XÂY DỰNG MÔ HÌNH GIÁ VẬN TẢI
ÁP DỤNG THUẬT TOÁN TRUNG BÌNH ĐỘNG PHÁT HIỆN KHOẢNG CÁCH
(MOVING AVERAGE DISTANCE ALGORITHM)

- *Kiến thức được tổng hợp từ internet với sự hỗ trợ của AI – ChatGPT (mode: deep research)*
- *Kết quả từ nghiên cứu Capstone Project – chương trình Master of Science in Supply Chain & Logistics Management (Ths Nguyễn Minh Ngọc)*
- *ERX Vietnam phát hành*

A - Thuật toán trung bình động phát hiện khoảng cách (Moving Average Distance Algorithm)

1. **Phương pháp Moving Average trong phát hiện pattern:**

   Thuật toán sử dụng **trung bình động (moving average)** để phát hiện sự thay đổi xu hướng giá theo khoảng cách. Cụ thể, với mỗi tuyến vận chuyển và loại xe, thuật toán duyệt qua các điểm dữ liệu được sắp xếp theo cự ly (km) tăng dần. Tại mỗi điểm, thuật toán tính **giá trung bình động** của **giá cước hiện tại** (ví dụ giá cước của nhà cung cấp tham chiếu) trên tất cả các điểm trước đó trong cùng nhóm. Sau đó, tính **độ lệch phần trăm của giá hiện tại so với giá trung bình này**. Công thức chung cho chỉ số này là: **(giá hiện tại – giá trung bình) / giá trung bình**, thường biểu diễn dưới dạng phần trăm (Distance From Moving Average | ChartSchool | StockCharts.com). Chỉ số "khoảng cách tới trung bình" này cho biết giá hiện tại đang cao hơn hay thấp

hơn bao nhiêu so với mức giá trung bình trước đó (Distance From Moving Average | ChartSchool | StockCharts.com). Nhờ biểu diễn dưới dạng %, thuật toán so sánh được mức dao động giá ở các quy mô khác nhau một cách công bằng (Distance From Moving Average | ChartSchool | StockCharts.com).

2. **Xác định khoảng cách và mức giá phù hợp:**

Thuật toán sẽ so sánh **độ lệch phần trăm** vừa tính với một ngưỡng cho trước (tham ví dụ 5% hoặc 10%). Nếu độ lệch vượt quá ngưỡng này, nghĩa là **giá cước tại cự ly đó đã tăng/giảm đột biến so với xu hướng trước**. Khi đó, thuật toán nhận định rằng **một "pattern" mới đã xuất hiện** – tức là biểu phí thay đổi đáng kể – và tiến hành **tách khoảng km mới**. Cụ thể, thuật toán kết thúc "khoảng" cự ly cũ tại điểm trước đó và bắt đầu một **khoảng km mới từ cự ly hiện tại**. Ngược lại, nếu giá hiện tại chỉ chênh lệch nhỏ (dưới ngưỡng) so với trung bình, thì được coi là vẫn **thuộc cùng một khoảng giá** ổn định, và thuật toán tiếp tục mở rộng khoảng km hiện tại để bao trùm điểm dữ liệu này. Như vậy, các **"khoảng km" (segment)** được xác định hoàn toàn dựa trên việc giá cước có **thay đổi đột biến** hay không. Phương pháp trung bình động giúp làm trơn các biến động nhỏ và chỉ ra **khi nào một sự thay đổi đủ lớn xảy ra, báo hiệu xu hướng mới** (Distance From Moving Average | ChartSchool | StockCharts.com).

3. **Tác động lên dữ liệu giá cước logistics:**

Kết quả của thuật toán là phân chia dải dữ liệu cự ly vận chuyển thành các **khoảng cự ly** liên tục (ví dụ: "0-50 km", "51-100 km", v.v. tuỳ theo dữ liệu) kèm mức giá đặc trưng cho mỗi khoảng. Thuật toán gán nhãn mỗi bản ghi dữ liệu với số hiệu **"Khoảng"** tương ứng và tính toán **"Price Range"** – khoảng cự ly cụ thể (từ km nhỏ nhất đến km lớn nhất) cho nhóm đó. Đồng thời, thuật toán có thể tính **mức giá đại diện** cho mỗi khoảng, chẳng hạn **giá trung vị trên mỗi km của từng nhà cung cấp**

