ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA

TRANG HỒNG SƠN

MỘT SỐ PHƯƠNG PHÁP TIẾP CẬN CHO BÀI TOÁN LẬP LỊCH CÁ NHÂN

Chuyên ngành: Khoa học máy tính
Mã số chuyên ngành: 62480101

TÓM TẮT LUẬN ÁN TIẾN SĨ

TP. HỒ CHÍ MINH - NĂM 2021

Công trình được hoàn thành tại **Trường Đại học Bách Khoa – ĐHQG-HCM**

Người hướng dẫn 1: PGS. TS. Trần Văn Lăng
Người hướng dẫn 2: PGS. TS. Huỳnh Tường Nguyên

Phản biện độc lập 1:
Phản biện độc lập 2:

Phản biện 1:
Phản biện 2:
Phản biện 3:

Luận án sẽ được bảo vệ trước Hội đồng đánh giá luận án họp tại

.....

.....

vào lúc giờ ngày tháng năm

Có thể tìm hiểu luận án tại thư viện:

- Thư viện Trường Đại học Bách Khoa – ĐHQG-HCM
- Thư viện Đại học Quốc gia Tp.HCM
- Thư viện Khoa học Tổng hợp Tp.HCM

DANH MỤC CÔNG TRÌNH ĐÃ CÔNG BỐ

Tạp chí quốc tế

[CT1] T. H. Son, T. V. Lang, N. Huynh-Tuong, and A. Soukhal, "Resolution for bounded-splitting jobs scheduling problem on a single machine in available time-windows", *Journal of Ambient Intelligence and Humanized Computing (SCIE Q1 IF=7.104)*, vol. 12, no. 1, pp. 1179-1196, 2021.

Tạp chí trong nước

[CT2] T. H. Son, T. V. Lang, and N. Huynh-Tuong, "A mathematical model for teamwork scheduling problem in available time windows", *Science & Technology Development Journal - Engineering and Technology*, vol. 3, no. SI1, pp. 50-58, 2020.

[CT3] T. H. Son, T. V. Lang, and N. Huynh-Tuong, "Minimizing makespan of personal scheduling problem in available time-windows with split-min and setup-time constraints", *Journal of Computer Science and Cybernetics*, vol. 34, no. 2, pp. 97–111, 2018.

Kỷ yếu hội nghị quốc tế

[CT4] T. H. Son, N. V. Huy, N. Huynh-Tuong, T. V. Lang, and A. Soukhal, "An approach based on max flow resolution for minimizing makespan of personal scheduling problem", *in Addendum Proceedings of the 2016 IEEE RIVF International Conference on Computing and Communication Technologies: Research, Innovation, and Vision for the Future (RIVF'2016)*, pp. 7–11, Hanoi, Vietnam: IEEE, 7-9 Nov. 2016.

Báo cáo hội nghị trong nước

[CT5] T. H. Son, T. V. Lang, and N. Huynh-Tuong, "A mathematical model for teamwork scheduling problem in available time windows", *in Symposium on Computer Science and Engineering (SCSE'2019)*, Ho Chi Minh City, Vietnam, 15-16 Oct. 2019.

[CT6] T. H. Son, N. Huynh-Tuong, and T. V. Lang, "Minimizing makespan of personal scheduling problem in available time-windows with split-min and setup-time constraints", *in The 11th National Conference on Fundamental and Applied IT Research (FAIR'2018)*, Hanoi, Vietnam, 9-10 Aug. 2018.

CHƯƠNG 1 GIỚI THIỆU VỀ ĐỀ TÀI LUẬN ÁN

1.1 Giới thiệu chung

Lập lịch công việc rất cần thiết trong cuộc sống cá nhân (personal job scheduling), không chỉ giúp các cá nhân xử lý nhiều công việc phức tạp mà còn giảm căng thẳng. Trong thời đại công nghệ hiện nay, mọi người phải đối phó với hàng trăm công việc, email và nhiều vấn đề phức tạp cần giải quyết từng ngày. Theo (David Allen, 2003), một chuyên gia về cải thiện năng suất công việc, hầu hết chúng ta luôn có khoảng 50 đến 150 nhiệm vụ lớn nhỏ cần phải được xử lý ở bất cứ thời điểm nào. Bên cạnh đó, mỗi công việc sẽ có nhiều thuộc tính và ràng buộc khác nhau như là thời điểm bắt đầu, thời hạn phải hoàn thành, thời gian xử lý thích hợp để thực hiện công việc hiệu quả hơn, ... Một vấn đề nữa là với rất nhiều công việc và những ràng buộc phải được thỏa mãn, đặc biệt hơn cả là vì sự thay đổi liên tục trong cuộc sống thực, mọi người sẽ rất vất vả và tốn thời gian khi liên tục phải tự sắp xếp lại các công việc đã được sắp xếp của mình mỗi khi các công việc bị thay đổi.

Do đó bài toán lập lịch công việc cá nhân có thể tự động phân chia các công việc nhỏ hơn trong những khung thời gian làm việc là rất quan trọng để áp dụng cho các ứng dụng quản lý công việc cá nhân. Đối với một tổ chức hoặc một nhóm người cùng làm việc, vấn đề lập lịch cũng được đặt ra sao cho hoạt động phối hợp trong nhóm và mỗi thành viên được hiệu quả. Việc lập lịch riêng cho mỗi thành viên hay còn gọi là lập lịch cá nhân là một bài toán quan trọng và cơ bản cho việc lập lịch cho cả tập thể nhóm (teamwork job scheduling).

Trong phạm vi nghiên cứu, luận án tập trung chủ yếu vào bài toán lập lịch công việc của một cá nhân, xem xét các phương pháp tiếp cận như là những nghiên cứu cơ bản để có thể làm nền tảng cho các bài toán lập lịch công việc đặc thù khác và cho bài toán lập lịch công việc trong một tập thể hoặc một nhóm người có quan hệ xã hội.

1.2 Động cơ nghiên cứu

Đối với việc lập lịch công việc cá nhân, bài toán sẽ có hai đặc điểm chính. Thứ nhất, mỗi người đều có những khung thời gian làm việc (time-windows) khác nhau, có thể linh động sắp xếp những công việc của họ vào đấy. Thứ hai, mỗi người có thể muốn chia một công việc lớn thành nhiều công việc nhỏ để dễ dàng sắp xếp vào các khung làm việc của mình, nhưng nếu các công việc được chia quá nhỏ thì lại không thể hiệu quả khi không đủ thời gian để xử lý, vì vậy cần phải có thêm ràng buộc là các công việc không được

chia nhỏ hơn một ngưỡng xác định (bounded-splitting) để việc xử lý công việc được hiệu quả hơn, và ràng buộc này lại thường không được đề cập đến trong các bài toán lập lịch hiện nay (xem trình bày tại Bảng 1.1).

Bảng 1.1: Một số ràng buộc về công việc được đề cập trong các tài liệu về lập lịch

	precedence	preemption	batching	lot-sizing	bounded-splitting
Handbook of Scheduling (Leung, 2004)	✓	✓	✓	N/A	N/A
Multicriteria Scheduling (Vincent & Billaut, 2006)	✓	✓	✓	N/A	N/A
Scheduling Algorithms (Brucker, 2007)	✓	✓	✓	N/A	N/A
Introduction to Scheduling (Robert & Vivien, 2009)	✓	✓	✓	N/A	N/A
Handbook on Project Management and Scheduling (Schwindt & Zimmermann, 2015)	✓	✓	✓	✓	N/A
Scheduling: Theory, Algorithms, and Systems (Pinedo, 2016)	✓	✓	✓	✓	N/A
Handbook on Scheduling (Blazewicz et al., 2019)	✓	✓	✓	✓	N/A
Mathematical programming formulations for machine scheduling: A survey (Blazewicz et al., 1991)	✓	✓	✓	N/A	N/A
Scheduling with processing set restrictions: A survey (Leung & Li, 2008)	✓	✓	N/A	N/A	N/A
A survey of case studies in production scheduling: Analysis and perspectives (Fuchigami & Rangel, 2018)	✓	N/A	✓	✓	N/A

Chính vì điều này nên có rất ít công bố trước đây liên quan tới bài toán lập lịch công việc cá nhân. Cụ thể ba công trình nghiên cứu trước đây chỉ mới đặt ra bài toán lập lịch công việc cá nhân (Quan et al., 2010), cũng như đã chứng minh bài toán này thuộc lớp strongly NP-hard (Huy et al., 2013a), và đã đề xuất một mô hình MILP để có thể sử dụng các MILP solver tìm ra lời giải tối ưu cho bài toán với dữ liệu đầu vào có kích thước nhỏ (Huy et al., 2013b). Do đó bài toán lập lịch công việc cá nhân này còn nhiều vấn đề tồn đọng cần phải được xem xét, giải quyết và trả lời các câu hỏi như cần có một khung tổng quát các bước để các giải quyết các bài toán liên quan tới hai ràng buộc $split_{min}$ và $available-windows$ hay không? Nếu có thì các phương pháp tiếp cận trong các bước này là gì? Có các phương pháp nào có thể giải quyết bài toán một cách hiệu quả hơn với dữ liệu đầu vào có kích thước lớn cỡ vài trăm hoặc thậm chí vài ngàn công việc? Có thể xem xét thêm các ràng buộc phù hợp với từng bài toán đặc thù cụ thể khác nhau hay không? Có thể áp dụng bài toán lập lịch công việc cá nhân này trên một nhóm nhiều người được không?

1.3 Mục tiêu, phạm vi và đối tượng nghiên cứu

Mục tiêu của luận án là xem xét và giải quyết bài toán lập lịch công việc cá nhân:

O1. Nghiên cứu và đề xuất một số phương pháp tiếp cận để giải quyết bài toán lập lịch công việc cá nhân trên các bộ dữ liệu đầu vào có kích thước nhỏ và lớn, qua đó đề

xuất lựa chọn phương pháp hiệu quả đối với từng loại dữ liệu đầu vào khác nhau.

O2. Nghiên cứu giải quyết các bài toán lập lịch công việc cá nhân đặc thù có một số ràng buộc thường được sử dụng trong thực tế.

O3. Nghiên cứu ứng dụng mở rộng bài toán lập lịch công việc cá nhân trên một nhóm nhiều người.

Dựa trên ba mục tiêu nghiên cứu ở trên, luận án đã xác định:

- Phạm vi nghiên cứu: giải quyết bài toán lập lịch công việc của từng cá nhân.
- Đối tượng nghiên cứu: nghiên cứu một số phương pháp tiếp cận để giải quyết bài toán lập lịch các công việc của từng cá nhân này.

