Mục lục

[Mục lục 1](#_Toc406198944)

[**I.** **MỞ ĐẦU** 2](#_Toc406198945)

[**1.** **Giới thiệu và mục tiêu của đề tài** 2](#_Toc406198946)

[**2.** **Đối tượng nghiên cứu** 2](#_Toc406198947)

[**3.** **Phương pháp nghiên cứu** 2](#_Toc406198948)

[**II.** **NỘI DUNG, CẤU TRÚC ĐỀ CƯƠNG** 3](#_Toc406198949)

[**Chương 1 Đặc tả bài toán** 4](#_Toc406198950)

[**Chương 2 Những giải thuật và công trình liên quan** 7](#_Toc406198951)

[**Chương 3 Giải thuật tìm kiếm Tabu** 11](#_Toc406198952)

[**Chương 4 Giải pháp đề xuất** 12](#_Toc406198953)

[**III.** **KẾT LUẬN** 15](#_Toc406198954)

[**IV.** **DỰ KIẾN TIẾN ĐỘ THỰC HIỆN** 16](#_Toc406198955)

[**TÀI LIỆU THAM KHẢO** 17](#_Toc406198956)

1. **MỞ ĐẦU**
2. **Giới thiệu và mục tiêu của đề tài**

Bài toán lập lịch là một lớp bài toán phổ biến trong ngành khoa học máy tính.

Trong giới hạn đề cương này, sẽ tập trung tìm hiểu, giải quyết bài toán lập lịch cho máy đơn, với hai loại công việc có chu kỳ và không có chu kỳ.

Trong đó, công việc không chu kỳ được thực hiện đúng một lần và không có sự ngắt quãng trong quá trình thực hiện, công việc có chu kỳ được thực hiện theo một chu kỳ *T*, tức là trong khoảng thời gian *T* sẽ có đúng một công việc có chu kỳ được thực hiện.

Các công việc có chu kỳ thường có kỳ hạn cứng (hard deadline), trong khi các công việc không chu kỳ thường có kỳ hạn mềm (soft deadline).

Mục tiêu đề tài là ứng dụng kỹ thuât song song hóa vào việc lập lịch, nhằm tăng hiệu năng và chất lương nghiệm đạt được.

1. **Đối tượng nghiên cứu**

* Bài toán lập lịch cho công việc có chu kỳ và không chu kỳ trong môi trường máy đơn.
* Các giải thuật lập lịch cổ điển.
* Các công trình liên quan
* Giải thuật tìm kiếm Tabu để đưa ra giải pháp lập lịch tối ưu đa mục tiêu.

1. **Phương pháp nghiên cứu**

* Tìm hiểu các giải thuật cơ bản có sẵn của bài toán
* Tìm hiểu các công trình liên quan
* Đề xuất giải thuật cải tiến chất lương nghiệm.
* Ứng dụng framework lập trình song song để hiện thực và cải tiến giải pháp.
* Đánh giá hiệu năng của giải thuật dựa trên đã đặt ra và so sánh với công trình có sẵn.

1. **NỘI DUNG, CẤU TRÚC ĐỀ CƯƠNG**

Phần này của đề cương sẽ được trình bày theo bố cục như sau:

**Chương 1** Trình bày mô tả bài toán, các ràng buộc, hàm mục tiêu của bài toán.

**Chương 2** Trình bày các giải thuật cơ bản, nội dung và kết quả làm việc của công trình trước.

**Chương 3** Giới thiệu giải thuật tìm kiếm Tabu.

**Chương 4** Trình bày giải pháp ứng dụng song song hóa và giải thuật tìm kiếm Tabu để giải quyết bài toán

**Chương 1 Đặc tả bài toán**

1. Giới thiệu

Cho tập *N* gồm *n* công việc độc lập không thể bắt đầu lại (non resumeable job).Công việc không thể bắt đầu lại ở đây là một khi công việc bị ngắt giữa chừng phải thực hiện lại từ đầu.

Mỗi công việc có thời gian xử lý (processing time) và kỳ hạn công việc (deadline) riêng. Các công việc trên được gọi là công việc không có chu kỳ (aperiodic task).

Các công việc có chu kỳ được thực hiện theo một chu kỳ *T* , tức công việc được thực hiện nằm trong khoảng chu kỳ *T* nhưng thời điểm thực hiện không cố định. Thời gian thực hiện các công việc có chu kỳ là *p*.