trong khoảng đó (như code đã tính toán). Nhờ vậy, dữ liệu giá cước vốn hỗn hợp được **nhóm lại có trật tự**, giúp các nhà quản lý dễ dàng **quan sát biểu phí theo từng khoảng cách**. Chẳng hạn, có thể phát hiện rằng **đến một cự ly nhất định, đơn giá vận chuyển tăng lên** – điều này hàm ý một ngưỡng tính giá (pricing tier) trong chính sách của nhà vận chuyển. Việc phân tích như vậy giúp doanh nghiệp logistics **hiểu rõ cấu trúc biểu phí** hiện tại và có thể điều chỉnh chiến lược giá. Ngoài ra, do mỗi khoảng cự ly có thể đi kèm **mức giá trung bình/trung vị**, doanh nghiệp dễ dàng **so sánh giá giữa các nhà cung cấp** cho cùng khoảng cách, nhận biết nhà cung cấp nào có giá cạnh tranh hơn ở từng khúc thị trường. Tóm lại, thuật toán trung bình động không chỉ phát hiện tự động các **điểm gãy** trong biểu phí theo km mà còn **chuẩn hoá dữ liệu** thành dạng bậc thang dễ phân tích, giúp tối ưu quyết định về giá cước và lựa chọn nhà cung cấp.

B - Triển khai ứng dụng trên nền tảng Google App Engine

1. **Chuyển đổi ứng dụng Flask sang môi trường cloud:**

   Để đưa ứng dụng Flask hiện tại lên Google App Engine, trước tiên cần đóng gói ứng dụng dưới dạng một **dịch vụ web WSGI** tương thích. Điều này bao gồm việc đảm bảo file Python khởi chạy (ví dụ `app.py` hoặc `main.py`) tạo ra một đối tượng `app` Flask. Sau đó, tạo file cấu hình `app.yaml` mô tả môi trường triển khai (chọn runtime Python phù hợp, ví dụ `python3.9`, và cấu hình lượng tài nguyên, auto-scaling...). Ứng dụng Flask có thể được triển khai trên **App Engine Standard** để tận dụng miễn phí và tự động mở rộng, hoặc **App Engine Flexible** nếu cần tùy chỉnh môi trường nhiều hơn. Việc chuyển đổi này đòi hỏi kiểm tra lại các đường dẫn tệp, cấu trúc thư mục để phù hợp với quy định của App Engine (ví dụ: mọi thư viện bên ngoài phải được liệt kê trong `requirements.txt` để Google cài đặt tự động).

2. **Yêu cầu về cấu hình và cơ sở dữ liệu:**

Trên App Engine, hệ thống tập tin cục bộ là không ghi lâu dài (chỉ cho phép ghi tạm thời vào thư mục /tmp). Vì vậy, nếu ứng dụng hiện tại lưu file đầu vào/đầu ra trực tiếp, ta cần chuyển sang sử dụng các dịch vụ lưu trữ cloud. Chẳng hạn, sử dụng **Google Cloud Storage** để lưu file Excel tải lên và kết quả đầu ra (Excel, hình ảnh, file zip) thay vì lưu trên đĩa cục bộ. Về cơ sở dữ liệu, nếu ứng dụng Flask dùng SQLite hoặc file CSV cục bộ, khi lên cloud nên chuyển sang **dịch vụ database quản lý**. Google cung cấp **Cloud SQL** (MySQL/PostgreSQL) cho ứng dụng quan hệ, hoặc **Firestore/Datastore** cho NoSQL. Việc này đảm bảo dữ liệu được lưu trữ bền vững và truy cập được từ nhiều instance ứng dụng. Cần cấu hình **biến môi trường** (environment variables) trong app.yaml để chứa thông tin kết nối DB, khóa API... Thêm nữa, nên bật **Google Secret Manager** cho các thông tin nhạy cảm thay vì ghi cứng trong mã nguồn.