1.4 Nội dung công việc của luận án

Với mục tiêu cũng như phạm vi và đối tượng nghiên cứu ở trên, luận án tập trung vào những nội dung công việc sau nhằm để giải quyết những thách thức đang đặt ra của bài toán lập lịch công việc cá nhân, từ đó góp phần vào việc giải quyết vấn đề mang tính nghiên cứu cơ bản của bài toán này:

- Đặc tả nhóm bài toán lập lịch công việc cá nhân bao gồm bài toán lập lịch công việc cá nhân cơ bản với hai ràng buộc đó là các công việc có thể cắt nhỏ bị chặn dưới và các công việc chỉ được sắp xếp vào những khung thời gian trống, bài toán lập lịch công việc cá nhân đặc thù với các ràng buộc như mỗi công việc đều có thời gian chuẩn bị khác nhau, mỗi công việc đều có thời điểm bắt buộc phải hoàn thành khác nhau, và bài toán lập lịch công việc cá nhân cho một nhóm nhiều người có các khung thời gian làm việc khác nhau.
- Đề xuất và hiện thực sơ đồ tổng quát các bước tiếp cận để giải quyết các bài toán lập lịch công việc cá nhân đang nghiên cứu bao gồm: (1) phân tích độ khó của bài toán, (2) xem xét một số trường hợp đặc biệt, (3) đưa ra một số tính chất trong lời giải tối ưu, (4) xác định miền nghiệm của bài toán, (5) xây dựng mô hình Mixed-Integer Linear Programming (MILP), (6) sử dụng phương pháp chính xác với MILP solver để xác định lời giải tối ưu cho bài toán, (7) sử dụng các phương pháp xấp xỉ dạng heuristic/metaheuristic/matheuristic để xác định lời giải khả thi cho bài toán.
- Áp dụng một số heuristics đặc thù thừa kế các giải thuật cổ điển như giải thuật Assignment, giải thuật Flow, giải thuật Matching, áp dụng các quy tắc lập lịch ưu tiên

First Come First Serve (FCFS), Shortest Processing Time (SPT), Longest Processing Time (LPT), ..., và một số metaheuristics thường dùng trong công nghiệp như là Simulated Annealing (SA), Genetic Algorithm (GA), Tabu Search (TABU) trên cơ sở rút giảm không gian tìm kiếm dựa trên các tính chất được đưa ra, với mong muốn xác định lời giải khả thi có chất lượng tốt trong giới hạn thời gian chấp nhận được.

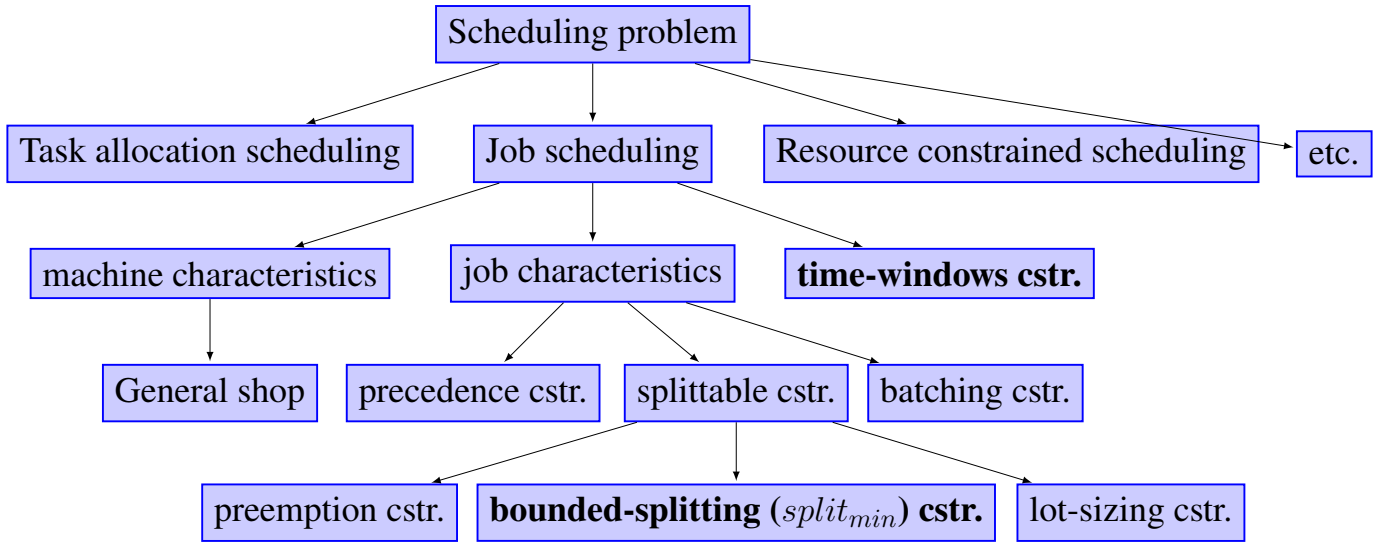
- Đề xuất hướng tiếp cận matheuristics bằng cách kết hợp giải thuật (meta)heuristic cùng với việc xử lý từng bài toán con nhỏ bằng công cụ MILP solver nhằm xác định lời giải khả thi có chất lượng tốt nhất có thể.

1.5 Cấu trúc luận án

Từ những nội dung công việc cần nghiên cứu được nêu ở trên, cấu trúc luận án được trình bày bao gồm năm chương. *Chương thứ nhất* nhằm giới thiệu một cách khái quát những vấn đề chung của một luận án như mục tiêu, phạm vi và đối tượng nghiên cứu, cũng như nội dung công việc phải giải quyết. *Chương thứ hai* trình bày tổng quan về bài toán lập lịch công việc, trong đó đưa ra những khảo sát về tình hình nghiên cứu trong và ngoài nước, từ đó chỉ ra những thách thức của bài toán lập lịch công việc cá nhân, làm rõ hơn vì sao có mục tiêu và nội dung trình bày trong chương thứ nhất. Ngoài ra luận án cũng trình bày những kiến thức mang tính nền tảng khi cần giải quyết phương pháp tính toán cho bài toán lập lịch công việc cá nhân bao gồm mô tả bài toán, các ký hiệu sử dụng, một số minh hoạ, phân tích độ khó của bài toán và các phương pháp giải quyết bài toán như phương pháp chính xác và phương pháp xấp xỉ. *Chương thứ ba* trình bày chi tiết bài toán lập lịch công việc cá nhân gồm hai ràng buộc, đây là bài toán cơ bản cần tiếp cận đầu tiên. Trong chương này các nội dung sẽ được trình bày như đặc tả bài toán bao gồm phát biểu và minh hoạ bài toán, các phương pháp tiếp cận để giải quyết bài toán bao gồm độ khó bài toán, các tính chất của lời giải tối ưu, miền đánh giá nghiệm, mô hình toán học, các phương pháp xấp xỉ như heuristic, metaheuristic, matheuristic, và đánh giá kết quả thực nghiệm các phương pháp tiếp cận này. *Chương thứ tư* trình bày một số bài toán lập lịch công việc cá nhân đặc thù với việc mở rộng các ràng buộc cho bài toán lập lịch công việc cá nhân cơ bản như ràng buộc setup-time, ràng buộc deadline, hoặc hướng đến việc khai thác những kết quả của bài toán lập lịch công việc cá nhân để mở rộng cho bài toán lập lịch công việc nhóm. *Chương thứ năm* là chương tổng kết để đưa ra những kết quả thực hiện của luận án như là những đóng góp, đồng thời cũng trình bày những hướng còn bỏ ngõ khi giải quyết các bài toán lập lịch công việc cá nhân này.

CHƯƠNG 2 TỔNG QUAN VỀ BÀI TOÁN LẬP LỊCH CÔNG VIỆC

2.1 Tình hình nghiên cứu



Hình 2.1: Các hướng nghiên cứu và các ràng buộc liên quan

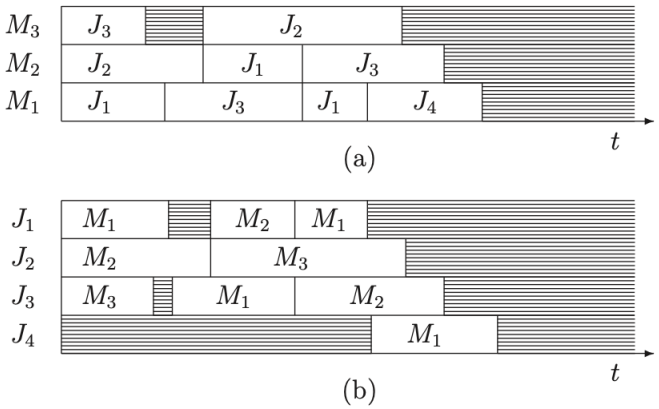
Bài toán lập lịch có rất nhiều hướng nghiên cứu và ứng dụng trong thực tế (xem chi tiết Hình 2.1), cụ thể có thể liệt kê vài hướng nghiên cứu như lập lịch cấp phát các tác vụ (task allocation scheduling) được ứng dụng trong môi trường IoT, lĩnh vực robot, lĩnh vực phương tiện không người lái, ..., lập lịch ràng buộc nguồn lực (resource constrained scheduling) được ứng dụng trong việc quản lý dự án, sắp xếp thời khoá biểu, phân bổ phòng ký túc xá, ..., lập lịch cần cẩu quay (quay crane scheduling) được ứng dụng trong việc bốc dỡ hàng hoá tại cảng biển (seaports), các bãi container, ..., định tuyến và lập lịch phương tiện giao thông (vehicle routing and scheduling) được ứng dụng trong việc giao nhận hàng hoá, việc quản lý lịch trình của tài xế xe công nghệ, ..., lập lịch công việc (job scheduling) được ứng dụng trong việc quản lý nhân sự, quản lý các máy sản xuất, ...

Trong hướng nghiên cứu lập lịch công việc, nếu khảo sát các ràng buộc trên đặc điểm của môi trường máy thực thi thì sẽ có các bài toán như open shop, flow shop, job shop, và mixed shop là những trường hợp đặc biệt của bài toán general shop. Còn nếu khảo sát các ràng buộc trên đặc điểm của công việc thì sẽ có các ràng buộc như preemption, precedence, batching, lot-sizing, ... Các ràng buộc này đã được nghiên cứu từ lâu và đã được trình bày trong các sách kinh điển về lập lịch, cũng như trong các survey về bài toán lập lịch (xem chi tiết tại Bảng 1.1). Tuy nhiên qua các nghiên cứu ở trên, bài toán lập lịch công việc vẫn không được giải quyết trọn vẹn với một lý do chính đó là bài toán lập lịch công việc cá nhân mang tính thực tế và gần gũi với chúng ta đã không được đề cập đến.

2.2 Giới thiệu bài toán lập lịch công việc

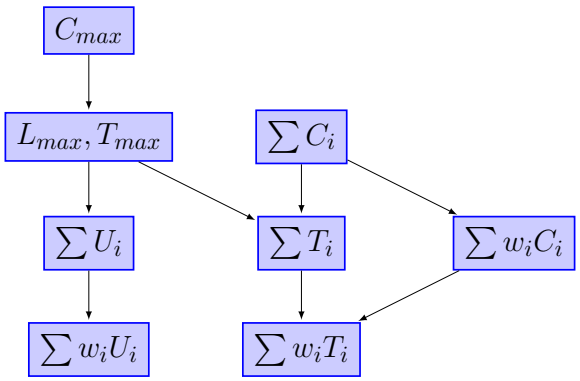
2.2.1 Mô tả bài toán

Giả sử có n công việc (job) J_i $(i = 1, \ldots, n)$ được thực thi trên m máy (machine) M_j $(j = 1, \ldots, m)$. Một lịch trình (schedule) là ứng mỗi công việc được gán vào một hoặc nhiều máy trong một khoảng thời gian nào đó, và có thể được biểu diễn bằng sơ đồ Gantt như Hình 2.2. Tuỳ thuộc vào bài toán cụ thể, mỗi job sẽ có các thông tin như p_{ij} là thời gian xử lý (processing time) của job J_i trên máy M_j, r_i là thời điểm có thể bắt đầu thực thi (release date) của job J_i, st_i là thời gian chuẩn bị (setup time) trước khi thực thi của job J_i, d_i là thời điểm đến hạn (due date) của job J_i, $\overline{d}_i$ là thời điểm bắt buộc phải hoàn thành (deadline) của job J_i, w_i là trọng số (weight) của job J_i, ...



