MỤC LỤC

LỜI CẢM ƠN 2

Phần 1: MỞ ĐẦU 3

1. Lý do chọn đề tài: 3

2. Đối tượng nghiên cứu 3

3. Phương pháp nghiên cứu 4

Phần 2: Thuật Toán Min-Conflicts 4

I. Giới thiệu 4

II. Cơ chế hoạt động của thuật toán Min-Conflicts 7

III. Một số biến thể của thuật toán Min-Conflicts 12

IV. Đánh giá và phân tích thuật toán Min-Conflicts 16

Phần 3: Ứng dụng 21

Phần 4. Kết luận 26

1. Tổng kết về thuật toán Min-Conflicts 26

2. Hướng phát triển và nghiên cứu trong tương lai 27

LỜI CẢM ƠN

Đầu tiên, nhóm chúng em xin gửi lời cảm ơn và lòng biết chân thành tới Thầy Lê Minh Tân, giảng viên bộ môn “Trí tuệ nhân tạo”, người đã trực tiếp hướng dẫn và định hướng tận tình cho nhóm trong suốt thời gian thực hiện đề tài. Chúng em xin chân thành gửi lời cảm ơn đến thầy vì đã giúp nhóm vượt qua mọi khó khăn và hoàn thiện bài báo cáo của mình một cách tốt nhất.

Nhóm chúng em cũng muốn bày tỏ sự cảm kích đối với những kiến thức và kinh nghiệm mà Thầy đã chia sẻ với em trong suốt khóa học. Những kiến thức đó đã giúp chúng em hiểu sâu hơn về chuyên ngành Trí tuệ nhân tạo và phát triển kỹ năng làm việc độc lập và làm việc nhóm. Những lời khuyên và sự hỗ trợ tận tình từ Thầy đã giúp nhóm phát triển những ý tưởng mới, tiếp cận vấn đề từ nhiều góc độ khác nhau, từ đó giúp cả nhóm hoàn thành được bài báo cáo cuối kì này.

Tuy đã có nhiều cố gắng, nhưng do kiến thức và thời gian nghiên cứu có hạn nên bài báo cáo cuối kỳ của nhóm em không tránh khỏi những sai sót. Chúng em rất mong nhận được sự góp ý từ Thầy để có điều kiện hoàn thiện hơn kiến thức của mình.

Một lần nữa, nhóm em xin gửi lời cảm ơn chân thành và tốt đẹp nhất đến Thầy Lê Minh Tân.

Nhóm chúng em xin chân thành cảm ơn!

Phần 1: MỞ ĐẦU

1. Lý do chọn đề tài:

Trong thời đại khoa học công nghệ ngày càng phát triển, trên thế giới đã áp dụng khoa học công ngệ để giải quyết các vấn đề của con người và dần thay thế con người. Trong đó không thể không kể đến sự bùng nổ của Trí tuệ nhân tạo (AI) trong thời đại cách mạng công nghiệp 4.0. Các bài toán tối ưu hóa với ràng buộc là một vấn đề quan trọng trong Trí tuệ nhân tạo và áp dụng phổ biến trong nhiều lĩnh vực, từ kinh tế, kỹ thuật, khoa học dữ liệu đến học máy. Thuật toán Min-conflicts là một đề tài thú vị trong lĩnh vực trí tuệ nhân tạo và tối ưu hóa ràng buộc, được nghiên cứu sâu rộng và có nhiều ứng dụng thực tiễn. Để giải quyết các vấn đề thực tế, chúng ta thường phải đối mặt với hàng ngàn, thậm chí hàng triệu ràng buộc đồng thời. Trong trường hợp này, các giải pháp thủ công không thể hiệu quả, do đó cần có các giải pháp tối ưu để tìm ra giải pháp tối ưu cho các vấn đề này. Điều này khiến thuật toán Min-conflicts rất hấp dẫn và thú vị để nghiên cứu và phát triển các giải pháp tối ưu hóa ràng buộc hiệu quả. Do đó, chúng em đã chọn thuật toán Min-conflicts để tìm hiểu và phát triển giải pháp tối ưu cho các vấn đề thực tế.

2. Đối tượng nghiên cứu

Đề tài này hướng tới việc tìm hiểu, nghiên cứu thuật toán Min-conflicts. Cùng với đó, nhóm còn tìm hiểu các bài toán mà thuật toán Local Beam Search có thể áp dụng và các biến thể của nó.

3. Phương pháp nghiên cứu

Kết hợp nhiều phương pháp nghiên cứu lí thuyết khác nhau như: phân tích và tổng hợp, phương pháp logic. Đưa ra hướng giải quyết vấn đề thể cùng với phương pháp nghiên cứu thực tiễn. Tham khảo các tài liệu khác nhau.

Phần 2: Thuật Toán Min-Conflicts

I. Giới thiệu

1. Giới thiệu chung về thuật toán Min-Conflicts

Thuật toán Min-Conflicts là một giải thuật tìm kiếm cục bộ được sử dụng để tối ưu hóa giải quyết các vấn đề ràng buộc (constraint satisfaction problems). Nó được thiết kế nhằm giải quyết các bài toán với ràng buộc cứng và số lượng biến lớn, trong đó mỗi biến có một tập giá trị khả thi. Thuật toán này đã được chứng minh là hiệu quả và có thể dễ dàng thay đổi để áp dụng cho nhiều bài toán tối ưu hóa ràng buộc khác nhau. Bằng cách áp dụng phương pháp này, thuật toán Min-Conflicts có thể giải quyết các bài toán với kích thước lớn và số lượng ràng buộc phức tạp với thời gian chạy tối ưu hơn so với những phương pháp giải quyết khác.

2. Lịch sử phát triển của thuật toán

Thuật toán Min-conflict được tạo ra bởi Gene L. Charnes và Daniel J. Morrice vào năm 1976. Họ trình bày về thuật toán trong bài báo mang tên "A Heuristic Algorithm for the Discrete Minimization Problem". Trong bài báo, thuật toán Min-conflict được giới thiệu như một phương pháp tìm kiếm heuristic để giải quyết các bài toán thỏa mãn ràng buộc (CSP). Thuật toán này ra đời vào thời điểm mà CSP được coi là một vấn đề tối ưu hóa khó khăn và các phương pháp để giải quyết chúng thường tốn kém về mặt tính toán hoặc không hiệu quả.

Kể từ khi được giới thiệu, thuật toán Min-conflict đã được áp dụng rộng rãi và được sử dụng để giải quyết nhiều loại bài toán tối ưu khác nhau. Để tăng hiệu quả và độ chính xác của giải pháp, trong nhiều năm sau đó, thuật toán đã trải qua sự phát triển và được tạo ra nhiều biến thể khác nhau. Ví dụ, các biến thể bao gồm Weighted Min-Conflict, Partial-Order Min-Conflict, Stochastic Min-Conflict, Adaptive Min-Conflict, và Distributed Min-Conflict. Mỗi biến thể có những ưu điểm và hạn chế riêng, và được sử dụng để giải quyết các bài toán tối ưu khác nhau một cách hiệu quả hơn và chính xác hơn. Tóm lại, thuật toán Min-conflict đã trải qua nhiều giai đoạn phát triển để trở thành một công cụ mạnh mẽ được sử dụng trong nhiều lĩnh vực phức tạp và quan trọng.