Cho một máy đơn thực hiện các công việc không có chu kỳ và có chu kỳ trên.

1. Các ràng buộc cuả bài toán:

* Tất cả các công việc đều có thể bắt đầu tại thời điểm 0
* Khi một công việc không chu kỳ được thực hiện, nó phải được thực hiện liên tục đến khi kết thúc, không có sự gián đoạn
* Trong mỗi khoảng thời gian *T* phải có đúng 1 công việc có chu kỳ được thực hiện.
* Các công việc không có chu kỳ phải có thời gian thực hiện nhỏ hơn hoặc bằng *2\*(T-p)*

1. Các tiêu chí đo độ hiệu quả của việc xếp lịch:

* *Total* *tardiness*:

Khi thực hiện 1 công việc không có chu kỳ, nếu công việc được hoàn thành trước hoặc ngay kỳ hạn của nó, thì độ trễ của công việc bằng 0, ngược lại nếu công việc được thự hiện hoàn tất sau deadline của công việc thì tardiness(độ trễ) được tính bằng công thức:

Trong đó:

: là độ trễ của từng task trong giải pháp lập lịch

: là thời điểm hoàn thành công việc

: là thời điểm kỳ hạn của công việc

*Total tardiness* được tính bằng tổng độ trễ của các công việc không có chu kỳ.

Trong đó:

n là tổng số các công việc không có chu kỳ

* *Cmax*:

Là thời điểm kết thúc của việc lập lịch.

* *Total earliness*:

Khi công việc không chu kỳ được thực hiện sớm hơn kỳ hạn của nó, thì earliness được tính bằng công thức:

Trong đó:

*n* là tổng số các công việc không có chu kỳ

Ngoài ra còn có nhiều độ đo hiệu quả của giải pháp lập lịch như tối thiểu hóa độ trễ tối đa của các công việc…

1. Cận của các tiêu chí:

* Thời gian thực hiện tối thiểu:

Trường hợp tốt nhất khi giải pháp lập lịch đưa ra một mà trong đó máy không có thời gian rỗi nào trong quá trình chạy.

+(.

* Thời gian thực hiện tối đa :

Trong trường hợp xấu nhất với giải pháp lập lịch mà trong mỗi chu kỳ chỉ thực hiện được đúng một công việc.

.

* Tổng độ trễ tối thiểu:

Trong trường hợp tốt nhất, tổng độ trễ bằng 0 khi tất cả công việc được hoàn thành trước kỳ hạn.

* Tổng độ trễ tối đa:

Trong trường hợp xấu nhất với mỗi công việc, công việc có độ trễ đạt cực đại khi được thực hiện sau cùng trong thứ tự lập lịch:

1. Định nghĩa hàm mục tiêu.

Một giải thuật đề xuất cho bài toán nhằm thỏa mãn một hoặc nhiều các mục tiêu nêu đã nêu trên.Trong giới hạn đề cương này chúng tôi tập trung vào việc tối ưu hóa 2 tiêu chí là *Cmax* và *Totaltardiness*.

Hàm mục tiêu của bài toán được định nghĩa như sau:

Với *Sched* là một giải pháp lập lịch.

**Chương 2 Những giải thuật và công trình liên quan**

Chương này giới thiệu về những giải thuật cơ bản để sắp xếp một danh sách các công việc. Sau đó là trình bày nội dung luận văn của tác giả Hà Thái Vũ [1], những gì đã làm được và những mặt cần được cải tiến.

1. Các giải thuật cơ bản:

* *Longest Processing Time* (*LPT*): Sắp xếp các công việc theo thời gian thực hiện giảm dần.
* *Earliest* *Deadline First* (*EDF*): sắp xếp công việc theo sự tăng dần về kỳ hạn.
* *Shortest Processing Time (SPT)* Sắp xếp các công việc theo thời gian thực hiện tăng dần.
* *Random List* Sắp xếp các công việc theo thứ tự ngẫu nhiên.
* *Minimum Slack Time (MST)* Giải thuật *MST* tính toán mức độ “cấp thiết “của công việc bằng một độ đo slack time. Cho *d[i]* và *t[i]* lần lượt là kỳ hạn và thời gian thực hiện của công việc. Giải thuật *MST* thực hiện sắp xếp các công việc theo thứ tự thỏa mãn:

*d[1]-t[1] <= d[2]-t[2] <= �<= d[n]-t[n]*

Ngoài ra còn có các giải thuật khác như: *Weighted Shortest Processing Time (WSPT)…*

1. Những công trình liên quan:

Trong bài báo cáo trước[1], tác giả Hà Thái Vũ đã thực hiện những công việc như sau:

Tác giả đề xuất áp dụng 2 giải thuật *LPT* và *EDF* để tìm giải pháp lập lịch cho lần lượt 2 mục tiêu là *Cmax* và *Totaltardiness*]. Để tăng tính hiệu quả, 2 heuristic được đề xuất kèm với 2 giải thuật *LPT* và *EDF* nhằm giảm thời gian rỗi của máy.

* ***LPT***: (longest processing time): Sắp xếp các tác vụ theo thời gian thực hiện giảm dần. Nếu 2 tác vụ có thời gian thực hiện ngang nhau, sắp xếp công việc có kỳ hạn nhỏ hơn phía trước.

***HEURISTIC1***: Tại một thời điểm *t* bất kỳ khi công việc được xem xét *Jk* theo thứ tự *LTP* không thể được bắt đầu( sẽ tạo thời gian rỗi), sẽ tìm công việc có thời gian xử lý lâu nhất có thể thực hiện tại thời điểm *t.*

* ***EDF*** (earliest deadline first): sắp xếp công việc theo sự tăng dần về kỳ hạn. Nếu 2 công việc có cùng kỳ hạn, sắp xếp công việc có thời gain xử lý dài hơn ở phía trước.

***HEURISTIC2***:Tại một thời điểm *t* bất kỳ khi công việc được xem xét *Jk* theo thứ tự *EDF* không thể được bắt đầu( sẽ tạo thời gian rỗi), tìm một công việc có kỳ hạn nhỏ nhất có thể bắt đầu tại thời điểm *t* mà không tạo thời gian rỗi.

Như vậy, giải thuật *LPT* và *EDF* đã được sử dụng để tìm các nghiệm tối ưu từng mục tiêu là *Cmax* và *Totaltardiness.*

Tiếp theo, tác giả đề xuất một giải thuật meta-heuristic lai giữa 2 heuristic là *EDF* và *LPT* nhằm đưa ra một giải pháp lập lịch thỏa mãn 2 mục tiêu *Cmax* và *Totaltardiness*. Kết quả của meta-heuristic này có *Cmax* nhỏ hơn kết quả của *EDF* và *Totaltardiness* nhỏ hơn của *LPT*.

Giải pháp được mô tả như sau:

* Đầu tiên sắp xếp các công việc theo thứ tự *LPT* và *EDF*. Như vậy ta sẽ có 2 tập thứ tự khác nhau làvà
* Hai mục tiêu sẽ có độ ưu tiên khác nhau là và . Tại một thời điểm, độ ưu tiên nào cao hơn thì ta sẽ xem xét lập lịch cho tập thứ tự tương ứng. Sau khi công việc được lập lịch thì độ ưu tiên sẽ được tính toán lại.
* Khi một công việc được đưa vào lập lịch thì ta sẽ loại bỏ nó ra khỏi tập và .
* Cách tính toán độ ưu tiên dựa vào 2 tham số α và β. Công thứ tính là: *Hệ số ưu tiên= Độ ưu tiên \*(Số công việc chưa lập lịch- Số công việc đã lập lịch của tập thứ tự).*

Ví dụ:

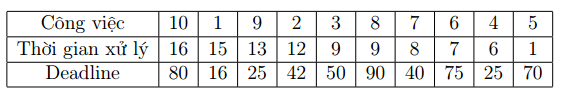
*N*={1,2,3,4,5,6,7,8,9,10}.

*P*={15,12,9,6,1,7,8,9,13,16}

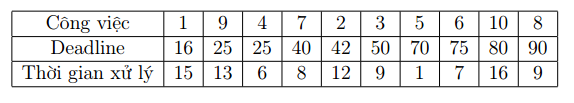
*D*={16,42,50,25,70,75,40,90,25,80}

*T*=10, *p*=1

Tập công việc *LPT* với thời gian thục hiện công việc giảm dần:



Tập công việc *EDF* với kỳ hạn công việc tăng dần:



Với hệ số ưu tiên *α*=0,6 và *β*=0,4.

