TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

Đề xuất thuật toán tìm kiếm cục bộ dựa trên ràng buộc cho bài toán phân chia hình chữ nhật mềm (Soft Rectangle Packing)

ĐÀO TUẤN DŨNG
dung.dt173050@sis.hust.edu.vn

Ngành Công nghệ thông tin

Giảng viên hướng dẫn: TS. Bùi Quốc Trung _____

Bộ môn: Khoa học máy tính
Viện: Công nghệ thông tin – Truyền thông

HÀ NỘI, 06/2021

Lời cam kết

Họ và tên sinh viên: Đào Tuấn Dũng

Điện thoại liên lạc: 0969821876 Email: tuandung0901@gmail.com

Lớp: Công nghệ thông tin 09-K62 Hệ đào tạo: chính quy

Tôi – Đào Tuấn Dũng – cam kết Đồ án Tốt nghiệp (ĐATN) là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của TS. Bùi Quốc Trung. Các kết quả nêu trong ĐATN là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong ĐATN – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

Hà Nội, ngày 18 tháng 6 năm 2021

Tác giả ĐATN

Dũng

Đào Tuấn Dũng

Lời cảm ơn

Lời đầu tiên, em xin gửi lời cảm ơn đến các thầy, cô giáo của trường Đại học Bách khoa Hà Nội và Viện Công nghệ thông tin và Truyền thông đã dạy dỗ, trang bị những kiến thức, kỹ năng cho em trong những năm học vừa qua.

Đặc biệt, em xin chân thành cảm ơn thầy TS. Bùi Quốc Trung, giảng viên bộ môn Khoa học máy tính, trường Đại học Bách khoa Hà Nội đã luôn tận tình hướng dẫn, chỉ bảo em trong quá trình làm đồ án.

Mặc dù em đã cố gắng hoàn thiện đồ án trong khả năng cho phép, nhưng không thể tránh khỏi mắc những sai sót. Em kính mong nhận được sự nhận xét, góp ý từ các thầy, cô giáo để Đồ án Tốt nghiệp của em được hoàn thiện hơn.

Em xin chân thành cảm ơn!

Tóm tắt

Việt Nam là quốc gia có nền nông nghiệp lâu đời, việc áp dụng khoa học kỹ thuật trong sản xuất nông nghiệp là vấn đề quan trọng trong giai đoạn hiện nay. Năm 2009, trong "chương trình mục tiêu quốc gia xây dựng nông thôn mới" đã tiến hành chia ruộng, dồn điền đổi thửa, để giải quyết phần nào việc ruộng manh mún, nhỏ lẻ tại Việt Nam. Tuy nhiên, tôi nhận thấy vấn đề chia ruộng, dồn điền đổi thửa chưa được được giải quyết một cách khoa học, là tiền đề cho việc phát triển nông nghiệp tại nước ta. Tôi tìm hiểu bài toán chia ruộng theo đúng tiêu chí chia ruộng của chính phủ và xem xét các tiêu chí đảm bảo tính công bằng, thuận lợi trong việc cơ giới hoá nông nghiệp và sản xuất.

Trong bài báo khoa học [1], đã đề cập đến bài toán phân chia hình chữ nhật và một số hàm mục tiêu giải quyết vấn đề chia ruộng tại Việt Nam. Nhóm tác giả bài báo đã đề xuất thuật toán quy hoạch nguyên tuyến tính (MIP) và tìm kiếm nhị phân (Binary Search) giải bài toán, kết quả thực nghiệm giải tốt bài toán với quy mô vừa và nhỏ. Trong đồ án, tôi đề xuất thuật toán tìm kiếm cục bộ dựa trên ràng buộc (Constraint-Based Local Search - CBLS) và Metaheuristics giải bài toán phân chia hình chữ nhật lớn thành các hình chữ nhật nhỏ với diện tích cho trước (Soft Rectangle Packing - SRP) với mục tiêu giải hiệu quả bài toán có kích thước lớn hơn. Thuật toán CBLS sử dụng ràng buộc điều khiển quá trình tìm kiếm cục bộ (Local Search), đây là mô hình hiệu quả để giải các bài toán tối ưu tổ hợp, giới hạn về mặt thời gian và tài nguyên tính toán. Tôi cài đặt thuật toán và làm thực nghiệm để so sánh với kết quả trong bài báo [1].

Tôi đã đề xuất thuật toán CBLS và kỹ thuật Metaheuristics giải hai bài toán Col-Peri-Max và Col-Aspect-Ratio và làm thực nghiệm so sánh với kết quả trong bài báo [1]. Thuật toán cho kết quả lời giải tối ưu khi giải bài toán quy mô vừa, nhỏ (từ 10 đến 25 hình chữ nhật) và kết quả giải tốt trong trường hợp quy mô lớn hơn (từ 30 đến 40 hình chữ nhật). Tôi đã chứng minh cận dưới của bài toán Col-Peri-Max và kết quả thực nghiệm đạt được tốt so với bài báo [1]. Trong một số trường hợp, bài toán Col-Aspect-Ratio cho kết quả thực nghiệm tốt hơn bài báo [1] nhưng vẫn còn trường hợp chưa đạt đến kết quả tối ưu trong bài báo nhưng về cơ bản thuật toán đề xuất cho kết quả tốt so với bài báo. Tôi đã công bố mã nguồn trên "GitHub" và chia sẻ quá trình làm đồ án, đây có thể là tài liệu tham khảo cho mọi người tiếp tục quá trình nghiên cứu của tôi, giải bài toán có kích thước lớn và đặc biệt là bài toán chia ruộng trong thực tế.

Bố cục của báo cáo đồ án được tổ chức như sau:

Chương 1: Giới thiệu đề tài. Chương 1 giới thiệu bài toán, mục tiêu và phạm vi đề tài, định hướng giải pháp và đóng góp của đồ án.

Chương 2: Cơ sở lý thuyết. Trong chương 2 giới thiệu cơ sở lý thuyết: bài toán tối ưu hoá tổ hợp, giải thuật tìm kiếm cục bộ (Local Search), tìm kiếm cục bộ dựa trên ràng buộc (CBLS) và các Metaheuristics đã sử dụng.

Chương 3: Đề xuất thuật toán CBLS cho bài toán phân chia hình chữ nhật mềm (SRP). Chương này trình bày mô hình toán học và đề xuất thuật toán CBLS giải bài toán SRP.

Chương 4: Cài đặt và kết quả thực nghiệm. Chương 4 trình bày kết quả cài đặt và đánh giá kết quả thực nghiệm.

Chương 5: Kết luận và hướng phát triển. Chương 5 trình bày kết quả công việc đã làm trong quá trình làm đồ án và hướng phát triển trong tương lai.

Mục lục

Danh mục hình vẽ

Danh mục bảng

Danh mục các từ viết tắt

SRP	Soft Rectangle Packing Bài toán phân chia hình chữ nhật mềm
LS	Local Search Tìm kiếm cục bộ
CBLS	Constraint-Based Local Search Tìm kiếm cục bộ dựa trên ràng buộc
ĐATN	Đồ án tốt nghiệp
TS	Tabu Search
ILS	Iterated Local Search
MIP	Mixed Integer Programming Quy hoạch nguyên tuyến tính hỗn hợp

Chương 1 Giới thiệu đề tài

1.1 Đặt vấn đề

Việt Nam là quốc gia có nền nông nghiệp, văn minh lúa nước lâu đời. Đất nước ngày càng phát triển, việc áp dụng khoa học kỹ thuật, công nghệ thông tin giải quyết các bài toán thực tế ngày càng được quan tâm. Trước năm 2009, việc chia ruộng tại nước ta dựa trên việc chia thành nhiều loại đất và mỗi hộ dân được một mảnh ruộng cho mỗi loại đất, dẫn đến tình trạng một cánh đồng nhỏ cũng có thể được phân chia cho nhiều hộ dân và mỗi hộ có nhiều mảnh ruộng. Ví dụ nổi bật, tại tỉnh Vĩnh Phúc: một hộ gia đình có tới 47 mảnh ruộng và có mảnh ruộng diện tích trung bình chỉ mười mét vuông [7]. Mỗi hộ dân có nhiều mảnh ruộng dẫn đến khó khăn trong việc quản lý, canh tác và đặc biệt ruộng nhỏ thì máy móc không vào được và khó khăn trong việc cơ giới hoá nông nghiệp. Trong "chương trình mục tiêu quốc gia xây dựng nông thôn mới" năm 2009, nước ta đã tiến hành chia ruộng, dồn điền đổi thửa để giải quyết phần nào việc ruộng manh mún, mỗi hộ dân có nhiều mảnh ruộng ở các khu khác nhau. Dồn điền đổi thửa giúp thuận lợi trong sản xuất, dễ cơ giới hoá nông nghiệp, sử dụng máy cấy, máy gặt và mang lại hiệu quả kinh tế cao hơn [8]. Tại một số địa phương, việc dồn điền đổi thửa trong chương trình xây dựng nông thôn mới vẫn dựa trên bốc số tại các khu ruộng. Bài toán chia ruộng là một vấn đề không mới nhưng tôi nhận thấy nó chưa được giải quyết một cách khoa học, là tiền đề phát triển nông nghiệp tại nước ta. Việc chia ruộng, dồn điền đổi thửa có thể dựa trên một số tiêu chí để tạo ra sự công bằng cho các hộ dân, tối ưu về diện tích làm bờ, thời gian di chuyển và đặc biệt là vấn đề cơ giới hoá nông nghiệp và áp dụng khoa học kỹ thuật vào sản xuất nông nghiệp trong giai đoạn hiện nay.

Trong bài báo khoa học [1], nhóm tác giả Bùi Quốc Trung đã đề xuất ba biến thể của bài toán phân chia hình chữ nhật (Three Soft Rectangle Packing) với ba hàm mục tiêu. Nhóm tác giả đã chứng minh hai bài toán Col-Peri-Max và Col-Aspect-Ratio thuộc lớp bài toán NP-khó và đề xuất thuật toán quy hoạch nguyên tuyến tính (MIP) và tìm kiếm nhị phân (Binary Search) để giải quyết bài toán này. Đây là bài toán có thể được áp dụng để giải quyết việc chia ruộng trên một cánh đồng lớn hay tại các khu ruộng chuyển đổi chia thành các hình chữ nhật nhỏ trồng các loại cây, rau củ hay các loại hoa với các diện tích xác định trước. Trong đồ án, tôi đề xuất thuật toán mới cho bài toán SRP, cài đặt thuật toán và so sánh với kết quả trong bài báo để đánh giá kết quả thực nghiệm đạt được. Do đó, tôi đã chọn và thực hiện đề tài: "Thuật toán tìm kiếm cục bộ dựa trên ràng buộc (CBLS) cho bài toán phân chia

hình chữ nhật lớn thành các hình chữ nhật nhỏ với diện tích cho trước (Soft Rectangle Packing)".

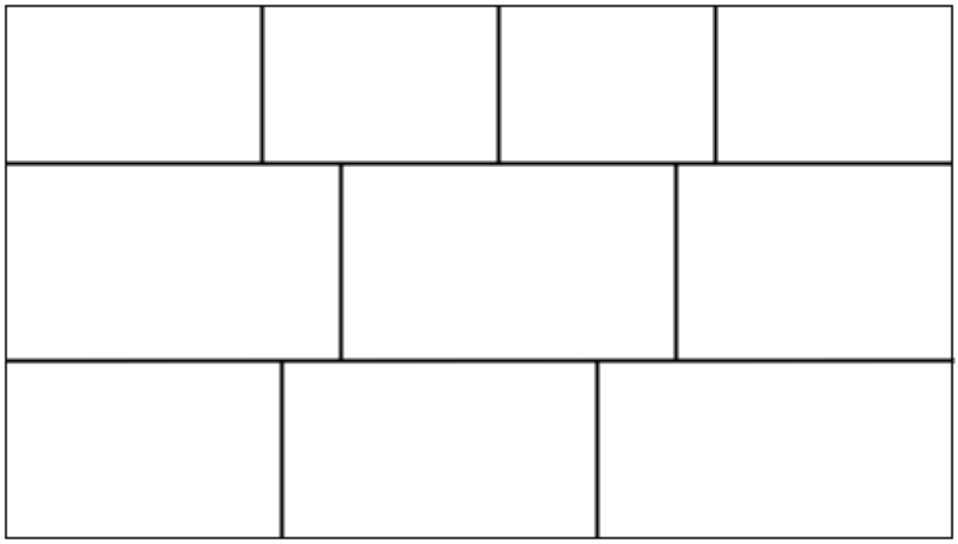
1.2 Bài toán phân chia hình chữ nhật mềm (SRP)

1.2.1 Định nghĩa bài toán

Bài toán phân chia hình chữ nhật mềm (SRP) được định nghĩa như sau: chia hình chữ nhật có kích thước chiều dài là L_1 và chiều rộng L_2 thành các hình chữ nhật với diện tích cho trước (a_1, …, a_n) sử dụng nhát cắt (tham lam) "guillotine". Nhát cát "guillotine" là một cắt thẳng từ cạnh của hình chữ nhật sang cạnh đối diện (edge-to-edge cut). Nhát cắt này đảm bảo theo đúng tiêu chí chia ruộng của chính phủ, phù hợp với bài toán thực tế chia ruộng tại một cánh đồng lớn cho các hộ dân với diện tích ruộng của các hộ dân đã xác định trước. Nhát cắt "guillotine" được thực hiện gồm hai giai đoạn, ví dụ cụ thể trong Hình 1.

Diện tích của các hình chữ nhật là cố định, do đó vị trí, chiều dài, chiều rộng của chúng tạo thành các biến quyết định của bài toán. Hai mục tiêu của bài toán là tối thiểu (i) chu vi hình chữ nhật có chu vi lớn nhất, (ii) tỷ lệ giữa chiều dài và chiều rộng của hình chữ nhật có tỷ lệ chiều dài và chiều rộng lớn nhất, tương ứng với hai bài toán Col-Peri-Max và Col-Aspect-Ratio. Hai hàm mục tiêu trên dựa trên cơ sở: giúp cho việc thuận lợi trong sản xuất, cơ giới hoá nông nghiệp và có sự công bằng trong việc chia ruộng cho các hộ dân, cũng như giảm thiểu diện tích làm bờ.

Bài toán SRP dựa trên cơ sở và có điểm tương đồng với bài toán cắt nguyên vật liệu (2D-strip Packing). Bài toán cho một tập các hình chữ nhật nhỏ có kích thước chiều dài và chiều rộng cố định và một hình chữ nhật lớn có chiều rộng cho trước và chiều cao vô hạn. Bài toán cắt nguyên vật liệu tìm cách xếp các hình chữ nhật nhỏ vào hình chữ nhật lớn sao cho các hình chữ nhật nhỏ không chồng lấn lên nhau và các cạnh song song với hình chữ nhật lớn, sao cho chiều cao của hình chữ nhật chiếm chỗ là nhỏ nhất. Bài toán SRP sử dụng nhát cắt "guillotine" để phân chia một hình chữ nhật lớn thành các hình chữ nhật nhỏ với diện tích cho trước, kích thước của các hình chữ nhật nhỏ có thể co dãn được nên đây là điểm khác với bài toán cắt nguyên vật liệu ở trên. Bố cục của các hình chữ nhật trong bài toán SRP phải tuân theo các quy tắc, sử dụng nhát cắt "guillotine". Như được minh họa trong Hình 1, nhát cắt "guillotine" gồm hai giai đoạn: đầu tiên cắt diện tích hình chữ nhật theo chiều ngang để tạo ra một số lớp (ba lớp như trên hình) và sau đó cắt từng lớp theo chiều dọc để có được các hình chữ nhật nhỏ (kết quả như hình vẽ).



