,k} = 0$ và $b_{i,k} > 0, \forall k \in 1..v, \forall i \in 1..n$

Một lịch trình (schedule) là một vector chứa các số tự nhiên $S = (S_0, S_1, .., S_n + 1)$ trong đó S_i là thời gian bắt đầu của công việc i. Không làm mất tính tổng quát, ta giả sử $S_0 = 0$. Một lời giải tối ưu của bài toán RCPSP là một lịch trình S cực tiêu hóa được giá trị makespan S_{n+1} , đồng thời thỏa mãn các điều kiện:

- Điều kiện về thứ tự: Mọi công việc chỉ được phép bắt đầu sau khi các công việc trước nó hoàn thành.
- Điều kiện về tài nguyên: Tại bất kì thời điểm nào, tổng nhu cầu về số lượng tài nguyên trên một tài nguyên của tất cả các công việc đang được thực hiện tại thời điểm đó không được vượt quá lượng tài nguyên khả dụng của tài nguyên đó.

2 Tiền xử lí

2.1 Đồ thị thứ tự mở rộng (Extended precedence relations)

Kí hiệu $G^* = (V, E^*)$ là đồ thị thứ tự mở rộng, trong đó, E^* là tập hợp các cạnh có trọng số. Với mỗi cặp công việc (i, j) trong V thỏa mãn giữa chúng có đường đi trong G bắt đầu từ i và kết thúc tại j , tồn tại một cạnh $(i, j, l_{i,j})$ tương ứng trong E^*. Time lag $l_{i,j}$ là giá trị nhỏ nhất (lower bound) của hiệu giữa thời gian bắt đầu thực hiện công việc i và j. Giá trị này có thể được tính toán bằng cách sử dụng thuật toán Floyd–Warshall.

2.2 Energetic reasoning on precedences

Chú ý rằng với tất cả các công việc a thỏa mãn $(i, a, l_{i,j}) \in E^* \wedge (a, j, l_{a,j}) \in E^*$ phải được thực hiện và hoàn thành trong khoảng thời gian $[S_i + p_i, S_j - 1]$. Vì vậy, khoảng thời gian này phải đủ rộng để có thể thực hiện tất cả các công việc a nêu trên mà không vượt quá giới hạn tài nguyên của bất kì tài nguyên nào. Do đó, với mỗi tài nguyên $k \in R$, giá trị nhỏ nhất của khoảng cách giữa thời gian kết thúc của công việc i và thời gian bắt đầu của công việc j có thể được tính bằng:

$$RLB_{i,j,k} = \left\lceil \frac{1}{B_k} \times \sum_{\substack{a \in A \text{ s.t.} \\ (i,a,l_{i,a}) \in E^* \\ (a,j,l_{a,j}) \in E^*}} (p_a \cdot b_{a,k}) \right\rceil \tag{1}$$

Dựa vào công thức này, ta có thể cặp nhật giá trị time lag tính được ở phần 2.1

$$l^*_{i,j} = \max(l_{i,j}, p_i + \max_{k \in R}(RLB_{i,j,k})) \qquad \forall (i, j, l_{i,j} \in E^*) \tag{2}$$

2.3 Cửa sổ thời gian (Time windows)

Trong bài toán RCPSP, cho trước đồ thị thứ tự mở rộng, với mỗi công việc trong project ta có thể tính được thời gian bắt đầu sớm nhất, thời gian bắt đầu muộn nhất, thời gian kết thúc sớm nhất, thời gian kết thúc muộn nhất của nó. Cụ thể:

$$ES(i) = l_{0,i}$$
$$EC(i) = l_{0,i} + p_i$$
$$LS(i) = UB - l_{i,n+1}$$
$$LS(i) = UB - l_{i,n+1} + p_i$$

Ta cũng định nghĩa:

$$RTW(i) = [ES(i), LC(i) - 1]$$
$$STW(i) = [ES(i), LS(i)]$$

3 Công thức biểu diễn các ràng buộc

Công việc 0 bắt đầu tại thời điểm 0

$$s_{0,0} \tag{3}$$

Công việc bắt đầu tại đúng một thời điểm trong $STW(i)$

$$\sum_{t \in STW(i)} s_{i,t} = 1 \quad \forall i \in V \setminus \{0\} \tag{4}$$