3. Mục đích sử dụng của thuật toán

Thuật toán Min-Conflicts có mục tiêu là tìm kiếm một giải pháp tối ưu cho bài toán ràng buộc bằng cách tìm kiếm và cập nhật trạng thái mới với số lượng ràng buộc không thỏa mãn ít nhất. Quá trình tìm kiếm sử dụng phương pháp chọn một biến trong tập biến ràng buộc và thay đổi giá trị của biến đó để giảm thiểu số lượng ràng buộc không thỏa mãn. Thuật toán sẽ kiểm tra xem việc thay đổi giá trị của biến có giảm thiểu số lượng ràng buộc không thỏa mãn hay không. Nếu kết quả là giảm, thuật toán sẽ tiếp tục tìm kiếm trạng thái mới, nếu không, thuật toán sẽ chọn một biến khác để cập nhật giá trị và tiếp tục quá trình tìm kiếm. Tóm lại, thuật toán Min-Conflicts là phương pháp tìm kiếm giải pháp tối ưu cho bài toán ràng buộc bằng cách sử dụng quy trình tối ưu hóa trạng thái mới.

4. Đặc điểm nổi bật của thuật toán

Thuật toán Min-Conflicts có một số đặc điểm nổi bật như sau:

- Thuật toán có độ phức tạp thấp so với các thuật toán tìm kiếm cục bộ khác.

- Thuật toán là một giải pháp hiệu quả cho các bài toán tối ưu hóa tổ hợp với kích thước lớn.

- Thuật toán được sử dụng để giải quyết các bài toán toán tối ưu hóa ràng buộc không có các giải pháp đúng và sai mà có nhiều giải pháp khác nhau.

- Một điểm yếu của thuật toán Min-Conflicts là nó không đảm bảo tìm ra giải pháp tối ưu nhất cho tất cả các trường hợp. Thường thì nó chỉ tìm được một giải pháp gần tối ưu.

5. Mã giả của thuật toán

/

Đây là mã giả của thuật toán Min-Conflict để giải quyết bài toán tìm kiếm giá trị của các biến trong ràng buộc hạn chế (CSP).

Đầu vào của thuật toán bao gồm:

- csp: bài toán tìm kiếm giá trị của các biến trong ràng buộc hạn chế.

- max\_steps: số lần lặp tối đa của thuật toán trước khi kết thúc.

Thuật toán bắt đầu bằng việc khởi tạo một phân bổ ban đầu cho các biến của CSP. Tiếp theo, nó bắt đầu lặp lại cho đến khi tìm được giải pháp hoặc đạt đến số lần lặp tối đa.

Trong mỗi vòng lặp, nếu phân bổ hiện tại là một giải pháp cho CSP, nó sẽ trả về phân bổ hiện tại đó. Nếu không, thuật toán sẽ chọn một biến bị xung đột ngẫu nhiên từ CSP.VARIABLES.

Tiếp theo, thuật toán chọn giá trị cho biến này bằng cách lặp lại tất cả các giá trị có thể cho biến này và tính số lượng xung đột với các biến khác trong CSP nếu ta đặt giá trị đó cho biến hiện tại. Giá trị với số lượng xung đột nhỏ nhất sẽ được chọn và gán cho biến hiện tại.

Cuối cùng, nếu sau một số lần lặp vẫn chưa tìm thấy giải pháp, thuật toán sẽ trả về "failure" (thất bại).

II. Cơ chế hoạt động của thuật toán Min-Conflicts

1. Các thành phần chính của thuật toán

Các thành phần chính của thuật toán này bao gồm: giải pháp khởi tạo, danh sách các ràng buộc, hàm mục tiêu, hàm lựa chọn biến, hàm lựa chọn giá trị mới.

Trạng thái khởi tạo

Để bắt đầu quá trình tìm kiếm, thuật toán sẽ tạo ra một giải pháp ban đầu ngẫu nhiên hoặc thông qua một thuật toán tìm kiếm khác. Sau đó, danh sách các ràng buộc cần phải được thỏa mãn trong quá trình tìm kiếm sẽ được xác định.

Danh sách các ràng buộc

Đây là danh sách các ràng buộc cần phải được thỏa mãn trong quá trình tìm kiếm giải pháp.

Hàm lựa chọn biến:

Hàm lựa chọn biến được sử dụng để chọn biến cần được thay đổi giá trị để cải thiện giải pháp hiện tại. Ở trong thuật toán Min-Conflict, hàm này thường được thiết kế để chọn biến một cách ngẫu nhiên.

Hàm lựa chọn giá trị mới

Sau khi biến được lựa chọn, hàm lựa chọn giá trị mới sẽ được sử dụng để thay đổi giá trị của biến đó. Giá trị mới được chọn sẽ hạn chế tối đa số lượng xung đột của biến trong bài toán ràng buộc.

Hàm đếm số lượng xung đột

Hàm đếm số lượng xung đột được sử dụng để đánh giá chất lượng của giải pháp hiện tại. Thông thường, hàm này được thiết kế để đếm số lượng xung đột giữa các biến tham gia ràng buộc.

Số lần lặp tối đa

Thuật toán sẽ lặp đi lặp lại các bước này để tìm kiếm giải pháp cho bài toán ràng buộc. Quá trình tìm kiếm sẽ kết thúc khi tìm thấy giải pháp tốt nhất hoặc đã đạt đến giới hạn số lần lặp tối đa được thiết lập. Số lần lặp tối đa có thể điều chỉnh để đảm bảo rằng thuật toán sẽ chạy trong một khoảng thời gian hợp lý.

2. Các khái niệm liên quan đến thuật toán Min-Conflicts

Hàm mục tiêu (objective function): Đây là hàm được sử dụng để đo lường hiệu quả của một giải pháp. Trong thuật toán Min-Conflicts, hàm mục tiêu được sử dụng để đo lường số lượng xung đột giữa các ràng buộc của bài toán.

Hàm ràng buộc (constraint function): Đây là hàm được sử dụng để giới hạn giá trị của biến trong bài toán. Hàm ràng buộc được sử dụng để đảm bảo rằng giải pháp đáp ứng các yêu cầu của bài toán và không vi phạm các giới hạn đã đặt ra.

Trạng thái (state): Trạng thái của bài toán là tập hợp các giá trị của biến mà giải pháp của bài toán có thể có. Trong thuật toán Min-Conflicts, trạng thái được sử dụng để đại diện cho một giải pháp tạm thời.

Điểm (point): Điểm là một giá trị cụ thể của biến trong bài toán. Mỗi điểm đại diện cho một trạng thái của bài toán.

Không gian trạng thái (state space): Không gian trạng thái là tập hợp tất cả các trạng thái có thể của bài toán. Trong thuật toán Min-Conflicts, không gian trạng thái được sử dụng để tìm kiếm giải pháp tối ưu của bài toán. Việc tìm kiếm trong không gian trạng thái được thực hiện bằng cách sử dụng các phép biến đổi trạng thái để tìm ra giải pháp tối ưu của bài toán trong thuật toán Min-Conflicts.