Hình 1 Ví dụ nhát cắt "guillotine"

1.2.2 Các nghiên cứu trước đây

Phân chia hình chữ nhật thành các hình đa giác có diện tích cho trước là một trong những bài toán được quan tâm nhiều trong lĩnh vực vận trù học và tính toán hình học (geometry computation). Năm 2002, nhóm tác giả Beaumont [9] xác định hai vấn đề tối ưu hóa tìm cách phân chia hình vuông có kích thước đơn vị thành một số hình chữ nhật với các diện tích nhất định, để tối ưu hóa các thuật toán nhân ma trận song song trong các nền tảng tính toán song song không đồng nhất. Bài toán đầu tiên nhằm mục đích tối thiểu tổng tất cả các chu vi hình chữ nhật, trong khi bài toán thứ hai nhằm mục đích tối thiểu hình chữ nhật có chu vi lớn nhất. Bài toán thứ hai là trường hợp đặc biệt Col-Peri-Max trong đó vùng hình chữ nhật chung là hình vuông. Sau đó năm 2007, Nagamochi và Abe [10] đã xem xét tổng quát bài toán Col-Peri-Max và Col-Aspect-Ratio nhưng không sử dụng lát cắt "guillotine".

Bài toán đã từng được đề cập đến trong một số bài báo được nêu trên. Tuy nhiên, hàm mục tiêu trong các bài báo này không giống với hai toán Col-Peri-Max và Col-Aspect-Ratio và chưa chứng minh độ phức tạp lời giải bài toán. Trong bài báo [1] năm 2019, nhóm tác giả Bùi Quốc Trung đã chứng minh hai bài toán Col-Peri-Max và Col-Aspect-Ratio thuộc lớp bài toán NP-khó và đề xuất thuật toán MIP và tìm kiếm nhị phân (Binary Search) để giải quyết bài toán. Kết quả thực nghiệm được thực hiện trên ba bộ dữ liệu với số lượng từ 10 đến 40 hình chữ nhật. Trong bài báo đã đánh giá độ tối ưu trong từng bài toán, thuật toán giải tốt trong trường hợp bài toán quy mô vừa, nhỏ và câu hỏi mở về cách tiếp cận giải quyết bài toán quy mô lớn hơn.

1.3 Mục tiêu và phạm vi đề tài

Bài báo [1] sử dụng hai phương pháp MIP và tìm kiếm nhị phân để giải hai bài toán Col-Peri-Max và Col-Aspect-Ratio, kết quả thực nghiệm giải tốt bài toán kích thước vừa và nhỏ (từ 10 đến 25 hình chữ nhật). Trong trường hợp bài toán có kích thước lớn hơn thì thuật toán không trả ra lời giải tối ưu trong giới hạn thời gian chạy là 60 phút. Do đó, tôi đề xuất thuật toán CBLS, kết hợp với các kỹ thuật Metaheuristics cho hai bài toán Col-Peri-Max và Col-Aspect-Ratio với mục tiêu giải tốt hai bài toán với quy mô lớn hơn và đảm bảo về giới hạn thời gian thực hiện.

Báo cáo đồ án trình bày về bài toán tối ưu hoá tổ hợp, giải thuật tìm kiếm cục bộ (Local Search), tìm kiếm cục bộ dựa trên ràng buộc (CBLS), các Metaheuristics sử dụng trong đồ án (Iterated Local Search và Tabu Search) và đề xuất thuật toán CBLS cho hai bài toán Col-Peri-Max và Col-Aspect-Ratio. Tôi cài đặt và tiến hành làm thực nghiệm, so sánh kết quả cài đặt với kết quả thực nghiệm trong bài báo [1].

1.4 Định hướng giải pháp

Tôi đề xuất thuật toán tìm kiếm cục bộ dựa trên ràng buộc (CBLS) để giải hai bài toán Col-Peri-Max và Col-Aspect-Ratio. CBLS là một mô hình quan trọng trong bài toán tối ưu hoá tổ hợp. CBLS là mô hình hiệu quả, dựa trên tìm kiếm cục bộ (LS) được dẫn dắt bởi bởi mục tiêu giảm số lượng vi phạm ràng buộc. Tôi sử dụng giải thuật tìm kiếm dựa trên các vùng hàng xóm (Neighborhood Search) để lựa chọn lời giải trong quá trình tìm kiếm. Ý tưởng chính của giải thuật Neighborhood Search là tìm lời giải tối ưu dựa trên việc tìm kiếm các vùng hàng xóm lân cận của lời giải hiện tại và tiến hành chuyển sang lời giải hàng xóm tốt được chọn, lặp đi lặp lại quá trình này để tìm được lời giải tối ưu. Các giải thuật LS có ưu điểm về thời gian thực hiện và tài nguyên tính toán nên được nghiên cứu giải các bài toán quy mô lớn. Tuy nhiên, nhược điểm của tìm kiếm cục bộ là lời giải tìm được có thể là không tối ưu nhất mà bị kẹt ở một tối ưu cục bộ, do đó tôi sử dụng các Metaheuristics 2.3 để có thể thoát ra tối ưu cục bộ và hướng đến tìm kiếm tối ưu toàn cục.

1.5 Đóng góp đồ án

Trong đồ án, tôi đề xuất thuật toán CBLS và Metaheuristic giải hai bài toán Col-Peri-Max và Col-Aspect-Ratio. Đây là hai bài toán được mô hình hoá dựa trên bài toán dồn điền đổi thửa theo đúng tiêu chí của chính phủ và phù hợp với bài toán thực tiễn chia ruộng trên một cánh đồng lớn tại Việt Nam.

Tôi đã cài đặt thuật toán và làm thực nghiệm để so sánh với kết quả trong bài báo [1] và đánh giá hiệu quả của thuật toán. Kết quả thực nghiệm giải hai bài toán Col-Peri-Max và

Col-Aspect-Ratio, thuật toán đề xuất giải tốt bài toán có quy mô lớn hơn so với bài báo [1]. Thuật toán cho kết quả tối ưu khi giải bài toán có kích thước vừa, nhỏ (từ 10 đến 25 hình chữ nhật) và kết quả giải tốt hơn bài báo [1] trong trường hợp kích thước bài toán lớn hơn (từ 30 đến 40 hình chữ nhật). Tôi đã chứng minh cận dưới của bài toán Col-Peri-Max và bài toán cho kết quả thực nghiệm tốt, bài toán Col-Aspect-Ratio trong một số trường hợp có kết quả thực nghiệm tốt hơn bài báo [1] nhưng vẫn còn trường hợp chưa đạt đến kết quả tối ưu trong bài báo. Kết quả cho thấy, cách tiếp cận tìm kiếm cục bộ dựa trên ràng buộc (CBLS) giải tốt bài toán với số lượng hình chữ nhật lớn hơn và đảm bảo giới hạn thời gian thực hiện.

Chương 2 Cơ sở lý thuyết

2.1 Bài toán tối ưu hoá tổ hợp

2.1.1 Một số khái niệm

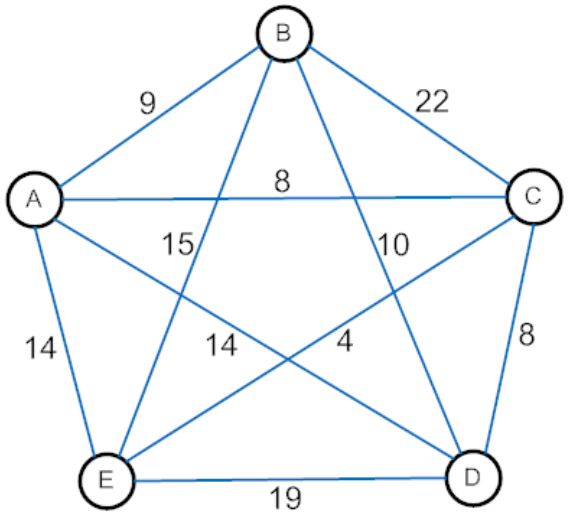
Bài toán tối ưu hoá tổ hợp là bài toán tìm ra một tập lời giải nhỏ tốt nhất dựa theo một hoặc nhiều tiêu chí nào đó trong một tập hữu hạn lời giải (thường rất lớn). Những lời giải này thường thỏa mãn một tập ràng buộc nào đó. Bài toán tối ưu hoá tổ hợp thường được mô hình hoá gồm ba thành phần chính: biến quyết định, các ràng buộc và hàm mục tiêu. Một số khái niệm được sử dụng trong báo cáo và trong bài toán tối ưu hoá tổ hợp sẽ được định nghĩa cụ thể hơn trong phần phụ lục A.

Bài toán tối ưu hoá tổ hợp là một bài toán quan trọng trong lý thuyết Khoa học máy tính, một số bài toán kinh điển như: bài toán người du lịch, bài toán cái túi, bài toán quân hậu, bài toán tô màu đồ thị hay bài toán lập lịch [2]. Trong thực tế, bài toán tối ưu hoá tổ hợp được ứng dụng trong nhiều lĩnh vực như lập kế hoạch, giao thông vận tải, tài chính, các hoạt động quản lý và điều hành trong các tổ chức, doanh nghiệp.

Một ví dụ về tối ưu hóa tổ hợp đó là bài toán người du lịch (Traveling Salesman Problem - TSP), đây là bài toán thuộc lớp bài toán NP-khó, có nguồn gốc lâu đời từ năm 1930, bài toán có ứng dụng rộng rãi trong nhiều lĩnh vực như lập kế hoạch, hậu cần (logistic), thiết kế vi mạch, ... Bài toán được phát biểu như sau: cho trước một danh sách các thành phố và giữa các thành phố có một chi phí di chuyển đã được xác định, cần tìm chu trình ngắn nhất xuất phát từ một thành phố đi qua mỗi thành phố đúng một lần, sau đó trở về vị trí xuất phát.

Bài toán TSP có thể được mô hình hoá dưới dạng đồ thị đầy đủ có trọng số $G = (V, E)$. Trong đó V là tập đỉnh, mỗi đỉnh $v \in V$ đại diện cho một thành phố mà người du lịch phải đi qua. E là tập cạnh nối các thành phố, với mỗi một cung (i, j) thì độ dài cung $d(i, j)$ chính là độ dài đường đi giữa thành phố i và thành phố j. Nhiệm vụ của bài toán là tìm một chu trình đi lần lượt qua các đỉnh của đồ thị, cuối cùng quay về điểm xuất phát, sao cho tổng độ dài của chu trình là nhỏ nhất.

Hình 2 là ví dụ minh hoạ bài toán TSP với 5 thành phố.



Hình 2 Minh họa bài toán TSP với 5 thành phố

Bài toán TSP với số lượng n thành phố thì có tất cả 1/2 * (n − 1)! đường đi, khi giá trị n > 15 thì số lượng lời giải sẽ rất lớn. Nên các hướng tiếp cận giải bài toán là rất quan trọng, tôi sẽ giới thiệu một số phương pháp giải bài toán tối ưu hoá tổ hợp ở mục 2.1.2.

2.1.2 Các phương pháp giải quyết

Các phương pháp giải bài toán tối ưu tổ hợp có thể chia thành 2 dạng chính: các phương pháp giải đúng và các phương pháp dựa trên kinh nghiệm.

(i) Các phương pháp giải đúng (exact approach) có thể kể đến các thuật toán toán điển hình như thuật toán nhánh cận, quy hoạch động, quy hoạch ràng buộc, quy hoạch tuyến tính. Hướng tiếp cận giải đúng bao gồm các phương pháp đảm bảo khi lời giải kết quả đưa ra là lời giải tối ưu của bài toán, bằng cách tìm kiếm trên tất cả không gian lời giải. Do đó, đối với các bộ dữ liệu lớn như hiện nay thì thời gian tính toán và các chi phí liên quan là rất lớn.

(ii) Các phương pháp dựa trên kinh nghiệm (heuristic) là các phương pháp tìm ra lời giải gần với lời giải tối ưu. Trong đó, giải thuật tìm kiếm cục bộ (LS) đang được quan tâm nghiên cứu để giải các bài toán quy mô lớn và thời gian hữu hạn cho phép. LS lặp đi lặp lại quá trình tìm kiếm như sau: tìm kiếm trên các vùng lời giải hàng xóm của một lời giải hiện tại và chuyển sang lời giải hàng xóm để tìm lời giải tốt hơn.

Để tránh rơi vào bẫy tối ưu cục bộ trong LS, có thể sử dụng một số Metaheuristics để khắc phục điều này. Một số Metaheuristics thường được sử dụng trong bài toán tối ưu hoá tổ hợp như: Tabu Search, giải thuật di truyền (Genetic Algorithms), Variable Neighborhood Search, Simulated Annealing hay Tối ưu hóa đàn kiến (Ant Colony Optimization).

2.2 Tìm kiếm cục bộ

Giải thuật tìm kiếm cục bộ (Local Search - LS) xuất phát từ một lời giải, tìm trong tập các lời giải hàng xóm, chọn lời giải hàng xóm tốt theo hàm mục tiêu và thực hiện bước chuyển (move) từ lời giải hiện tại sang lời giải hàng xóm đó. LS sẽ thực hiện lặp đi lặp lại việc tìm kiếm lời giải trong miền không gian tìm kiếm nhằm mục đích tìm ra lời giải tối ưu. Tại mỗi bước lặp, thuật toán sẽ tìm kiếm và chỉ lựa chọn một lời giải duy nhất để làm cơ sở cho bước lặp tiếp theo.

2.2.1 Thuật toán tìm kiếm cục bộ

Trong phần giới thiệu thuật toán tìm kiếm cục bộ, tôi sẽ giới thiệu các bước để thiết kế một thuật toán tìm kiếm cục bộ (LS) hiệu quả cho một bài toán tối ưu hoá tổ hợp nói chung.

Mô hình toán học bài toán

Bước đầu tiên của việc thiết kế một thuật toán tìm kiếm cục bộ (LS) cho bài toán tối ưu hoá tổ hợp là đưa ra một mô hình toán học cho bài toán đó. Đối với một bài toán, có thể tồn tại nhiều mô hình khác nhau. Một mô hình toán học tốt sẽ giúp thuật toán LS tương ứng tìm ra các lời giải chất lượng tốt trong thời gian tính toán nhỏ. Thông thường, đề xuất một mô hình cho bài toán tối ưu hoá tổ hợp gồm hai bước sau. Trong bước đầu tiên, chúng ta phải xác định các biến quyết định của bài toán. Bước thứ hai, chúng ta xây dựng tất cả các ràng buộc và hàm mục tiêu của bài toán đối với các biến quyết định.