3. **Tích hợp các Google APIs:**

Triển khai trên App Engine cho phép ứng dụng dễ dàng **gọi các API của Google**. Ví dụ, nếu cần tích hợp Google Drive API (để tự động tải file kết quả lên Drive) hoặc Google Maps API (cho tính toán khoảng cách), ta phải kích hoạt những API này trong Google Cloud Console và cấp quyền cho ứng dụng (qua tài khoản dịch vụ). Môi trường App Engine cung cấp sẵn **chứng chỉ** để xác thực nội bộ với các dịch vụ Google khác, hoặc ta có thể sử dụng thư viện OAuth2 của Google cho các API bên ngoài phạm vi dự án. Khi triển khai, cần chắc chắn include các thư viện Google Cloud (như google-cloud-storage, google-api-python-client...) trong tệp yêu cầu. Việc cấu hình đúng quyền IAM cho tài khoản dịch vụ App Engine default cũng rất quan trọng, nhằm cho phép ứng dụng đọc/ghi dữ liệu trên các dịch vụ liên quan một cách an toàn.

4. **Lợi ích khi mở rộng ứng dụng trên App Engine:**

Triển khai ứng dụng lên Google App Engine mang lại nhiều lợi ích về **mở rộng và vận hành**. Trước hết, App Engine có khả năng **tự động scale** – khi lưu lượng người dùng tăng, nền tảng sẽ tự động khởi tạo thêm instance ứng dụng để đáp ứng, và giảm xuống khi lưu lượng giảm. Điều này giúp ứng dụng luôn đáp ứng nhanh mà không cần quản trị thủ công máy chủ. Thêm vào đó, App Engine là dịch vụ **serverless** managed, nên đội ngũ phát triển không phải lo lắng về quản lý máy chủ, cập nhật hệ điều hành hay bảo mật cấp thấp; Google sẽ lo các phần đó. Ứng dụng có thể đạt **tính sẵn sàng cao** bởi App Engine phân phối trên nhiều vùng và tự động cân bằng tải. Việc tích hợp với hệ sinh thái Google Cloud cũng rất thuận tiện – ứng dụng có thể gửi log đến **Stackdriver Logging**, theo dõi hiệu năng qua **Monitoring**, và dễ dàng kết nối các dịch vụ khác (như Cloud SQL, BigQuery) với độ trễ mạng nội bộ thấp. Nhờ tận dụng hạ tầng của Google, ứng dụng có thể **phục vụ người dùng toàn cầu** một cách ổn định, mở rộng thị trường mà không cần đầu tư thêm hạ tầng vật lý.

5. **Thách thức và lưu ý khi triển khai:**

Bên cạnh lợi ích, việc đưa ứng dụng Flask lên App Engine cũng có một số thách thức. Trước hết, cần **kiểm thử kỹ** để đảm bảo ứng dụng chạy được trong **môi trường sandbox** của App Engine Standard – ví dụ như đã đề cập, hạn chế ghi file đòi hỏi thay đổi nhỏ trong mã nguồn. Một thách thức khác là **chi phí**: App Engine có mức miễn phí nhất định, nhưng khi ứng dụng mở rộng và sử dụng nhiều dịch vụ (ví dụ Cloud SQL, API tính phí) thì chi phí có thể tăng. Do đó, nhóm phát triển cần thiết lập phương thức giám sát chi phí và tối ưu (như tắt bớt tính năng không dùng, hoặc thiết lập auto-scaling hợp lý để tránh over-provision). Thêm nữa, quá trình triển khai lên App Engine đòi hỏi làm quen với một số **công cụ CI/CD của Google Cloud** (như gcloud CLI, Cloud Build), nhưng sau khi thiết lập pipeline, việc nâng cấp ứng dụng sẽ được tự động hoá và tin cậy hơn. Cuối cùng, cần quan tâm đến **bảo mật**: mặc dù

App Engine bảo mật phần hạ tầng, lập trình viên vẫn phải đảm bảo an toàn ở tầng ứng dụng (các endpoint Flask) và dữ liệu (mã hóa thông tin nhạy cảm, sử dụng HTTPS, cấu hình CORS đúng...). Nếu vượt qua được những điều chỉnh ban đầu này, ứng dụng sẽ vận hành trơn tru trên cloud và hưởng lợi dài lâu từ kiến trúc linh hoạt mà App Engine mang lại.

