Đại học quốc gia TP.HCM
Trường đại học công nghệ thông tin

Ngành Khoa học Máy tính

Môn học: CS112.Q11.CTTN

Báo cáo bài tập Nhóm 12

Nhóm thực hiện:
Nhóm 7

Giảng viên:
Nguyễn Thanh Sơn

1 Subtask 1 ($N \leq 20$): Tìm nghiệm tối ưu tuyệt đối

1.1 Phương pháp thiết kế thuật toán

Phương pháp sử dụng: Completed Search (Vét cạn) kết hợp kỹ thuật Bitmasking và nguyên lý EDD (Earliest Due Date).

Mô tả chi tiết nguyên lý: Bài toán được mô hình hóa dưới dạng tìm một tập con các tác vụ $S \subseteq \{1, 2, \ldots, N\}$ và một hoán vị σ của S sao cho tổng lợi nhuận là lớn nhất và thỏa mãn ràng buộc thời gian.

1. **Biểu diễn không gian trạng thái (Bitmasking):** Vì N nhỏ, ta sử dụng một số nguyên $mask$ từ 0 đến $2^N - 1$. Nếu bit thứ i của $mask$ là 1, tác vụ i được chọn vào tập ứng viên.

2. **Kiểm tra tính khả thi (Feasibility Check):** Với một tập hợp các tác vụ được chọn, để kiểm tra xem có tồn tại một thứ tự thực hiện thỏa mãn deadline hay không, ta áp dụng **Định lý Jackson (Jackson's Rule)** mở rộng: *"Nếu một tập hợp các công việc có thể hoàn thành đúng hạn, thì việc sắp xếp chúng theo thứ tự deadline tăng dần (Earliest Due Date - EDD) cũng sẽ hoàn thành đúng hạn."*

   Quy trình kiểm tra:

   - Sắp xếp các tác vụ trong tập con theo t_i tăng dần.
   - Duyệt qua từng tác vụ j trong danh sách đã sắp xếp, tính thời gian hoàn thành $C_j = \sum_{k=1}^{j} r_k$.
   - Điều kiện thỏa mãn: $\forall j, C_j \leq t_j$.

3. **Cập nhật tối ưu:** Nếu tập hợp thỏa mãn, tính tổng lợi nhuận $P = \sum p_i$ và cập nhật $MaxProfit = \max(MaxProfit, P)$.

1.2 Tính phù hợp của phương pháp

- **Đảm bảo tính chính xác (Correctness):** Subtask 1 yêu cầu tìm nghiệm tối ưu tuyệt đối. Các phương pháp Heuristic hoặc Greedy đơn thuần không đảm bảo điều này. Chỉ có vét cạn toàn bộ không gian nghiệm mới đưa ra kết quả chính xác 100%.

- **Khả năng tính toán:** Với $N \leq 20$, số lượng trạng thái là $2^{20} \approx 1,048,576$. Máy tính hiện đại có thể xử lý 10^8 phép tính/giây. Với giới hạn thời gian 5s, việc duyệt 1 triệu trạng thái kèm thao tác sắp xếp là hoàn toàn khả thi và an toàn.

1.3 Phân tích độ phức tạp

- **Độ phức tạp thời gian:** $O(2^N \cdot N \log N)$.
  - Số lượng trạng thái (tập con): 2^N.
  - Chi phí kiểm tra mỗi trạng thái: Sắp xếp k phần tử ($k \leq N$) mất $O(N \log N)$ và duyệt kiểm tra mất $O(N)$.
- **Độ phức tạp không gian:** $O(N)$. Ta chỉ cần lưu trữ mảng các tác vụ và một vector tạm thời để lưu tập con đang xét. Không sử dụng mảng quy hoạch động hay đệ quy sâu, do đó tiết kiệm bộ nhớ.

2 Subtask 2 ($N \leq 500$): Tìm nghiệm gần đúng tốt nhất

2.1 Phương pháp thiết kế thuật toán

Với $N = 500$ và trọng số lớn, bài toán trở thành NP-Hard. Nhóm áp dụng phương pháp **Meta-heuristic - Thuật toán Gần đúng**, cụ thể là **Simulated Annealing (Tôi luyện mô phỏng)**.

2.1.1 Cấu trúc thuật toán chi tiết

1. Khởi tạo đa chiến lược (Hybrid Initialization): Thay vì khởi tạo ngẫu nhiên, thuật toán tạo ra 3 phương án ban đầu dựa trên các chiến lược Greedy (Tham lam) để chọn ra điểm xuất phát tốt nhất:

- **Strategy 1 (Density):** Ưu tiên tỉ lệ p_i/r_i giảm dần.
- **Strategy 2 (Profit):** Ưu tiên lợi nhuận p_i giảm dần.
- **Strategy 3 (Tightness):** Ưu tiên deadline t_i tăng dần.