Khi chưa có công việc,

Vì nên công việc đầu tiên được lập lịch theo

Vậy công việc đầu tiên được lập lịch là công việc 10, với thời gian xử lý 16, deadline 80.

Tiếp theo, độ ưu tiên được cập nhật với

Vì nên công việc thứ 2 được lập lịch theo

Vì công việc tiếp theo trong danh sách *LPT* là 1, không thể bắt đầu ngay sau khi kết thúc công việc 10, nên ta áp dụng *HEURISTIC1*, tìm công việc có thời gian xử ý lâu nhất có thể bắt đầu ở thời điểm kết thúc công việc 10. Vậy, công việc thứ 2 được lập lịch là công việc 3, với thời gian xử lý 9, kỳ hạn 50.

Tiếp tục, độ ưu tiên được cập nhật với

Vì nên công việc thứ 3 được lập lịch theo

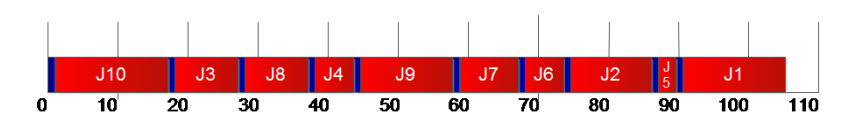
Tương tự ta sử dụng *HEURISTIC1*, và công việc thứ 3 được lập lịch là công việc 8, thời gian xử lý 9, kỳ hạn 90

Sau đó độ ưu tiên được cập nhật:

Vì nên công việc thứ 3 được lập lịch theo

Sử dụng *HEURISTIC2*, công việc tiếp theo được lập lịch là côngviệc 4, với thời gian xử lý 6, kỳ hạn 25.

Tiếp tục như vậy, ta được giải pháp lập lịch như sau:



Với *Cmax*=107, *Total tardiness*=235

Nhận xét: Giải thuật meta-heuristic trên đã cho một kết quả với giá trị *Cmax* nhỏ hơn kết quả của *EDF* và *Totaltardiness* nhỏ hơn của *LPT*.

Tuy nhiên công việc trên vẫn còn những chỗ có thể cải tiến như sau:

* Chỉ tìm ra 1 lời giải duy nhất cho mỗi đầu vào của bài toán. Trong khi với mỗi đầu vào của bài toán, có thể có những nghiệm khả thi khác nhau.
* Chất lương nghiệm của giải thuật có thể được cải tiến tốt hơn.

Trong chương 4 sẽ trình bày đề xuất để cải tiến chất lượng nghiệm đạt được.

## Chương 3 Giải thuật tìm kiếm Tabu

**Giới thiệu**

*Giải thuật Tabu* được đề xuất bởi Glover những năm cuối thập kỷ 1980. Từ “*Tabu*” bắt nguồn từ *Tongan*, một ngôn ngữ ở *Polynesia*, được những thổ dân ở *Tonga* dung để chỉ những thứ cấm kỵ, không thể chạm đến. [5]

*Tìm kiếm Tabu (TS)* có thể được sử dụng để giải quyết bài toán tối ưu tổ hợp (những vấn đề mà kết quả mong muốn là một trật tự và sự chọn lọc tối ưu).

Những ứng dụng hiện tại cuả tìm kiếm Tabu trong các bài toán lên kế hoạch, viễn thông, VLSI, phân tích tài chính, lập lịch, vận tải….

**Ý tưởng của tìm kiếm Tabu.**

Giải thuật sử dụng một thủ tục tìm kiếm lân cận, thực hiện các *bước* *dịch chuyển (Tabu Move)* lặp đi lặp lại từ một giải pháp tiềm năng *x* tới một giải pháp *x’* tốt hơn, với *x’* nằm trong các lân cận của *x*.

Trong quá trình tìm kiếm, *giải thuật Tabu* lưu những vết đi chuyển mới nhất vào một dữ liệu gọi là *Tabu List*. Những di chuyển đã được lưu này sẽ không được thực hiện lại trong một khoảng thời hạn nhất định, gọi là *nhiệm kỳ Tabu (Tabu tenure).*

Trong khoảng *nhiệm kỳ Tabu* của *bước dịch chuyển*, *bước dịch chuyển* đó ở trong *trạng thái Tabu* bị cấm thực hiện. Ngược lại thì *bước dịch chuyển* được phép thực hiện.