C - Mở rộng ứng dụng phát hiện pattern trong quản lý chuỗi cung ứng

Thuật toán phát hiện pattern dựa trên trung bình động có tiềm năng ứng dụng rộng trong quản lý chuỗi cung ứng, không chỉ giới hạn ở phân tích giá cước. Dưới đây là một số hướng mở rộng:

1. **Giám sát xu hướng hiệu suất máy móc:**

   Trong sản xuất và logistics, máy móc thiết bị (như xe nâng, băng chuyền, robot kho) thường có các **chỉ số hiệu suất** hoặc **tình trạng vận hành** được ghi nhận liên tục (ví dụ: nhiệt độ, độ rung, tốc độ, hiệu suất OEE...). Bằng cách áp dụng phương pháp trung bình động, hệ thống có thể **phát hiện sớm xu hướng bất thường** trong các chỉ số này. Ví dụ, nếu **tiêu thụ năng lượng của máy tăng dần vượt quá mức trung bình động** hoặc **hiệu suất giảm quá ngưỡng cho phép**, thuật toán sẽ cảnh báo rằng máy có thể sắp gặp sự cố. Việc **theo dõi xu hướng thiết bị** giúp lên kế hoạch **bảo trì dự phòng (predictive maintenance)** thay vì chờ hỏng hóc. Như một báo cáo ngành thực phẩm đã chỉ ra, việc **giám sát số liệu hiệu suất máy móc theo thời gian thực** cho phép thấy rõ **xu hướng suy giảm hiệu quả thiết bị, thời gian ngừng máy**..., từ đó **dự đoán vấn đề trước khi xảy ra và bảo trì vào lúc ít ảnh hưởng nhất** (A changing digital landscape | Miller Magazine). Điều này nâng cao **độ tin cậy vận hành** và giảm gián đoạn chuỗi cung ứng do sự cố máy móc.

2. **Đánh giá hiệu suất nhà cung cấp:**

   Doanh nghiệp thường theo dõi nhiều **chỉ số KPI của nhà cung cấp** (như **thời gian giao hàng**, **tỷ lệ giao đúng hạn**, **chất lượng hàng hóa**, **giá cả**). Bằng cách thu thập dữ liệu lịch sử các KPI này và áp dụng thuật toán phát hiện pattern, hệ thống có thể **nhận diện xu hướng thay đổi trong hiệu suất của từng nhà cung cấp**. Chẳng hạn, nếu một nhà cung cấp bắt đầu **giao hàng chậm hơn đáng kể so với trung bình trước đây** hoặc **tỷ lệ lỗi sản phẩm tăng đột biến**, đó là **tín hiệu cảnh báo sớm**. Việc phát hiện sớm những **biến động bất thường** cho phép doanh nghiệp làm việc với nhà cung cấp để tìm nguyên nhân và cải thiện, trước khi vấn đề trở nên trầm trọng. Theo các chuyên gia, giải pháp phát hiện bất thường trong dữ liệu hiệu suất nhà cung cấp có thể đóng vai trò như một **hệ thống cảnh báo sớm**, giúp nhận biết vấn đề **trước khi chúng leo thang thành gián đoạn lớn** (Anomaly Detection in Supplier Performance | AI/ML Development Solutions). Ví dụ, **phân tích dữ liệu giao hàng, chất lượng** bằng AI có thể tự động **gắn cờ những nhà cung cấp có hiệu suất giảm sút hoặc dao động bất thường** (Anomaly Detection in Supplier Performance | AI/ML Development Solutions). Nhờ đó, công ty có thể đánh giá lại nhà cung cấp, yêu cầu cải thiện hoặc chuyển nguồn cung ứng nếu cần, góp phần **đảm bảo tính liên tục và ổn định của chuỗi cung ứng**.

3. **Quản lý kho và tối ưu chuỗi cung ứng:**

   Trong hoạt động kho bãi và phân phối, việc nhận biết **pattern trong dữ liệu hoạt động** có giá trị lớn để tối ưu hiệu suất. Một ứng dụng cụ thể là **phân tích xu hướng tồn kho**: Theo dõi mức tồn kho theo thời gian và **phát hiện các mẫu nhu cầu** theo mùa vụ hoặc chu kỳ. Thuật toán trung bình động có thể giúp **làm mượt** dữ liệu bán hàng hằng ngày và **chỉ ra xu hướng thực** (loại bỏ nhiễu), từ đó hỗ trợ dự báo nhu cầu chính xác hơn. Tài liệu cho thấy bằng việc **phân tích dữ liệu bán hàng lịch sử**, doanh nghiệp có thể **phát hiện mẫu hình nhu cầu** và **điều chỉnh mức tồn kho phù hợp**, tránh thiếu hàng hoặc tồn kho dư thừa (Data-Driven Warehousing: Leveraging