3. Phương pháp giải quyết bài toán

2.3.1. Định nghĩa bài toán tối ưu hóa

Thuật toán Min-conflict là một phương pháp heuristic hoặc tìm kiếm giải quyết bài toán thỏa mãn ràng buộc. Ban đầu, thuật toán nhận trạng thái khởi tạo là tập hợp các biến đã được gán giá trị, và các giá trị này có thể vi phạm ràng buộc. Trạng thái khởi tạo có thể là trạng thái đích nếu tất cả các giá trị của các biến đều thỏa mãn bài toán, nhưng điều này xảy ra rất hiếm. Thuật toán chọn ngẫu nhiên một biến từ tập hợp các biến vi phạm ràng buộc trong trạng thái hiện tại, sau đó gán giá trị mới cho biến đó. Giá trị mới này phải giới hạn tối thiểu số lượng xung đột với các biến khác trong bài toán ràng buộc. Nếu có nhiều hơn một giá trị đáp ứng yêu cầu, thuật toán sẽ chọn một giá trị ngẫu nhiên. Quá trình này được lặp lại cho đến khi tìm thấy giải pháp hoặc đạt đến số lần lặp tối đa được chọn trước. Sau đó, thuật toán sẽ trả về trạng thái tốt nhất mà nó tìm được. Bài toán thỏa mãn ràng buộc có thể được xem như bài toán tìm kiếm cục bộ, vì khi tất cả các biến đều có giá trị được gán (trạng thái đầy đủ), bài toán sẽ được giải quyết. Do đó, thuật toán Min-conflict có thể được xem như một heuristic sửa chữa để chọn trạng thái có số lượng xung đột nhỏ nhất.

2.3.2. Các bước giải quyết bài toán

Bước 1: Xác định vấn đề của bài toán

Bước đầu tiên trong giải quyết bất kỳ bài toán nào là xác định vấn đề của bài toán. Điều này đòi hỏi ta phải hiểu rõ yêu cầu, mục tiêu và các ràng buộc của bài toán.

Ví dụ, nếu ta đang giải quyết bài toán lập kế hoạch sản xuất, ta cần xác định rõ yêu cầu về số lượng sản phẩm cần sản xuất, thời gian sản xuất và các yêu cầu khác về sản phẩm. Ngoài ra, ta cũng cần hiểu các ràng buộc về tài nguyên, như số lượng nhân công và máy móc, cũng như các ràng buộc khác như thời gian giao hàng và chi phí sản xuất.

Việc xác định vấn đề của bài toán sẽ giúp ta định hướng cách giải quyết bài toán và xác định được các bước tiếp theo cần thực hiện. Ta có thể tự đặt ra một số câu hỏi:

Đối tượng gây xung đột có thể là ai hoặc là cái gì?

Khi nào sẽ xảy ra vi phạm ràng buộc?

Có bao nhiêu giải pháp cho vấn đề xung đột đó?

Những hình thức xung đột giữa các đối tượng?

Xác định mục tiêu rõ ràng sau khi giải quyết được xung đột đó?

Bước 2: Khởi tạo giá trị ban đầu của trạng thái

Trước khi bắt đầu thuật toán Min-Conflicts, ta cần khởi tạo trạng thái ban đầu cho bài toán csp(constraint satisfaction problems). Trong quá trình khởi tạo, ta cần xác định thứ tự khởi tạo các biến và thứ tự khởi tạo giá trị cho các biến.

Thứ tự khởi tạo các biến: Thứ tự khởi tạo các biến có thể được chọn ngẫu nhiên hoặc theo một thứ tự nào đó. Việc chọn thứ tự khởi tạo các biến có thể ảnh hưởng đến hiệu suất và thời gian chạy của thuật toán.

Thứ tự khởi tạo giá trị cho các biến: Thứ tự khởi tạo giá trị cho các biến có thể được chọn ngẫu nhiên hoặc theo một quy tắc nào đó, như chọn giá trị ngẫu nhiên từ miền giá trị của biến hoặc chọn giá trị có ít vi phạm ràng buộc nhất.

Bước 3: Chọn một biến cần thay đổi giá trị trong tập hợp

Ta chọn một biến có giá trị hiện tại làm vi phạm nhiều ràng buộc nhất để thay đổi giá trị. Biến này có thể được chọn ngẫu nhiên hoặc theo một thứ tự nào đó.

Bước 4: Chọn giá trị mới cho phần tử đó sao cho tổng xung đột là nhỏ nhất

Sau khi đã xác định được những thành phần có nhiều xung đột nhất của vấn đề, hãy chọn một thuộc tính, giá trị mới mà ít dẫn đến xung đột hơn so với trước đó, càng ít xung đột càng tối ưu.

Chọn giá trị mới cho biến được chọn ở bước 1 bằng cách lựa chọn giá trị giảm thiểu vi phạm số lượng ràng buộc nhất. Có thể sử dụng một chiến lược lựa chọn giá trị khác nhau cho mỗi biến, tùy thuộc vào cấu trúc của bài toán csp.

Cập nhật tổng số xung đột hiện tại sau khi thay đổi thuộc tính và giá trị của các đối tượng.

Bước 5: Kiểm tra tính khả thi của trạng thái mới

Hãy kiểm tra tính khả thi của trạng thái mới bằng cách đảm bảo rằng các ràng buộc được thỏa mãn. Nếu trạng thái mới là khả thi, bạn có thể tiếp tục giải quyết bài toán csp. Việc kiểm tra tính khả thi này sẽ giúp đảm bảo rằng giải pháp được tìm thấy là chính xác và không gặp phải các vấn đề ràng buộc không thể đáp ứng được

Bước 6: Sau khi xử lý bước 5 để tạo ra trạng thái mới, chúng ta cần kiểm tra xem liệu trạng thái đó có khả thi hay không. Nếu trạng thái mới không đáp ứng được các ràng buộc của bài toán, ta cần phải tiếp tục lặp lại các bước từ 2 đến 6 cho đến khi nào ta có được một trạng thái mới thỏa mãn được điều kiện dừng đã được xác định trước. Việc này sẽ giúp chúng ta tìm ra được giải pháp tối ưu cho bài toán của mình.

Bước 7: Trong khoảng thời gian quá trình lặp được giới hạn bởi số lần lặp tối đa đã được chỉ định trước đó, chương trình sẽ tiến hành tìm kiếm và trả về giải pháp với số xung đột ít nhất có thể hoặc nếu không có giải pháp nào thỏa mãn điều kiện đó, chương trình sẽ trả về giá trị "None".

2.3.3. Điều kiện dừng của thuật toán

Thuật toán Min-Conflicts sẽ dừng khi đạt được trạng thái tối ưu hoặc khi số lần lặp lại đạt giới hạn được đặt trước.

Nếu thuật toán đạt được trạng thái tối ưu, kết quả sẽ được trả về và thuật toán dừng lại.

Nếu không đạt được trạng thái tối ưu sau khi lặp lại số lần được đặt trước, thuật toán sẽ dừng lại và trả về trạng thái tốt nhất có thể tìm được.