Hình 2.2: Sơ đồ Gantt theo hướng máy (a) và hướng công việc (b) (Brucker, 2007)

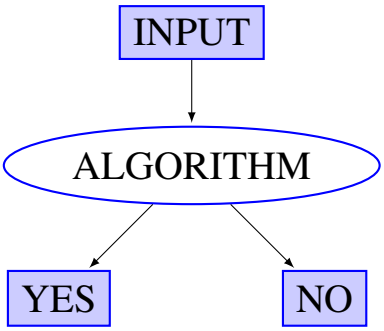
Một bài toán lập lịch công việc có ba yếu tố đặc trưng, đó là: môi trường máy thực thi (machine environment), các đặc điểm công việc (job characteristics) và tiêu chí tối ưu (optimality criterion) hay còn gọi là hàm mục tiêu (objective function). Theo (Graham et al., 1979) thì ba đặc trưng này được ký hiệu thành ba tham số: $\alpha|\beta|\gamma$. Ngoài ra, (Brucker, 2007) cũng đã mô tả mối quan hệ giữa các hàm mục tiêu như trong Hình 2.3.



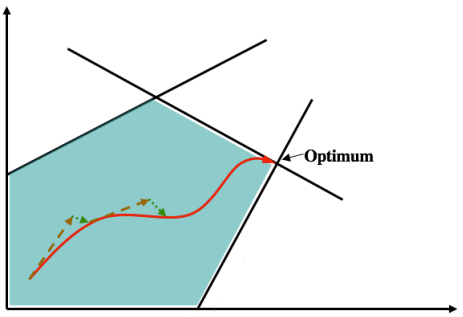
Hình 2.3: Mối quan hệ giữa các hàm mục tiêu

2.2.2 Độ khó bài toán

Theo (Leung, 2004), bài toán quyết định (decision problem) là bài toán có đầu ra (output) chỉ là YES hoặc NO như Hình 2.4. Còn bài toán tối ưu (optimization problem) là bài toán xác định lời giải tốt nhất (best solution) từ tất cả các lời giải khả thi (feasible solution) như Hình 2.5.



Hình 2.4: Bài toán quyết định



Hình 2.5: Bài toán tối ưu

Bên cạnh đó, một nhận xét quan trọng cũng được (Leung, 2004) đưa ra tại mục 2.3.3:

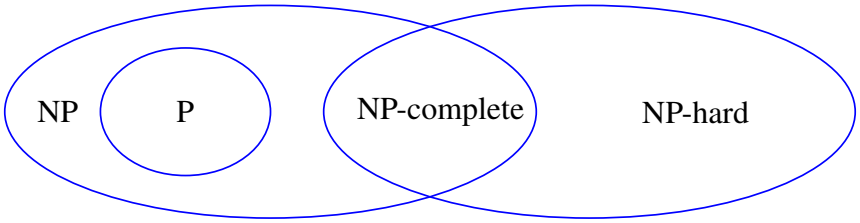
Nhận xét 2.1 *Các bài toán tối ưu (cực tiểu hoá hoặc cực đại hoá) có thể được chuyển đổi thành bài toán quyết định tương ứng bằng cách cung cấp bổ sung một tham số ω và chỉ cần đặt câu hỏi liệu có lời giải khả thi nào để chi phí của lời giải là $\leq \omega$ (cực tiểu hoá) hoặc $\geq \omega$ (cực đại hoá).*

Theo (Cook, 2006), một bài toán quyết định được gọi là thuộc lớp P nếu tồn tại một thuật toán giải bài toán trong thời gian $O(n^c)$, với một hằng số c không phụ thuộc vào kích thước đầu vào n, đôi khi người ta còn thay $O(n^c)$ bởi $poly(n)$ để nói rõ đây là lớp bài toán có độ phức tạp đa thức. Và một bài toán quyết định được gọi là thuộc lớp NP nếu tồn tại một bằng chứng (certificate) dễ kiểm tra cho bài toán đó. Trong đó, bằng chứng dễ kiểm tra được hiểu như là ta có thể dễ dàng kiểm tra một dữ liệu (instance) cụ thể nào đó của bài toán có đầu ra (output) là YES trong thời gian đa thức $poly(n)$. Nói ngắn gọn, P là lớp bài toán quyết định mà chúng ta có thể giải trong thời gian đa thức, còn NP là lớp bài toán quyết định mà chúng ta có thể kiểm tra lời giải trong thời gian đa thức. Về mặt trực quan, một bài toán dễ giải thì cũng dễ kiểm tra lời giải, do đó $P \subseteq NP$, hay nói cách khác, bất kì một bài toán P nào cũng thuộc NP.

Trong các bài toán NP, bài toán NP đầy đủ (NP-complete) là bài toán khó nhất. Theo (Leeuwen, 1990), một bài toán quyết định C được gọi là thuộc lớp bài toán NP-complete nếu $C \in NP$ và với mọi bài toán $X \in NP$, ta có $X \preceq C$ (ta nói X dễ giải quyết hơn C). Ngoài ra, người ta còn đưa ra khái niệm NP-complete mạnh (strongly NP-complete) để nhấn mạnh về độ khó của bài toán quyết định như sau: một bài toán quyết định C được

gọi là thuộc lớp bài toán strongly NP-complete nếu mọi dữ liệu input của bài toán đều là nguyên và $\max(input) \leq poly(length(input))$.

Bên cạnh đó, còn có bài toán NP-hard là bài toán ít nhất là khó ngang bất kì bài toán nào trong NP. Theo (Leeuwen, 1990), một bài toán quyết định H được gọi là thuộc lớp bài toán NP-hard nếu với mọi bài toán $X \in NP$, ta có $X \preceq H$. Ta có thể xem, NP-complete = $NP \cap NP$-hard.



Hình 2.6: P vs. NP vs. NP-complete vs. NP-hard

Một nhận xét khác cũng được (Leung, 2004) đưa ra tại mục 2.4:

Nhận xét 2.2 *Nếu bài toán quyết định tương ứng của bài toán tối ưu thuộc lớp bài toán NP-complete thì bài toán tối ưu thuộc lớp bài toán NP-hard.*

(Knust & Brucker, 2009) cũng đã tổng hợp độ khó/độ phức tạp của một số bài toán lập lịch khác và công bố tại địa chỉ: http://www.informatik.uni-osnabrueck.de/knust/class.

2.3 Các phương pháp giải quyết

2.3.1 Phương pháp chính xác

2.3.1.1 Quy hoạch toán học

Trong (Castillo et al., 2002) tại chương 1 và 7, quy hoạch toán học MP (mathematical programming) là một kỹ thuật mô hình hoá được sử dụng như một công cụ mạnh mẽ trong việc xác định lời giải cho các bài toán quyết định hoặc tối ưu bằng cách xây dựng các mô hình toán học (mathematical model). Quy hoạch toán học bao gồm 3 thành phần là các biến quyết định (decision variables), các ràng buộc (constraints) và một hàm mục tiêu (objective function) được cực đại hoá (maximized) hoặc cực tiểu hoá (minimized). Quy hoạch toán học thường có các dạng sau: quy hoạch tuyến tính LP (linear programming) là quy hoạch toán học trong đó hàm mục tiêu và các ràng buộc đều phải tuyến tính, quy hoạch tuyến tính nguyên ILP (integer linear programming) là quy hoạch tuyến tính trong đó tất cả các biến quyết định đều phải là số nguyên, quy hoạch tuyến tính nhị phân BLP (binary linear programming) là quy hoạch tuyến tính trong đó tất cả các biến quyết

định đều phải là số nhị phân, quy hoạch tuyến tính nguyên hỗn hợp MILP (mixed-integer linear programming) là quy hoạch tuyến tính trong đó phải có ít nhất một biến quyết định là số nguyên. Kỹ thuật mô hình hóa bài toán dưới dạng quy hoạch toán học với các mô hình toán học thường được áp dụng cho bài toán tối ưu và sau đó sử dụng các solver hoặc sử dụng phương pháp nhát cắt để tìm ra lời giải tối ưu cho bài toán (Bixby et al., 2000).

2.3.1.2 Phương pháp nhát cắt

Phương pháp nhát cắt (cutting plane) thường được sử dụng để tìm lời giải nguyên cho các bài toán quy hoạch tuyến tính nguyên ILP, quy hoạch tuyến tính nguyên hỗn hợp MILP, cũng như các bài toán tối ưu hóa lồi. Ý tưởng chính của phương pháp nhát cắt là để giải một bài toán quy hoạch tuyến tính nguyên, chúng ta sẽ giải một chuỗi các bài toán quy hoạch tuyến tính theo các bước như sau: (1) chuyển bài toán quy hoạch tuyến tính nguyên về bài toán quy hoạch tuyến tính bằng cách bỏ đi các ràng buộc nguyên (gọi là integer-relaxation); (2) tìm lời giải tối ưu x^* cho bài toán quy hoạch tuyến tính: nếu x^* là lời giải nguyên (integer solution) thì dừng lại và x^* chính là lời giải tối ưu cho bài toán quy hoạch tuyến tính nguyên, ngược lại, tạo ra một nhát cắt bằng cách sử dụng giải thuật (Gomory, 1958), để từ đó tạo ra một ràng buộc mới thoả mãn tất cả các lời giải nguyên khả thi khác x^*; (3) thêm ràng buộc mới này vào mô hình, quay trở lại bước 2 và giải lại bài toán.

2.3.1.3 Phương pháp nhánh cận

Phương pháp nhánh cận (branch-and-bound) là một phương pháp để giải các bài toán tối ưu tổ hợp (combinatorial optimization problems). Việc đánh giá được thực hiện theo từng bước, nếu không có khả năng tìm thấy kết quả tốt hơn thì sẽ cắt bỏ nhánh đó, không thực hiện tìm tiếp mà chuyển sang nhánh khác. Lời giải của bài toán sẽ tốt dần lên do khi tìm ra kết quả tốt hơn thì sẽ cập nhật lại giá trị hiện thời của bài toán. Phương pháp này được giới thiệu lần đầu tiên bởi (Land & Doig, 1960) cho quy hoạch rời rạc (discrete programming), và tên "branch-and-bound" được xuất hiện lần đầu tiên tại công bố của (Little et al., 1963) về bài toán người đi giao hàng.

2.3.2 Phương pháp xấp xỉ

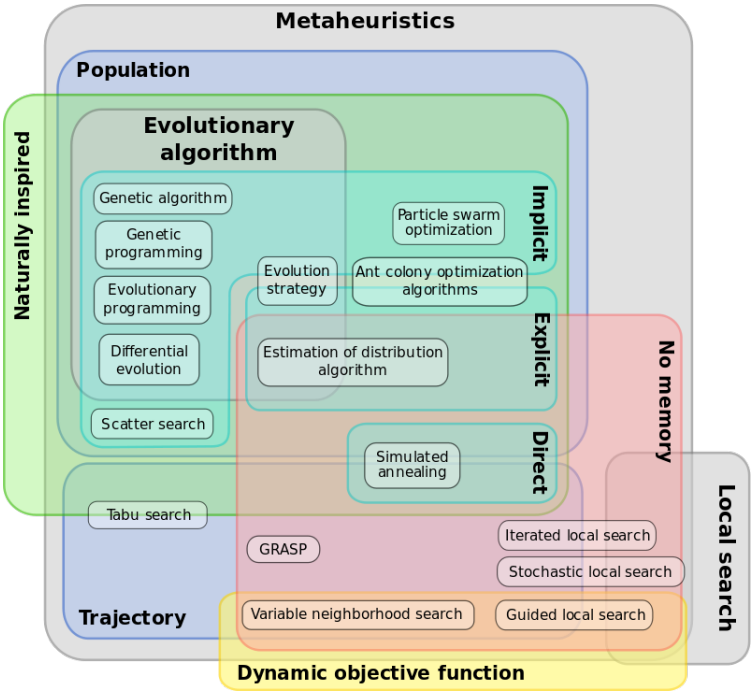
2.3.2.1 Phương pháp heuristic

Heuristic là các kỹ thuật dựa trên kinh nghiệm để giải quyết vấn đề, học hỏi hay khám phá nhằm đưa ra một lời giải mà không được đảm bảo là tối ưu. Các phương pháp heuristic được dùng nhằm tăng nhanh quá trình tìm kiếm với các lời giải hợp lý thông qua các suy

nghĩ rút gọn để giảm bớt việc nhận thức vấn đề khi đưa ra quyết định (Myers, 2010). Có nhiều phương pháp để xây dựng một giải thuật heuristic, trong đó thường dựa vào một số nguyên lý cơ sở như nguyên lý vét cạn (brute-force), nguyên lý tham lam (greedy), nguyên lý thứ tự (order), ... Theo (Swamidass, 2000), nguyên lý thứ tự trong bài toán lập lịch được thể hiện bởi các quy tắc lập lịch ưu tiên (priority scheduling rules) như First Come First Serve (FCFS), Shortest Processing Time (SPT), Longest Processing Time (LPT), Earliest Due Date (EDD), Slack Time Remaining (STR) = due date - processing time, Critical Ratio (CR) = due date / processing time.