Những cải tiến trong giải thuật *tìm kiếm Tabu*:

*Kỳ vọng (**Aspiration Level Conditions)* Việc lưu vết các bước di chuyển trong *Tabu List* sẽ ngăn cấm việc lặp lại bước di chuyển này trong một vài bước tìm tiếp theo.

Việc này giúp giải thuật tìm kiếm không lặp lại những bước đi đã cũ, nhằm ngăn việc tìm kiếp lặp đi lặp lại những giải pháp cũ, nhưng nó cũng ngăn cản việc tìm kiếm đến những lời giải tốt hơn.

Vì vậy, có một cơ chế được đưa ra gọi là tiêu chí khát vọng (*aspiration criterion).* Nó cho phép thay đổi trạng thái của phép dịch chuyển Tabu Move nếu: Tại một lời giải *s*, một *bước dịch chuyển* *m* làm tăng hiệu quả lời giải một cách rõ rệt, thì *bước dịch chuyển* *m* có thể được chấp nhận bất kể *trạng thái Tabu* của *m*.

**Chương 4 Giải pháp đề xuất**

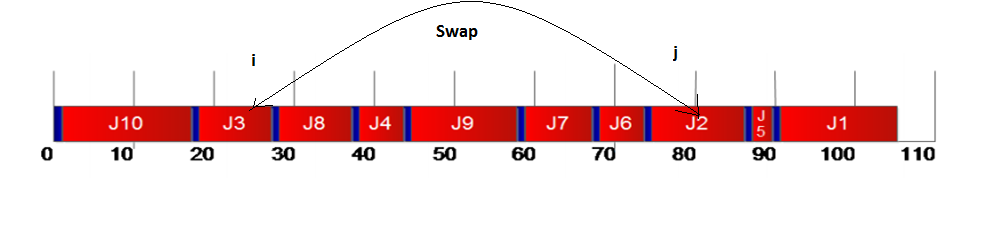
Chương này trình bày giải pháp hiện thực hóa *giải thuật Tabu* và ứng dụng tính toán song song để giải quyết bài toán.

1. Ý tưởng đề xuất

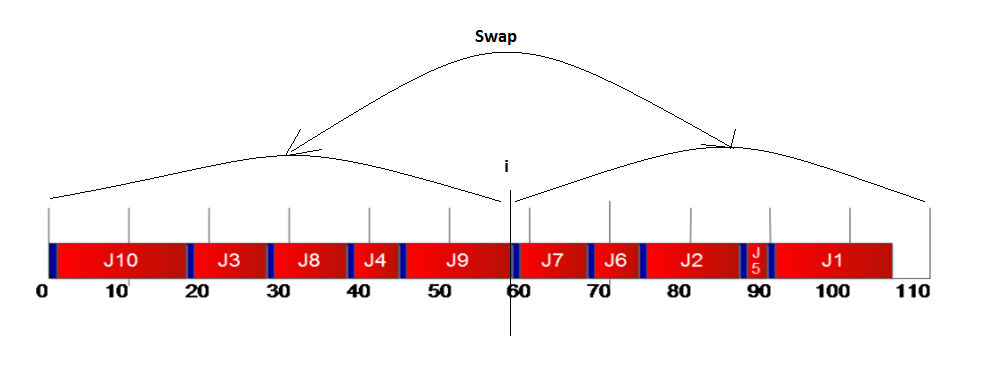
Ý tưởng của giải pháp là dựa trên những kết quả của các heuristic *LPT* và *EDF*, meta-heuristic, thực hiện tìm kiếm *Tabu* với các toán tử *TabuMove* khác nhau.

Các toán tử *TabuMove* được định nghĩa:

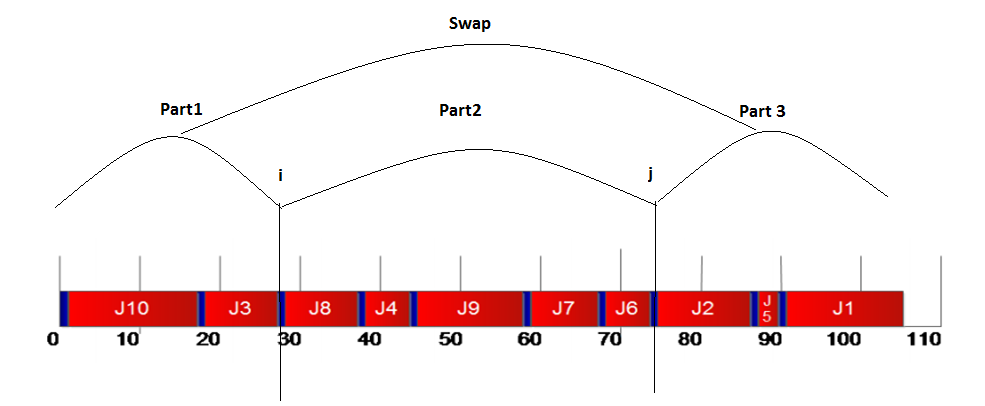
* *TabuMove1* (*Sched, pos1, pos2*): Thực hiện một phép hoán đổi vị trí giữa 2 phần tử *pos1* và *pos2* trong kết quả lập lịch *Sched.*



* *TabuMove2* (*Sched, pos*): Chia kết quả lập lịch *Sched* ra thành 2 phần: 0..*.pos* và *pos*...*n*. Thực hiện phép hoán đổi 2 phần trên.



* *TabuMove3 (Sched, pos1, pos2)*: Chia kết quả lập lịch thành 3 phần. Thực hiện phép hoán đổi vị trí của 2 phần 0…*pos1* và *pos2*…*n*.



. Ngược lại khi *TabuList* [*n, n*] có giá trị lớn 0 thì *phép di chuyển* tương ứng với 2 phần tử *i* và *j* bị ngăn cấm.

*Nhiệm kỳ Tabu (TabuTenure)* được định nghĩa là một hằng số nguyên dương *T,* biểu diễn số bước lặp cần thiết để một *phép dịch chuyển* thay đổi *trạng thái Tabu.*

Các bước thực hiện chính:

* Thực hiện các giải thuật *EDF* và *LPT*.
* Thực hiện meta-heuristic lai giữa *EDF* và *LPT* với *α* lần lượt là 0 0.1 0.2 .. 1.0 và tương ứng .
* Định nghĩa kiểu dữ liệu mảng *BestSolution* để lưu các kết quả lập lịch tốt nhất trong quá trình tìm kiếm. Khởi tạo *BestSolution* với 13 phần tử là kết quả của các giải thuật *LPT*, *EDF*, meta-heuristic.
* Thực hiện tìm kiếm song song từng tiến trình với mỗi tiến trình thực hiện tìm kiếm Tabu với phép dịch chuyển *TabuMove* khác nhau.
* Các tiến trình thực hiện việc đồng bộ trao đổi kết quả tốt nhất qua vùng nhớ chung *BestSolution*[3].

1. Nội dung hiện thực và kết quả dự kiến

Đề tài sẽ được hiện thực trên một framework hỗ trợ lập trình song song. Kết quả của chương trình thực hiện là một tập hợp các nghiệm thỏa mãn 2 tiêu chí lập lịch.

Giải thuật sẽ đưa ra một tập các nghiệm tương ứng với các phương án lập lịch khác nhau. Một nghiệm *R(Sched,Cmax,Totaltardiness)* tìm được của giải thuật gồm có:

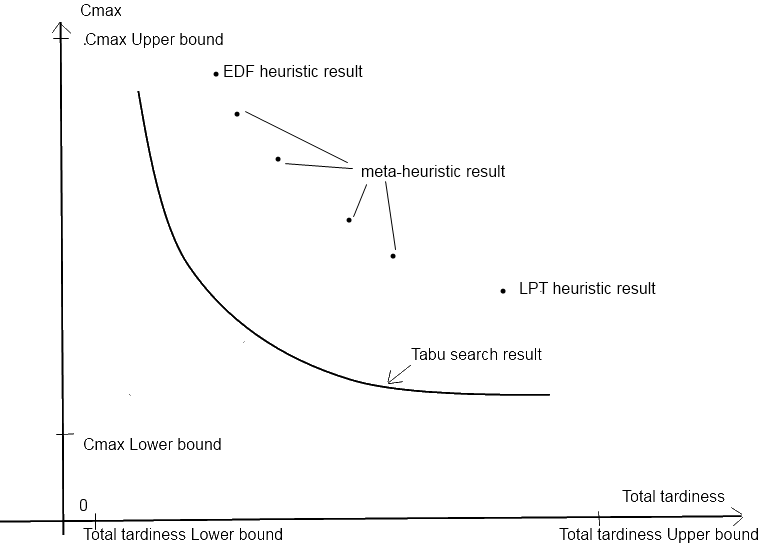
* Một phương án lập lịch *Sched* là thứ tự sắp xếp các công việc cho máy.
* Các tham số hiệu năng *Cmax* và *Total tardiness.*