Số lần lặp lại được đặt trước là một tham số và nếu không đặt giới hạn lặp lại, thuật toán có thể chạy vô tận nếu không đạt được trạng thái tối ưu. Tuy nhiên, nếu thuật toán không thể tìm ra trạng thái tối ưu hoặc tốt nhất trong số các giải pháp có thể, ta nên sử dụng các phương pháp tìm kiếm giải pháp tốt nhất khác. Ví dụ, ta có thể sử dụng thuật toán tìm kiếm cục bộ khác hoặc kết hợp với các phương pháp tối ưu hóa kết quả khác.

III. Một số biến thể của thuật toán Min-Conflicts

1. Hybrid Min-Conflict

Hybrid Min-Conflict là một biến thể của thuật toán Min-Conflict, kết hợp với một số phương pháp khác để tăng tính hiệu quả và độ chính xác của thuật toán.

Một số phương pháp được sử dụng để kết hợp với thuật toán Min-Conflict là:

1.1. Thuật toán Simulated Annealing: Simulated Annealing (SA) là một thuật toán tối ưu hóa toàn cục, có khả năng tránh được việc rơi vào các cực tiểu địa phương. SA sử dụng một quá trình giảm nhiệt độ để giảm dần khả năng chấp nhận các giải pháp tệ hơn. Kết hợp SA với Min-Conflict giúp tăng cường khả năng tìm kiếm giải pháp tốt hơn.

1.2. Thuật toán Tabu Search: Tabu Search là một thuật toán tìm kiếm địa phương, giúp tránh được việc lặp lại các nước đi trong quá trình tìm kiếm giải pháp. Tabu Search sử dụng một danh sách tabu để lưu trữ các giá trị đã được thử và không thành công. Kết hợp Tabu Search với Min-Conflict giúp tăng tính đa dạng của các giải pháp được tìm kiếm.

1.3. Thuật toán Genetic Algorithm: Genetic Algorithm (GA) là một thuật toán tối ưu hóa toàn cục, được lấy cảm hứng từ cơ chế sinh sản của các cá thể trong tự nhiên. GA sử dụng một quá trình lựa chọn, lai ghép và đột biến để tạo ra các thế hệ giải pháp tối ưu hơn. Kết hợp GA với Min-Conflict giúp tìm kiếm giải pháp tối ưu hơn và giảm thời gian tìm kiếm.

1.4. Thuật toán Hill Climbing: Hill Climbing là một thuật toán tìm kiếm địa phương, thực hiện di chuyển từ vị trí hiện tại đến vị trí kế tiếp tốt hơn. Kết hợp Hill Climbing với Min-Conflict giúp tăng khả năng tìm kiếm giải pháp tốt hơn và giảm thời gian tìm kiếm.

Tất cả các phương pháp kết hợp này đều nhằm tăng cường khả năng tìm kiếm giải pháp tốt hơn, giảm thiểu thời gian tìm kiếm và đạt được độ chính xác cao hơn.

2. Adaptive Min-Conflict

Adaptive Min-Conflict là một biến thể của thuật toán Min-Conflict được sử dụng để giải quyết các bài toán ràng buộc động, trong đó các ràng buộc có thể thay đổi trong quá trình giải quyết bài toán. Thuật toán này giải quyết vấn đề này bằng cách sử dụng một hàm đánh giá độ ưu tiên để xác định biến nào cần được thay đổi giá trị trước.

Trong Adaptive Min-Conflict, mỗi biến được gán một độ ưu tiên tương ứng với mức độ quan trọng của nó trong giải pháp. Mức độ ưu tiên này được cập nhật liên tục trong quá trình tìm kiếm giải pháp. Các biến có độ ưu tiên cao hơn sẽ được chọn để thay đổi giá trị trước.

Thuật toán Adaptive Min-Conflict có thể được áp dụng trong nhiều lĩnh vực khác nhau, từ lập kế hoạch sản xuất cho đến lập kế hoạch vận tải. Các ứng dụng cụ thể của thuật toán này bao gồm lập lịch công việc, quản lý tài nguyên, quản lý hệ thống mạng và nhiều ứng dụng khác.

3. Min-Conflict with Random Restart

Min-Conflict with Random Restart là một biến thể của thuật toán Min-Conflict, trong đó sử dụng việc khởi tạo lại ngẫu nhiên giải pháp sau một số lần lặp không tìm thấy giải pháp tối ưu.

Cụ thể, thuật toán Min-Conflict with Random Restart được thực hiện bằng cách lặp lại các bước sau:

Bước 1: Khởi tạo giải pháp ban đầu ngẫu nhiên.

Bước 2: Kiểm tra xem giải pháp hiện tại đã thỏa mãn các ràng buộc hay chưa. Nếu đã thỏa mãn, thuật toán kết thúc và trả về giải pháp tối ưu. Nếu chưa thỏa mãn, tiếp tục với bước 3.

Bước 3: Chọn một biến ngẫu nhiên trong giải pháp hiện tại và thay đổi giá trị của biến đó thành một giá trị mới được chọn ngẫu nhiên từ tập giá trị có thể có của biến đó.

Bước 4: Kiểm tra xem giải pháp mới có tốt hơn giải pháp hiện tại hay không. Nếu giải pháp mới tốt hơn, tiếp tục với bước 2 và 3. Nếu không, thuật toán sẽ thực hiện việc khởi tạo lại giải pháp ngẫu nhiên và tiếp tục với bước 2.

Quá trình này sẽ tiếp tục cho đến khi giải pháp tối ưu được tìm thấy hoặc đã đạt đến giới hạn số lần lặp tối đa. Việc khởi tạo lại giải pháp ngẫu nhiên sau mỗi số lần lặp được thiết lập để tránh việc rơi vào vùng giải pháp cục bộ.

4. Weighted Min-Conflict

Weighted Min-Conflict là một biến thể của thuật toán Min-Conflict, trong đó mỗi biến được gán một trọng số dựa trên mức độ quan trọng của nó trong bài toán. Trọng số này được sử dụng để xác định giá trị mới của biến trong quá trình tìm kiếm giải pháp.

Cụ thể, thuật toán Weighted Min-Conflict bao gồm các bước sau:

Bước 1: Khởi tạo giải pháp ban đầu ngẫu nhiên hoặc thông qua một thuật toán tìm kiếm khác.

Bước 2: Đánh giá mức độ quan trọng của mỗi biến trong bài toán và gán trọng số cho từng biến.

Bước 3: Chọn một biến có giá trị xung đột cao nhất và tính toán các giá trị mới của biến dựa trên trọng số của nó và các giá trị có thể thay đổi được.

Bước 4: Thay đổi giá trị của biến với giá trị mới tốt nhất và tiếp tục quá trình tìm kiếm giải pháp bằng cách lặp lại các bước 3 và 4 cho đến khi không còn xung đột nào hoặc đạt được giới hạn số lần lặp tối đa.

Việc gán trọng số cho từng biến có thể được thực hiện dựa trên các yếu tố như tần suất xuất hiện của biến trong các ràng buộc, tầm quan trọng của biến đối với mục tiêu của bài toán, hoặc theo cách tùy chỉnh khác tùy thuộc vào tính chất của bài toán cụ thể.