2.3.2.2 Phương pháp metaheuristic

Theo (Blum & Roli, 2003), metaheuristic là các chiến lược cấp cao để khai phá các không gian tìm kiếm bằng cách sử dụng các phương pháp khác nhau được trình bày trong Hình 2.7.



Hình 2.7: Các phương pháp được sử dụng trong metaheuristic [1]

Sự khác nhau cơ bản giữa heuristic và metaheuristic được (Sörensen & Glover, 2013) nêu ra đó là heuristic là một giải thuật phụ thuộc vào bài toán, nghĩa là chúng ta phải xác định một heuristic cho một bài toán cụ thể cho trước, trong khi metaheuristic là một khung giải thuật (algorithmic framework) độc lập với bài toán, cung cấp các hướng dẫn (guidelines) hoặc các chiến lược (strategies) để phát triển các giải thuật heuristics bên trong.

[1] J. Dréo, C. Candan, *Different classifications of metaheuristics*, 2011. [Online]. Available: https://commons.wikimedia.org/wiki/File:Metaheuristics_classification.svg (visited on 08/28/2011).

2.3.2.3 Phương pháp matheuristic

Theo (Fischetti, 2016), matheuristic là sự lai tạo (hybridization) giữa mathematical programming (MP) với (meta)heuristic. Thuật ngữ "model-based metaheuristics" đã xuất hiện trong nhiều tiêu đề của một chuỗi hội nghị quốc tế dành riêng cho matheuristics như Matheuristics 2006 & 2008 - Bertinoro (Italia), Matheuristics 2010 - Vienna (Austria), Matheuristics 2012 - Rio de Janeiro (Brazil), Matheuristics 2014 - Hamburg (Germany), Matheuristics 2016 - Brussel (Belgium), Matheuristics 2018 - Tours (France). Theo khảo sát của (Ball, 2011), cũng như khảo sát của (Archetti & Speranza, 2014) về việc sử dụng phương pháp (meta)heuristic kết hợp với MP được phân thành ba nhóm: (1) phương pháp phân rã (decomposition approaches): bài toán ban đầu được chia thành các bài toán con nhỏ và đơn giản hơn có thể được giải quyết thông qua mô hình MP; (2) phương pháp cải tiến heuristics (improvement heuristics approaches): kết hợp phương pháp heuristic với lời giải chính xác của mô hình MP nhằm mục đích cải thiện lời giải đạt được; (3) phương pháp cải tiến nhánh (improvement branching approaches): các phương pháp chính xác về nhánh được sửa đổi để tăng tốc độ hội tụ (ví dụ như dừng sớm giai đoạn tạo cột) để tạo ra lời giải gần đúng. Hiện nay phương pháp matheuristic được rất nhiều nhà khoa học nghiên cứu cho nhiều bài toán tối ưu ứng dụng trong nhiều lĩnh vực khác nhau. Cụ thể như nhóm bài toán routing problems bao gồm vehicle routing problem, inventory routing problem, production routing problem, location routing problem đã có rất nhiều công bố liên quan tới phương pháp tiếp cận matheuristic.

CHƯƠNG 3 BÀI TOÁN LẬP LỊCH CÔNG VIỆC CÁ NHÂN

3.1 Đặc tả bài toán

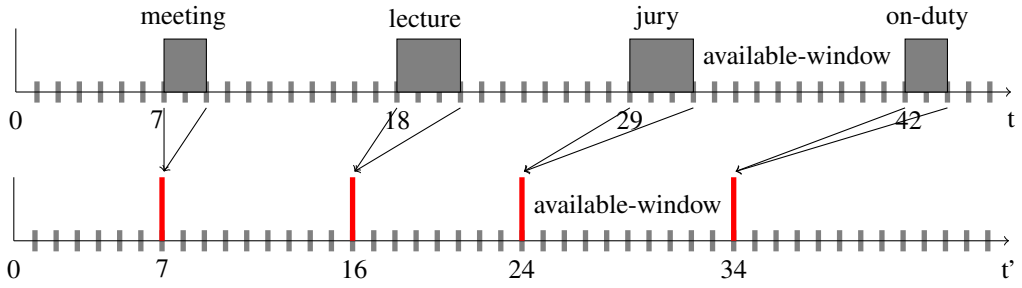
3.1.1 Phát biểu bài toán

Bài toán lập lịch công việc cá nhân là bài toán lập lịch các công việc ***có thể cắt nhỏ bị chặn dưới*** *vào những* ***khung thời gian trống*** *sao cho thời điểm hoàn thành tất cả các công việc là nhỏ nhất (gọi là bài toán PSP).*

Bài toán PSP có ba đối tượng sau:

- Mỗi cá nhân (gọi là machine) là đối tượng cần được xếp lịch.
- Các công việc cần phải lên lịch (gọi là job), mỗi công việc đều có thời gian thực thi (gọi là processing time), thời gian chuẩn bị (gọi là setup time), thời điểm bắt buộc phải hoàn thành (gọi là deadline), ...

- Những khung thời gian trống có thể sắp xếp các công việc vào đó (gọi là available time-window) và những khung thời gian không trống hoặc không cần sắp lịch (gọi là unavailable time-window). Để đơn giản cho việc mô hình hóa bài toán, những khung thời gian không trống hoặc không cần lên lịch (unavailable time-window) được thu giảm thành những cột mốc (gọi là break-time), xem ví dụ tại Hình 3.1.



Hình 3.1: Các khung cửa sổ thời gian sau khi được thu giảm

Hai ràng buộc của bài toán PSP là:

Ràng buộc 1 *Các jobs có thể được cắt ra thành những phần nhỏ (gọi là sub-job) nhưng không thể nhỏ hơn một ngưỡng xác định (gọi là $split_{min}$).*

Ràng buộc 2 *Các jobs/sub-jobs chỉ được sắp xếp vào những khung thời gian trống mà không được cắt qua các cột mốc break-time.*

Bài toán PSP được mô tả như sau:

- Dữ liệu:
  - một tập hợp n công việc $J = \{J_1, \ldots, J_n\}$ và một giá trị $split_{min}$.
  - một tập hợp m khung thời gian trống $W = \{W_1, \ldots, W_m\}$.
- Ràng buộc:
  - các công việc có thể được cắt nhỏ (splittable) nhưng không thể nhỏ hơn $split_{min}$.
  - các công việc chỉ được sắp xếp vào những khung thời gian trống (available-windows).
- Câu hỏi: xác định một lịch trình sắp xếp các công việc vào những khung thời gian trống sao cho thời điểm hoàn thành tất cả các công việc C_{max} là nhỏ nhất?

Bài toán PSP được ký hiệu như sau:

$$1|splittable; split_{min}; available - windows|C_{max}$$

3.1.2 Minh họa bài toán

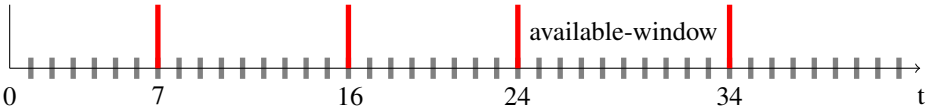
Bảng 3.1: Dữ liệu đầu vào cho bài toán PSP

(a) Jobs ($split_{min} = 3$)

Job	Processing time
J_1	5
J_2	8
J_3	4
J_4	6
J_5	13

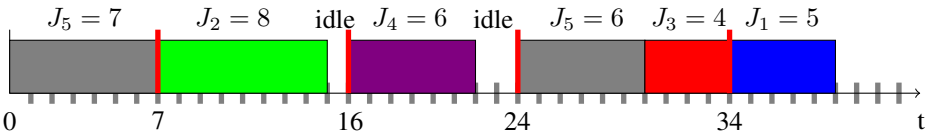
(b) Windows

Window	Available time
W_1	$[0, 7]$
W_2	$[7, 16]$
W_3	$[16, 24]$
W_4	$[24, 34]$
W_5	$[34, +\infty)$

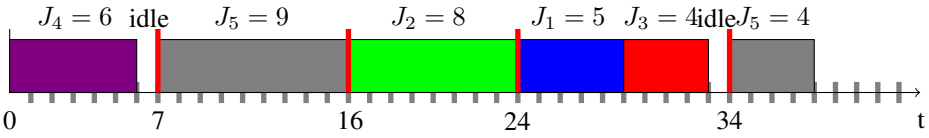


Hình 3.2: Các khung cửa sổ thời gian của bài toán PSP

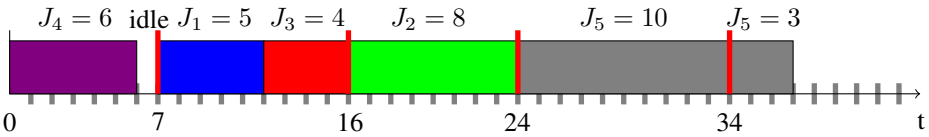
Các lời giải có thể có của bài toán PSP tương ứng với dữ liệu đầu vào tại Bảng 3.1:



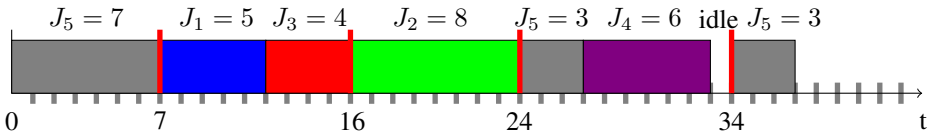
Hình 3.3: Một lời giải khả thi với $C_{max} = 39$



Hình 3.4: Một lời giải khả thi tốt hơn với $C_{max} = 38$



Hình 3.5: Một lời giải tối ưu với $C^*_{max} = 37$



Hình 3.6: Một lời giải tối ưu khác với $C^*_{max} = 37$

Qua ví dụ minh hoạ trên chúng ta có ba nhận xét sau:

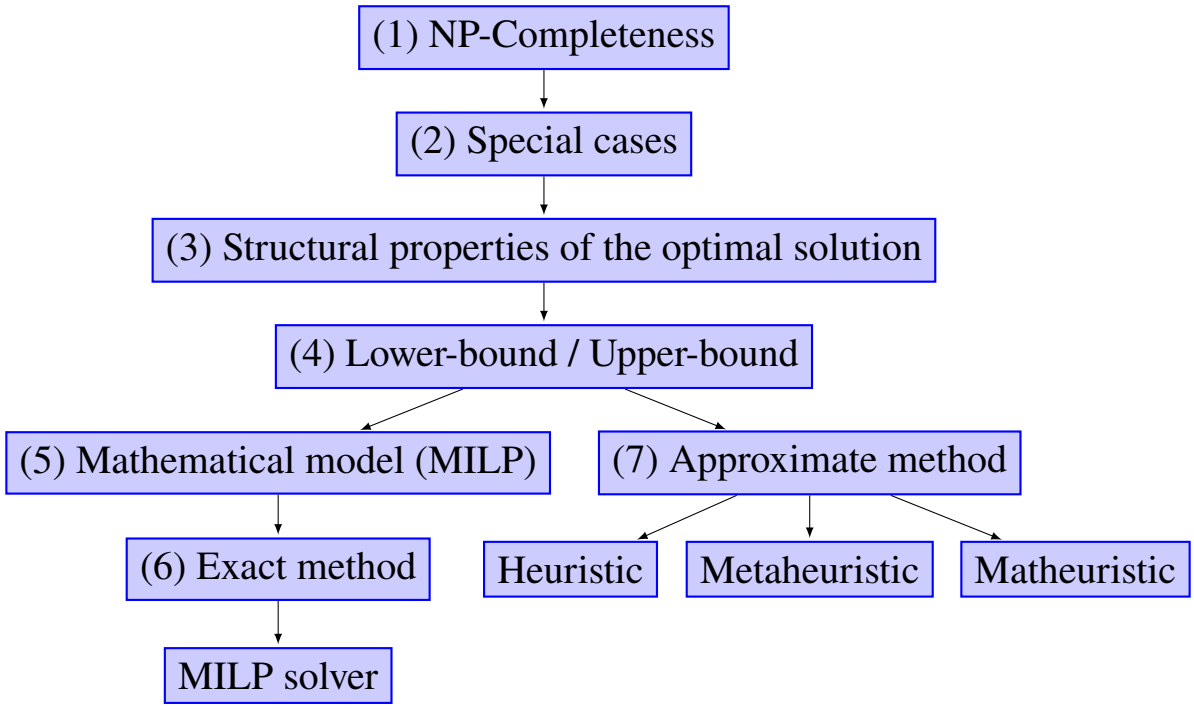
Nhận xét 3.1 *Bài toán luôn có lời giải khả thi bởi vì kích thước khung cửa sổ cuối cùng là không giới hạn.*

Nhận xét 3.2 *Bài toán có thể có nhiều lời giải tối ưu với giá trị C^*_{max} bằng nhau nhỏ nhất.*

Nhận xét 3.3 *Một lời giải không có idle-time chắc chắn là một lời giải tối ưu, nhưng điều ngược lại không chắc chắn đúng bởi vì lời giải tối ưu trong một số trường hợp vẫn có idle-time.*

Từ nhận xét 3.3 cho thấy độ khó của bài toán, là khi chúng ta tìm ra được một lời giải với giá trị C_{max} tương ứng, thì chúng ta không biết được liệu đây có phải là lời giải tối ưu hay không tương ứng với giá trị C_{max} nhỏ nhất hay chưa?

3.2 Các phương pháp tiếp cận



Hình 3.7: Sơ đồ các bước tiếp cận cho các bài toán lập lịch đang nghiên cứu

3.2.1 Độ khó bài toán

Nhóm tác giả (Huy et al., 2013a) đã chứng minh bài toán PSP thuộc lớp strongly NP-hard theo các bước:

(i) Từ nhận xét 2.1, phát biểu bài toán quyết định Splitsche tương ứng với bài toán tối ưu PSP.

(ii) Chứng minh bài toán quyết định Splitsche thuộc lớp strongly NP-complete bằng cách đưa ra một sự thu giảm đa thức từ bài toán quyết định 3-Partition đến bài toán quyết định Splitsche (3-Partition $\leq_p$ Splitsche).

(iii) Từ nhận xét 2.2, do bài toán quyết định Splitsche thuộc lớp strongly NP-complete nên bài toán tối ưu PSP thuộc lớp strongly NP-hard.

3.2.2 Một số trường hợp đặc biệt

3.2.2.1 Trường hợp các công việc không thể cắt nhỏ

Trong trường hợp tất cả các công việc đều không thể cắt nhỏ ($p_i < 2 \times split_{min}$), bài toán được ký hiệu như sau: $1|available - windows|C_{max}$.

Theo (Ji et al., 2007), bài toán này có $C_{LPT} \leq 2 \times C_{OPT}$. Như vậy tỷ lệ trường hợp xấu nhất của lời giải đạt được bằng giải thuật LPT so với lời giải tối ưu là 2.

3.2.2.2 Trường hợp các khung cửa sổ có kích thước bằng nhau

Trong trường hợp tất cả các khung cửa sổ đều có kích thước bằng nhau ($w_t = w$), bài toán được ký hiệu như sau: $1|splittable; split_{min}; available - windows; w_t = w|C_{max}$.

Khi đó bài toán này gần giống với bài toán Bin-packing, với các sự tương ứng như sau: các items $\sim$ các jobs/sub-jobs có kích thước trong khoảng $[split_{min}, 2 \times split_{min})$, các bins $\sim$ các windows có sức chứa là w, giải thuật FFD $\sim$ giải thuật LPT.

Theo (Dosa et al., 2013) áp dụng giải thuật FFD cho bài toán Bin-packing, trong đó b là số lượng bin được tìm thấy bởi giải thuật FFD và b^* là số lượng bin tối ưu, ta có: $b \leq \frac{11}{9}b^* + \frac{6}{9}$. Sau đó qua một số bước tính toán, ta được: $C_{LPT} \leq \frac{28}{9}C_{OPT}$. Như vậy tỷ lệ trường hợp xấu nhất của lời giải đạt được bằng giải thuật LPT so với lời giải tối ưu là $\frac{28}{9}$.

3.2.3 Các tính chất của lời giải tối ưu

Sáu tính chất của lời giải tối ưu của bài toán PSP đã được trình bày trong (Huy et al., 2013a, 2013b), đó là tồn tại một lời giải tối ưu sao cho trong một khung cửa sổ thời gian:

Tính chất 1 *Thứ tự các jobs/sub-jobs được sắp xếp tùy ý.*

Tính chất 2 *Chỉ có 0 hoặc 1 sub-job thuộc cùng một job.*

Tính chất 3 *Có nhiều nhất một idle-time.*

Tính chất 4 *Nếu có idle-time thì nên ở cuối khung cửa sổ thời gian.*

Tính chất 5 *Không có idle-time nào có kích thước lớn hơn hoặc bằng* $2 \times split_{min}$.

Tính chất 6 *Một job có thể được chia nhỏ tối đa là* $nsub_i = \min(\left\lfloor \frac{p_i}{split_{min}} \right\rfloor, m)$.

Ngoài ra, một tính chất nữa được bổ sung thêm trong luận án, đó là tồn tại một lời giải tối ưu sao cho:

Tính chất 7 *Thứ tự các jobs/sub-jobs có cùng kích thước được sắp xếp tùy ý.*

3.2.4 Miền đánh giá nghiệm

Theo (Lane & Birkhoff, 1999) thì giá trị C_{max} của các lời giải sẽ nằm trong khoảng:

$$LB \leq GLB \leq C^*_{max} \leq C_{max} \leq LUB \leq UB$$

Đối với lower-bound LB, dựa vào nhận xét 3.3 thì một LB đề xuất là:

$$LB = \sum_{i=1}^{n} p_i$$

Đối với upper-bound UB, trường hợp xấu nhất của một lời giải đó là tất cả các khung cửa sổ đều chứa idle-time (trừ khung cửa sổ cuối cùng), khi đó dựa vào tính chất 5 thì một UB đề xuất là:

$$UB = LB + (m - 1) \times (2 \times split_{min})$$

3.2.5 Mô hình MILP

3.2.5.1 Mô hình MILP 1

Dựa vào các tính chất được trình bày tại mục 3.2.3, một lời giải của bài toán được xác định bằng cách trả lời hai câu hỏi sau: (1) một job/sub-job có được gán cho một window không? (2) nếu có thì kích thước của job/sub-job này là bao nhiêu? Để trả lời hai câu hỏi này, mô hình MILP 1 được đề xuất trong (Huy et al., 2013b) sẽ có hai biến quyết định là $x_{i,t} \in \{0, 1\}$ (tương ứng cho câu hỏi số 1) và $y_{i,t} \in \mathbb{N}$ (tương ứng cho câu hỏi số 2), cụ thể như sau:

- Biến quyết định: $x_{i,t} \in \{0,1\}$, $y_{i,t} \in \mathbb{N}$
- Biến trung gian: $\alpha_t = \left\lceil \frac{\sum_{i=1}^{n} x_{i,t}}{n} \right\rceil \in \{0,1\}$, $\beta_t = \alpha_t \times b_{t-1} \in \mathbb{N}$, $\gamma_t = \beta_t + \sum_{i=1}^{n} y_{i,t} \in \mathbb{N}$, $C_{max} = \max(\gamma_t) \in \mathbb{N}$
- Hàm mục tiêu: $\min(C_{max})$
- Các ràng buộc:

$$\sum_{t=1}^{m} y_{i,t} = p_i; \quad \forall i = 1, \ldots, n \tag{3.1}$$

$$\sum_{i=1}^{n} y_{i,t} \leq w_t; \quad \forall t = 1, \ldots, m \tag{3.2}$$

$$split_{min} \times x_{i,t} \leq y_{i,t} \leq p_i \times x_{i,t}; \quad \forall i = 1, \ldots, n; \quad \forall t = 1, \ldots, m \tag{3.3}$$

3.2.5.2 Mô hình MILP 2

Đối với mô hình MILP 1, chúng ta sẽ biết được một job có được gán vào một window hay không và kích cỡ của job này là bao nhiêu nếu được gán vào window đó. Tuy nhiên chúng ta sẽ không biết được thêm thông tin là job này được gán vào vị trí nào và sẽ kết thúc ở đâu trong window này. Để biết thêm những thông tin này thì mô hình MILP 2 được đề xuất trong (Son et al., 2021) sẽ có thêm một biến quyết định $s_{i,t} \in \mathbb{N}$ và một biến trung gian $c_{i,t} = s_{i,t} + y_{i,t} \in \mathbb{N}$, cụ thể như sau:

- Các biến quyết định: $x_{i,t} \in \{0,1\}$, $y_{i,t} \in \mathbb{N}$, $s_{i,t} \in \mathbb{N}$, $v_{i,j,t} \in \{0,1\}$
- Các biến trung gian: $c_{i,t} = s_{i,t} + y_{i,t} \in \mathbb{N}$, $C_i = \max(c_{i,t}) \in \mathbb{N}$, $C_{max} = \max(C_i) \in \mathbb{N}$
- Hàm mục tiêu: $\min(C_{max})$
- Các ràng buộc:

$$\sum_{t=1}^{m} y_{i,t} = p_i; \quad \forall i = 1, \ldots, n \tag{3.4}$$

$$\sum_{i=1}^{n} y_{i,t} \leq w_t; \quad \forall t = 1, \ldots, m \tag{3.5}$$

$$split_{min} \times x_{i,t} \leq y_{i,t} \leq p_i \times x_{i,t}; \quad \forall i = 1, \ldots, n; \quad \forall t = 1, \ldots, m \tag{3.6}$$

$$b_t \times x_{i,t} \leq s_{i,t} \leq UB \times x_{i,t}; \quad \forall i = 1, \ldots, n; \quad \forall t = 1, \ldots, m \tag{3.7}$$

$$b_t \leq s_{i,t} \leq b_{t+1} - y_{i,t}; \quad \forall i = 1, \ldots, n; \quad \forall t = 1, \ldots, m \tag{3.8}$$

$$\begin{cases} c_{i,t} - s_{j,t} \leq UB \times v_{i,j,t}; & \forall i, j = 1, \ldots, n \ / \ i \neq j; \quad \forall t = 1, \ldots, m \\ c_{j,t} - s_{i,t} \leq UB \times (1 - v_{i,j,t}); & \forall i, j = 1, \ldots, n \ / \ i \neq j; \quad \forall t = 1, \ldots, m \end{cases} \tag{3.9}$$