Khởi tạo lời giải ban đầu

Từ mô hình toán học đã đề xuất ở phần trước, chúng ta khởi tạo một lời giải cho bài toán. Lời giải là một phép gán giá trị cho các biến quyết định. Một lời giải khả thi là một lời giải không vi phạm bất kỳ ràng buộc nào của bài toán. Thuật toán LS bắt đầu từ một lời giải ban đầu, vì vậy chúng ta cần khởi tạo một lời giải ban đầu. Các lời giải ban đầu có thể được tạo ngẫu nhiên hoặc bằng cách áp dụng một số giải thuật heuristics để thu được các lời giải ban đầu chất lượng tốt. Lời giải ban đầu nên được khởi tạo ở các vùng khác nhau của không gian tìm kiếm.

Xây dựng vùng hàng xóm

Trong thuật toán LS, bước chính là bước chuyển (move) từ một lời giải hiện tại sang một lời giải hàng xóm. Vì vậy, định nghĩa hàng xóm của bài toán là một giải pháp rất quan trọng trong việc thiết kế một thuật toán LS. Một hàng xóm có thể xây dựng bằng cách thay đổi một số thành phần (hoặc bộ phận hoặc tính năng) của lời giải. Hơn nữa, quy mô của vùng hàng xóm có ảnh hưởng hiệu quả đến hiệu suất của thuật toán LS. Nếu một vùng hàng xóm quy mô nhỏ có thể dẫn đến thuật toán LS đi vào vùng có lời giải chất lượng thấp. Nếu một

vùng hàng xóm có kích thước lớn sẽ cần thời gian tính toán tốn kém hơn để chọn một lời giải hàng xóm trong quá trình di chuyển ở bước tiếp theo.

Lựa chọn hàng xóm

Nhiệm vụ của bước này là chọn một lời giải hàng xóm từ một hoặc các vùng hàng xóm khác nhau. Bước này có thể được thực hiện ngẫu nhiên hoặc dựa trên việc đánh giá chất lượng của lời giải hàng xóm. Một thuật toán LS thiết kế tốt nên có sự cân bằng giữa quy mô của các vùng hàng xóm và độ phức tạp của thuật toán trong việc lựa chọn hàng xóm. Metaheuristics thường được triển khai trong phần lựa chọn hàng xóm để hỗ trợ trong quá trình tìm kiếm. Một số Metaheuristics sẽ được giới thiệu ở mục 2.3.

Bước chuyển sang hàng xóm (move)

Bước này sẽ chuyển từ lời giải hiện tại sang lời giải hàng xóm của nó đã được chọn ở bước trên. Bước này được thực hiện bằng các thay đổi cục bộ của các thành phần lời giải và giá trị của hàm mục tiêu.

Kết thúc thuật toán LS

Chúng ta cần quyết định thời điểm kết thúc thuật toán LS, ví dụ: một số lần lặp nhất định đã bị vượt quá. Để có được các lời giải chất lượng tốt với thời gian thích hợp, các tiêu chí kết thúc quá trình kiếm kiếm nên được quyết định cẩn thận. Ví dụ: thuật toán LS không thể có được giải pháp tốt nếu nó kết thúc trước thuật toán hội tụ.

Tìm kiếm cục bộ (LS)

Trong giải thuật LS, hai vấn đề quan trọng nhất cần quan tâm là tính tăng cường (intensification) và tính đa dạng (diversification) của quá trình tìm kiếm. Tính tăng cường là khả năng tập trung tìm kiếm sâu ở những vùng không gian mà ta dự đoán là sẽ chứa lời giải tối ưu, tính đa dạng là khả năng tìm đến những vùng không gian lời giải mới nhằm thoát ra khỏi các vùng chứa điểm tối ưu cục bộ. Các LS khác nhau sẽ đưa ra các chiến lược khác nhau để đảm bảo sự tồn tại và cân bằng giữa hai yếu tố này.

2.2.2 Tìm kiếm cục bộ dựa trên ràng buộc (CBLS)

Thuật toán CBLS sử dụng các ràng buộc để mô tả và điều khiển quá trình tìm kiếm cục bộ (LS). Trong bài toán tối ưu hoá tổ hợp, ngoài các hàm mục tiêu, có một tập các ràng buộc. Vì vậy một lời giải cho bài toán tối ưu tổ hợp phải thỏa mãn tập hợp các ràng buộc và có giá trị hàm mục tiêu tối ưu.

Tìm kiếm cục bộ dựa trên ràng buộc (CBLS) nhằm mục đích triển khai tầm nhìn được thực hiện dựa trên: “LocalSearch = Model + Search”

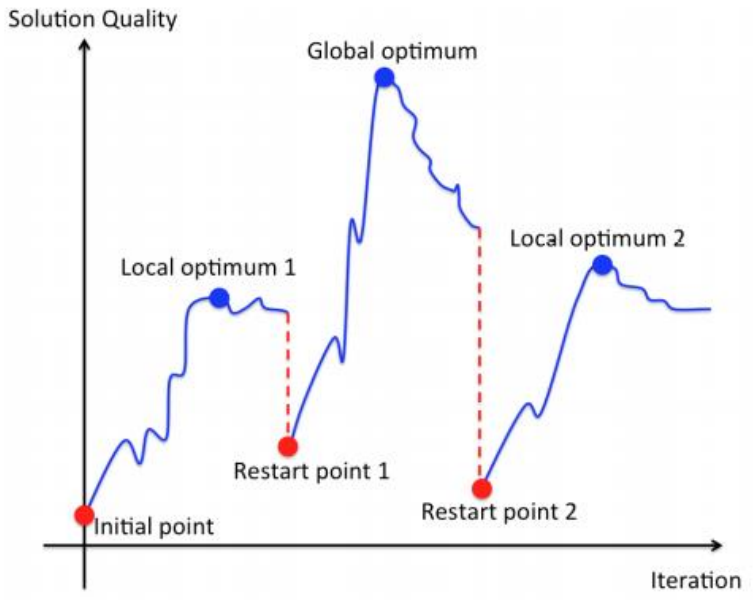
Điều đó thể hiện rằng một thuật toán tìm kiếm cục bộ được xem tốt nhất là thành phần của mô hình khai báo với thành phần tìm kiếm. Sự tách biệt các mối quan tâm này là trọng tâm: nó giả định tầm quan trọng của việc thể hiện các cấu trúc của vấn đề đang được giải quyết một cách khai báo, có cấu trúc và cung cấp một thành phần tìm kiếm khai thác các cấu trúc đó và hướng đến tìm kiếm tối ưu cục bộ chất lượng cao.

2.3 Metaheuristics

Metaheuristics có vai trò cơ bản là chúng hướng đến việc thoát ra tối ưu cục bộ và hướng đến tìm kiếm tối ưu toàn cục. Mục tiêu này có thể được tiếp cận theo nhiều cách khác nhau, điều này giải thích sự đa dạng và phong phú của các kết quả sử dụng Metaheuristics để giải quyết vấn đề này. Trong báo cáo, tôi trình bày các Metaheuristics đã sử dụng trong đồ án để minh hoạ giải thuật CBLS mà không xem xét toàn diện các Metaheuristics. Phần này, tôi sẽ giới thiệu hai Metaheuristics là Iterated Local Search và Tabu Search.

2.3.1 Iterated Local Search

Iterated Local Search (ILS) là Metaheuristic phổ biến, lặp đi lặp lại một miền tìm kiếm cục bộ cụ thể từ các điểm xuất phát khác nhau, để lấy các vùng khác nhau của không gian tìm kiếm và để tránh trả về các điểm tối ưu cục bộ (local optimum) chất lượng không tốt. Các điểm xuất phát khác nhau làm tăng tính đa dạng trong quá trình tìm kiếm. Hình 3 mô tả một ví dụ về ILS. Với các điểm xuất phát khác nhau, LS có thể tìm kiếm đến các vùng khác nhau của không gian tìm kiếm. Do đó, cơ hội để tìm kiếm được điểm tối ưu toàn cục (global optimum) sẽ lớn hơn.



Hình 3 Iterated Local Search

2.3.2 Tabu Search

Tabu Search (TS) là một Metaheuristic phổ biến và hiệu quả, giúp tìm kiếm đa dạng trong mỗi bước tìm kiếm của nó. TS cố gắng để ngăn việc tìm kiếm truy cập vào các điểm giống nhau trong không gian tìm kiếm. Có hai đặc điểm của TS: thứ nhất, định nghĩa các di chuyển hợp lệ (legal move), không áp đặt phải di chuyển sang lời giải tốt hơn giá trị mục tiêu và cho phép LS lựa chọn di chuyển làm giảm chất lượng của lời giải hiện tại, do đó thoát cực tiểu cục bộ; thứ hai, bản chất tham lam của TS đảm bảo rằng giá trị hàm mục tiêu không giảm quá nhiều ở bất kỳ bước nào.

TS không thể theo dõi tất cả các lời giải đã truy cập (do thiếu bộ nhớ và tính toán thời gian tốn kém để kiểm tra việc xem lại), vì vậy TS sử dụng bộ nhớ ngắn hạn để ngăn việc tìm kiếm trả về lời giải đã truy cập gần đây. Thành phần chính quan trọng nhất của giải thuật TS là "Tabu list", đây là một danh sách chứa một số bước chuyển (move) đã được sử dụng trong quá khứ, một bước chuyển sang lời giải hàng xóm sẽ không được phép áp dụng lên lời giải hiện tại chừng nào bước chuyển này còn nằm trong "Tabu list". Nhiệm vụ của "Tabu list" là để tránh quay trở lại những lời giải đã tìm trước đó, nhằm tăng tính đa dạng của quá trình tìm kiếm. Tuy nhiên, nếu một bước chuyển đang nằm trong "Tabu list" nhưng nó lại có thể giúp cải thiện chất lượng của lời giải tốt nhất hiện tại thì bước chuyển này vẫn được chấp nhận sử dụng (aspiration criteria).

Chương 3 Đề xuất thuật toán CBLS cho bài toán phân chia hình chữ nhật mềm (SRP)

3.1 Mô hình hoá bài toán

Mô hình hoá bài toán tôi tham khảo dựa trên mô hình được đề xuất trong bài báo [1].

3.1.1 Biến quyết định

Biến quyết định của bài toán là gán mỗi hộ nông dân vào một lớp (tương ứng là gán hình chữ nhật có diện tích a_i vào mỗi lớp, điều kiện số lớp phải nhỏ hơn hoặc bằng tổng số hình chữ nhật).

Biến quyết định được lưu trong mảng x[1…n]: mảng n biến quyết định, gán giá trị x[i] = p, tương ứng mỗi hình chữ nhật i được gán vào một lớp p. Miền xác định: Dx[i] = {1, . . . , n}, ∀i ∈ {1, . . . , n}.

3.1.2 Các ràng buộc của bài toán

Mô hình toán học này mô tả một giải pháp có n lớp trong đó một số lớp có thể trống. Tôi sử dụng một biến nhị phân x_{ik} và một biến liên tục w_{ik} cho mỗi hình chữ nhật i và lớp k. Biến x_{ik} nhận giá trị 1 khi và chỉ khi hình chữ nhật i thuộc lớp k và biến w_{ik} đại diện cho chiều dài của hình chữ nhật i khi được đặt trong lớp k, và bằng 0 nếu ngược lại. Cuối cùng, mỗi biến nhị phân y_k nhận giá trị 1 nếu lớp k không trống, và bằng 0 nếu ngược lại.

Mô hình toán học của bài toán Col-Peri-Max và Col-Aspect-Ratio được đưa ra thỏa mãn các ràng buộc (4) – (16):

minimize Φ_2 (1)

minimize Φ_3 (2)

$\sum_{k=1}^{n} x_{ik} = 1$ i, k ∈ {1, . . . , n} (3)

$\sum_{i=1}^{n} x_{ik} \geq y_k$ i, k ∈ {1, . . . , n} (4)

$w_{ik} \leq y_k$ i, k ∈ {1, . . . , n} (5)

$\sum_{i=1}^{n} w_{ik} = L_1 y_k$ i, k ∈ {1, . . . , n} (6)

$w_{ik} \leq L_1 x_{ik}$ $i, k \in \{1, \ldots, n\}$ (7)

$a_i x_{ik} \leq L_2 x_{ik}$ $i, k \in \{1, \ldots, n\}$ (8)

$a_j w_{ik} - a_i w_{jk} \leq a_j L_1 (2 - x_{ik} - x_{jk})$ $i, j, k \in \{1, \ldots, n\}, i \neq j$ (9)

$a_i w_{jk} - a_j w_{ik} \leq a_i L_1 (2 - x_{ik} - x_{jk})$ $i, j, k \in \{1, \ldots, n\}, i \neq j$ (10)

$x_{ji} \in \{0, 1\}$ $i, j \in \{1, \ldots, n\}$ (11)

$w_{ik} \geq 0$ $i, k \in \{1, \ldots, n\}$ (12)

$y_k \in \{0, 1\}$ $k \in \{1, \ldots, n\}$ (13)

$\Phi_2 \geq 0$ (14)

$\Phi_3 \geq 0$ (15)

Ràng buộc (3) - (5) đảm bảo rằng mọi hình chữ nhật đều được nằm trong một lớp và y_k sẽ nhận giá trị 1 khi có ít nhất một hình chữ nhật trong lớp k. Ràng buộc (6) nói rằng tổng chiều dài của các hình chữ nhật của mỗi lớp k bằng L_1 nếu lớp này được sử dụng ($y_k = 1$) và ngược lại ($y_k = 0$). Ràng buộc (7) và (8) chỉ ra ($w_{ik} = 0$) ⇔ ($x_{ik} = 0$). Cuối cùng, những ràng buộc (9) và (10) đảm bảo rằng nếu hai hình chữ nhật i và j nằm trong cùng một lớp k thì $a_i / w_{ik} = a_j / w_{jk}$.

Mô hình bài toán có thể được tăng cường với việc bổ sung một số ràng buộc để loại bỏ các lớp trống và các lời giải đối xứng:

$y_k \geq y_k + 1$ $k \in \{1, \ldots, n - 1\}$ (16)

$x_{ik} = 0$ $i \in \{1, \ldots, n\}, k \in \{i + 1, \ldots, n\}$ (17)

Ràng buộc (16), đảm bảo việc sử dụng các lớp theo thứ tự của các chỉ số của chúng. Ràng buộc (17) đảm bảo bất kỳ hình chữ nhật i nào thuộc một lớp có chỉ số $k \in \{1, \ldots, i\}$.