Điều này giúp thuật toán bắt đầu tìm kiếm từ một vùng "đồi cao" thay vì dưới "vực sâu", tăng tốc độ hội tụ.

2. Cơ chế sinh láng giềng (Neighbor Generation): Từ cấu hình hiện tại S, cấu hình mới S' được sinh ra bằng kỹ thuật *Ruin and Recreate*:

- **Ruin (Phá hủy):** Loại bỏ ngẫu nhiên một số lượng công việc khỏi S.
- **Recreate (Tái tạo):** Cố gắng chèn các công việc chưa được chọn (trong tập pool) vào S theo thứ tự ngẫu nhiên hoặc theo heuristic, đảm bảo tính khả thi ngay khi chèn.

3. Hàm năng lượng và Tiêu chuẩn Metropolis: Gọi $E(S)$ là tổng lợi nhuận của cấu hình S. Ta muốn tối đa hóa $E(S)$. Xét $\Delta E = E(S') - E(S)$.

- Nếu $\Delta E > 0$ (nghiệm mới tốt hơn): Luôn chấp nhận $S \leftarrow S'$.
- Nếu $\Delta E \leq 0$ (nghiệm mới tồi hơn): Chấp nhận S' với xác suất $P = e^{\frac{\Delta E}{T}}$.

Việc chấp nhận nghiệm tồi hơn cho phép thuật toán "trèo xuống dốc" để thoát khỏi các điểm tối ưu cục bộ (Local Optima), điều mà thuật toán Greedy hay Hill Climbing không làm được.

4. Lịch trình làm lạnh (Cooling Schedule): Nhiệt độ giảm dần theo công thức: $T_{k+1} = \alpha \cdot T_k$. Trong đó, $\alpha = 0.9999$ giúp quá trình giảm nhiệt diễn ra rất chậm, đảm bảo không gian tìm kiếm được duyệt đủ rộng.

2.2 Tính phù hợp của phương pháp

1. **Tại sao là Simulated Annealing (SA)?**
    - **Tránh tối ưu cục bộ:** Với hàm mục tiêu phức tạp của bài toán lập lịch, không gian nghiệm có nhiều "đỉnh giả". SA vượt trội hơn Hill Climbing nhờ khả năng chấp nhận bước đi xuống.
    - **Hiệu quả bộ nhớ:** So với Quy hoạch động (cần bảng mảng lớn không khả thi với $t_i \approx 10^{12}$) hay Giải thuật di truyền (cần lưu quần thể lớn), SA chỉ cần lưu trạng thái hiện tại và trạng thái tốt nhất ($O(N)$), cực kỳ tiết kiệm bộ nhớ.
    - **Linh hoạt với thời gian:** Thuật toán được thiết kế để chạy liên tục và kiểm tra điều kiện dừng dựa trên thời gian thực (‘chrono‘). Điều này tận dụng tối đa 5 giây cho phép để tìm nghiệm tốt nhất có thể.

- **Nhiệt độ khởi đầu $T = 500$:** Đủ lớn để xác suất chấp nhận nghiệm xấu ở giai đoạn đầu cao, giúp thuật toán "nhảy" khắp không gian tìm kiếm.
- **Hệ số làm lạnh $\alpha = 0.9999$:** Tốc độ giảm nhiệt rất chậm nhằm duy trì quá trình tìm kiếm lâu nhất có thể, tránh việc hội tụ quá sớm (Premature Convergence).
- **Điều kiện dừng:** Dựa trên thời gian thực (Time-based termination). Thuật toán sẽ chạy liên tục và chỉ dừng khi đạt ngưỡng 4.9 giây (sát giới hạn 5s của đề bài) để tận dụng tối đa tài nguyên CPU.

2.3 Phân tích độ phức tạp

- **Độ phức tạp thời gian:** $O(K \cdot N)$. Trong đó K là số bước lặp. Với cài đặt tối ưu, K có thể đạt tới hàng triệu bước trong 5 giây. Trong mỗi bước, thao tác xóa và chèn lại mất $O(N)$ (do thao tác trên vector). Tuy nhiên, vì đây là thuật toán heuristic bị giới hạn thời gian, độ phức tạp thực tế là hằng số thời gian T_{limit}.
- **Độ phức tạp không gian:** $O(N)$. Chỉ tốn bộ nhớ lưu trữ danh sách công việc và vector nghiệm. Hoàn toàn thỏa mãn giới hạn bộ nhớ 256MB/1024MB của đề bài.