5. Min-Conflict with Tabu Search

Tabu Search Min-Conflict là một biến thể của thuật toán Min-Conflict kết hợp với thuật toán Tabu Search. Trong thuật toán này, các bước tìm kiếm mới được thực hiện bằng cách lựa chọn các biến và giá trị mới một cách ngẫu nhiên như trong thuật toán Min-Conflict. Tuy nhiên, các bước này phải tuân theo các ràng buộc của thuật toán Tabu Search.

Các biến đã được thay đổi trước đó sẽ được đánh dấu và được coi là "cấm" trong một số lượt tìm kiếm tiếp theo. Việc sử dụng danh sách tabu như vậy giúp tránh việc tìm kiếm trong cùng một vùng của không gian giải pháp quá nhiều lần và nhằm tránh bị rơi vào các vòng lặp vô hạn và giúp thuật toán tìm được các giải pháp tốt hơn.

6. Partial Min-Conflict

Partial Min-Conflict là một biến thể của thuật toán Min-Conflict được sử dụng để giải quyết bài toán về sắp xếp các công việc trong lịch làm việc. Trong bài toán này, có nhiều công việc cần được xếp vào lịch làm việc trong một khoảng thời gian nhất định, và mỗi công việc có một số lượng thời gian khác nhau để hoàn thành. Các công việc cũng có thể có các ràng buộc về thứ tự hoặc trùng lặp, do đó cần phải tìm kiếm một lịch làm việc hợp lý nhất để tối ưu hóa thời gian và đáp ứng các ràng buộc của bài toán.

Thuật toán "Partial Min-Conflict" bắt đầu với một lịch làm việc khởi tạo ngẫu nhiên. Trong mỗi vòng lặp, thuật toán sẽ chọn ngẫu nhiên một số công việc và tìm cách di chuyển chúng đến một vị trí mới trên lịch làm việc. Việc di chuyển sẽ được thực hiện bằng cách sử dụng hàm "Partial Min-Conflict", trong đó thuật toán sẽ chọn một số công việc trong các công việc được chọn và cố gắng thay đổi thứ tự của chúng để giảm thiểu số lượng xung đột trong lịch làm việc.

Thuật toán sử dụng một bộ đếm để giới hạn số lần lặp tối đa và sẽ dừng lại khi đạt đến giới hạn này hoặc khi tìm được lịch làm việc tốt nhất. Với mỗi lịch làm việc mới được tạo ra, thuật toán sẽ lưu giữ nó vào một danh sách và sử dụng nó để khởi tạo các lịch làm việc khác trong tương lai.

Partial Min-Conflict được xem là một biến thể hiệu quả của thuật toán Min-Conflict trong việc giải quyết bài toán về sắp xếp các công việc trong lịch làm việc.

IV. Đánh giá và phân tích thuật toán Min-Conflicts

Độ phức tạp tính toán của thuật toán

+ Độ phức tạp về thời gian:

Thời gian thực hiện thuật toán Min-Conflicts phụ thuộc vào số lượng biến và các ràng buộc trong bài toán csp. Thời gian chạy thuật toán Min-Conflicts có thể nhanh hơn so với thuật toán tìm kiếm cục bộ khác nhưng tùy vào độ phức tạp của bài toán. Thời gian trung bình của thuật toán Min-Conflicts có thể là O(n \* k), với n là số lượng biến và k là số lượng giá trị khác nhau có thể gán cho mỗi biến. Trong trường hợp tốt nhất, thuật toán Min-Conflicts có thể chạy nhanh hơn với độ phức tạp là O(n), nếu giải pháp tối ưu có thể tìm được trong một số lần thay đổi giá trị của biến nhỏ hơn k.

+ Độ phức tạp về không gian:

Thuật toán Min-Conflicts không yêu cầu lưu trữ một cấu trúc dữ liệu phức tạp. Nó chỉ cần lưu trữ trạng thái hiện tại của bài toán CSP và các giá trị khả thi cho các biến. Do đó, độ phức tạp không gian của thuật toán là O(n), với n là số biến. Vì vậy, độ phức tạp không gian của thuật toán là O(n), với n là số lượng biến trong bài toán. Điều này làm cho thuật toán Min-Conflicts phù hợp để giải quyết các bài toán lớn với số lượng biến và giá trị lớn.

Thuật toán Min-Conflicts có độ phức tạp thời gian tương đối tốt trong các thuật toán giải quyết bài toán CSP bằng phương pháp tìm kiếm cục bộ (local search).Và độ phức tạp không gian của thuật toán cũng là rất tốt và yêu cầu không nhiều bộ nhớ để lưu trữ trạng thái hiện tại của bài toán.

So sánh với các thuật toán khác

1) So sánh với thuật toán Quay lui (Backtracking)

Cả hai thuật toán min-conflicts và quay lui (backtracking) được sử dụng để giải quyết các vấn đề tối ưu hoá hoặc tìm kiếm. Tuy nhiên, chúng có những điểm khác nhau trong cách tiếp cận và hiệu quả giải quyết các vấn đề khác nhau.

Thuật toán quay lui (backtracking) là một thuật toán tìm kiếm theo chiều sâu trong không gian tìm kiếm của vấn đề. Thuật toán này cố gắng duyệt qua tất cả các giá trị của các biến trong vấn đề, và thử từng giá trị của từng biến cho đến khi tìm được giải pháp tối ưu hoặc không còn giá trị nào khả dĩ. Vì vậy, quay lui có thể tốn nhiều thời gian và không hiệu quả nếu không có phương án cắt tỉa (pruning) tối ưu.

Trong khi đó, thuật toán min-conflicts tìm kiếm giải pháp thông qua việc thay đổi giá trị của một số biến có liên quan đến vấn đề đang giải quyết. Thuật toán này thực hiện nhiều lần lặp lại, mỗi lần cố gắng tìm cách cải thiện giá trị của một số biến cho đến khi tìm được giải pháp tối ưu hoặc không thể cải thiện được nữa. Như vậy, thuật toán min-conflicts thường nhanh hơn và hiệu quả hơn trong các vấn đề có không gian tìm kiếm lớn.

Tóm lại, hai thuật toán có sự khác biệt trong cách tiếp cận giải quyết vấn đề. Quay lui tìm kiếm theo chiều sâu, trong khi min-conflicts tìm kiếm thông qua việc thay đổi giá trị của biến. Vì vậy, sự lựa chọn giữa hai thuật toán này phụ thuộc vào tính chất của vấn đề cần giải quyết.

2) So sánh với thuật toán Các nhánh và cận (Branch and Bound)

Thuật toán Branch and Bound là một phương pháp tìm kiếm tối ưu bằng cách liệt kê tất cả các khả năng của vấn đề và loại bỏ các khả năng không tối ưu nhằm cắt tỉa (prune) không gian tìm kiếm. Thuật toán này sử dụng một hàm cận để ước lượng giá trị tối ưu của vấn đề và tiếp tục liệt kê các khả năng tiếp theo chỉ khi chúng có khả năng tối ưu hơn giá trị hiện tại của vấn đề. Vì vậy, thuật toán Branch and Bound thường tốn nhiều thời gian, nhưng có thể tìm được giải pháp tối ưu trong một không gian tìm kiếm lớn.