Bảng 3.2: So sánh MILP 1 và MILP 2

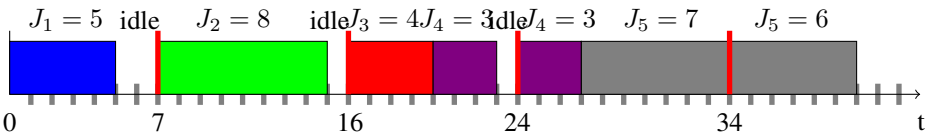
	MILP 1	MILP 2
Nhóm biến quyết định	$x_{i,t}$, $y_{i,t}$	$x_{i,t}$, $y_{i,t}$, $s_{i,t}$
Nhóm biến trung gian	α_t, β_t, γ_t, C_{max}	$c_{i,t}$, C_i, C_{max}
Nhóm ràng buộc	3 nhóm (3.1 - 3.3)	6 nhóm (3.4 - 3.9)
Hàm mục tiêu	$\min(C_{max})$	$\min(C_{max})$
Số lượng biến quyết định	$2 \times n \times m$	$3 \times n \times m$
Số lượng ràng buộc	$n + m + (2 \times n \times m)$	$n + m + (8 \times n \times m)$
Không gian nghiệm	$O(2^{n \times m} \times UB^{n \times m})$	$O(2^{n \times m} \times UB^{2 \times n \times m})$
Phù hợp với các bài toán	không quan tâm thứ tự của các jobs trong một khung cửa sổ thời gian	quan tâm thứ tự của các jobs trong một khung cửa sổ thời gian
Khả năng mở rộng (r_i, $\overline{d}_i$, $\sum C_i$)	thêm biến quyết định và ràng buộc	thêm ràng buộc

3.2.6 Phương pháp heuristic

3.2.6.1 Phân bổ dựa trên quy tắc FCFS

Ý tưởng chính của giải thuật Assignment based on FCFS (ASGN) là duyệt qua từng window từ trái sang phải; tại mỗi window, duyệt qua từng job theo quy tắc FCFS; tại mỗi job, cố gắng gán job này vào window tương ứng tuỳ theo kích thước trống của window (chi tiết trong Algorithm 3, 4).

Minh hoạ Giải thuật ASGN với dữ liệu tại Bảng 3.1.



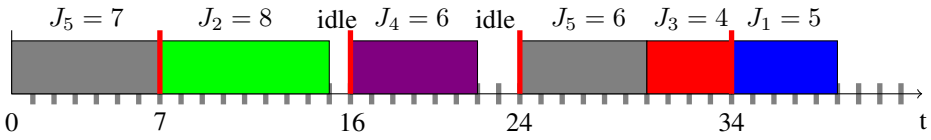
Hình 3.8: Giải thuật ASGN với $C_{max} = 40$

3.2.6.2 Phân bổ dựa trên quy tắc SPT/LPT

Ý tưởng chính của giải thuật Assignment based on SPT (ASPT) / Assignment based on LPT (ALPT) tương tự như giải thuật ASGN. Điểm khác biệt duy nhất là ở mỗi window, các jobs được sắp xếp theo thời gian xử lý tăng/giảm dần. Điều này có nghĩa là các jobs

có thời gian xử lý nhỏ/lớn hơn sẽ được ưu tiên để gán vào các windows trống (chi tiết trong Algorithm 5).

Minh hoạ Giải thuật ALPT với dữ liệu tại Bảng 3.1.

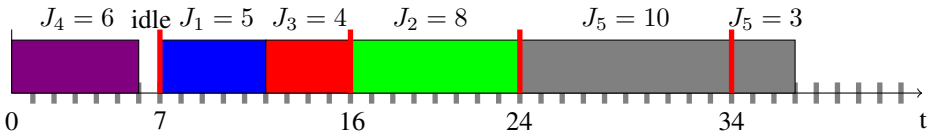


Hình 3.9: Giải thuật ALPT với $C_{max} = 39$

3.2.6.3 So khớp và phân bổ

Lưu ý rằng trong việc giảm thiểu "makepan", lời giải nào có ít thời gian rảnh rỗi idle-time hơn thì sẽ tốt hơn. Do đó, việc cố gắng lấp đầy các windows bằng các jobs sẽ làm giảm thời gian rảnh rỗi trong lời giải. Giải thuật Matching and Assignment (MAAS) sẽ lấp đầy các windows bằng cách so khớp (matching) kích thước của từng window với kích cỡ của các jobs trước, sau đó các windows trống còn lại sẽ lần lượt phân bổ (assignment) các jobs vào. Ý tưởng ban đầu của heuristic này được chia thành hai bước chính là matching, assignment, và một bước kiểm tra bổ sung (verification) (chi tiết trong Algorithm 6, 7, 8, 9, 10).

Minh hoạ Giải thuật MAAS với dữ liệu tại Bảng 3.1.

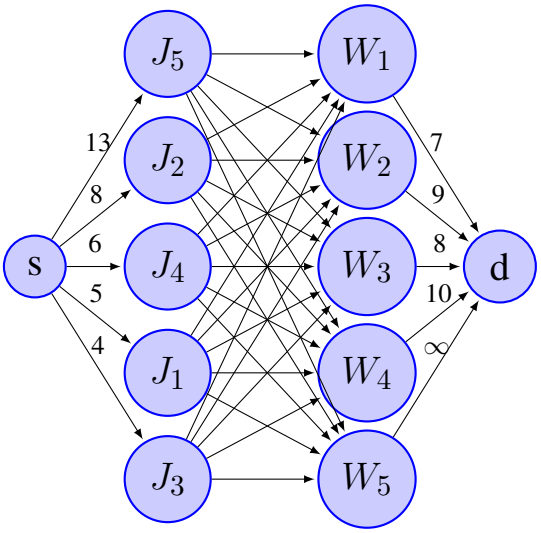


Hình 3.10: Giải thuật MAAS với $C_{max} = 37$

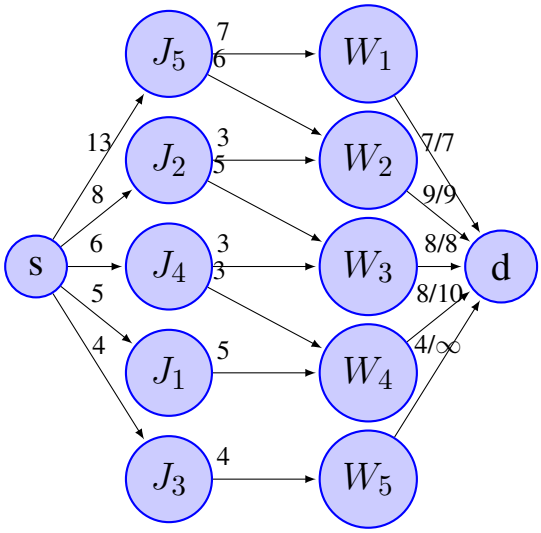
3.2.6.4 Phân bổ dựa trên luồng cực đại

Việc xác định các jobs có kích thước bao nhiêu được gán vào window nào gần giống như việc xác định dòng chảy (flow) có kích thước bao nhiêu đi qua các đỉnh nào của đồ thị trong bài toán luồng cực đại (Max Flow problem), ở đây chúng ta không quan tâm thứ tự job được gán vào window hay thứ tự dòng chảy qua đỉnh đồ thị. Chính vì vậy, ý tưởng của giải thuật Assignment based on Max Flow (BMF) là chuyển đổi bài toán PSP về bài toán Max Flow có thêm một số ràng buộc liên quan đến $split_{min}$, và có thể sử dụng giải thuật Ford-Fulkerson (có hiệu chỉnh) để xác định lời giải cho bài toán PSP.

Minh hoạ Đồ thị N có trọng số được xây dựng với dữ liệu tại Bảng 3.1.



Hình 3.11: Đồ thị N



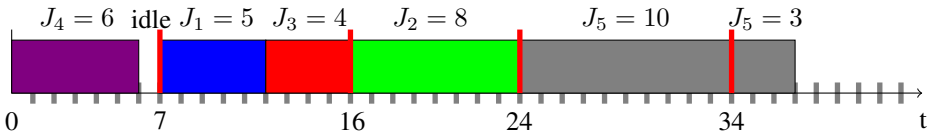
Hình 3.12: Flow trên đồ thị N với $C_{max} = 38$

3.2.7 Phương pháp metaheuristic

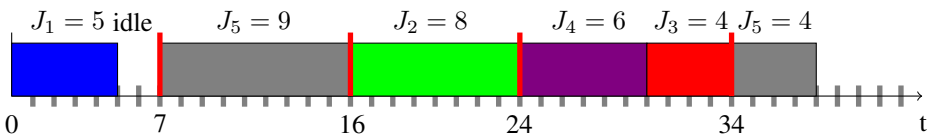
Quan sát 1 *Thứ tự của các jobs sẽ ảnh hưởng đến lời giải đạt được bởi các giải thuật như ASGN, BMF và MAAS.*

Minh hoạ Giải thuật MAAS với dữ liệu tại Bảng 3.1.

- $J = \{J_1 = 5, J_2 = 8, J_3 = 4, J_4 = 6, J_5 = 13\} \Longrightarrow C_{max} = 37$



- $J = \{J_4 = 6, J_2 = 8, J_3 = 4, J_1 = 5, J_5 = 13\} \Longrightarrow C_{max} = 38$



Hình 3.13: Giải thuật MAAS với thứ tự của các jobs khác nhau

Dựa trên quan sát 1, một cách tiếp cận khác được đề xuất để giải quyết bài toán này là cố gắng thay đổi thứ tự của các jobs để tìm ra các lời giải tốt hơn. Phương pháp này sẽ đưa bài toán đang xem xét về dạng bài toán tối ưu tổ hợp với độ thích nghi (fitness value) chính là giá trị C_{max}. Có nhiều metaheuristics có thể được áp dụng để giải quyết bài toán

tối ưu tổ hợp này như Simulated Annealing (SA), Genetic Algorithm (GA), Tabu Search (TABU), ... Một cách tổng quát thì các metaheuristics như GA hoặc TABU có thể được thực hiện bằng cách xem xét thứ tự của các jobs như một nhiễm sắc thể (chromosome) trong GA hoặc một lời giải mã hoá (solution encoding) trong TABU.

Minh hoạ Cách mã hóa trong GA và TABU với dữ liệu tại Bảng 3.1.