Analytics for Smarter Supply Chain Decisions - TVS Supply Chain Solutions). Ngoài ra, pattern detection còn hỗ trợ **tối ưu bố trí kho**: phân tích dữ liệu di chuyển hàng hóa trong kho để tìm mẫu (mặt hàng nào hay được lấy cùng nhau, tuyến đường nào thường di chuyển) nhằm sắp xếp kho hợp lý, giảm thời gian lấy hàng. Trong **vận tải**, các thuật toán tương tự có thể theo dõi **dữ liệu vận chuyển theo thời gian thực** và phát hiện bất thường (như lộ trình giao hàng bị chậm hơn thường lệ) (Machine Learning in the Logistics Industry: 7 Use Cases), từ đó gợi ý tối ưu tuyến đường hoặc điều phối lại phương tiện. Tóm lại, **nhận diện pattern** trong kho vận và phân phối giúp **ra quyết định dựa trên dữ liệu**: từ điều chỉnh tồn kho theo xu hướng nhu cầu, cải thiện **quy trình lấy hàng, đóng gói**, đến tối ưu **lộ trình vận tải** – tất cả nhằm **nâng cao hiệu quả chuỗi cung ứng** và **giảm chi phí hoạt động**.

D - Tham khảo nghiên cứu và case study liên quan

Để bổ sung cơ sở khoa học và thực tiễn cho các nội dung trên, nhóm đã tham khảo một số nguồn nghiên cứu và ví dụ triển khai từ bên ngoài:

1. **Phát hiện pattern trong dữ liệu logistics:**

   Nhiều nghiên cứu gần đây tập trung vào **phát hiện bất thường (anomaly detection)** trong chuỗi cung ứng, coi đó như một cách nhận biết pattern quan trọng. Chẳng hạn, thuật toán machine learning có thể sàng lọc dữ liệu vận chuyển để **phát hiện các mẫu bất thường** (ví dụ: **phí vận chuyển tăng đột biến hoặc lộ trình giao hàng chậm bất thường**) nhanh hơn con người rất nhiều (Anomaly Detection in Logistics: Precise Analytics for Cost Allocation). Các **doanh nghiệp logistics hàng đầu** cũng đang đầu tư mạnh vào công nghệ này – theo một khảo sát, những công ty dẫn đầu **có khả năng đầu tư vào phân tích dữ liệu & phát hiện bất thường cao hơn 50%** so với doanh nghiệp trung bình (Anomaly Detection in Logistics: Precise Analytics for Cost Allocation). Lý do là các công nghệ này giúp họ **dự báo nhu cầu, tối ưu tuyến vận tải, quản lý tồn kho** hiệu quả hơn, cũng như **phát hiện sự cố** trước khi chúng

gây thiệt hại lớn. Thực tế cho thấy ứng dụng AI/ML trong logistics giúp **giảm lỗi giao hàng tới 60% và cắt giảm 40% thời gian chờ xếp dỡ** (theo số liệu báo cáo của ngành) nhờ phát hiện sớm vấn đề và tối ưu quy trình (Anomaly Detection in Logistics: Precise Analytics for Cost Allocation). Những kết quả này khẳng định **giá trị của việc phát hiện pattern/bất thường** trong việc **nâng cao hiệu quả và giảm rủi ro** cho chuỗi cung ứng.