Hai thuật toán có sự khác biệt trong cách tiếp cận giải quyết vấn đề. Min-conflicts tìm kiếm thông qua việc thay đổi giá trị của biến, trong khi Branch and Bound liệt kê tất cả các khả năng của vấn đề và cắt tỉa không gian tìm kiếm. Vì vậy, sự lựa chọn giữa hai thuật toán này phụ thuộc vào tính chất của vấn đề cần giải quyết. Nếu không gian tìm kiếm lớn và giải pháp tối ưu là yếu tố quan trọng, thì Branch and Bound là một lựa chọn tốt hơn. Trong khi đó, nếu giải pháp nhanh hơn và không yêu cầu tìm kiếm tất cả các khả năng của vấn đề, thì thuật toán min-conflicts là một lựa chọn tốt hơn.

Ngoài ra, cả hai thuật toán này đều có thể được kết hợp với các phương pháp khác để tăng hiệu quả giải quyết vấn đề. Ví dụ, Branch and Bound có thể được kết hợp với một số thuật toán tìm kiếm như tìm kiếm theo chiều rộng (BFS) hoặc tìm kiếm theo chiều sâu (DFS) để cải thiện hiệu suất của nó. Tương tự, min-conflicts có thể được kết hợp với một số phương pháp tìm kiếm như tìm kiếm địa phương hoặc tìm kiếm Monte Carlo để tăng tốc quá trình tìm kiếm.

Tóm lại, sự lựa chọn giữa thuật toán min-conflicts và Branch and Bound phụ thuộc vào tính chất của vấn đề cần giải quyết. Mỗi thuật toán có ưu điểm và hạn chế riêng, và có thể được kết hợp với các phương pháp khác để tăng hiệu quả giải quyết vấn đề.

Ưu điểm và hạn chế của thuật toán

- Ưu điểm của thuật toán Min-Conflicts

+ Hiệu quả: Thuật toán Min-conflict thường cho kết quả nhanh chóng và hiệu quả, đặc biệt là trong việc giải quyết bài toán tối ưu với ràng buộc. Giải thuật thường chỉ tìm kiếm giải pháp tốt nhất, và do đó không tốn nhiều thời gian so với các phương pháp tìm kiếm toàn bộ không gian giải pháp.

+ Đơn giản và dễ triển khai: Thuật toán Min-conflict dễ hiểu và dễ triển khai, do đó không đòi hỏi nhiều kiến thức chuyên môn để sử dụng.

+ Độ phức tạp tính toán thấp: Giải thuật thường không tốn nhiều bộ nhớ và độ phức tạp tính toán thấp hơn so với các giải thuật khác, vì chỉ cần lưu trữ trạng thái hiện tại của bài toán.

+ Phù hợp với bài toán tối ưu với ràng buộc: Thuật toán Min-conflict là một giải pháp hiệu quả cho bài toán tối ưu với ràng buộc, như bài toán Sudoku, vì giải thuật tìm kiếm giải pháp tốt nhất trong số các giải pháp khả thi.

- Hạn chế của thuật toán Min-Conflicts

+ Không đảm bảo tìm kiếm giải pháp tối ưu: thuật toán Min-conflict không đảm bảo tìm kiếm giải pháp tối ưu cho bài toán, chỉ tìm kiếm giải pháp tốt nhất trong số các giải pháp khả thi.

+ Phụ thuộc vào khởi tạo ban đầu: Nếu khởi tạo ban đầu không tốt, giải thuật có thể tốn nhiều thời gian để tìm kiếm giải pháp tối ưu. Ví dụ, nếu các ô đã được điền sẵn trong bảng Sudoku có quá nhiều giá trị trùng lặp, thuật toán Min-conflict có thể tìm kiếm lâu hơn so với các giải thuật khác.

+ Không thể áp dụng cho tất cả các loại bài toán: thuật toán Min-conflict phù hợp với bài toán tối ưu với ràng buộc, nhưng không thể áp dụng cho tất cả các loại bài toán khác.

Phần 3: Ứng dụng

1. Bài toán N-Queens

Năm 1850, Carl Gauss đề xuất vấn đề N-queens lần đầu tiên. Mục tiêu là sắp xếp n quân hậu trên một bàn cờ nxn sao cho không có hai quân hậu nào tấn công lẫn nhau. Vấn đề này thuộc loại vấn đề NP-complete, nghĩa là không có giải pháp thời gian đa thức. Bởi vì tất cả các giải pháp đều phức tạp nên không có cách giải quyết cụ thể nào. Một số giải pháp được khám phá bằng cách áp dụng phép xoay hoặc đối xứng cho nhau. Vấn đề n quân hậu là một biến thể mở rộng của vấn đề nữ hoàng 8 quân hậu cổ điển với N quân hậu và bàn cờ nxn.

Vấn đề N-Queens, là một dạng vấn đề tối ưu hoá tổ hợp, là một vấn đề rất quan trọng, giải quyết được các thách thức trong cuộc sống thực. Các vấn đề tối ưu sắp xếp tổ hợp là một dạng vấn đề toán học có rất nhiều ứng dụng trong thực tế, bao gồm tự động hóa thiết kế VLSI, thiết kế và điều khiển mạng thông tin, lập lịch công việc, và nhiều hơn nữa. Chúng thường có rất nhiều biến để giải quyết. Ví dụ, tự động hóa thiết kế VLSI yêu cầu ít nhất một triệu biến. Bên cạnh đó, do độ khó của NP, sự phức tạp tính toán thường là không thể khắc phục được. Do việc tính toán song song và sự thuận tiện cho việc thực hiện phần cứng, các mạng neuron cung cấp các giải pháp như những thuật toán xấp xỉ cho nhiều thách thức khó khăn.

Min-Conflicts giải quyết vấn đề N-Queens bằng cách gán lại các quân hậu vào một cột trên bàn cờ một cách ngẫu nhiên. Phương pháp sẽ tìm số xung đột trong mỗi ô trong mỗi nước đi có thể có. Chương trình đưa quân hậu đến ô có ít xung đột nhất, phá vỡ trật tự ngẫu nhiên nếu có sự cố. Lưu ý rằng mỗi hướng mới từ một quân hậu có thể tấn công sẽ tăng số lượng xung đột. Cuộc đấu chỉ được tính một lần nếu hai quân hậu tấn công từ cùng một hướng. Lưu ý rằng nếu một quân hậu đang ở trong tình huống mà một nước đi sẽ đặt cô ấy vào nhiều xung đột hơn so với hiện tại, cô ấy sẽ không di chuyển. Do đó, nếu một quân hậu đang ở trong tình huống ít xung đột, cô ấy không cần phải di chuyển. Thời gian để giải quyết N-Queens bằng thuật toán này độc lập với kích thước của vấn đề.

2. Bài toán tô màu đồ thị

Bài toán tô màu đồ thị là bài toán tìm cách sử dụng một số lượng màu ít nhất để tô màu các đỉnh của đồ thị sao cho hai đỉnh kề nhau không được tô màu cùng một màu. Thuật toán Min-Conflicts được sử dụng để giải quyết bài toán tô màu đồ thị bằng cách sử dụng tìm kiếm cục bộ để tìm một giải pháp tối ưu cho bài toán.

