Đại Học Quốc Gia TP.HCM
Trường Đại học Bách Khoa
Khoa KH&KT Máy Tính

Vietnam National University – HCMC
Hochiminh University of Technology
Faculty of Computer Science and Engineering

CẤU TRÚC DỮ LIỆU & GIẢI THUẬT
ASSIGNMENT 02
ĐỊNH VỊ VÀ DỰ BÁO

1 Giới thiệu

Trong hầu hết các phương tiện vận chuyển từ dân dự đến quân sự hiện nay, người ta luôn trang bị các thiết bị hỗ trợ GPS (hệ thống định vị toàn cầu). Việc trang bị này giúp theo dõi lộ trình cũng như giúp thu thập các thông tin cần thiết cho các nhà phân tích dữ liệu. Các thiết bị này ghi nhận vị trí GPS tại các thời điểm lấy mẫu và gởi về máy chủ (server) liên tục. Vì số lượng thông tin gởi về server khá nhiều nên để giảm thiểu tải cho server, người ta quyết định lưu dữ liệu tạm thời trên cấu trúc cây để thao tác nhanh chóng và lưu dữ liệu đó xuống database vào cuối ngày. Dữ liệu trong ngày cũng tương đối lớn, do đó cây được chọn pahải có cấu trúc cân bằng nhằm đảm bảo truy xuất nhanh. Do đó trong trường hợp này AVL được chọn làm cấu trúc dữ liệu lưu trữ.

2 Yêu cầu

Trong bài tập lớn này, sinh viên sẽ được cung cấp 2 tập tin chứa dữ liệu nhập: **data.csv** và **request.txt**. Chi tiết mô tả dữ liệu nhập, xuất và các công việc sinh viên phải làm được nêu trong các mục dưới đây.

2.1 Dữ liệu nhập

- File **data.csv**: chứa các thông tin về vị trí của các phương tiện vận chuyển, trong đó chúng ta quan tâm 4 thông số quan trọng sau:
  - COLLECTED_TIME (timestamp): thời điểm mà dữ liệu được gởi về.
  - TAG: ID của phương tiện.
  - LONGITUDE (long): kinh độ.
  - LATITUDE (lat): vĩ độ.
- File **request.txt**: chứa các yêu cầu xử lý dữ liệu. Các yêu cầu này được mô tả trong phần sau. Nội dung tập tin dữ liệu nhập request.txt sẽ mô tả các yêu cầu theo thứ tự từ trái sang phải. Số lượng các yêu cầu là không cố định, có thể thay đổi tùy theo test case. Một yêu cầu có thể xuất hiện nhiều lần trong tập tin dữ liệu nhập. Nếu số yêu cầu nhiều, các yêu cầu có thể trình bày thành nhiều dòng. Mỗi yêu cầu được mô tả sẽ cách nhau bởi một dấu khoảng trắng, kết thúc các yêu cầu bằng dấu chấm phẩy. Nếu mã yêu cầu không tồn tại thì bỏ qua yêu cầu đó và đi tiếp đến yêu cầu tiếp theo.

2.2 Dữ liệu xuất

Dữ liệu phải được xuất theo từng dòng riêng biệt với mỗi yêu cầu. Dữ liệu xuất theo cú pháp:

<mã yêu cầu>: <kết quả>

trong đó, <**kết quả**> được in ra theo quy tắc sau:

- Ký tự: không cần định dạng.
- Số nguyên: không cần định dạng.
- Số thực: độ chính xác 12 chữ số.

- Danh sách (số nguyên hoặc số thực): mỗi phần tử cách nhau bởi một dấu khoảng trắng và không có dấu khoảng trắng ở cuối danh sách. Các phần tử trong danh sách vẫn phải tuân theo quy tắc về số nguyên và số thực ở trên.