2. **Ứng dụng Moving Average trong chuỗi cung ứng:**

Phương pháp **trung bình động** từ lâu đã được sử dụng trong lĩnh vực chuỗi cung ứng, đặc biệt là trong **dự báo nhu cầu và quản lý tồn kho**. Các tài liệu dự báo cho thấy trung bình động là **một trong những phương pháp đơn giản và thông dụng nhất**: phương pháp này lấy trung bình dữ liệu các khoảng thời gian gần nhất để **dự báo cho giai đoạn kế tiếp**, nhờ đó **làm trơn biến động ngẫu nhiên và nổi bật xu hướng chính** (Supply Chain Forecasting: Methods & Technology | Coupa). Chính nhờ loại bỏ được các nhiễu ngẫu nhiên, moving average giúp doanh nghiệp **nhận ra mô hình nhu cầu thật** – ví dụ phân biệt được xu hướng tăng trưởng đều đặn so với dao động nhất thời. Điều này rất hữu ích để **ổn định việc lập kế hoạch tồn kho và sản xuất ngắn hạn** (Supply Chain Forecasting: Methods & Technology | Coupa), tránh các quyết định dựa trên dao động bất thường. Ngoài dự báo, trung bình động còn được dùng trong **kiểm soát tồn kho** (tính mức tồn kho trung bình để tái đặt hàng), và trong phân tích vận tải (ví dụ tính thời gian giao hàng trung bình để so sánh hiệu suất). Việc thuật toán của chúng tôi áp dụng trung bình động để phát hiện ngưỡng thay đổi giá cước thực chất là một biến thể của ý tưởng trên: dùng trung bình động để **phát hiện khi nào một dữ liệu mới không còn "phù hợp" với lịch sử**, từ đó ra quyết định (dự báo hay phân đoạn dữ liệu).

3. **Case study triển khai hệ thống tương tự trên cloud:**

Triển khai các hệ thống phân tích chuỗi cung ứng và phát hiện pattern trên nền tảng cloud đang trở thành xu hướng chung. **DHL Supply Chain** – công ty logistics hàng

đầu thế giới – là một ví dụ tiêu biểu về ứng dụng cloud và AI. Họ đã **tập trung toàn bộ dữ liệu chuỗi cung ứng lên đám mây** như bước đầu tiên trong chiến lược ứng dụng AI (). Trên nền tảng dữ liệu cloud đó, DHL phát triển các mô hình AI để theo dõi hoạt động kho, vận tải theo thời gian thực và đề xuất tối ưu mỗi ngày. Kết quả giúp DHL **chủ động ứng phó** với gián đoạn, đồng thời **cải thiện hiệu suất vận hành** ở quy mô hàng ngàn kho hàng trên toàn cầu. Bên cạnh đó, các công ty công nghệ cung cấp giải pháp cho logistics cũng tận dụng cloud: chẳng hạn **Intelligent Audit** – một nhà cung cấp giải pháp phân tích vận tải – dùng machine learning trên hạ tầng đám mây để **phát hiện các bất thường về chi phí vận chuyển và thời gian giao hàng** cho khách hàng của họ, giúp tiết kiệm đáng kể chi phí logistics (Anomaly Detection in Logistics: Precise Analytics for Cost Allocation). Những **case study thành công** này cho thấy việc đưa ứng dụng phân tích chuỗi cung ứng lên cloud (như Google Cloud, AWS) mang lại **khả năng mở rộng tài nguyên** để xử lý **dữ liệu lớn** và tích hợp dễ dàng với các dịch vụ AI tiên tiến. Nhờ đó, hệ thống có thể phân tích dữ liệu từ nhiều nguồn trong thời gian thực, **nhận diện pattern nhanh hơn** và hỗ trợ ra quyết định kịp thời trong chuỗi cung ứng – một yếu tố sống còn trong bối cảnh thị trường biến động nhanh.

Tài liệu tham khảo:

Các nội dung trên được tổng hợp từ nhiều nguồn uy tín, bao gồm hướng dẫn kỹ thuật (ví dụ StockCharts về chỉ báo khoảng cách so với trung bình động (Distance From Moving Average | ChartSchool | StockCharts.com) (Distance From Moving Average | ChartSchool | StockCharts.com)), các bài nghiên cứu và blog chuyên ngành logistics (ví dụ bài viết của Intelligent Audit về phân tích chi phí vận tải (Anomaly Detection in Logistics: Precise Analytics for Cost Allocation) (Anomaly Detection in Logistics: Precise Analytics for Cost Allocation)), cũng như case study thực tiễn từ doanh nghiệp lớn (DHL và các đơn vị cung cấp giải pháp phân tích chuỗi cung ứng). Điều này nhằm đảm bảo rằng đề xuất và phân tích trong báo cáo không chỉ dựa trên lý thuyết suông mà còn được ***hậu thuẫn bởi dữ liệu và kinh nghiệm thực tế*** *trong ngành.*