Chromosome 1	J_2	J_3	J_5	J_4	J_1
Chromosome 2	J_3	J_2	J_4	J_1	J_5

Hình 3.14: Mã hóa chromosome trong GA

Solution encoding 1	J_2	J_3	J_5	J_4	J_1
Solution encoding 2	J_3	J_2	J_4	J_1	J_5

Hình 3.15: Mã hóa solution trong TABU

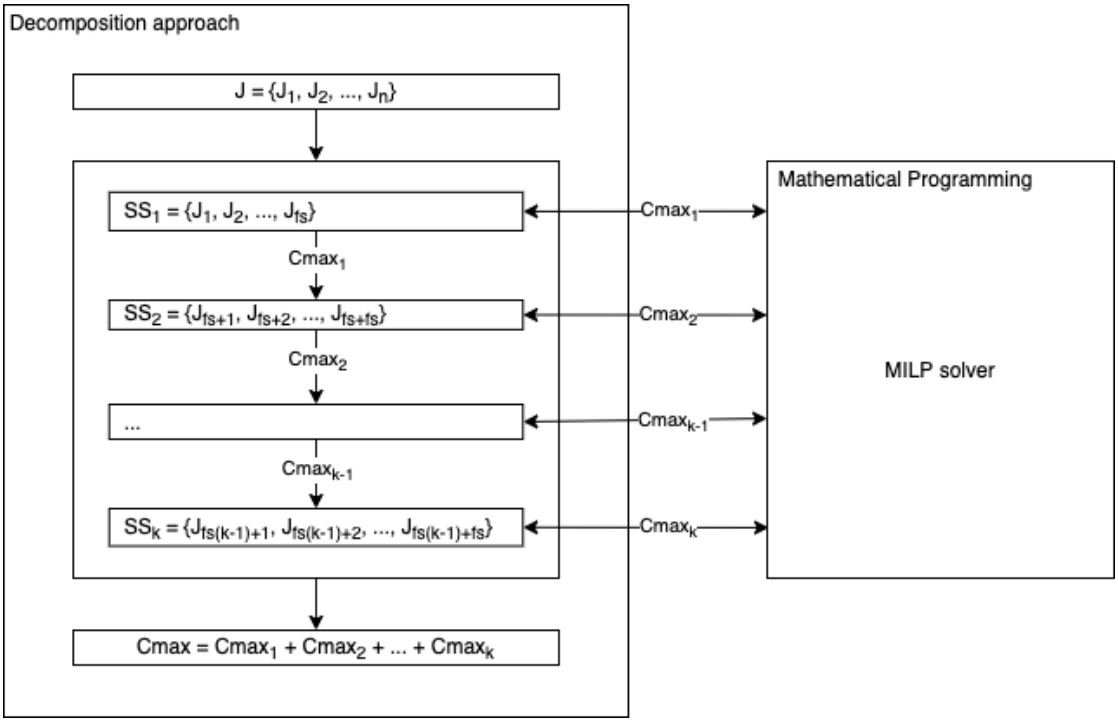
Tuy nhiên với một thứ tự của các jobs cho trước, không thể xác định trực tiếp một lời giải cụ thể cho bài toán đang xem xét. Nhờ có quan sát 1, một cách tiếp cận được đề xuất bằng cách pha trộn (mixing) giữa một metaheuristic như GA hoặc TABU với một heuristic đã được đề xuất ở trên. Do đó chúng tôi có quan sát tiếp theo, đó là:

Quan sát 2 *Một thuật toán tiến hóa (evolutionary algorithm) có thể được áp dụng để xác định lời giải cho bài toán bằng cách kết hợp một metaheuristic và một heuristic đã được đề xuất.*

Ý tưởng chính của thuật toán tiến hóa đề xuất là sử dụng các chiến lược lặp đối với TABU hoặc chiến lược tiến hoá đối với GA để tạo ra các thế hệ mới tốt hơn so với thế hệ trước. Tại mỗi lần lặp hoặc tiến hóa, mỗi lời giải mã hoá sẽ có một giá trị thích nghi tương ứng với giá trị C_{max} được tính bởi một heuristic đề xuất. Sau đó ở mỗi lần lặp hoặc tiến hóa tiếp theo, các lời giải mã hoá mới được tạo ra bằng cách sử dụng các thao tác như SWAP, EBSR, EFSR trong TABU hoặc selection, crossover, mutation trong GA.

3.2.8 Phương pháp matheuristic

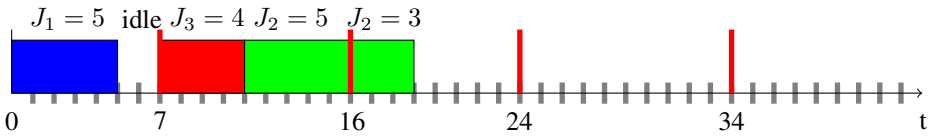
Giải thuật matheuristic đề xuất có tên gọi là Exact for SubSet-Jobs (E4SSJ) với ý tưởng là chia tập các jobs đầu vào ban đầu thành các tập các jobs nhỏ hơn không giao nhau (subset-jobs), sau đó sử dụng MILP solver để xác định lời giải tối ưu trên các subset-jobs này, và lời giải cuối cùng của bài toán chính là tổng hợp các lời giải tối ưu cho từng subset-jobs (chi tiết trong Algorithm 11). Hình 3.16 trình bày ý tưởng của giải thuật E4SSJ.



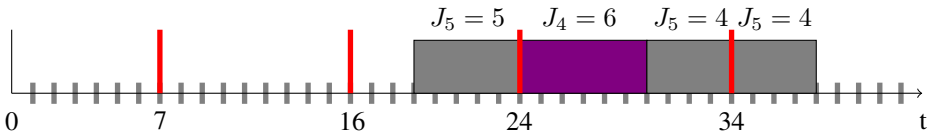
Hình 3.16: Giải thuật E4SSJ

Minh hoạ Giải thuật E4SSJ với dữ liệu tại Bảng 3.1, cho trước fixed-size $= 3$ và áp dụng quy tắc lập lịch ưu tiên FCFS, chia tập jobs ban đầu $J = \{J_1 = 5, J_2 = 8, J_3 = 4, J_4 = 6, J_5 = 13\}$ thành 2 subset-jobs $SS_1 = \{J_1 = 5, J_2 = 8, J_3 = 4\}, SS_2 = \{J_4 = 6, J_5 = 13\}$, sau đó sử dụng MILP solver để xác định 2 lời giải tối ưu cho 2 subset-jobs này, ta được:

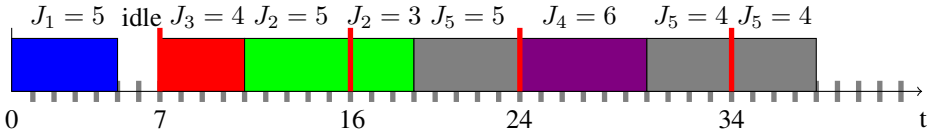
- $SS_1 = \{J_1 = 5, J_2 = 8, J_3 = 4\} \Longrightarrow C_{max_1} = 19$



- $SS_2 = \{J_4 = 6, J_5 = 13\} \Longrightarrow C_{max_2} = 19$



- $J = \{J_1 = 5, J_2 = 8, J_3 = 4, J_4 = 6, J_5 = 13\} \Longrightarrow C_{max} = C_{max_1} + C_{max_2} = 38$



Hình 3.17: Giải thuật E4SSJ với $C_{max} = 38$

3.3 Đánh giá

Trong trường hợp dữ liệu đầu vào có kích thước nhỏ (khoảng dưới 40 jobs) nên chọn phương pháp chính xác sử dụng MILP solver để có thể xác định được lời giải tối ưu cho bài toán trong thời gian chấp nhận được (dưới 10 phút). Ngược lại, trong trường hợp dữ liệu đầu vào có kích thước lớn thì phương pháp matheuristic với giải thuật E4SSJ là lựa chọn tốt nhất vì chất lượng của lời giải được tìm thấy là rất tốt và thời gian tính toán lại nhanh.

CHƯƠNG 4 MỘT SỐ BÀI TOÁN LẬP LỊCH CÔNG VIỆC CÁ NHÂN ĐẶC THÙ

4.1 Bài toán lập lịch công việc nhóm

4.1.1 Phát biểu bài toán

Bài toán lập lịch công việc nhóm là bài toán PSP áp dụng trên một nhóm nhiều người có ***các khung thời gian làm việc khác nhau*** *(gọi là bài toán TWSP).*

Bài toán TWSP được ký hiệu như sau:

$$P|splittable; split_{min}; available - windows|C_{max}$$

4.1.2 Minh hoạ bài toán

Bảng 4.1: Dữ liệu đầu vào cho bài toán TWPSP

(a) Jobs ($split_{min} = 3$)

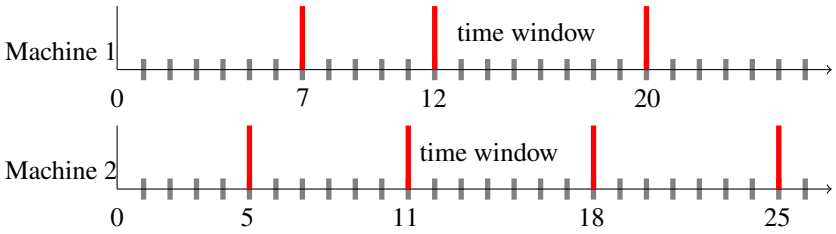
Job	Processing time
J_1	6
J_2	7
J_3	8
J_4	5
J_5	10
J_6	4

(b) Machine 1

Window	Available time
W_1^1	$[0, 7]$
W_2^1	$[7, 12]$
W_3^1	$[12, 20]$
W_4^1	$[20, +\infty)$

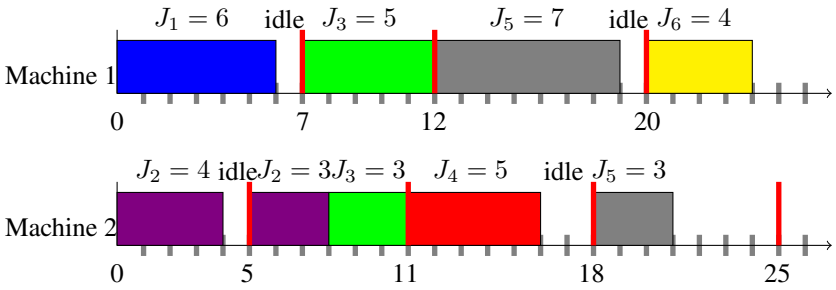
(c) Machine 2

Window	Available time
W_1^2	$[0, 5]$
W_2^2	$[5, 11]$
W_3^2	$[11, 18]$
W_4^2	$[18, 25)$
W_5^2	$[25, +\infty)$

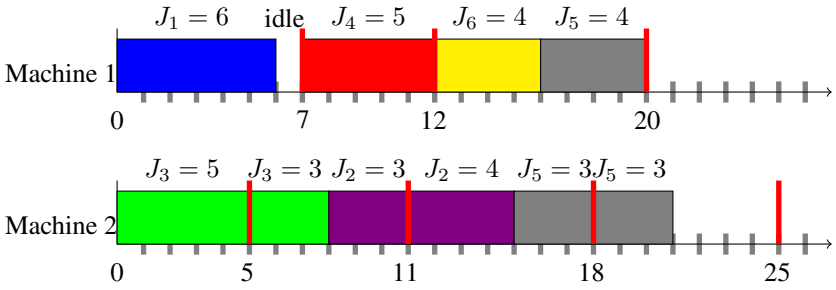


Hình 4.1: Các khung cửa sổ thời gian của bài toán TWSP

Các lời giải có thể có của bài toán TWSP tương ứng với dữ liệu đầu vào tại Bảng 4.1:



Hình 4.2: Một lời giải khả thi với $C_{max} = 24$



Hình 4.3: Một lời giải tối ưu với $C^*_{max} = 21$

Qua ví dụ minh hoạ trên chúng ta thấy bài toán TWSP cũng có ba đặc điểm giống các nhận xét 3.1, 3.2, 3.3 trong bài toán PSP.

4.1.3 Độ khó bài toán

Dễ dàng nhận thấy bài toán PSP: $1|splittable; split_{min}; available - windows|C_{max}$ là trường hợp đặc biệt của bài toán TWSP: $P|splittable; split_{min}; available-windows|C_{max}$ với số lượng $machine = 1$. Ta thấy các instances của bài toán PSP ($machine = 1$) chỉ là tập hợp con trong tập hợp các instances của bài toán TWSP ($machine = k$), và các instances của bài toán PSP này được ánh xạ $1 - 1$ đến các instances của bài toán TWSP, do đó: PSP $\leq_p$ TWSP. Mà bài toán PSP đã được chứng minh thuộc lớp strongly NP-hard tại mục 3.2.1 nên bài toán TWSP cũng thuộc lớp strongly NP-hard.