3.1.3 Xây dựng hàm mục tiêu

Bài toán thuộc lớp bài toán tối ưu hoá tổ hợp. Bất kỳ lời giải nào của những vấn đề này đều có thể được mô tả dưới dạng phân vùng $\{S_1, \ldots, S_m\}$ của tập hình chữ nhật $S = \{1, \ldots, n\}$ thành m lớp. Vì chiều dài của mỗi lớp được cố định là L_1 nên chiều rộng $w(S_k)$ của một lớp S_k (và tất cả các hình chữ nhật có trong lớp đó) nhận giá trị:

$$w(S_k) = \frac{\Sigma_{i \epsilon S_k} a_i}{L_1} \quad (18)$$

và chiều còn lại của mỗi hình chữ nhật i ∈ S_k là a_i / w(S_k). Do đó, mục tiêu của những lời giải này là tìm {S_1 ,. . . , S_m} để giảm thiểu:

Col-Peri-Max: $$\Phi_2 = 2 \times \max_k \max_{i \in S_k} \left(w(S_k) + \frac{a_i}{w(S_k)} \right) \quad (19)$$

Col-Aspect-Ratio: $$\Phi_3 = \max_k \max_{i \in S_k} \max \left(\frac{a_i}{w(S_k)^2}, \frac{w(S_k)^2}{a_i} \right) \quad (20)$$

3.2 Đề xuất thuật toán CBLS giải bài toán Soft Rectangle Packing

3.2.1 Định nghĩa hàng xóm

Trong thuật toán LS, dựa vào vùng lân cận hàng xóm để tìm lời giải tối ưu, bước chính là chuyển từ một lời giải hiện tại sang một lời giải hàng xóm. Vì vậy, định nghĩa hàng xóm của bài toán là một giải pháp rất quan trọng trong việc thiết kế thuật toán LS. Thuật toán CBLS cho hai bài toán Col-Peri-Max và Col-Aspect-Ratio, tôi đặt lần lượt là P_1, P_2. Lời giải S của bài toán P_1, P_2 được biểu diễn như sau: S = (sp_1, . . . , sp_i, …, sp_j, . . . , sp_n), trong đó sp_i là vị trí của hình chữ nhật i hoặc S = {S_1, . . . , S_i , . . . , S_j , . . . , S_n} trong đó S_i là tập các hình chữ nhật thuộc lớp i.

Trong bài toán P_1, P_2, tôi định nghĩa bốn loại hàng xóm cho giải thuật LS như sau:

1. Change-based neighborhood: Thay đổi một hình chữ nhật từ lớp này sang lớp khác. Có hai trường hợp xảy ra, số lớp không đổi và trường hợp tăng thêm một lớp khi hình chữ nhật đó thay đổi tạo thành lớp mới.

   $N_1(S) = \{(sp_1, \ldots, sp_i', \ldots, sp_n) | sp_i' \neq sp_i , sp_i' \in F \}$

2. Swap-based neighborhood: Đổi chỗ hai hình chữ nhật ở hai lớp khác nhau, số lượng lớp sẽ không thay đổi.

   $N_2(S) = \{(sp_1, \ldots, sp_{i+j} , \ldots , sp_i , \ldots , sp_n) | sp_i \neq sp_{i+j} , 1 \leq i < i + j \leq n\}$

3. Split-based neighborhood: Tách một lớp có số hình chữ nhật lớn thành 2 lớp nhỏ, gồm lớp hiện tại và lớp (n+1). Do đó, số lượng lớp sẽ tăng thêm một.

   $N_3(S) = \{S_1, \ldots, S_i' , \ldots , S_{n,}, S_{n+1}\} | S_i' = S_i \backslash \{sp_k\}, S_{n+1} = S_{n+1} \cup \{sp_k\}, 0 < k < i, sp_k \in S_i\}$

4. Merge-based neighborhood: Ghép hai lớp khác nhau thành một lớp, lớp mới có số lượng hình chữ nhật lớn hơn, số lượng lớp sẽ giảm đi một.

   $N_4(S) = \{S1, \ldots, S_i' , \ldots , S_j' , \ldots , S_n\} | S_i' = S_i \setminus S_i, S_j' = S_j \cup S_i, i \neq j\}$

3.2.2 Thuật toán CBLS cho bài toán Col-Peri-Max và Col-Aspect-Ratio

Chú thích mã giả Algorithm 1:

Bảng 1 Chú thích mã giả Algorithm 1

Kí hiệu	**Chú thích**
S	Lời giải khởi tạo ban đầu (được sinh ngẫu nhiên)
S	Lời giải hiện tại đang xét
S_1, S_2, S_3, S_4	Lời giải hàng xóm tốt nhất được chọn để so sánh với lời giải S hiện tại
bestSol	Lời giải tốt nhất hiện tại
obj	Giá trị hàm mục tiêu của lời giải hiện tại
obj*	Giá trị hàm mục tiêu lời giải hàng xóm được chọn
bestObj	Giá trị hàm mục tiêu của lời giải tối ưu hiện tại
N(S)	Tập hàng xóm của lời giải S
maxIt	Số vòng lặp tối đa của bài toán
it	Số vòng lặp đã thực hiện
stable	Số lượng vòng lặp (iterations) mà tính từ thời điểm gần nhất thuật toán có trả ra lời gian tốt hơn lời giải lời giải hiện tại
stableLimit	Số lượng vòng lặp (iterations) được giới hạn mà thuật toán không tìm ra lời giải tốt hơn lời giải hiện tại

Algorithm 1: Col-Peri-Max: A constraint-based local search algorithm for $\mathbf{P_1}$

```
1  S ← A random solution to P1; obj ← objective of S;
2  bestSol ← S; bestObj ← obj ;
3  while it < maxIt do
4      S1 ← the best neighbor of S in N1(S) that is obtained by changing field of a
       household such that this household is not in tabu;
5      if S1 is better than S then
6          S ← S1;
7      else
8          S2 ← the best neighbour of S in N2(S) that is obtained by swapping the
               position of a pair of households such that this pair is not in tabu;
9          if S2 is better than S then
10             S ← S2;
11         else
12             S3 ← the best neighbour of S in N3(S) that is obtained by splitting
               the position of the largest zone;
13             if S3 is better than S then
14                 S ← S3;
15             else
16                 S4 ← the best neighbour of S in N4(S) that is obtained by
                   merging the position of a pair of zones;
17                 if S4 is better than S then
18                     S ← S4;
19                 else
20                     Swap the position of two random households in N2(S);
21     obj* ← objective of S;
22     if obj* < obj then
23         stable = 0;
24     else
25         stable++;
26     obj = obj*;
27     if obj < bestObj then
28         bestObj ← obj; bestSol ← S;
29     if stable = stableLimit then
30         S ← a random solution; obj ← objective of S;
31     it = it + 1;
```

Thuật toán CBLS cho bài toán Col-Peri-Max và Col-Aspect-Ratio được thực hiện như sau:

1. Khởi tạo lời giải ban đầu (sinh ngẫu nhiên lời giải S) và tính giá trị hàm mục tiêu obj.
2. bestSol, bestObj lần lượt là lời giải tốt nhất và giá trị hàm mục tiêu tốt nhất đến thời điểm đang tìm kiếm, ban đầu được gán là lời giải S và giá trị hàm mục tiêu obj.
3. Bắt đầu vòng lặp quá trình tìm kiếm, số lượng vòng lặp đã thực hiện phải nhỏ hơn giá trị vòng lặp tối đa maxIt.
4. Tìm trong tập lời giải hàng xóm $N_1(S)$ được thực hiện bởi thay đổi vị trí một hộ dân (không thuộc hàng đợi tabu) từ lớp hiện tại sang một lớp khác, S_1 là lời giải hàng xóm tốt nhất thuộc $N_1(S)$.
5. Nếu lời giải S_1 tốt hơn lời giải S hiện tại (tương ứng giá trị hàm mục tiêu của lời giải S_1 nhỏ hơn giá trị hàm mục tiêu của lời giải S), thực hiện
6. Gán lời giải S_1 thành lời giải S hiện tại và chuyển đến dòng lệnh 21.
7. Nếu không,
8. Tìm trong tập lời giải hàng xóm $N_2(S)$ được thực hiện bởi đổi chỗ vị trí hai hộ dân ở hai lớp khác nhau (điều kiện hai hộ nông dân đó không thuộc hàng đợi tabu), S_2 là lời giải hàng xóm tốt nhất thuộc $N_2(S)$.
9. Nếu lời giải S_2 tốt hơn lời giải S hiện tại, thực hiện
10. Gán lời giải S_2 thành lời giải S hiện tại và chuyển đến dòng lệnh 21.
11. Nếu không,
12. Tìm trong tập lời giải hàng xóm $N_3(S)$ được thực hiện tách một lớp có số lượng hộ dân lớn nhất thành hai lớp (lớp hiện tại và một lớp được tạo mới), S_3 là lời giải hàng xóm tốt nhất thuộc $N_3(S)$.
13. Nếu lời giải S_3 tốt hơn lời giải S hiện tại, thực hiện
14. Gán lời giải S_3 thành lời giải S hiện tại và chuyển đến dòng lệnh 21.
15. Nếu không,
16. Tìm trong tập lời giải hàng xóm $N_4(S)$ được thực hiện bởi dồn hai lớp thành một lớp lớn hơn, S_4 là lời giải hàng xóm tốt nhất thuộc $N_4(S)$.
17. Nếu lời giải S_4 tốt hơn lời giải S hiện tại, thực hiện
18. Gán lời giải S_4 thành lời giải S hiện tại và chuyển đến dòng lệnh 21.
19. Nếu không,
20. Gán một lời giải thuộc tập $N_2(S)$ thành lời giải hiện tại.
21. Tính giá trị hàm mục tiêu obj* của lời giải hàng xóm của S được chọn.
22. Nếu obj* < obj thì
23. stable = 0.
24. Nếu không,
25. stable = 1.
26. Cập nhật obj = obj*
27. Nếu obj < bestObj

28. Cập nhật, lời giải bestSol là lời giải S và besbObj = obj
29. Nếu stable = stableLimit
30. Khởi tạo lời giải S mới (được sinh ngẫu nhiên) và giá trị obj.
31. Tăng số lượng vòng lặp it = it +1.
32. Kiểm tra điều kiện vòng lặp while (it < maxIt) và tiếp tục quá trình tìm kiếm từ dòng lệnh số 3.
33. Khi giá trị it = maxIt, kết thúc quá trình kiếm. Tìm được lời giải tốt nhất bestSol và giá trị hàm mục tiêu bestObj.

Tôi sẽ minh hoạ thuật toán CBLS trong một số bước tìm kiếm (ví dụ Hình 4):

```
Randomly generated solution: 1 2 3 1 4 4 2 5 4 2 Objecttive = 79.2
(1) 1 2 3 1 4 4 2 1 4 2 Objecttive = 71.314285 Tabu list[7] Stable = 0
(2) 1 2 2 1 4 4 2 1 4 2 Objecttive = 63.451477 Tabu list[7, 2] Stable = 0
(3) 1 2 2 1 4 4 2 1 4 1 Objecttive = 52.82823 Tabu list[7, 2, 9] Stable = 0
(4) 1 2 2 1 4 1 2 1 4 1 Objecttive = 52.00016 Tabu list[7, 2, 9, 5] Stable = 0
(5) 1 2 2 1 1 1 2 1 1 1 Objecttive = 51.291817 Tabu list[7, 2, 9, 5] Stable = 0
(6) 1 2 2 1 1 1 1 1 2 1 Objecttive = 51.272728 Tabu list[2, 9, 5, 6, 8] Stable = 0
(7) 1 1 1 1 1 1 1 1 1 1 Objecttive = 50.833336 Tabu list[2, 9, 5, 6, 8] Stable = 0
(8) 1 1 2 1 2 1 1 1 2 1 Objecttive = 51.62772 Tabu list[2, 9, 5, 6, 8] Stable = 1
(9) 1 2 2 1 1 1 1 1 2 1 Objecttive = 51.272728 Tabu list[5, 6, 8, 1, 4] Stable = 0
(10)1 2 1 1 1 1 1 1 2 1 Objecttive = 50.936836 Tabu list[6, 8, 1, 4, 2] Stable = 0
```

Hình 4 Minh họa quá trình tìm kiếm thuật toán CBLS

Bước 1: Khởi tạo lời giải ban đầu S = (1 2 3 1 4 4 2 5 4 2), giá trị hàm mục tiêu Objective = 79.2

Bước 2: Tìm trong các vùng lời giải hàng xóm, lời giải hàng xóm tốt nhất $S_1 \in N_1(S)$ và tiến hành bước chuyển (move) từ lời giải hiện tại sang lời giải S_1, cập nhật lời giải hiện tại S = (1 2 3 1 4 4 2 1 4 2).

Tabu list lưu trạng thái các bước chuyển (move) thực hiện trước đó (tôi đặt kích thước hàng đợi lưu trữ Tabu list là 5), cập nhật Tabu list [7] tương ứng chỉ số (index) 7 hay hình chữ nhật số 8 được chuyển từ lớp số 5 về lớp số 1.

Bước 3: Tương tự, cập nhật lời giải S = (1 2 2 1 4 4 2 1 4 2), lưu ý từ bước này trở đi sẽ tìm kiếm lời giải tốt nhất mà vị trí thay đổi không thuộc Tabu list, cập nhật Tabu list [7,2] tương ứng hình chữ nhật số 3 được chuyển từ lớp số 3 sang lớp số 2.

Lặp đi lặp lại quá trình tìm kiếm, đến bước 6, lời giải hàng xóm tốt nhất $S_2 \in N_2(S)$ và tiến hành bước chuyển (move) từ lời giải hiện tại sang S_2, cập nhật lời giải hiện tại S = (1 2 2 1 1 1 1 1 2 1), cập nhật Tabu list [2, 9, 5, 6, 8] tương ứng đổi chỗ vị trí hình chữ nhật số 7 từ lớp số 2 sang lớp số 1 và hình chữ nhật số 9 từ lớp số 1 sang lớp số 2.

Bước 7: Tìm trong các vùng lời giải hàng xóm, lời giải hàng xóm tốt nhất $S_4 \in N_4(S)$ và tiến hành bước chuyển (move) từ lời giải hiện tại sang S_4, cập nhật lời giải hiện tại S = (1 1 1 1 1 1 1 1 1 1). Bước này đã tiến hành hợp hai lớp số 1 và số 2.

Bước 8: Không tìm được lời giải hàng xóm tốt hơn, nên lời giải hiện tại được chuyển sang lời giải tốt nhật của $N_2(S)$, cập nhật lời giải hiện tại S = (1 1 2 1 2 1 1 1 2 1). Lời giải hàng xóm được chuyển (move) không tốt hơn lời giải hiện tại nên cập nhật stable = 1 (tăng giá trị biến stable thêm 1).

Bước 9: Tìm trong các vùng lời giải hàng xóm, lời giải hàng xóm tốt nhất $S_2 \in N_2(S)$ và tiến hành bước chuyển (move) từ lời giải hiện tại sang S_2, cập nhật lời giải hiện tại S = (1 2 2 1 1 1 1 1 2 1). Cập nhật Tabu list [5, 6, 8, 1, 4] tương ứng đổi chỗ vị trí hình chữ nhật số 2 từ lớp số 1 sang lớp số 2 và hình chữ nhật số 5 từ lớp số 2 sang lớp số 1.

Tiếp tục, quá trình tìm kiếm. Sau một khoảng thời gian (stableLimit) mà thuật toán không tìm ra lời giải tốt hơn lời giải hiện tại, tiến hành chuyển sang vùng khác, bằng cách sinh ra lời giải ngẫu nhiên mới và bắt đầu tìm kiếm trong vùng đó.

Thuật toán lặp đi, lặp lại quá trình tìm kiếm ở trên sẽ tìm được lời giải tối ưu (Thuật toán kết thúc khi số lượng vòng lặp (iterations) số lần tìm kiếm đạt giá trị maxIt).