3 Quy ước và yêu cầu

3.1 Quy ước

- Phương tiện được xem là đang đứng yên nếu trong hai lần liên tiếp mà tọa độ GPS có độ lệch từ 5m ($\leq$ 5m) trở xuống.
- Phương tiện được xem là đang kiểm tra nếu trong 5 lần liên tiếp mà độ lệch toạ độ GPS từ 5m trở xuống ($<$ 5m).
- Phương tiện được xem là đang bảo trì nếu trong 20 lần liên tiếp mà độ lệch toạ độ GPS từ 5m trở xuống ($<$ 5m).
- Hướng theo kinh độ: vị trí X ($long_X$, lat_X) nằm về phía Đông (East, được kí hiệu bởi kí tự 'E' khi xuất kết quả) so với vị trí Y ($long_Y$, lat_Y) nếu $long_X - long_Y \geq 0$ hoặc nằm về phía Tây (West, được kí hiệu bởi kí tự 'W' khi xuất kết quả) nếu ngược lại.
- Hướng theo vĩ độ: vị trí X ($long_X$, lat_X) nằm về phía Bắc (North, được kí hiệu bởi kí tự 'N' khi xuất kết quả) so với vị trí Y ($long_Y$, lat_Y) nếu $lat_X - lat_Y \geq 0$ hoặc nằm về phía Nam (South, được kí hiệu bởi kí tự 'S' khi xuất kết quả) nếu ngược lại.

3.2 Yêu cầu truy xuất

Các yêu cầu truy xuất có mã và mô tả về việc truy xuất lần lượt sau đây:

1. **Xác định vị trí tương đối và khoảng cách của hai phương tiện tại một thời điểm**
   Mã yêu cầu: **1_X_Y_hhmmss**
   Trong đó **X, Y** là số hiệu của phương tiện.
   Ví dụ: mã yêu cầu là **1_1526_1782_123048** tức là cần xác định vị trí tương đối và khoảng cách của phương tiện mang số hiệu **1526** so với phương tiện mang số hiệu **1782** lúc **12 giờ 30 phút 48 giây**. Kết quả của yêu cầu này là một danh sách gồm có 3 thành phần có thứ tự sau:
   - Hướng theo kinh độ: 'E' hoặc 'W'.
   - Hướng theo vĩ độ: 'N' hoặc 'S'.
   - Khoảng cách tính theo km và luôn luôn là số thực.

   Trong trường hợp, không có dữ liệu về vị trí của một trong 2 phương tiện hoặc cả hai phương tiện mang số hiệu trong yêu cầu tại thời điểm đó thì xuất ra **-1**.
2. **Xác định số phương tiện chỉ di chuyển một phía theo kinh độ của một vị trí xác định**
   Mã yêu cầu: **2_**A_{long}**_H**
   Ví dụ: mã yêu cầu là **2_-122.3879_W**, tức là cần xác định số lượng phương tiện mà lộ trình đi của các phương tiện đó chỉ nằm về phía Tây so với vị trí có kinh độ là **-122.3879**. Kết quả của yêu cầu này là một số nguyên thể hiện số lượng phương tiện thỏa mãn yêu cầu.
3. **Xác định số phương tiện chỉ di chuyển một phía theo vĩ độ của một vị trí xác định**
   Mã yêu cầu: **3_**A_{lat}**_H**
   Ví dụ: mã yêu cầu là **2_37.70786_S**, tức là cần xác định số lượng phương tiện mà lộ trình đi của các phương tiện đó chỉ nằm về phía Nam so với vị trí có vĩ độ là **37.70786**. Kết quả của yêu cầu này là một số nguyên thể hiện số lượng phương tiện thỏa mãn yêu cầu.

4. **Xác định số phương tiện nằm trong tầm kiểm soát của một trạm quan sát trong một khoảng thời gian xác định**
   Mã yêu cầu: **4**_A_{long}_A_{lat}_**R**_H_1_H_2
   Xác định số lượng phương tiện nằm trong tầm kiểm soát của một trạm quan sát A có tọa độ (A_{long}, A_{lat}), bán kính kiểm soát là R, được tính theo km từ trong khoảng thời gian từ H_1 (giờ) đến H_2 (giờ). Một phương tiện bị kiểm soát là nếu trong khoảng thời gian này, tồn tại một vị trí nằm trong vùng kiểm soát của trạm quan sát.
   Ví dụ: với mã yêu cầu **5_-122.3879_37.70786_3.02_12_15**, cần xác định số lượng phương tiện có lộ trình đi qua nằm trong hình tròn tâm A (-122.3879, 37.70786), bán kính 3.02 (km) từ lúc 12 giờ đến lúc 15 giờ.

   Kết quả của yêu cầu này là một số nguyên thể hiện số lượng phương tiện thỏa mãn yêu cầu.

5. **Xác định số lần phương tiện đi vào vùng kiểm soát của trạm quan sát**
   Mã yêu cầu: 5_X_A_{long}_A_{lat}_R. Xác định số lần mà phương tiện mang số hiệu X đi vào trong vùng kiểm soát bán kính R của làng du kích có tọa độ (A_{long}, A_{lat}).
   Kết quả của mã yêu cầu này là số lần mà phương tiện trên đi vào vùng kiểm soát của trạm quan sát.