Trong tập kết quả, với hai nghiệm *R1 (Sched1, Cmax1, Totaltardiness1)* và *R2* *(Sched2, Cmax2, Totaltardiness2)* bất kỳ đều thỏa mãn ràng buộc sau đây:

* Nếu và thì nghiệm *R1* sẽ bị loại bỏ, vì nghiệm *R2* ưu việt hơn *R1* ở cả hai mục tiêu.

Vậy, hai nghiệm *R1 (Sched1, Cmax1, Totaltardiness1),* *R2 (Sched2, Cmax2, Totaltardiness2)* bất kỳ đều thỏa mãn:

Dựa vào tính chất ràng buộc trên, chúng tôi dự đoán kết quả của giải thuật sẽ được biểu diễn dưới dạng đồ thị hyperpol như sau:



1. **KẾT LUẬN**

Đề tài sẽ đi sâu vào tìm hiểu, đặc tả vấn đề lập lịch trong môi truờng máy đơn.

Bài toán sẽ đuợc biểu diễn bằng các ràng buộc toán học cụ thể.

Dựa vào đó, chúng tôi xây dựng mô hình toán học cho bài toán. Tiếp đến chúng tôi xác định độ phức tạp của giải thuật và đề xuất giải thuật heuristic cụ thể, ứng dụng song song hóa để tăng hiệu năng của giải thuật.

Hiệu năng của giải thuật sẽ được đánh giá dựa trên thời gian chạy và chất lượng nghiệm đạt được.

Xa hơn nữa chúng tôi mong muốn mở rộng bài toán của mình. Giải thuật đã đề xuất có thể đuợc cải tiến hơn nữavới các kỹ thuật tinh chỉnh.

1. **DỰ KIẾN TIẾN ĐỘ THỰC HIỆN**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | CÔNG VIỆC | TUẦN BẮT ĐẦU | TUẦN KẾT THÚC | TỔNG THỜI GIAN |
| 1 | Đặc tả bài toán, xác định tập mẫu đầu vào cho bài toán | T1 | T2 | 2 |
| 2 | Mô hình hóa bài toán, xác định độ phức tạp | T3 | T6 | 4 |
| 3 | Hiện thực giải thuật, áp dụng các kỹ thuật song song hóa đã đề xuất. | T7 | T12 | 6 |
| 4 | Đánh giá hiệu năng giải thuật | T13 | T16 | 4 |
| 5 | Tinh chỉnh các tham số của giải thuật. | T17 | T20 | 4 |
| 6 | Viết luận văn | T21 | T24 | 4 |
| TỔNG CỘNG | | | | 24 |

**TÀI LIỆU THAM KHẢO**

[1] Hà Thái Vũ, “*Lập lịch công việc có chu kỳ và không có chu kỳ trong môi trường máy đơn để tối ưu hóa đa mục tiêu thời gian hoàn thành và độ trễ*”, Luân văn Thạc sĩ KHMT, Đại học BK TP.HCM, 2013.

[2] Min Ji, Yong He, T. C. E. Cheng*“*Single-machine Scheduling with Periodic Maintenance to Minimize Makespan”*,* *Computers & Operations Research,*Volume 34, Issue 6, Pages 1764–1770, June 2007.

[3] JianyongJin, TeodorGabrielCrainic, ArneLøkketangen,*”* A cooperative parallel metaheuristic for the capacitated vehicle routing problem”. *Computers & Operations Research,* Volume 44, Pages 33–41, April 2014.

[4] Manuel Laguna, Rafael Marti, Vicente Campos,” Intensification and Diversification with Elite Tabu Search Solutions for the Linear Ordering Problem” *Computers & Operations Research,* Volume 26, Issue 12, Pages 1217–1230,Octorber 1999.

[5] Fred Glover, “*Tabu Search: A Tutorial”,* Center for Applied Artificial Intelligent, University of Colordo, 1990.