**Tô màu đồ thị**

**I. Bài toán: Tô màu đồ thị**

* Input: Nhập vào đồ thị vô hướng G
* Output: In ra đồ thị đã được tô màu và số màu cần dùng để tô sao cho:
  + Tất cả các đỉnh của đồ thị đều được tô màu
  + Không có 2 đỉnh liền kề nào cùng màu
  + Số màu sử dụng là tối thiểu

**II. Bộ test và số liệu đánh giá**

1. **Bộ test**

Chạy chương trình trên đồ Erdos-Renyi được tạo ngẫu nhiên . Thay vì sử dụng các đồ thị từ thế giới thực, mà thường quá lớn để chạy, chúng ta tạo ra các đồ thị nhỏ nhưng khó.

Trong trường hợp của bài toán tô màu đồ thị, có một ngưỡng đặc biệt để đo lường khả năng tô màu k của các đồ thị, với mỗi giá trị k ≥ 3. Ngưỡng này phụ thuộc vào tỷ lệ c = m/n = p × n, trong đó m là số cạnh của đồ thị, n là số đỉnh, và p là xác suất một cạnh được tạo giữa bất kỳ cặp đỉnh nào. Theo thử nghiệm đã chỉ ra rằng đối với các đồ thị có sắc độ tô màu 3 và kích thước n = 10, 12, 14, 16 và 18, ngưỡng chuyển đổi thường xảy ra ở khoảng c = 4.5. Do đó, sử dụng giá trị này để tạo ra tập dữ liệu thử nghiệm trong bài.

Chạy thử nghiệm chương trình với các giá trị n = 10, 20, 30, 40, 50, 60 và quan sát. Mỗi giá trị của n sẽ tương ứng với một giá trị của p được tính theo công thức ở trên với c = 4.5.

1. **Số liệu đánh giá**

**III. Chương trình**

1. **ILP**

Cho x, w là các biến nhị phân. Mỗi biến x[i][v] bằng 1 nếu đỉnh v được tô màu i và bằng 0 nếu không. Mỗi biến w[i] bằng 1 nếu màu i đã được dùng và bằng 0 nếu chưa.

Mục tiêu của bài toán cần tối thiểu số màu sử dụng. Ta xác định được hàm mục tiêu:

với i = 0, 1, 2,…

Sau đó thêm các ràng buộc:

* Mỗi đỉnh chỉ tô 1 màu duy nhất:

với i = 0, 1, 2,…

* Cập nhật w[i]

* Không có 2 đỉnh liền kề nào cùng màu:

với v và u là hai đỉnh liền kề

1. **Annealing**

Ý tưởng để giải bài toán như sau:

* Tìm tập con độc lập lớn nhất trong đồ thị (maximum independent set – MIS)
* Tô màu cho tập con này và loại bỏ tập con này khỏi đồ thị
* Lặp lại 2 bước trên cho tới khi không còn đỉnh nào trong đồ thị
* Số lượng tập con lớn nhất tìm được chính là số màu tối thiểu cần tìm

Để giải được bài toán, cần tìm được tập con lớn nhất trong một đồ thị cho trước. Ở đây sử dụng hàm phạt QUBO tìm kiếm giải pháp cho mức năng lượng thấp nhất. Hàm QUBO có dạng:

Trong đó, x là biến nhị phân. là hệ số tuyến tính, là hệ số bậc hai. Để tìm tập con độc lập lớn nhất, ứng dụng hàm trên như sau:

với

Mục tiêu của hàm là tìm được tập con lớn nhất trong đồ thị ban đầu sao cho không có hai nút nào liền kề nhau. Trong hàm trên, ban đầu hệ số tuyến tính được đặt bằng -1 để khuyến khích chọn đỉnh i, trong khi đó hệ số bậc hai α dùng để phạt khi có 2 đỉnh liền kề được chọn. Sau khi thử nghiệm với n và α khác nhau, không có sự khác biệt khi tăng α lên vượt quá giá trị tối thiểu là 2.

Trong quá trình chạy, một đồ thị có thể có thể tìm được nhiều tập con độc lập lớn nhất có cùng kích thước. Để tối ưu bài toán (số màu sử dụng là ít nhất), chọn tập con độc lập có tổng số bậc của các đỉnh lớn nhất để bỏ khỏi đồ thị (bỏ đi nhiều liên kết nhất có thể)

Chương trình chạy thử nghiệm sử dụng SimulatedAnnealingSampler() để tìm giải pháp cho bài toán.

**IV. Kết quả**

Với n = 10, 20, 30, 40, 50, 60 chạy chương trình thử trên cả 2 thuật toán nhận được kết quả như sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | **ILP** | | **SA** | |
| Số màu tối thiểu | Thời gian chạy chương trình (s) | Số màu tối thiểu | Thời gian chạy chương trình (s) |
| 10 | 3 | 0.15642714500427246 | 3 | 0.2719745635986328 |
| 20 | 4 | 1.927896499633789 | 4 | 0.632854700088501 |
| 30 | 4 | 138.2559459209442 | 4 | 0.9029972553253174 |
| 40 | 4 | 1535.2528820037842 | 5 | 1.7444288730621338 |
| 50 | - | - | 4 | 1.7941296100616455 |
| 60 | - | - | 5 | 1.8805954456329346 |

Dựa vào kết quả trên, có thể thấy khi sử dụng SA, chương trình chạy cho ra thời gian nhanh hơn nhiều so với ILP với các bộ test lớn hơn.

Sử dụng các test n = 10, 20, 30 để kiểm tra tiếp. Chạy 10 lần mỗi test và so sánh kết quả giữa 2 phương pháp.

**V. Tài liệu tham khảo**

[1] Julia Kwok, Kristen Pudenz. Graph Coloring with Quantum Annealing. *rXiv:2012.04470v1 [quant-ph] 8 Dec 2020*

[2] Hausdorff School: Computational Combinatorial Optimization, September 12-16, 2022. Lecture 1: ILP Formulations for the Graph Coloring Problem