6. **Dự đoán quá tải trạm quan sát**
   Mã yêu cầu: **6**_A_{long}_A_{lat}_**M**_**hhmm**
   Trong đó (A_{long}, A_{lat}) là tọa độ của trạm A, số lượng phương tiện đang trong trạm là **M**, **hhmm** là thời điểm phương tiện vào trạm A, được tính chính xác đến phút.

   Nếu trong khoảng thời gian từ trước thời điểm phương tiện vào trạm 15 phút đến lúc vào trạm, trong bán kính 2km tính từ trạm, số lượng phương tiện có mặt trong bán kính đó (được xác định là có ít nhất một vị trí xuất hiện trong bán kính nói trên) nhỏ hơn số lượng phương tiện đang có trong trạm thì toàn bộ các phương tiện được phép vào trạm.

   Ngược lại, trong thời gian đó, số lượng phương tiện lớn hơn hoặc bằng số lượng phương tiện trong trạm thì chỉ có những phương tiện có khoảng cách tới trạm dưới 500m mới được vào trạm, số còn lại không được đi vào trạm. Nếu số phương tiện có khoảng cách tới trạm là 300m chiếm hơn 0.75M thì toàn bộ các phương tiện không được vào trạm do quá tải.

   Kết quả của yêu cầu: <**In**> - <**Out**>
   Trong đó <**In**> là danh sách các phương tiện được vào trạm, <**Out**> là danh sách các phương tiện không được vào trạm. Lưu ý thứ tự danh sách được sắp xếp theo ID.

7. **Dự báo ùn tắc khi phương tiện ra khỏi trạm**
   Mã yêu cầu: **7**_A_{long}_A_{lat}_**M**_**R**_**hhmm**
   Trong đó (A_{long}, A_{lat}) là tọa độ của trạm quan sát A, số lượng phương tiện trong trạm là M, bán kính quan sát là R, hhmm là thời điểm phương tiện từ trạm A đi ra, được tính chính xác đến từng phút.

   Nếu số lượng phương tiện xuất hiện trong bán kính 500m tính từ trạm trong khoảng thời gian từ **hhmm** đến sau đó 30 phút nhỏ hơn 0.7M thì không xảy ra ùn tắc, ngược lại thì sẽ xảy ra ùm tắc. Trong trường hợp xảy ra ùm tắc, 75% những phương tiện (ưu tiên các phương tiện có khoảng cách xa hơn) có khoảng cách nằm từ 2km đến 1km tính từ trạm sẽ không mắc kẹt.

   Kết quả của yêu cầu: <**S**> - <**NS**>
   Trong đó <**S**> là danh sách các phương tiện bị kẹt (-1 nếu không có phương tiện nào kẹt), <**NS**> là danh sách các phương tiện không bị kẹt. Lưu ý thứ tự danh sách được sắp xếp theo ID.

8. **Phương tiện gặp trục trặc**
   Mã yêu cầu: **8**_A_{long}_A_{lat}_**R**_**hhmm**
   Trong một số tình huống đặc biệt, ví dụ như thiên tai, tai nạn, v.v các phương tiện có thể không hoạt động theo lộ trình dự kiến được. GIả sử rằng có cơn bão với tâm (A_{long}, A_{lat}) với bán kính **R** xuất hiện tại thời điểm **hhmm** với cường độ đủ mạnh có thể làm hỏng phương tiện. Người ta cần dự đoán các phương tiện có lộ trình đi vào vùng bão gây thiệt hại để cảnh báo sớm. Khi gặp yêu cầu này, xuất ra danh sách các phương tiện rơi vào vùng bị bão ảnh hưởng. Hệ quả là những phương tiện này sẽ bị đưa ra khỏi danh sách xử lý vì lộ trình sẽ bị huỷ.

9. **Phục hồi lộ trình**
   Mã yêu cầu: **9_**A_{long}**_**A_{lat}**_R_hhmm**
   Trong trường hợp cảnh báo trong yêu cầu số 8 không thể xảy ra như dự báo, lộ trình của phương tiện sẽ được phục hồi. Chú ý toạ độ và vùng ảnh hưởng của yêu cầu này không nhất thiết trùng khớp với yêu cầu số 8.