Chương 4 Cài đặt và đánh giá kết quả thực nghiệm

4.1 Môi trường và cài đặt thực nghiệm

4.1.1 Bộ dữ liệu thực nghiệm

Bộ dữ liệu được lấy từ bài báo [1], https://w1.cirrelt.ca/~vidalt/en/research-data.html.

Mô tả bộ dữ liệu: gồm ba tập dữ liệu và mỗi tập dữ liệu có 21 trường hợp, số lượng hình chữ nhật n ∈ {10, 15, 20, 25, 30, 35, 40}.

- Tập U - Diện tích trong mỗi trường hợp được lấy mẫu theo phân bố đều: X ~ U (1, 200).
- Tập MU - Diện tích trong mỗi trường hợp được lấy mẫu trong một hỗn hợp của ba phân bố đều: $X \sim \frac{1}{3}$[U(1, 10) + U(11, 50) + U(51, 150)].
- Tập MN - Diện tích trong mỗi trường hợp được lấy mẫu trong hỗn hợp của ba phân phối chuẩn: $X \sim \frac{1}{3}$ [N (5, 2) + N (25, 10) + N (125, 50)], nhưng một mẫu khác được lấy bất cứ khi nào khu vực lớn hơn 200.

4.1.2 Môi trường cài đặt

Thuật toán CBLS cho bài toán Soft Rectangle Packing được cài đặt bằng ngôn ngữ lập trình Java, môi trường phát triển tích hợp (IDE) là Eclipse IDE 2020-06. Java là là ngôn ngữ lập trình hướng đối tượng, đơn giản, hiệu quả và an toàn, sử dụng thư viện Apache POI để xử lý ghi kết quả chạy thực nghiệm vào file Excel.

Tất cả các kết quả thực nghiệm trong đồ án được thực hiện trên máy tính Dell G3-3579 Intel Core i7-8750H CPU 2.20GHz, RAM 16GB, hệ điều hành Windows 10 Education.

4.2 Kết quả và đánh giá kết quả thực nghiệm

4.2.1 Kết quả thực nghiệm

Kết quả thực nghiệm được thực hiện như sau: tôi tiến hành chạy chương trình trong mỗi trường hợp 10 lần, sau đó ghi kết quả xử lý về thời gian chạy trung bình, giá trị hàm mục

tiêu trung bình, giá trị hàm mục tiêu lớn nhất và nhỏ nhất trong 10 lần chạy ra file Excel. Kết quả thực nghiệm của hai bài toán Col-Peri-Max và Col-Aspect-Ratio được trình bày trong bảng 2-7 (từ Bảng 2 đến Bảng 7).

Những thuật ngữ, định nghĩa sử dụng trong các bảng từ 2-7 như sau:

Kết quả thực nghiệm của đồ án, được ký hiệu như sau: Time: Thời gian trung bình chạy chương trình trên CPU, tính bằng giây (s); AV: Average of Objective Value – Giá trị hàm mục tiêu trung bình; MaxV: Maximum Objective Value – Giá trị hàm mục tiêu lớn nhất trong 10 lần chạy; MinV: Minimum Objective Value – Giá trị hàm mục tiêu nhỏ nhất trong 10 lần chạy. LB_1: Lower Bound – Giới hạn dưới của bài toán Col-Peri-Max.

Kết quả trong bài báo (Paper) [1], PTime: thời gian trên CPU, tính bằng giây (s); TL: Giới hạn thời gian chạy trên CPU là 3600 (s); LB: Lower Bound – Giới hạn dưới (Feasible Solution – Lời giải trong MIP); UP: Upper Bound – Giới hạn trên hay kết quả lời giải đạt được; MIP_Paper: Phương pháp quy hoạch nguyên tuyến tính; BS_Paper: Thuật toán tìm kiếm nhị phân.

Bảng 2 Kết quả thực nghiệm bài toán Col-Peri-Max - tập dữ liệu MN

Data		Col-Peri-Max				Paper			
#	Size	Time	AV	MaxV	MinV	PTime	LB	UB	LB_1
p01	10	0.14	**50.83**	**50.83**	**50.83**	0.31	**50.83**	**50.83**	50.75
p02	10	0.39	**51.5**	**51.5**	**51.5**	0.1	**51.5**	**51.5**	47.67
p03	10	0.08	**51**	**51**	**51**	0.27	**51**	**51**	50.91
p04	15	1.58	**39.8**	**39.8**	**39.8**	20.37	**39.8**	**39.8**	**39.8**
p05	15	1.11	**40.99**	**40.99**	**40.99**	94.84	**40.99**	**40.99**	**40.99**
p06	15	1.16	**45.96**	**45.96**	**45.96**	TL	44.67	**45.96**	**45.96**
p07	20	0.96	**53.17**	**53.17**	**53.17**	9.13	**53.17**	**53.17**	52.92
p08	20	1.01	53.99	54.25	**53.38**	35	**53.38**	**53.38**	**53.38**
p09	20	3.06	**54.55**	**54.55**	**54.55**	TL	36.61	**54.55**	**54.55**
p10	25	9.72	**43.08**	**43.08**	**43.08**	TL	29.04	**43.08**	**43.08**
p11	25	3.59	54.6	54.75	**54.55**	TL	40.91	**54.55**	**54.55**
p12	25	8.07	**53.67**	**53.67**	**53.67**	1699.02	**53.67**	**53.67**	**53.67**
p13	30	3.52	**54.12**	**54.12**	**54.12**	139.13	**54.12**	**54.12**	**54.12**
p14	30	13.3	**54.41**	**54.41**	**54.41**	TL	35	**54.41**	**54.41**
p15	30	3.47	**51.23**	**51.23**	**51.23**	42.77	**51.23**	**51.23**	**51.22**
p16	35	15.83	**55.43**	**55.43**	**55.43**	TL	39.18	**55.43**	**55.43**
p17	35	42.63	**53.33**	**53.33**	**53.33**	TL	40.41	**53.33**	53.07
p18	35	10.68	**52.38**	52.63	**52.31**	TL	44.24	52.57	**52.31**
p19	40	21.86	**56.43**	**56.43**	**56.43**	TL	34.26	**56.43**	**56.43**
p20	40	51.53	**55.28**	**55.28**	**55.28**	TL	30.57	**55.28**	**55.28**
p21	40	28.18	**54.55**	**54.55**	**54.55**	TL	9.94	**54.55**	**54.55**

Bảng 3 Kết quả thực nghiệm bài toán Col-Peri-Max - tập dữ liệu MU

Data		Col-Peri-Max				Paper			
#	Size	Time	AV	MaxV	MinV	PTime	LB	UB	LB_1
p01	10	0.08	**55.29**	**55.29**	**55.29**	0.31	**55.29**	**55.29**	**55.29**
p02	10	0.4	**55.78**	**55.78**	**55.78**	0.07	**55.78**	**55.78**	52.15
p03	10	0.12	**52.76**	**52.76**	**52.76**	1.22	**52.76**	**52.76**	**52.76**
p04	15	0.37	**56.17**	**56.17**	**56.17**	0.49	**56.17**	**56.17**	55.57
p05	15	1.36	**51.23**	**51.23**	**51.23**	TL	43.43	**51.23**	**51.23**
p06	15	0.58	**54.7**	**54.7**	**54.7**	111.49	**54.7**	**54.7**	**54.7**
p07	20	2.57	**55.86**	**55.86**	**55.86**	TL	40.1	**55.86**	**55.86**
p08	20	2.82	**53.81**	**53.81**	**53.81**	TL	28.26	**53.81**	**53.81**
p09	20	1.18	**55.86**	**55.86**	**55.86**	285.98	**55.86**	**55.86**	**55.86**
p10	25	5.51	**55.71**	**55.71**	**55.71**	TL	40.79	**55.71**	**55.71**
p11	25	2.23	54.99	55.5	**54.85**	TL	51.08	**54.85**	**54.85**
p12	25	1.58	**56.09**	56.9	**56**	TL	47.41	56.17	55.86
p13	30	12.35	**53.07**	**53.07**	**53.07**	TL	21.43	53.07	**53.07**
p14	30	7.51	**55.43**	**55.43**	**55.43**	TL	42.46	**55.43**	**55.43**
p15	30	8.84	**55.43**	**55.43**	**55.43**	TL	32.4	**55.43**	**55.43**
p16	35	9.13	**55.86**	**55.86**	**55.86**	TL	41.47	**55.86**	**55.86**
p17	35	17.18	**44.36**	**44.36**	**44.36**	TL	23.28	**44.36**	**44.36**
p18	35	17.23	**55.28**	**55.28**	**55.28**	TL	32.23	**55.28**	**55.28**
p19	40	32.89	**54.7**	**54.7**	**54.7**	TL	21.41	**54.7**	**54.7**
p20	40	33.28	**56**	**56**	**56**	TL	29.09	**56**	**56**
p21	40	32.82	**52.76**	**52.76**	**52.76**	TL	15.76	**52.76**	**52.76**

Bảng 4 Kết quả thực nghiệm bài toán Col-Peri-Max - tập dữ liệu U

Data		Col-Peri-Max				Paper			
#	Size	Time	AV	MaxV	MinV	PTime	LB	UB	LB_1
p01	10	0.09	**52.53**	**52.53**	**52.53**	0.53	**52.53**	**52.53**	52
p02	10	0.1	**55.17**	**55.17**	**55.17**	0.2	**55.17**	**55.17**	54.7
p03	10	0.1	**55.54**	**55.54**	**55.54**	0.2	**55.54**	**55.54**	**55.43**
p04	15	0.82	**55.14**	**55.14**	**55.14**	TL	52.27	**55.14**	**55.14**
p05	15	0.49	**51.38**	**51.38**	**51.38**	92.46	**51.38**	**51.38**	**51.38**
p06	15	1.05	**52.46**	**52.46**	**52.46**	TL	48.63	**52.46**	**52.46**
p07	20	1.45	56.58	56.62	**56.57**	TL	43.27	**56.57**	**56.57**
p08	20	2.88	**56.14**	**56.14**	**56.14**	TL	34.76	**56.14**	**56.14**
p09	20	1.06	**53.71**	**53.71**	**53.71**	3.37	**53.71**	**53.71**	53.67
p10	25	3.95	**55.86**	**55.86**	**55.86**	TL	49	**55.86**	**55.86**
p11	25	4.91	**55.14**	**55.14**	**55.14**	TL	45.82	**55.14**	**55.14**
p12	25	4.83	**53.96**	**53.96**	**53.96**	TL	33.01	**53.96**	**53.96**
p13	30	10.89	**54.7**	**54.7**	**54.7**	TL	25.86	**54.7**	**54.7**
p14	30	7.46	**54.15**	**54.43**	**54.11**	TL	40.41	**54.11**	**54.11**
p15	30	7.22	53.38	53.41	**53.37**	TL	33.64	**53.37**	**53.37**
p16	35	26.79	**56.43**	**56.43**	**56.43**	TL	40	**56.43**	**56.43**
p17	35	14.93	**56.43**	**56.43**	**56.43**	TL	36.01	**56.43**	**56.43**
p18	35	9.16	55.76	56.27	**55.71**	TL	41.35	**55.71**	**55.71**
p19	40	25.29	**56.43**	**56.43**	**56.43**	TL	19.3	**56.43**	**56.43**
p20	40	21.57	**54.7**	**54.7**	**54.7**	TL	44.48	54.79	**54.7**
p21	40	18.14	**56.14**	**56.14**	**56.14**	TL	44.12	**56.14**	**56.14**

Bảng 5 Kết quả thực nghiệm bài toán Col-Aspect-Ratio - tập dữ liệu MN

Data		Col-Aspect-Ratio				MIP_Paper				BS_Paper		
#	Size	Time	AV	MaxV	MinV	PTime	LB_4	UB_4	UB_3	PTime	LB	UB
p01	10	0.09	**2.75**	**2.75**	**2.75**	0.27	1.05	1.05	**2.75**	0.87	2.74	**2.75**
p02	10	0.27	**2.22**	**2.22**	**2.22**	0.06	0.82	0.82	**2.22**	0.41	2.21	**2.22**
p03	10	0.09	5.62	5.86	**5.6**	0.28	1.94	1.94	**5.6**	0.75	5.59	**5.6**
p04	15	0.78	**4.17**	**4.17**	**4.17**	18.08	1.55	1.55	**4.17**	147.71	4.17	**4.17**
p05	15	0.74	**2.68**	**2.68**	**2.68**	5.06	1.03	1.03	**2.68**	20.51	2.67	**2.68**
p06	15	0.53	**4.49**	**4.49**	**4.49**	0.86	1.65	1.65	**4.49**	5.18	4.49	4.5
p07	20	0.82	9.86	9.89	**9.83**	200.99	2.82	2.82	**9.83**	2866.39	9.83	**9.83**
p08	20	0.91	6.17	6.33	**6.09**	484.53	2.06	2.06	**6.09**	259.48	6.08	**6.09**
p09	20	1.38	2.99	3.16	**2.73**	290.84	1.05	1.05	**2.73**	554.92	2.73	**2.73**
p10	25	5.17	2.15	2.43	**1.94**	TL	0.56	0.67	**1.94**	TL	1.76	1.95
p11	25	2.39	**8.08**	**8.09**	**8.08**	11.2	2.49	2.49	8.09	76.6	8.09	8.09
p12	25	3.66	2.84	3.19	**2.45**	TL	0.55	0.93	**2.45**	661.27	2.44	**2.45**
p13	30	4.39	6.47	7.74	**6.18**	TL	1.39	2.08	**6.18**	TL	5.47	6.21
p14	30	6.72	4.27	4.84	**3.05**	TL	0.8	1.18	3.06	TL	2.28	3.55
p15	30	4.36	9.97	10.96	**9.45**	TL	1.85	2.81	9.81	TL	6.85	12.69
p16	35	10.78	7.45	8.57	**5.08**	TL	0.14	1.82	5.12	TL	1	14.14
p17	35	21.55	2.15	2.3	**1.73**	TL	0.14	0.56	**1.73**	TL	1.55	1.74
p18	35	7.12	6.64	7.8	**5.26**	TL	0.55	1.91	5.48	TL	1	9.88
p19	40	21.56	5.62	5.84	**5.42**	TL	0.59	1.93	5.53	TL	1	6.92
p20	40	30.9	**2.62**	**2.62**	**2.62**	TL	0.84	1	**2.62**	TL	2.31	2.64
p21	40	18.36	6.68	8.14	**4.64**	TL	1.09	1.71	4.7	TL	1	5.09