4.1.4 Các tính chất của lời giải tối ưu

Bài toán TWSP cũng có các tính chất của lời giải tối ưu giống bài toán PSP đã được trình bày tại mục 3.2.3, tuy nhiên tính chất 6 được thay đổi như sau:

Tính chất 6 *Tồn tại một lời giải tối ưu sao cho một job có thể được chia nhỏ tối đa là:*

$$nsub_i = \min(\left\lfloor \frac{p_i}{split_{min}} \right\rfloor, \sum_{j=1}^{k} m_j)$$

4.1.5 Miền đánh giá nghiệm

Dựa vào nhận xét 3.3 và tính chất 5 thì một LB và một UB đề xuất là:

$$LB = \left\lceil \frac{\sum_{i=1}^{n} p_i}{k} \right\rceil \qquad UB = LB + \left\lceil \frac{\sum_{j=1}^{k} (m_j - 1)}{k} \right\rceil \times (2 \times split_{min})$$

4.1.6 Mô hình MILP

Dựa vào các tính chất được trình bày tại mục 4.1.4, một lời giải của bài toán được xác định bằng cách trả lời ba câu hỏi sau: (1) một job/sub-job có được gán cho một window trên một machine không? (2) nếu có thì kích thước của job/sub-job này là bao nhiêu? (3) và nếu có thì thời điểm bắt đầu của job/sub-job này là lúc nào?

Để trả lời ba câu hỏi ở trên, mô hình MILP 5 cho bài toán TWSP được đề xuất sẽ có ba biến quyết định là $x_{i,j,t} \in \{0, 1\}$ (tương ứng cho câu hỏi số 1), $y_{i,j,t} \in \mathbb{N}$ (tương ứng cho câu hỏi số 2), và $s_{i,j,t} \in \mathbb{N}$ (tương ứng cho câu hỏi số 3), cụ thể như sau:

- Các biến quyết định: $x_{i,j,t} \in \{0, 1\}$, $y_{i,j,t} \in \mathbb{N}$, $s_{i,j,t} \in \mathbb{N}$, $v_{i_1,i_2,j,t} \in \{0, 1\}$
- Các biến trung gian: $c_{i,j,t} = s_{i,j,t} + y_{i,j,t} \in \mathbb{N}$, $C_i = \max(c_{i,j,t}) \in \mathbb{N}$, $C_{max} = \max(C_i) \in \mathbb{N}$
- Hàm mục tiêu: $\min(C_{max})$
- Các ràng buộc:

$$\sum_{j=1}^{k} \sum_{t=1}^{m} y_{i,j,t} = p_i; \quad \forall i = 1, \ldots, n \tag{4.1}$$

$$\sum_{i=1}^{n} y_{i,j,t} \leq w_{j,t}; \quad \forall j = 1, \ldots, k; \quad \forall t = 1, \ldots, m \tag{4.2}$$

$$split_{min} \times x_{i,j,t} \leq y_{i,j,t} \leq p_i \times x_{i,j,t};\ \forall i = 1, \ldots, n;\ \forall j = 1, \ldots, k;\ \forall t = 1, \ldots, m \tag{4.3}$$

$$b_{j,t} \times x_{i,j,t} \leq s_{i,j,t} \leq UB \times x_{i,j,t};\ \forall i = 1, \ldots, n;\ \forall j = 1, \ldots, k;\ \forall t = 1, \ldots, m \tag{4.4}$$

$$b_{j,t} \leq s_{i,j,t} \leq b_{j,t+1} - y_{i,j,t};\ \ \forall i = 1, \ldots, n;\ \ \forall j = 1, \ldots, k;\ \ \forall t = 1, \ldots, m \tag{4.5}$$

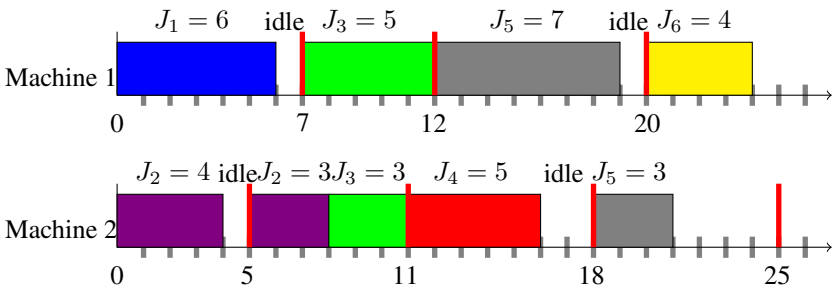
$$\begin{cases} c_{i_1,j,t} - s_{i_2,j,t} \leq UB \times v_{i_1,i_2,j,t}; & \forall i_1, i_2 = 1, \ldots, n \ / \ i_1 \neq i_2; \\ & \forall j = 1, \ldots, k;\ \forall t = 1, \ldots, m \\ c_{i_2,j,t} - s_{i_1,j,t} \leq UB \times (1 - v_{i_1,i_2,j,t}); & \forall i_1, i_2 = 1, \ldots, n \ / \ i_1 \neq i_2; \\ & \forall j = 1, \ldots, k;\ \forall t = 1, \ldots, m \end{cases} \tag{4.6}$$

4.1.7 Phương pháp heuristic

4.1.7.1 Phân bổ dựa trên quy tắc FCFS

Ý tưởng chính của giải thuật Assignment based on FCFS (ASGN) là duyệt lần lượt các jobs theo quy tắc FCFS; ứng với mỗi job, chọn window của machine hiện có thời điểm hoàn thành là ít nhất để gán job vào window này theo giải thuật AssignJob2Window (chi tiết trong Algorithm 18, 19).

Minh hoạ Giải thuật ASGN với dữ liệu tại Bảng 4.1.

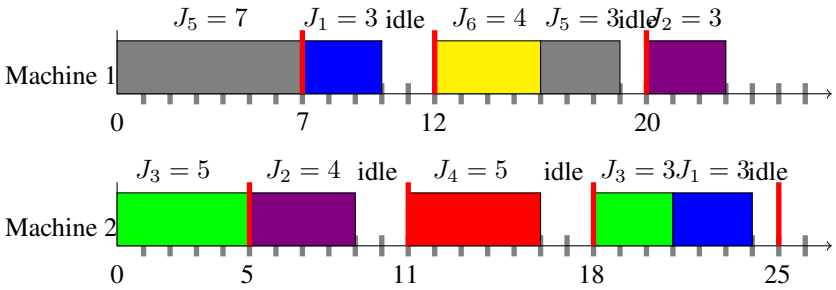


Hình 4.4: Giải thuật ASGN với $C_{max} = 24$

4.1.7.2 Phân bổ dựa trên quy tắc SPT/LPT

Ý tưởng chính của giải thuật Assignment based on SPT (ASPT) / Assignment based on LPT (ALPT) tương tự như giải thuật ASGN. Điểm khác biệt duy nhất là các jobs được sắp xếp theo thời gian xử lý tăng dần/giảm dần trước khi thực hiện việc duyệt lần lượt các jobs (chi tiết trong Algorithm 20).

Minh hoạ Giải thuật ALPT với dữ liệu tại Bảng 4.1.



Hình 4.5: Giải thuật ALPT với $C_{max} = 24$

4.1.8 Đánh giá

Trong trường hợp dữ liệu đầu vào có kích thước nhỏ (khoảng dưới 30 jobs) nên chọn phương pháp chính xác sử dụng MILP solver để có thể xác định được lời giải tối ưu cho bài toán trong thời gian chấp nhận được (dưới 10 phút). Ngược lại, trong trường hợp dữ liệu đầu vào có kích thước lớn thì giải thuật ALPT nên được lựa chọn vì chất lượng của lời giải khả thi được tìm thấy là tốt nhất trong số các giải thuật heuristics được đề xuất và thời gian tính toán lại rất nhanh.

CHƯƠNG 5 KẾT LUẬN

5.1 Đánh giá kết quả

Bài toán lập lịch công việc cá nhân có thể tự động phân chia các công việc nhỏ hơn (nhưng không nhỏ hơn một ngưỡng xác định trước) trong những khung thời gian làm việc là rất quan trọng để áp dụng cho các ứng dụng quản lý công việc cá nhân. Luận án đã tập trung giải quyết từ bài toán lập lịch công việc cá nhân cơ bản (chi tiết ở chương 3) cho đến một số bài toán lập lịch công việc cá nhân đặc thù với việc mở rộng các ràng buộc cho bài toán lập lịch công việc cá nhân cơ bản như ràng buộc setup-time, ràng buộc deadline, hoặc hướng đến việc khai thác những kết quả của bài toán lập lịch công việc cá nhân để mở rộng cho bài toán lập lịch công việc nhóm (chi tiết ở chương 4). Để từ đó các mục tiêu của luận án đã đạt được và các kết quả cũng đã được công bố tại các hội thảo, hội nghị, tạp chí trong nước và quốc tế, cụ thể như sau:

O1. Nghiên cứu và đề xuất một số phương pháp tiếp cận để giải quyết bài toán lập lịch công việc cá nhân cơ bản trên các bộ dữ liệu đầu vào có kích thước nhỏ và lớn, qua đó đề xuất lựa chọn phương pháp hiệu quả đối với từng loại dữ liệu đầu vào khác nhau [CT 1,4].

O2. Nghiên cứu giải quyết các bài toán lập lịch công việc cá nhân đặc thù có một số ràng buộc thường được sử dụng trong thực tế [CT 3,6].

O3. Nghiên cứu ứng dụng mở rộng bài toán lập lịch công việc cá nhân trên một nhóm nhiều người [CT 2,5].

5.2 Các đóng góp chính

Những đóng góp chính của luận án bao gồm:

C1. Xem xét và giải quyết những vấn đề tồn đọng của bài toán lập lịch công việc cá nhân (bài toán PSP) từ các nghiên cứu trước đây.

C2. Mở rộng nghiên cứu giải quyết các bài toán lập lịch công việc cá nhân đặc thù có một số ràng buộc thường được sử dụng trong thực tế như mỗi công việc đều có thời gian chuẩn bị khác nhau (bài toán PSP+setup-time), mỗi công việc đều có thời điểm bắt buộc phải hoàn thành khác nhau (bài toán PSP+deadline), và bài toán lập lịch công việc cá nhân cho một nhóm nhiều người có các khung thời gian làm việc khác nhau (bài toán TWSP).

C3. Đề xuất sơ đồ tổng quát 7 bước tiếp cận để giải quyết lớp các bài toán lập lịch công việc cá nhân bao gồm: (1) phân tích độ khó của bài toán, (2) xem xét một số trường hợp đặc biệt, (3) đưa ra một số tính chất của lời giải tối ưu, (4) xác định miền nghiệm của bài toán, (5) xây dựng mô hình MILP, (6) sử dụng phương pháp chính xác với MILP solver để xác định lời giải tối ưu cho bài toán, (7) sử dụng các phương pháp xấp xỉ dạng heuristic/metaheuristic/matheuristic để xác định lời giải khả thi cho bài toán.

C4. Trên cơ sở xem xét bài toán lập lịch công việc cá nhân như là một bài toán có thời gian là tài nguyên đặc biệt cần được cấp phát cho các công việc, đề xuất một số heuristics đặc thù thừa kế các giải thuật cổ điển, và một số metaheuristics thường dùng trong công nghiệp, qua đó xác định lời giải khả thi có chất lượng tốt trong giới hạn thời gian chấp nhận được (dưới 10 phút).

C5. Đề xuất hướng tiếp cận matheuristics bằng cách kết hợp giải thuật (meta)heuristic cùng với phương pháp chính xác sử dụng MILP solver nhằm xác định lời giải khả thi có chất lượng tốt nhất có thể.