4 Hiện thực chương trình

Sinh viên được cung cấp các tập tin sau đây:

- **main.cpp**: mã nguồn chính của chương trình.
- **dsaLib.h**: tập tin chứa định nghĩa thư viện về các cấu trúc dữ liệu cần thiết.
- **dbLib.h**: tập tin header chứa prototype cần thiết để quản lý dữ liệu đầu vào.
- **dbLib.cpp**: mã nguồn hiện thực các chức năng quản lý dữ liệu đầu vào.
- **requestLib.h**: tập tin header chứa các prototype cần thiết để quản lý các yêu cầu.
- **requestLib.cpp**: mã nguồn hiện thực các hàm xử lý yêu cầu.

Sinh viên được cho các hàm sau:

- **distanceEarth**: tính khoảng cách giữa 2 điểm trên mặt đất với toạ độ (latitude, longitude) cho trước.

Sinh viên được yêu cầu hiện thực các hàm sau:

- **loadRequests**: đọc thông tin các yêu cầu.
- **loadVDB**: đọc file và lấy dữ liệu vào danh sách.

Đối với tập tin thư viện **dsaLib.h**, sinh viên được cung cấp sẵn định nghĩa cơ bản. Ngoài ra, để build được chương trình hoàn chỉnh, sinh viên cần hoàn thiện các hàm còn lại trong danh sách.

processData.cpp: Sinh viên hiện thực việc xử lý các yêu cầu thông qua hàm processRequest. Không được thay đổi prototype của hàm này.

Sinh viên không được sử dụng các thư viện nào khác ngoài các thư viện đã được dùng trong code mẫu.

5 Cách dịch và thực thi chương trình

Sinh viên thực hiện build bằng lệnh make từ command line trên Linux và chạy file a02. Cú pháp chạy chương trình trên Linux như sau:

./a02 requests.txt data.csv

Trong đó:

- **requests.txt** là tập tin chứa danh sách các yêu cầu, cách nhau bởi khoảng trắng hoặc ký tự xuống dòng.
- **data.csv** là tập tin chứa dữ liệu của phương tiện, sinh viên có thể thử nghiệm với các tập tin dữ liệu khác nhau.

Đối với các bạn dùng Visual Studio trên Windows, các bạn có thể tạo một project và thêm các tập tin mã nguồn vào. Nếu không bạn có thể sử dụng hệ thống CMake để tự sinh project VS cho mình.

6 Nộp bài và xử lý gian lận

Khi nộp bài, sinh viên sử dụng account đã được cấp phát trên hệ thống Sakai để nộp bài qua mạng. Sinh viên nộp một tập tin nén duy nhất chứa tất cả các tập tin mã nguồn.

Có tất cả 100 test cases, điểm Bài tập lớn sẽ là số test cases chạy đúng chia cho 10.

Thời hạn nộp bài:

- Sẽ thông báo chi tiết trên hệ thống.
- Sinh viên phải dùng tài khoản trên hệ thống Sakai để nộp bài. KHÔNG nhận bài được gửi qua mail hoặc bất kỳ hình thức nào khác. Bài nộp trễ sẽ KHÔNG được nhận.

Bài tập lớn phải được sinh viên TỰ LÀM. Sinh viên sẽ bị coi là gian lận nếu:

1. Có sự giống nhau bất thường giữa mã nguồn của các bài nộp. Trong trường hợp này, TẤT CẢ các bài nộp đều bị coi là gian lận. Do vậy sinh viên phải bảo vệ mã nguồn bài tập lớn của mình.
2. Sinh viên không hiểu mã nguồn do chính mình viết, trừ những phần mã được cung cấp sẵn trong chương trình khởi tạo. Sinh viên có thể tham khảo từ bất kỳ nguồn tài liệu nào, tuy nhiên phải đảm bảo rằng mình hiểu rõ ý nghĩa của tất cả những dòng lệnh mà mình viết. Trong trường hợp không hiểu rõ mã nguồn của nơi mình tham khảo, sinh viên được đặc biệt cảnh báo là KHÔNG ĐƯỢC sử dụng mã nguồn này; thay vào đó nên sử dụng những gì đã được học để viết chương trình.

Trong trường hợp bị kết luận là gian lận, sinh viên sẽ bị điểm 0 cho toàn bộ môn học (không chỉ bài tập lớn). KHÔNG CHẤP NHẬN BẤT KỲ GIẢI THÍCH NÀO VÀ KHÔNG CÓ BẤT KỲ NGOẠI LỆ NÀO!

Sau mỗi bài tập lớn được nộp, sẽ có một số sinh viên được gọi phỏng vấn ngẫu nhiên để chứng minh rằng bài tập lớn vừa được nộp là do chính mình làm.