Quan hệ thứ tự: Có ba trường hợp có thể xảy ra với mỗi cạnh $(i, j, l_{i,j}) \in E^*$:

1. $S_i + l_{i,j} - 1 < ES(j)$ (Công việc i bắt đầu tại thời điểm mà nó sẽ hoàn thành trước khi công việc j bắt đầu): Ta không cần mã hóa cho trường hợp này
2. $ES(i) \leq S_i + l_{i,j} - 1 \leq LS(j)$ (Công việc i bắt đầu tại thời điểm mà nó sẽ hoàn thành trong $STW(j)$):

$$s_{i,t'} + \sum_{t \in [ES(j),\, k]} s_{j,t} \leq 1 \quad \forall (i,j) \in E, \tag{5}$$
$$\forall k \in STW(j) \text{ s.t. } k - l_{i,j} + 1 \leq LS(i),$$
$$t' = \max(ES(i),\, k - l_{i,j} + 1)$$

3. $S_i + l_{i,j} - 1 > LS(j)$ (Công việc i bắt tại thời điểm mà nó sẽ hoàn thành sau khi công việc j bắt đầu):

$$s_{i,t'} + \sum_{t \in STW(j)} s_{j,t} \leq 1 \quad \forall t' \text{ s.t. } \max(LS(j) - l_{i,j} + 1, ES(i) - 1) < t' \leq LS(i) \tag{6}$$

Nếu công việc i bắt đầu tại thời điểm t thì nó phải được thực hiện trong khoảng thời gian $[t, t + p_i - 1]$

$$\neg s_{i,t} \vee x_{i,t'} \quad \forall i \in V, \forall t \in STW(i), \forall t' \in [t, s + p_i - 1] \tag{7}$$

Điều kiện lan truyền ngược (back propagation) nhằm tăng tốc độ giải

$$\neg s_{i,t} \vee x_{i,t+1} \vee s_{i,t-d_i+1} \qquad \forall i \in V, \forall t \in RTW(i) \setminus \{LC(i) - 1\} \tag{8}$$

Giới hạn tài nguyên

$$\sum_{\substack{i \in A \text{ s.t.} \\ t \in RTW(i)}} r_{ik}.x_{it} \leq R_k \quad \forall k \in R, \forall t \in H \tag{9}$$

4 Mã hóa điều kiện bắt đầu và quan hệ thứ tự của các công việc

Kí hiệu $R_{i,a,b}$ là biểu diễn staircase tương đương của $\sum_{a \le t < b} s_{i,t} \le 1$. Ta có:

$$s_{i,a} \vee \neg R_{i,a,a} \tag{10}$$

$$\neg s_{i,t} \vee R_{i,a,t} \quad t \in [a,b] \tag{11}$$

$$\neg R_{i,a,t-1} \vee R_{i,a,t} \quad \forall t \in (a,b] \tag{12}$$

$$R_{i,a,t-1} \vee s_{i,t} \vee \neg R_{i,a,t} \quad \forall t \in (a,b] \tag{13}$$

$$\neg R_{i,a,t-1} \vee \neg s_{i,t} \quad \forall t \in (a,b] \tag{14}$$

Khi đó, mệnh đề 4, 5, 6 lần lượt trở thành:

$$R_{i,ES(i),LS(i)} \tag{15}$$

$$\neg s_{i,t'} \vee \neg R_{j,ES(j),k} \tag{16}$$

$$\neg s_{i,t'} \vee \neg R_{j,ES(j),LS(j)+1} \tag{17}$$

5 Tối ưu makespan

Sử dụng phương pháp incremental SAT. Sau khi tìm được lời giải khi makespan $S_{n+1} = T$, ta bổ sung thêm vào tập các assumption mệnh đề

$$\neg s_{n+1,T} \tag{18}$$

rồi tiếp tục tìm lời giải. Nếu UNSAT thì kết luận $S_{n+1} = T$, nếu tìm được lời giải thì tiếp tục thực hiện bước trên cho đến khi UNSAT hoặc gặp LB.