Bảng 6 Kết quả thực nghiệm Bài toán Col-Aspect-Ratio - tập dữ liệu MU

Data		Col-Aspect-Ratio				MIP_Paper				BS_Paper		
#	Size	Time	AV	MaxV	MinV	PTime	LB_4	UB_4	UB_3	PTime	LB	UB
p01	10	0.1	**3.65**	**3.65**	**3.65**	0.24	1.39	1.39	**3.65**	1.03	3.64	**3.65**
p02	10	0.31	**3.57**	**3.57**	**3.57**	0.18	1.36	1.36	**3.57**	1.38	3.56	**3.57**
p03	10	0.08	5.8	6.65	**5.71**	0.18	1.97	1.97	**5.71**	0.72	5.7	**5.71**
p04	15	0.31	**6.67**	**6.67**	**6.67**	1.67	2.2	2.2	**6.67**	12.05	6.66	**6.67**
p05	15	0.79	**3.57**	**3.57**	**3.57**	280.22	1.36	1.36	**3.57**	451.13	3.56	**3.57**
p06	15	0.29	6.91	7.9	**6.39**	1.87	2.13	2.13	**6.39**	8.82	6.39	**6.39**
p07	20	1.27	**6.19**	**6.19**	**6.19**	1885.98	2.09	2.09	**6.19**	1450.33	6.19	**6.19**
p08	20	2.06	2.75	2.81	**2.74**	TL	0.95	1.05	**2.74**	TL	2.69	4.38
p09	20	1.12	6.7	6.8	**6.61**	550.19	2.18	2.18	**6.61**	2355.92	6.61	**6.61**
p10	25	3.33	4.17	5.16	**3.48**	TL	0.76	1.33	**3.48**	TL	2.98	3.96
p11	25	1.75	7.05	9.18	**6.52**	TL	1.47	2.24	6.86	TL	1	12.49
p12	25	2.34	6.51	6.73	**6.16**	TL	1.77	2.13	6.39	TL	5.72	6.66
p13	30	7.15	3.85	4.55	**3.61**	TL	0.61	1.4	3.68	TL	1	7.02
p14	30	5.2	6.24	7.9	**5.24**	TL	1.28	1.96	5.67	TL	3.69	6.38
p15	30	6.94	5.67	6.75	**5.2**	TL	0.91	1.86	5.27	TL	1	7.1
p16	35	9.85	6.52	8.81	**6.09**	TL	0.27	2.06	**6.09**	TL	4.94	8.89
p17	35	16.94	**3.67**	**3.67**	**3.67**	297.25	1.39	1.39	**3.67**	471.57	3.67	**3.67**
p18	35	13.82	6.42	6.63	**5.59**	TL	0.04	1.96	5.68	TL	1	7.2
p19	40	21.66	4.09	4.19	**3.83**	181.11	1.48	1.48	3.92	TL	3.87	4.08
p20	40	23.42	4.36	4.58	**4.08**	TL	0.11	1.53	4.11	TL	3.88	5.32
p21	40	23.15	4.11	4.43	**3.31**	TL	0	1.3	3.39	TL	1	4.6

Bảng 7 Kết quả thực nghiệm Bài toán Col-Aspect-Ratio - tập dữ liệu U

Data		Col-Aspect-Ratio				MIP_Paper				BS_Paper		
#	Size	Time	AV	MaxV	MinV	PTime	LB_4	UB_4	UB_3	PTime	LB	UB
p01	10	0.1	2.54	2.55	**2.51**	0.39	0.95	0.95	**2.51**	1.21	2.51	**2.51**
p02	10	0.09	4.75	4.78	**4.75**	0.22	1.72	1.72	**4.75**	1.38	4.75	**4.75**
p03	10	0.09	**2.73**	**2.73**	**2.73**	0.08	1.05	1.05	**2.73**	0.93	2.72	**2.73**
p04	15	0.78	**2.22**	**2.22**	**2.22**	0.74	0.82	0.82	**2.22**	4.91	2.22	**2.22**
p05	15	0.49	**2.59**	**2.59**	**2.59**	1.85	0.99	0.99	**2.59**	4.27	2.59	2.6
p06	15	1.03	**5.76**	**5.76**	**5.76**	479.41	1.98	1.98	**5.76**	560.23	5.75	**5.76**
p07	20	1.33	8.64	8.66	**8.63**	3.38	2.6	2.6	8.64	29.04	8.63	8.64
p08	20	2.86	**1.59**	**1.59**	**1.59**	202.12	0.47	0.47	**1.59**	174.54	1.59	**1.59**
p09	20	0.99	4.16	4.33	**3.86**	12.31	1.46	1.46	**3.86**	35.94	3.85	**3.86**
p10	25	3.09	**2.07**	**2.07**	**2.07**	851.98	0.74	0.74	**2.07**	679.89	2.06	**2.07**
p11	25	5.75	4.49	4.66	**4.24**	12.37	1.6	1.6	4.33	TL	4.29	4.37
p12	25	4.03	**2.65**	**2.65**	**2.65**	TL	0.81	1.01	**2.65**	TL	2.62	4.25
p13	30	11.7	2.37	2.49	**2.21**	TL	0	0.86	2.3	TL	2.2	3.41
p14	30	7.64	5.17	5.59	**5**	TL	0.22	1.79	5.01	TL	1	7.22
p15	30	6.92	9.72	10.25	**8.05**	TL	1.71	2.59	8.57	TL	1	17.06
p16	35	25.31	**5**	**5**	**5**	TL	0	1.79	**5**	TL	1	5.34
p17	35	20.02	4.08	4.24	**3.89**	TL	0	1.47	3.92	TL	1	8.34
p18	35	11.69	8.73	10.54	**8.35**	TL	0.34	2.59	8.58	TL	1	28.7
p19	40	26.98	4.04	4.89	**3.76**	TL	0.64	1.42	3.76	TL	1	5.86
p20	40	25.73	3.55	3.9	**2.83**	TL	0.12	1.09	**2.83**	TL	1.93	2.86
p21	40	17.3	9.6	11.19	**9.18**	TL	0.05	2.73	9.35	TL	1	21.33

4.2.2 Đánh giá kết thực nghiệm

Kết quả thực nghiệm của hai bài toán Col-Peri-Max và Col-Aspect-Ratio được trình bày trong mục 4.2.1. Sau đây, tôi so sánh kết quả thực nghiệm trong đồ án và kết quả thực nghiệm trong bài báo [1].

Chú thích:

Bộ dữ liệu [6], gồm 3 tập dữ liệu là: tập MN, tập MU, tập U. Trong mỗi tập dữ liệu có 21 trường hợp, số lượng hình chữ nhật $n \in \{10, 15, 20, 25, 30, 35, 40\}$.

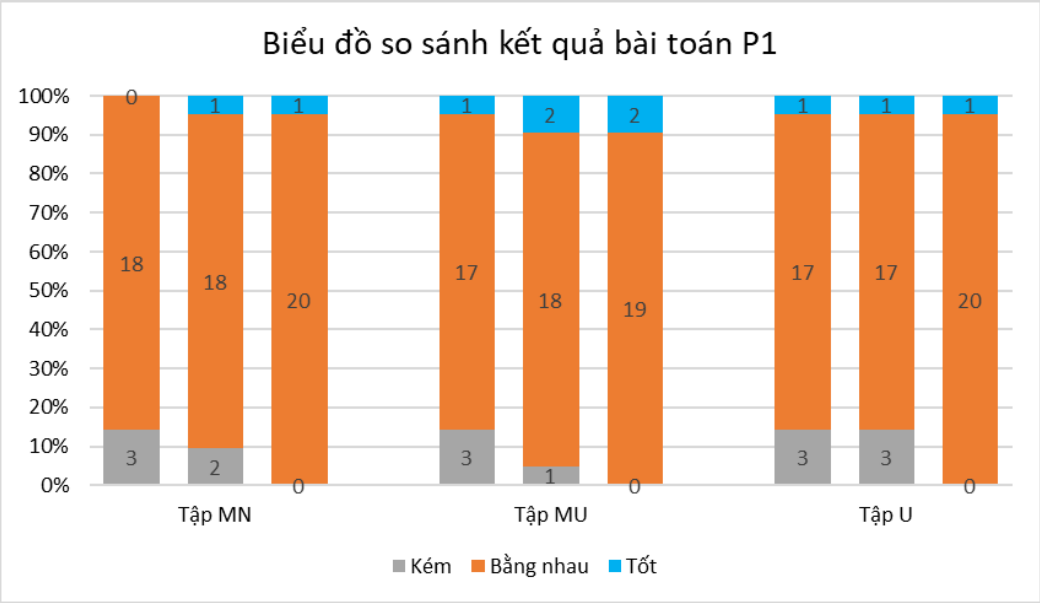
Trong biểu đồ Hình 5 và Hình 6, mỗi tập MN, MU, U có 3 cột tương ứng thể hiện trong trường hợp: giá trị hàm mục tiêu lớn nhất (MaxV), giá trị trung bình các lần thực hiện (AV) và giá trị hàm mục tiêu nhỏ nhất (MinV) trong 10 lần chạy thực nghiệm. Số liệu thực nghiệm trong bảng 2-7 mục 4.2.1.

(i) Kết quả thực nghiệm của thuật toán

Trong mục 4.2.1, tôi đã trình bày các bảng kết quả thực nghiệm của đồ án và bài báo [1], các phần được in đậm thể hiện thuật toán có kết quả tốt hơn trong từng hàng. Cụ thể, tôi so sánh số lượng trường hợp kết quả đồ án và bài báo [1]: kết quả đồ án tốt hơn (Tốt), kết quả bằng nhau (Bằng nhau) và kết quả đồ án kém hơn (Kém) trong Hình 5 và Hình 6.

Trong mỗi tập MN, MU, U có 3 cột tôi ký hiệu lần lượt là MaxV, AV, MinV, tôi đã trình bày trong chú thích. Kết quả thực nghiệm mục 4.2.1, tôi thực hiện chương trình 10 lần trong mỗi trường hợp, rồi xử lý kết quả tìm giá trị hàm mục tiêu lớn nhất trong 10 lần (MaxV), giá trị hàm mục tiêu trung bình (AV), giá trị hàm mục tiêu nhỏ nhất trong 10 lần (MinV). Tôi so sánh kết quả trong 3 trường hợp trên trong mỗi tập MN, MU, U với bài báo [1].

Trong mỗi tập, có 3 cột kết quả theo thứ tự MaxV, AV và Min V và kết quả số lượng trường hợp: thuật toán CBLS đề xuất tốt hơn bài báo (Tốt), thuật toán và bài báo có kết quả bằng nhau (Bằng nhau) và thuật toán CBLS kém hơn bài báo (Kém). Kết quả cụ thể trong Hình 5 và Hình 6.

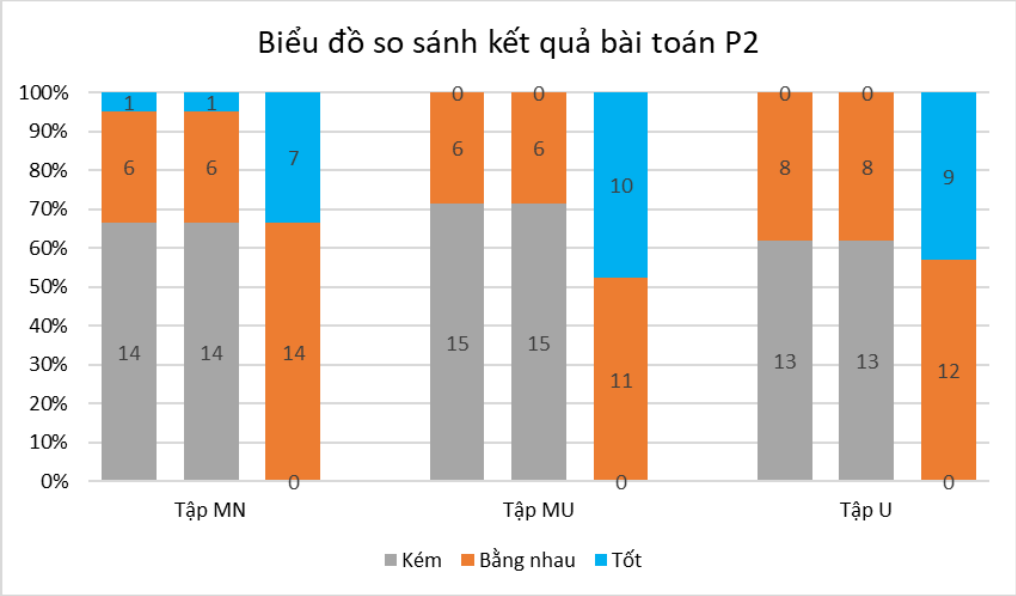


Hình 5 Biểu đồ so sánh kết quả bài toán Col-Peri-Max

Hình 5 thể hiện kết quả so sánh bài toán Col-Peri-Max của đồ án và bài báo [1], đa số kết quả bằng nhau và tốt hơn trong ba tập MN, MU, U. Trong cột MinV (kết quả hàm mục tiêu tốt nhất trong 10 lần chạy), kết quả đồ án luôn bằng và tốt hơn trong bài báo. Trong cột MaxV, AV trong ba tập, số lượng lời giải kém hơn từ 1 đến 3 trường hợp (tương ứng từ 4.76% đến 14.29%), vẫn có trường hợp tìm được lời giải tốt hơn trong bài báo. Tôi đã chứng minh trong phần phụ lục C, LB_1 là cận dưới hàm mục tiêu của bài toán Col-Peri-Max và có thể có trường hợp đạt đến giá trị LB_1, trường hợp đạt giá trị LB_1 chắc chắn là tối ưu toàn cục

(Global Optimum). Trong trường hợp kết quả đồ án và bài báo bằng nhau có nhiều trường hợp đạt giá trị LB_1. Trong bài báo [1], sử dụng thuật toán MIP khi giá trị LB = UB thì đã tìm được tối ưu toàn cục, bài báo giải tốt bài toán có kích thước nhỏ từ 10 đến 25 hình chữ nhật. Trong đồ án, các tập dữ liệu MN, MU, U lần lượt có 20/21, 20/21, 21/21 trường hợp tìm được giá trị hàm mục tiêu bằng với giá trị LB_1, giá trị tối ưu toàn cục khi so sánh với bài báo [1]. Trong một số trường hợp, tôi tìm ra lời giải tốt hơn bài báo và đạt giá trị LB_1 là MN-p18, MU-p13 và U-p20. Trường hợp, MU-p12 tôi tìm được lời giải tốt hơn bài báo, có hai trường hợp MN-p17 và MU-p12 là lời giải tối ưu, cần xem xét về lời giải tối ưu toàn cục.

Dựa trên kết quả thực nghiệm chỉ ra rằng: các trường hợp số lượng hình chữ nhật nhỏ, thuật toán CBLS cho kết quả lời giải tối ưu được kiểm chứng bằng thuật toán MIP trong bài báo [1] và trong các trường hợp có kích thước lớn hơn thì CBLS trả ra lời giải tốt hơn bài báo. Bài báo cho lời giải tốt bài toán với quy mô vừa và nhỏ (từ 10 đến 25 hình chữ nhật). Thuật toán CBLS tôi đề xuất giải tốt bài toán Col-Peri-Max trong quy mô từ 10 đến 40 hình chữ và chất lượng lời giải tốt trong các lần thực hiện.