Thuật toán Min-Conflicts được áp dụng bằng cách tạo một bảng ghi danh sách các đỉnh và màu được gán cho mỗi đỉnh. Mỗi lần lặp, thuật toán sẽ chọn một đỉnh không được tô màu và tìm kiếm một màu mới để gán cho đỉnh đó. Sau đó, thuật toán kiểm tra xem giải pháp mới này có hợp lệ hay không bằng cách kiểm tra xem hai đỉnh kề nhau có được gán cùng một màu hay không. Nếu không hợp lệ, thuật toán sẽ thử một màu khác cho đỉnh đó và lặp lại quá trình này cho đến khi tìm ra một giải pháp tối ưu.

3. Ứng dụng trong lập lịch sản xuất (Job-shop scheduling)

Lập lịch sản xuất là một bài toán quản lý lịch trình sản xuất trong các nhà máy. Mỗi công việc bao gồm một chuỗi các nhiệm vụ, cái mà phải được thực hiện theo một trật tự nhất định và mỗi nhiệm vụ phải được xử lý trên một máy cụ thể. Ví dụ, một công việc có thể sản xuất ra một món đồ tiêu dùng đơn lẻ, như một chiếc ô tô. Nhiệm vụ của bài toán là sắp xếp các nhiệm vụ trên các máy sao cho tối thiểu chiều dài của lịch (thời gian để tất cả các công việc được hoàn thành).

Bài toán Job-shop scheduling cần xác định thứ tự các công việc trên các máy móc sao cho:

- Không có nhiệm vụ nào trong một công việc có thể được bắt đầu cho đến khi nhiệm vụ trước cho công việc đó được hoàn thành.

- Một máy chỉ có thể làm một nhiệm vụ tại một thời điểm.

- Một công việc khi đã bắt đầu phải chạy cho đến khi hoàn thành

Bài toán Job-shop scheduling là bài toán có tính ứng dụng cao trong các ngành sản xuất, vận tải, vận hành hệ thống, linh kiện điện tử, chẩn đoán y khoa, quản lý tài nguyên, tài chính và nhiều ngành công nghiệp khác.

Thuật toán Min-conflicts có thể được áp dụng vào trong bài toán trên bằng các bước phía dưới đây:

Bước 1: Khởi tạo lịch trình ban đầu bằng cách sắp xếp các công việc trên máy móc một cách ngẫu nhiên.

Bước 2: Tính tổng thời gian hoàn thành của tất cả công việc để đánh giá chất lượng của lịch trình

Bước 3: Chọn ngẫu nhiên một công việc trong một máy móc và tìm kiếm một máy móc khác để thực hiện công việc đó.

Bước 4: Với mỗi máy móc được tìm kiếm, tính toán lại thời gian hoàn thành mới của công việc được di chuyển

Bước 5. Chọn máy có thời gian hoàn thành mới của công việc được di chuyển nhỏ nhất và di chuyển công việc tới máy móc đó

Bước 6. Lặp lại các bước 2 đến 6 cho đến khi đạt được lịch trình tối ưu

4. Ứng dụng trong lập lịch họp (Meeting Scheduling Problems)

Bài toán lập lịch họp (Meeting Scheduling Problem) là một vấn đề quan trọng đối với các cuộc họp tổ chức lớn. Một cuộc họp liên quan đến một nhóm những người đại diện và bao gồm ít nhất hai người tham dự. Những người đại diện sẽ thay mặt những người tham gia trong cuộc họp. Mục đích giải quyết là tìm kiếm thời gian bắt đầu cho mỗi cuộc họp với một số ràng buộc nhất định. Mỗi ràng buộc giới hạn hai cuộc họp không được trùng lặp với nhau nếu chúng có cùng đại lý cuối cùng và chúng không thể bắt đầu cùng một thời điểm. Ví dụ, nếu cuộc họp A kết thúc với đại lý L, và cuộc họp B bắt đầu với đại lý L thì chúng sẽ bị trùng lặp và không thể xảy ra đồng thời ngoài ra hai cuộc họp này không được bắt đầu cùng một lúc.

Trong ứng dụng này, thuật toán Min-conflict có thể được áp dụng như sau:

Bước 1: Tạo ra một lịch trình bằng cách sắp xếp các cuộc họp một cách ngẫu nhiên.

Bước 2: Chọn một cuộc họp ngẫu nhiên, tìm kiếm một thời gian bắt đầu mới tốt nhất mà làm giảm số lượng ràng buộc thời gian xung đột (có số lượng ràng buộc nhỏ nhất)

Bước 3: Nếu không có giá trị nào tốt hơn được tìm thấy, chọn ngẫu nhiên một cuộc họp khác và thay đổi thời gian bắt đầu của nó

Bước 4: Lặp lại cho đến khi tất cả các ràng buộc thời gian được thỏa mãn hoặc đến một lịch trình họp được chấp nhận

5. Ứng dụng trong lập lịch lực lượng lao động (Workforce Scheduling)

Vấn đề của lập lịch lực lượng lao động liên quan đến việc đáp ứng các vấn đề thỏa mãn ràng buộc khác nhau như số lượng nhân viên được cần mỗi ngày, các chuỗi ca làm việc bị cấm, độ dài của các khối làm việc và các ngày nghỉ,... Có hai loại chính của lịch làm việc lực lượng lao động: lịch xoay và lịch không chu kỳ. Trong lịch xoay, tất cả nhân viên có cùng lịch cơ bản nhưng bắt đầu ở các thời điểm khác nhau. Giả định trong một công ty có n nhân viên, một tập hợp m ca làm việc, độ dài của lịch trình, yêu cầu thời gian, các ràng buộc về các chuỗi ca làm việc cho phép để phân bổ cho nhân viên, độ dài tối đa và tối thiểu của chuỗi các ca liên tiếp, các khối ngày làm việc và nghỉ. Mục tiêu là tìm thấy một lịch xoay (phân bổ ca làm việc cho nhân viên) thỏa mãn ma trận yêu cầu và thực hiện tất cả các ràng buộc khó được chỉ định.

Thuật toán Min-Conflicts bắt đầu bằng cách tạo ra một phân công ngẫu nhiên ban đầu của nhân viên vào các ca làm việc. Sau đó, xác định bất kỳ biến (nhân viên) nào vi phạm các ràng buộc, chẳng hạn như một nhân viên được phân công quá nhiều hoặc quá ít ca làm việc. Thuật toán sẽ chọn một trong những biến này ngẫu nhiên và gán cho nó một giá trị mới có số lượng ràng buộc bị vi phạm ít nhất. Sau mỗi lần gán, thuật toán kiểm tra xem giải pháp có đáp ứng được tất cả các ràng buộc hay không; nếu không, nó sẽ lặp lại quy trình lặp đến khi tìm thấy một giải pháp đáp ứng tất cả các ràng buộc.

6. Ứng dụng trong bài toán Bệnh viện/Cư dân với sự ràng buộc (The Hospitals/Residents problem with Ties)

Bài toán Hospitals/Residents with Ties là một bài toán tối ưu hóa kết hợp thông qua việc các bệnh viện cần phải chọn một số lượng cư dân để phục vụ cũng như các cư dân cũng có thể lựa chọn và ưu tiên những bệnh viện mà họ muốn được phục vụ. Trong bài toán này, mỗi bệnh viên và cư dân đều được mô tả bằng một danh sách cách cư dân (bệnh viện) yêu thích của họ, thể hiện sự ưu tiên của họ đối với các điều kiện và mức độ của dịch vụ được cung cấp. Mục đích của bài toán là xây dựng một sơ đồ ghép đôi từ danh sách ưu tiên của các bệnh viện và cư dân và tối ưu hóa mức độ hài lòng của tất cả các bên trong quá trình lựa chọn bệnh viện hay đối tác của họ.

