1. **Thuật toán Havel – Hakimi trong xây dựng đồ thị đơn từ dãy bậc cho trước:**

Từ ý tưởng từ định lý Havel – Hakimi, **thuật toán Havel – Hakimi** được hình thành để xây dựng một đồ thị đơn từ dãy bật cho trước

Dưới đây là một cách giải quyết bài toán dựa trên **thuật toán Havel – Hakimi**:

* **Input:** Giả định đầu vào thuật toán là , tức là số lượng đỉnh, và dãy bậc, tức là một dãy số nguyên không âm .
* **Ý tưởng chính:** Sắp xếp dãy bậc đã cho thành dãy không tăng. Trong mỗi bướcxét các bậc lớn nhất trước để tạo các cạnh kết nối giữa các đỉnh có bậc cao nhất với các đỉnh khác. Sau đó, nó giảm bậc của các đỉnh liên quan và tiếp tục với các bậc tiếp theo cho đến khi tất cả các bậc đều bằng 0.
* **Mô tả thuật toán:**
* Vector deg đại diện cho số lượng đỉnh trong đồ thị; tạo một ma trận *adjacencyMatrix* có kích thước n x n và khởi tạo tất cả các giá trị ban đầu bằng 0.
* Tạo một bản sao của vector deg và sắp xếp nó theo thứ tự giảm dần, lưu vào *rsorted\_deg*.
* Duyệt qua các phần tử của *rsorted\_deg*.
* Với mỗi giá trị trong *rsorted\_deg* ứng với bậc cao nhất trong dãy bậc hiện tại, Duyệt qua các phần tử từ 1 đến giá trị của *rsorted\_deg[i]*.
* Gán giá trị 1 cho phần tử tại vị trí ***(i, i + j)*** trong adjacencyMatrix để biểu diễn một cạnh kết nối từ đỉnh i đến đỉnh *i + j.*
* Gán giá trị 1 cho phần tử tại vị trí ***(i + j, i)*** trong adjacencyMatrix để biểu diễn một cạnh kết nối từ đỉnh *i + j* đến đỉnh *i*.
* Giảm giá trị của rsorted\_deg[i + j] đi 1 khi duyệt xong.
* Lặp lại cho đến khi tất cả các bậc của đỉnh trong dãy bậc bằng 0
* **Minh họa thuật toán:**
* Giả sử cần xây dựng đồ thị gồm 6 đỉnh từ dãy bậc . Một mảng một chiều deg được khởi tạo để lưu các giá trị trong dãy bậc đã cho ứng với các chỉ số từ 1 đến 6. Bảng sau minh họa các bước của thuật toán:

A grid of numbers and symbols

Description automatically generated

* Đồ thị kết quả là:

A diagram of a triangle with points and lines

Description automatically generated

* **Mã giả của chương trình:**

A white screen with black text

Description automatically generated

* **Đánh giá độ phức tạp của thuật toán:**
* Độ phức tạp trung bình của thuật toán
* **Nhận xét:**
* Phương pháp trên cho phép xây dựng đồ thị từ một đỉnh bất kỳ, đồng thời, kiểm tra tính hợp lệ của dãy bậc thỏa mãn định lý Havel-Hakimi và kết thúc nếu dãy bậc còn lại không thỏa mãn các tính chất trên tại bất kỳ một thời điểm nào của quá trình xây dựng đồ thị.
* Quá trình tổng hợp dãy bậc còn lại cũng sẽ tạo ra một dãy bậc mới đã được sắp xếp thông qua một vài bước không tốn nhiều chi phí. Do đó,các bước sẽ tối ưu cho quá trình chọn đỉnh bất kỳ tiếp theo nếu có.
* Một phương pháp phổ biến dùng để biểu diễn thuật toán Havel – Hakimi khác là Trong mỗi bước ta chọn đỉnh i có bậc cao nhất trong số các đỉnh còn lại và giảm bậc của đỉnh có bậc cao nhất trong dãy bậc còn lại không chứa . Lặp lại quá trình trên cho đến khi tất cả các bậc đều bằng 0.
* **Mở rộng:**
* Như đã đề cập ở trên, đồ thị thu được là một đồ thị đơn vô hướng hết sức đặc biệt, trong đó đỉnh có bậc cao nhất kề với tất cả các đỉnh có bậc cao tiếp theo. Ta có thể mở rộng thuật toán trên cho đồ thị có hướng và các lớp đồ thị với những tính chất khác nhau.
* Ngoài ra, chúng ta có thể tìm các đồ thị bằng thuật toán sinh, duyệt qua tất cả các cấu hình có thể của các cạnh trong đồ thị, từ đó xây dựng các đồ thị tương ứng. Thuật toán sinh sử dụng các quy tắc sinh cẩn thận để đảm bảo rằng ta không tạo ra các đồ thị trùng lặp hoặc không thỏa mãn dãy bậc đã cho.
* Một vài phương pháp khác để xây dựng đồ thị từ dãy bậc ban đầu như: thuật toán , hoặc sử dụng chu trình Euler đan màu,…

1. **Tài liệu tham khảo:**

* On and Havel-Hakimi algorithms: <https://arxiv.org/pdf/1111.3282.pdf>
* Sử dụng chu trình euler xây dựng tất cả các đồ thị có dãy bậc cho trước: <https://vjol.info.vn/index.php/DHSPHN-KHTN/article/download/42614/34365/>
* Generating graphs from degree sequence:

<https://scholar.uwindsor.ca/cgi/viewcontent.cgi?article=9870&context=etd>

* Wikipedia:

<https://en.wikipedia.org/wiki/Havel%E2%80%93Hakimi_algorithm>

* ChatGPT.