Hình 6 Biểu đồ so sánh kết quả bài toán Col-Aspect-Ratio

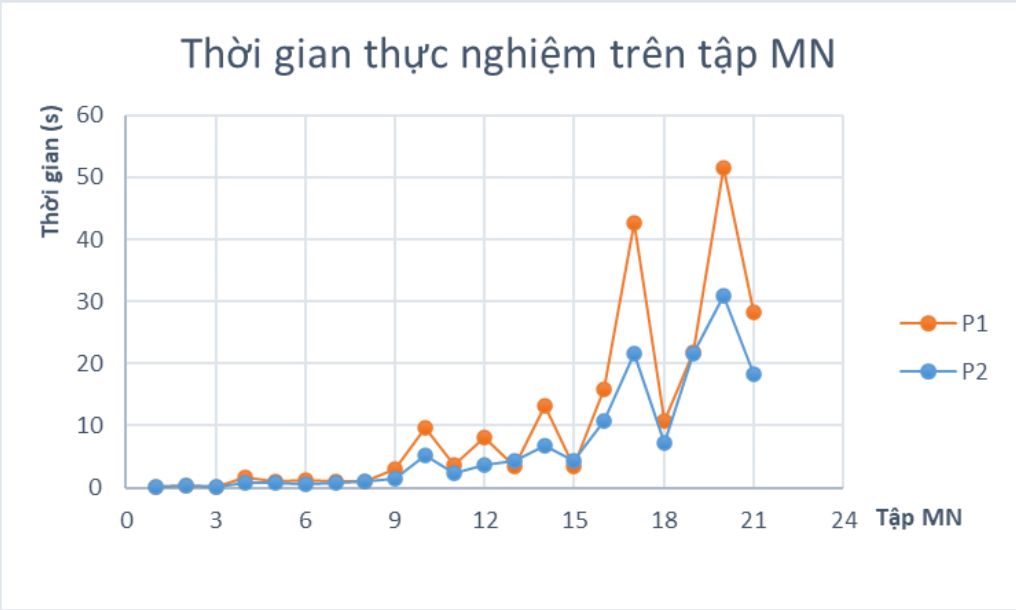
Hình 6 thể hiện kết quả so sánh của đồ án và bài báo [1] trong bài toán Col-Aspect-Ratio, kết quả thực nghiệm trong 3 tập dữ liệu và trong cùng tập dữ liệu thì các trường hợp MaxV, AV, MinV cũng có nhiều sự biến đổi. Trong trường hợp tốt nhất MinV, kết quả đồ án luôn bằng và tốt hơn trong bài báo và trong tập MU có 10/21 (47.62%) tốt hơn bài bào. Trong trường hợp tệ nhất MaxV, tập MU có 15/21 trường hợp kém hơn bài báo và có 6/21 trường hợp tốt bằng bài báo [1]. Trong bài toán Col-Aspect-Ratio, thuật toán CBLS đề xuất đã tìm được lời giải tốt hơn hoặc bằng bài báo trong cột MinV (kết quả hàm mục tiêu tốt nhất trong 10 lần thực nghiệm). Tuy nhiên, trong cột MaxV, AV vẫn còn nhiều trường hợp tìm được lời giải kém hơn trong bài báo [1]. Kết quả Hình 6 dựa trên so sánh kết quả thuật toán đề

xuất của đồ án với hai phương pháp được đề xuất trong bài báo là MIP và tìm kiếm nhị phân. Dựa trên kết quả thực nghiệm cho thấy, thuật toán CBLS đề xuất cho kết quả tốt trong bài toán kích thước nhỏ và có thể tìm được lời giải tốt hơn bài báo [1] trong một số trường hợp kích thước lớn hơn.

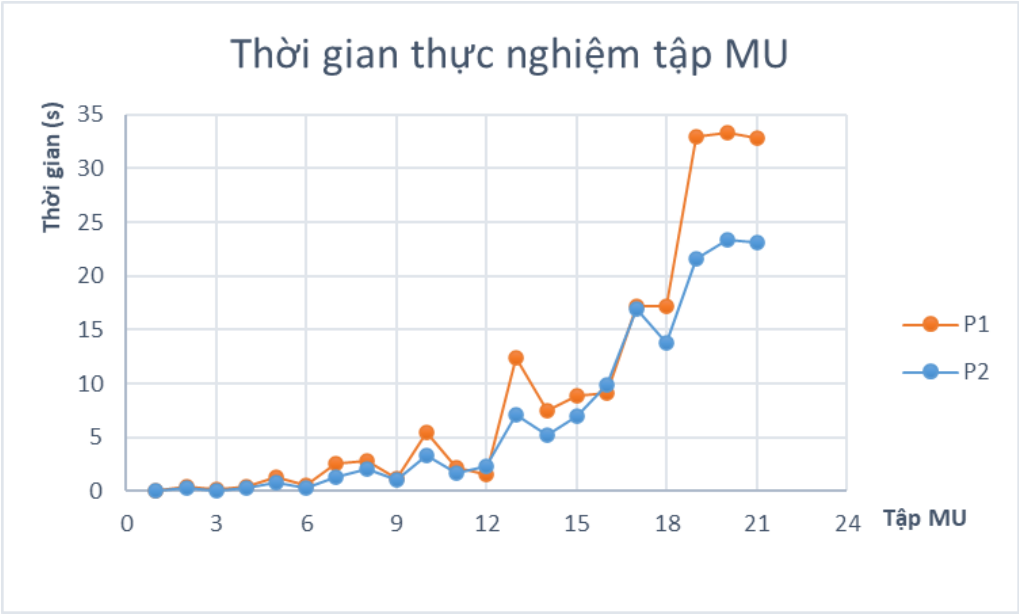
Bài toán Col-Aspect-Ratio có thể được phát biểu, tìm trong số các lời giải, lời giải mà tỷ lệ chiều dài và chiều rộng lớn nhất của một hình chữ nhật là nhỏ nhất. Theo phụ lục C, cận dưới hàm mục tiêu của bài toán P_2 là 1, khi tất cả các hình chữ nhật được chia đều là hình vuông. Tuy nhiên trong các trường hợp thực nghiệm, tỷ lệ của hai kích thước hình chữ nhật có nhiều sự thay đổi và khó có lời giải đạt giá trị bằng 1. Tôi có thể sử dụng ý tưởng của giải thuật tham lam để xét xem có tồn tại lời giải chia được tất cả các hình chữ nhật thành hình vuông hay không. Giả sử, tất cả các hình chữ nhật cần chia là hình vuông, ta được bài toán mới là xem có thể sắp xếp tất cả các hình chữ nhật này vào hình chữ nhật lớn hay không sao cho đảm bảo nhát cắt "guillotine" (được trình bày ở mục 1.2.1).

(ii) Thời gian thực hiện chương trình trên CPU

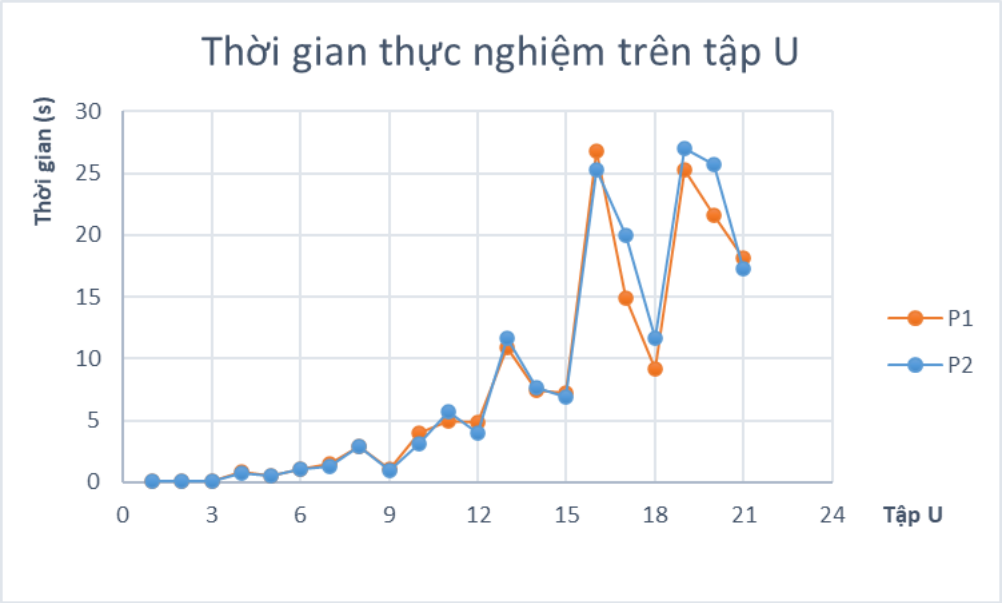
Thuật toán CBLS đề xuất có thể chạy trong giới hạn về thời gian thực hiện. Thời gian thực nghiệm thuật toán CBLS có giá trị lớn nhất là 52.53 giây trong bài toán Col-Peri-Max và 30.9 giây trong bài toán Col-Aspect-Ratio. Thời gian thực nghiệm bài toán Col-Peri-Max và Col-Aspect-Ratio trong ba tập dữ liệu MN, MU, U, được thể hiện trong Hình 7, Hình 8 và Hình 9.



Hình 7 Thời gian thực nghiệm trên tập MN



Hình 8 Thời gian thực nghiệm trên tập MU



Hình 9 Thời gian thực nghiệm trên tập U

(iii) Độ ổn định (stable) của giá trị hàm mục tiêu trong quá trình thực nghiệm

Để đánh giá độ ổn định của thuật toán, trong quá trình thực nghiệm, tôi tính độ chênh lệch của giá trị hàm mục tiêu lớn nhất và giá trị hàm mục tiêu nhỏ nhất mỗi trường hợp trong 10 lần thực nghiệm. Giá trị tỷ lệ của giá trị hàm mục tiêu lớn nhất và giá trị hàm mục tiêu nhỏ nhất (MaxV/MinV) trong ba tập dữ liệu được thể hiện trong Bảng 8 (lấy giá trị trung trình trong mỗi tập về 100%).

Bảng 8 Tỷ lệ giá trị hàm mục tiêu lớn nhất và nhỏ nhất trong thực nghiệm

Tập/ Tỷ lệ	Bài toán Col-Peri-Max		Bài toán Col-Aspect-Ratio	
	Trung bình (%)	Lớn nhất (%)	Trung bình (%)	Lớn nhất (%)
Tập MN	100	101.6	100	144.63
Tập MU	100	101.6	100	134.82
Tập U	100	101	100	126.61

Từ Bảng 8 cho thấy, thuật toán ổn định trong quá trình tìm kiếm bài toán Col-Peri-Max khi thực hiện trong nhiều trường hợp, giá trị tối ưu của hàm mục tiêu không có nhiều thay đổi và nhiều trường hợp đều tìm ra được cùng một lời giải tối ưu (đã nêu ở phần trên). Trong bài toán Col-Aspect-Ratio, tỷ lệ này có nhiều sự thay đổi hơn và cần xem xét các giá trị tối ưu của bài toán.

Trong thuật toán tìm kiếm cục bộ, hai vấn đề quan trọng nhất cần quan tâm là tính tăng cường (intensification) và tính đa dạng (diversification), tôi đã trình bày trong phần cơ sở lý thuyết mục 2.2.1. Tôi vừa trình bày, thuật toán đề xuất ổn định trong quá trình thực nghiệm và xem xét để đảm bảo hai tính tăng cương và tính đa dạng. Trong lời giải khởi tạo ban đầu, tôi sinh ngẫu nhiên các giá trị của biến quyết định từ 1 đến n và sử dụng một số ràng buộc để đảm bảo không tồn tại lớp trống và hạn chế các lời giải đối xứng, cũng như sử dụng một số Metaheuristics trong quá trình tìm kiếm để đảm bảo tính đa dạng của thuật toán trong quá trình tìm kiếm cục bộ. Tôi đã định nghĩa bốn loại hàng xóm để có thể tìm kiếm sâu hơn trong không gian tìm kiếm.

(iv) Đánh giá chung kết quả thực nghiệm của thuật toán

Trong ba phần được trình bày ở trên, tôi đã đánh giá độ tối ưu của thuật toán đề xuất, thời gian thực hiện và độ ổn định của thuật toán. Kết quả thực nghiệm đạt được tốt so với bài báo [1], cho lời giải tối ưu bài toán kích thước nhỏ và cho lời giải tốt bài toán kích thước lớn hơn trong bài báo [1]. Thuật toán CBLS đề xuất cho kết quả tốt trong hai bài toán Col-Peri-Max và Col-Aspect-Ratio trong quy mô từ 10 đến 40 hình chữ nhật.

Chương 5 Kết luận và hướng phát triển

5.1 Kết luận

Tôi đã đề xuất và cài đặt thuật toán tìm kiếm cục bộ dựa trên ràng buộc (CBLS) và Metaheuristic giải bài toán SRP. Kết quả thực nghiệm đạt được tốt khi so sánh với kết quả trong bài báo [1] trong thời gian tính toán giới hạn. Thuật toán cho kết quả tối ưu khi giải bài toán quy mô vừa, nhỏ (từ 10 đến 25 hình chữ nhật) và kết quả tốt trong trường hợp quy mô lớn hơn (từ 30 đến 40 hình chữ nhật). Tôi đã tìm cận dưới của hàm mục tiêu bài toán Col-Peri-Max và kết quả thực nghiệm tối ưu trong bài toán này, bài toán Col-Aspect-Ratio cho kết quả thực nghiệm tốt hơn bài báo [1] trong một số trường hợp nhưng vẫn còn trường hợp chưa đạt đến kết quả tối ưu trong bài báo.

Trong quá trình thực hiện đồ án, tôi đã cố gắng thực hiện đồ án theo từng giai đoạn tiếp cận vấn đề, tìm hiểu các vấn đề liên quan và đề xuất giải pháp. Đồ án Tốt nghiệp là kết quả quá trình cố gắng của bản thân và sự hướng dẫn tận tình của thầy Bùi Quốc Trung. Tuy nhiên, việc sắp xếp thời gian chưa được hợp lý và kiến thức của của bản thân còn hạn chế nên kết quả chưa thật tốt. Tôi cần hoàn thiện bản thân hơn để phát triển trong quá trình học tập và làm việc sau này.

5.2 Hướng phát triển

Tôi đã trình bày kết quả quá trình làm đồ án trong mục 5.1 ở trên. Kết quả thực nghiệm đạt được tốt khi so sánh với kết quả bài báo [1]. Thuật toán cho kết quả tối ưu trong bài toán Col-Peri-Max có kích thước từ 10 đến 40 hình chữ nhật. Tôi thấy một số vấn đề cần hoàn thiện: đánh giá trường hợp bài toán quy mô lớn hơn và tối ưu giải bài toán Col-Aspect-Ratio.

Tôi đã công bố mã nguồn trên “GitHub” và chia sẻ quá trình làm đồ án của mình (Đường dẫn https://github.com/tuandungdao/CBLS_SoftRectanglePacking), đây có thể là tài liệu tham khảo ban đầu cho mọi người tìm hiểu bài toán, hướng phát triển giải quyết bài toán quy mô lớn hơn và đặc biệt là bài toán chia ruộng trên thực tế trên nhiều cánh đồng và sẽ có nhiều ràng buộc hơn. Tôi hi vọng bài toán thực tế chia ruộng, dồn điền đổi thửa tại Việt Nam được thực hiện hiệu quả hơn và nước ta có thể phát triển theo mô hình nông nghiệp thông minh trên thế giới.

Tài liệu tham khảo

[1] Quoc Trung Bui, Thibaut Vidal, Minh Hoang Ha, On three soft rectangle packing problems with “guillotine” constraints, Journal of global optimization 74 (1), 45-62, 2019

[2] Nguyễn Đức Nghĩa, Nguyễn Tô Thành, Toán rời rạc, Nhà xuất bản Đại học Quốc gia Hà Nội, 2009.