Thuật toán Min-Conflicts được áp dụng bằng cách khởi tạo bảng I gồm danh sách tham chiếu của cư dân và danh sách tham chiếu của bệnh viện và bắt đầu với một cặp phù hợp ngẫu nhiên. Với mỗi cặp được lựa chọn, Min-Conflicts sẽ tìm kiếm một bộ các cặp chặn không bị áp đảo (có nghĩa là không có cặp nào khác ưu tiên hơn cả hai bên đối với nhau) để phù hợp với quan điểm của cư dân, chọn một cư dân được các bệnh viện yêu thích nhất trong bộ đó. Tuy nhiên, để tránh gặp vấn đế cực tiểu cục bộ, thuật toán này sẽ chọn ngẫu nhiên một cư dân trong bộ các cặp chặn không bị áp đảo với xác suất nhỏ. Sau đó, nó sẽ loại bỏ cặp (cư dân, bệnh viện) và tiếp tục cho đến khi tìm được sự sắp xếp tối ưu của bài toán.

Phần 4. Kết luận

1. Tổng kết về thuật toán Min-Conflicts

Thuật toán Min-Conflicts là giải quyết các bài toán tối ưu hóa tổ hợp, trong đó các giải pháp tối ưu được định nghĩa là tập hợp các giá trị cho các biến quyết định sao cho hàm mục tiêu đạt giá trị nhỏ nhất hoặc lớn nhất.

Thuật toán này mang lại những ưu điểm như hiệu quả, đơn giản dễ triển khai, độ phức tạp tính toán thấp nhưng bên cạnh đó cũng có những mặt hạn chế như không đảm bảo tìm kiếm giải pháp tối ưu, phụ thuộc vào khởi tạo ban đầu cũng như không thể áp dụng cho tất cả các loại bài toán.

Các bước để giải quyết bài toán bao gồm:

Bước 1: Xác định vấn đề của bài toán

Bước 2: Khởi tạo giá trị ban đầu của trạng thái

Bước 3: Chọn ngẫu nhiên một phần tử trong tập hợp

Bước 4: Cập nhật giá trị của phần tử đó sao cho tổng xung đột là nhỏ nhất

Bước 5: Tiếp tục lặp lại bước 2 đến bước 5 khi đáp ứng điều kiện dừng

Bước 6: Trả về giá trị

Các khái niệm liên quan đến thuật toán Min-Conflicts bao gồm: Hàm mục tiêu (objective function), hàm ràng buộc (constraint function), trạng thái (state), điểm (point), không gian trạng thái (state space).

Về độ phức tạp tính toán của thuật toán, thời gian của thuật toán là O(n\*k) với n là số lượng biến và k là số lượng giá trị khác nhau có thể gán cho mỗi biến còn về mặt không gian, độ phức tạp của thuật toán là O(n) với n là số biến. Min-conflicts thường được ứng dụng trong các bài toán như N-Queens, Tô màu đồ thị, các ứng dụng thực tiễn liên quan đến lập lịch: lập lịch sản xuất, lập lịch cuộc họp, lập lịch lực lượng lao động,…

Mặc dù thuật toán không đảm bào tìm được giải pháp tối ưu, nhưng nó thường đưa ra các giải pháp có chất lượng tương đối tốt và thường được sử dụng trong thực tế để giải quyết các bài toán có số lượng ràng buộc lớn.

2. Hướng phát triển và nghiên cứu trong tương lai

Hướng nghiên cứu tiếp theo cho thuật toán Min-Conflicts:

+ Cải tiến tính hiệu quả của thuật toán: Hiện tại, thuật toán Min-Conflicts chưa được chứng minh là hiệu quả trong mọi trường hợp. Vì thế, việc nghiên cứu các biến thể của thuật toán có thể được thực hiện để cải thiện hiệu suất và kết quả tìm kiếm.

+ Ứng dụng Học Máy vào Min-Conflicts: thuật toán Min-Conflicts trên các bài toán lớn đòi hỏi một lượng lớn dữ liệu đầu vào. Việc kết hợp với các kỹ thuật học máy có thể giúp cải thiện hiệu suất và độ chính xác của thuật toán.

+ Nghiên cứu các thuật toán tìm kiếm khác nhau: Thuật toán Min-Conflicts là một trong những thuật toán để giải quyết bài toán CSP. Vì thế có thể thực hiện các cuộc nghiên cứu liên quan đến việc so sánh, đánh giá hiệu quả với các thuật toán khác.

Các ứng dụng tiềm năng của thuật toán cho các lĩnh vực khác nhau:

+ Tối ưu hóa lịch trình sản xuất: Min-conflicts có thể được sử dụng để tối ưu hóa lịch trình sản xuất trong các doanh nghiệp hay công ty, giúp nâng cao hiệu suất và giảm thiểu các chi phí phát sinh có thể gây ra.

+ Lập lịch trình du lịch: Thuật toán này có thể được sử dụng để tối ưu hóa lịch trình du lịch cho các khách du lịch, tiết kiệm được thời gian, chi phí, đem lại trải nghiệm tốt hơn cho khách du lịch.

+ Giảm thiểu tắc nghẽn giao thông: Thuật toán Min-conflicts có thể được sử dụng để giảm thiểu tắc nghẽn giao thông thông qua việc thay đổi hướng đi, giảm thiểu số lượng xe trên các điểm hay xảy ra tắc nghẽn và tăng số lượng xe trên các đoạn đường trải dài. Như vậy có thể giúp đảm bảo tính an toàn cho các phương tiện cũng tính đồng bộ và hiệu quả giữa các kế hoạch quản lý giao thông và đường bộ.

+ Lập lịch chuyến bay: Min-conflicts có thể được sử dụng để tối ưu hóa lịch trình chuyến bay, giảm thiểu độ trễ, tiết kiệm thời gian và đem lại chất lượng dịch vụ tốt hơn cho khách hàng.

II. Thực nghiệm và kết quả

Link: https://developers.google.com/optimization/scheduling/job\_shop

Amitrakshar Sanyal, Surjyanee Halder, Tamoghna Ghosh, An approach to solve the N-Queens Problem using Artificial Intelligence algorithm

Nysret Musliu, Min-Conflicts Based Heuristics for Rotating Workforce Scheduling Problem

CHN and Min-Conﬂict Heuristic to Solve

Scheduling Meeting Problems

CHN and Min-Conﬂict Heuristic to Solve

Scheduling Meeting Problems

Adil Bouhouch, Chakir Loqman and Abderrahim El Qadi, CHN and Min-Conflict Heuristic to Solve Scheduling Meeting Problems

Nguyen Thi Uyen, Nguyen Long Giang, Truong-Thang Nguyen, Hoang Huu Viet, A min-conflicts algorithm for maximum stable matchings of the hospitals/residents problem with ties