[3] Pascal Van Hentenryck and Laurent Michel, Constraint-Based Local Search, The MIT Press, 2005.

[4] Michel L., Van Hentenryck P., Constraint-Based Local Search. In: Martí R., Panos P., Resende M. (eds) Handbook of Heuristics. Springer, Cham, 2017.

[5] Quoc Trung Bui, Modelling and solving complex combinatorial optimization problems: quorumcast routing, elementary shortest path, elementary longest path and agricultural land allocation, Ph.D. Thesis, Université Catholique de Louvain, Belgium, 2015.

[6] Thibaut Vidal, https://w1.cirrelt.ca/~vidalt/en/research-data.html last visited June, 2021.

[7] Báo Kinh tế Nông nghiệp, 2008, Cơ giới hoá nông nghiệp: Khi nào qua bước khởi động? http://ipsard.gov.vn/vn/tID2264_Co-gioi-hoa-nong-nghiep-Khi-nao-qua-buoc-khoi-dong-.html lần truy cập cuối 16/6/2021.

[8] Báo Kinh tế Thái Nguyên, 2021, Hiệu quả khi dồn điền đổi thửa, thực hiện cánh đồng lớn, https://thainguyentv.vn/hieu-qua-khi-don-dien-doi-thua-thuc-hien-canh-dong-lon-84262.html lần truy cập cuối 16/06/2021.

[9] Beaumont, O., V. Boudet, F. Rastello, Y. Robert, Partitioning a square into rectangles: NP Completeness and approximation algorithms, Algorithmica 34(3) 217–239, 2002.

[10] Nagamochi, H., Y. Abe., An approximation algorithm for dissecting a rectangle into rectangles with specified areas, Discrete Applied Mathematics 155(4) 523–537, 2007.

Phụ lục

A Một số khái niệm trong báo cáo

Định nghĩa 2.1: Biến quyết định của bài toán (Decision variable)

Biến quyết định của bài toán tối ưu hóa tổ hợp là một biến nằm trong mô hình của bài toán mà người ra quyết định có thể kiểm soát được giá trị của biến đó và cần xác định giá trị cho biến đó.

Biến quyết định có thể coi như những tế bào xây dựng nên bài toán tối ưu hóa tổ hợp. Đó không phải là những biến ngẫu nhiên, người ra quyết định hoàn toàn có thể kiểm soát và quyết định được giá trị của nó. Ví dụ: số lượng những chiếc xe sẽ được sản xuất, giá thành của một chiếc xe, … Biến quyết định của bài toán thường được ký hiệu là x, và tập hợp các biến quyết định của bài toán là X.

Định nghĩa 2.2: Miền xác định của biến quyết định (Domain)

Miền giá trị của biến quyết định là tập hợp các giá trị có thể gán tới các biến đó. Miền quyết định của biến có thể là một tập hữu hạn hoặc vô hạn các giá trị, miền đó có thể là liên tục hoặc rời rạc. Trong báo cáo đồ án, tôi chỉ xét các bài toán tối ưu hóa tổ hợp rời rạc, trong đó miền xác định của tất cả các biến quyết định cấu thành nên bài toán là hữu hạn và rời rạc. Miền xác định của biến quyết định x thường được ký hiệu là Dx.

Định nghĩa 2.3: Ràng buộc (Constraint)

Một ràng buộc của bài toán tối ưu hóa tổ hợp là một điều kiện mà một tập các biến quyết định của bài toán cần thỏa mãn.

Ví dụ: $x_1 + x_2 > 10$, $x_3 \geq x_4$, …

Một ràng buộc được định nghĩa trên các biến quyết định của bài toán. Trong thực tế, ràng buộc là điều chúng ta rất thường gặp, ví dụ: ngân sách chi tiêu không vượt quá một con số nhất định, giá bán của chiếc xe không được thấp hơn tổng chi phí sản xuất ra nó, … Ràng buộc của bài toán thường được ký hiệu là c, tập các ràng buộc của bài toán thường được ký hiệu là C.

Định nghĩa 2.4: Hàm mục tiêu của bài toán (Objective function)

Hàm mục tiêu của bài toán tối ưu hóa tổ hợp là một hàm được định nghĩa trên một tập các biến quyết định và cần được tối ưu hóa (cực đại hóa hoặc cực tiểu hóa).

Ví dụ: $f = x_1 \times x_2$ với:

- x_1 là biến quyết định thể hiện giá thành sản phẩm.
- x_2 là biến quyết định thể hiện số lượng của sản phẩm bán ra.
- f là hàm mục tiêu thể hiện doanh thu từ việc bán sản phẩm, cần được cực đại hóa.

Hàm mục tiêu cũng là khái niệm chúng ta rất thường gặp trong cuộc sống, như cực đại hóa doanh thu, cực tiểu hóa chi phí, … Hàm mục tiêu thường được ký hiệu là f, tập hợp các hàm mục tiêu của bài toán ký hiệu là F.

Định nghĩa 2.5: Lời giải của bài toán (Solution)

Một lời giải của bài toán tối ưu hóa tổ hợp là một bộ giá trị xác định gán cho tập các biến quyết định của bài toán: s = {<x_i, v_i >, $x_i \in X$, $v_i \in Dx_i$}.

Việc giải bài toán tối ưu tổ hợp thực chất là đi tìm lời giải tối ưu cho bài toán, do đó các thuật toán tìm kiếm như Tìm kiếm cục bộ có thể áp dụng được vào cho việc giải các bài toán dạng này. Lời giải của bài toán thường được ký hiệu là s.

Định nghĩa 2.6: Lời giải chấp nhận được của bài toán (Feasible Solution)

Một lời giải chấp nhận được của bài toán tối ưu hóa tổ hợp là một lời giải thỏa mãn mọi ràng buộc c trong tập ràng buộc C của bài toán.

Định nghĩa 2.7: Không gian lời giải của bài toán (Solution space)

Không gian lời giải của bài toán tối ưu hóa tổ hợp là tập tất cả các lời giải có thể của bài toán. Vì chúng ta đang chỉ xét các bài toán toán ưu hóa tổ hợp rời rạc, với miền giá trị của các biến là rời rạc và hữu hạn, do đó không gian lời giải của bài toán là hữu hạn. Tuy vậy, không gian lời giải của bài toán tối ưu hóa tổ hợp thường có kích thước vô cùng lớn, tăng với tốc độ hàm mũ theo kích thước của tập các biến quyết định, do đó việc xét hết toàn bộ lời giải trong không gian là không thể, đòi hỏi phải có những chiến lược tìm kiếm phù hợp. không gian lời giải của bài toán thường được ký hiệu là S.

Định nghĩa 2.8: Bài toán tối ưu hóa tổ hợp

Bài toán tối ưu hóa tổ hợp thường được cho dưới dạng bộ ba (X, C, F), trong đó:

- X là tập các biến quyết định của bài toán
- C là hợp các ràng buộc của bài toán
- F là tập các hàm mục tiêu của bài toán.

Nhiệm vụ của việc giải bài toán tối ưu hóa tổ hợp là tìm ra lời giải s, thỏa mãn mọi ràng buộc c ∈ C và đồng thời tối ưu các hàm mục tiêu f ∈ F.

Các lời giải thuộc tập S, thỏa mãn các ràng buộc C gọi là lời giải khả thi. Mục tiêu bài toán là tìm ra một lời giải s* với giá nhỏ nhất, nghĩa là f(s*) ≤ f(s) với mọi lời giải s ∈ S. Ngược lại bài toán tối ưu hóa cực đại là tìm lời giải s* với giá lớn nhất, nghĩa là f(s*) ≥ f(s) với mọi lời giải s ∈ S.

Định nghĩa 2.9: Hàng xóm (Neighborhood)

Hàng xóm thuộc tập hợp các lời giải hàng xóm, được thực hiện bởi các phép gán có thể truy cập thông qua một bước chuyển cục bộ từ lời giải hiện tại, các bước chuyển cục bộ có thể là những thay đổi nhỏ đối với lời giải hiện tại. Một hàng xóm có thể xây dựng bằng cách thay đổi một số thành phần (hoặc bộ phận hoặc tính năng) của lời giải.

B Danh mục thuật ngữ

Rectangle Packing là một bài toán đóng gói, trong đó mục tiêu là xác định xem có thể đặt một tập hợp các hình chữ nhật nhỏ nhất định vào bên trong một đa giác lớn nhất định hay không, sao cho không có hai hình chữ nhật nhỏ nào chồng lên nhau.

Heuristic là các kỹ thuật dựa trên kinh nghiệm để giải quyết vấn đề, học hỏi hay khám phá nhằm đưa ra một giải pháp mà không được đảm bảo là tối ưu.

Metaheuristic là một thủ tục cấp cao hơn hoặc heuristic được thiết kế để tìm, tạo hoặc chọn một phương pháp heuristic có thể cung cấp một giải pháp đủ tốt cho một vấn đề tối ưu hóa, đặc biệt với thông tin không đầy đủ hoặc không hoàn hảo hoặc khả năng tính toán hạn chế.

NP-khó là một tập hợp các bài toán trong lý thuyết độ phức tạp tính toán "ít nhất là khó ngang bất kì bài toán nào trong NP". Một bài toán H là NP-khó khi và chỉ khi có một bài toán NP-đầy đủ L quy về H trong thời gian đa thức.

Cauchy là bất đẳng thức so sánh giữa trung bình cộng và trung bình nhân của các số thực không âm, có tên gọi khác là bất đẳng thức AM-GM (Arithmetic Means - Geometric Means).

C Chứng minh cận dưới của hàm mục tiêu

Bất đẳng thức Cauchy được phát biểu như sau: trung bình cộng của n số thực không âm luôn lớn hơn hoặc bằng trung bình nhân của chúng, và trung bình cộng chỉ bằng trung bình nhân khi và chỉ khi n số đó bằng nhau.

Với 2 số thực dương a và b, ta có:

$\frac{a+b}{2} \geq \sqrt{ab}$ (1) Dấu "=" xảy ra khi và chỉ khi a = b

Ứng dụng của bất đẳng thức Cauchy, trong tất cả các hình chữ nhật có diện tích không đổi, hình vuông có chu vi nhỏ nhất (Dấu bằng của bất đẳng thức (1) xảy ra khi và chỉ khi a = b).

Bài toán Col-Peri-Max: cho một tập diện tích các hình chữ nhật được lưu trong mảng a = $\{a_1, \ldots, a_n\}$. Sử dụng thuật toán tìm kiếm, tìm được a_i là phần tử lớn rất trong mảng. Kết quả giá trị hàm mục tiêu tốt nhất có thể đạt được: $LB_1 = 4 * \sqrt{a_i}$. Giá trị LB_1 (Lower Bound - Giới hạn dưới của bài toán Col-Peri-Max) là cận dưới của hàm mục bài toán Col-Peri-Max, được định nghĩa như sau: với mọi hàm mục tiêu $f_i \in F$, $f_i \geq LB_1$, nghĩa là tất cả giá trị hàm mục tiêu tìm được sẽ lớn hơn hoặc bằng LB_1 và có thể tồn tại lời giải tốt nhất đạt giá LB_1 khi và chỉ khi hình a_i là hình vuông, lời giải đạt được chắc chắn là tối ưu toàn cục (Global optimum) và cũng có thể không thể phân chia a_i vào một lớp để a_i là hình vuông thì chúng ta cần tìm lời giải tối ưu có thể phân chia được. Kết quả thực nghiệm bài toán Col-Peri-Max được trình bày trong mục 4.2.1.

Bài toán Col-Aspect-Ratio: hàm mục tiêu của bài toán là tìm lời giải có tỷ lệ chiều dài/chiều rộng lớn nhất của một hình chữ nhật là nhỏ nhất. Giá trị tốt nhất có thể đạt được là 1, khi tất cả các hình chữ nhật sau khi được phân chia là hình vuông. Tuy nhiên, trong thực tế thì việc chia ruộng trong thực tế hay bài toán SRP thì trường hợp có thể chia tất cả các hình chữ nhật thành hình vuông khó xảy ra. Tôi đã trình bày kết quả thực nghiệm và đánh giá sự tối ưu của lời giải bài toán Col-Aspect-Ratio trong mục 4.2.1.

Kiểm chứng bài toán Col-Peri-Max trong bộ dữ liệu [5]:

1. MN-p01

Array: x_1 = {30, 4, 23, 31, 20, 22, 9, 161, 8, 112}

Sử dụng thuật toán tìm kiếm, phần tử lớn nhất là 161. Kết quả hàm mục tiêu tối ưu có thể đạt được là: $bestObj_1 = 4 * \sqrt{161} = 50.75$

Kết quả thực nghiệm MN-01: 50.83

2. MN-p21

Array: x_2 = {3, 22, 26, 33, 20, 167,30, 132, 4, 4, 27, 6, 13, 186, 2, 13, 26, 2, 4, 107, 123, 127, 5, 5, 1, 16, 3, 165, 3, 97, 125, 3, 2, 89, 35, 137, 34, 79, 152, 62}

Sử dụng thuật toán tìm kiếm, phần tử lớn nhất là 186. Kết quả hàm mục tiêu tối ưu có thể đạt được là: $bestObj_2 = 4 * \sqrt{186} = 54.55$

Kết quả thực nghiệm MN-21: 54.55

3. MU-p01

Array: x_3 = {32, 187, 191, 13, 58, 9, 44, 109, 19, 10}

Sử dụng thuật toán tìm kiếm, phần tử lớn nhất là 191. Kết quả hàm mục tiêu tối ưu có thể đạt được là: $bestObj_4 = 4 * \sqrt{191} = 55.28$

Kết quả thực nghiệm MU-01: 55.28

4. MU-p21

Array: x_4 = {27, 89, 31, 9, 19, 23, 17, 92, 174, 6, 49, 134, 4, 44, 9, 31, 23, 47, 6, 119, 8, 9, 15, 71, 10, 119,44, 2, 33, 38, 106, 32, 38, 11, 50, 6, 30, 8, 49, 6}

Sử dụng thuật toán tìm kiếm, phần tử lớn nhất là 174. Kết quả hàm mục tiêu tối ưu có thể đạt được là: $bestObj_4 = 4 * \sqrt{174} = 52.76$

Kết quả thực nghiệm MU-21: 52.76

5. U-p01

Array: x_5 = {117, 109, 114, 26, 148, 90, 21, 78, 169, 103}

Sử dụng thuật toán tìm kiếm, phần tử lớn nhất là 169. Kết quả hàm mục tiêu tối ưu có thể đạt được là: $bestObj_4 = 4 * \sqrt{169} = 52$

Kết quả thực nghiệm U-01: 52.53

6. U-p21

Array: x_5 = {137, 56, 177, 66, 23, 40, 66, 2, 82, 183, 136, 117, 167, 87, 124, 34, 171, 41, 22, 13, 169, 89, 110, 131, 130, 94, 125, 169, 40, 13, 118, 150, 142, 189, 163, 170, 1, 197, 191, 161}

Sử dụng thuật toán tìm kiếm, phần tử lớn nhất là 169. Kết quả hàm mục tiêu tối ưu có thể đạt được là: $bestObj_4 = 4 * \sqrt{197} = 56.14$

Kết quả thực nghiệm U-21: 56.14