Applied Data Mining

**Part I: Automata-based CER with FlinkCEP**

 

Alevizopoulou Sofia 2022201704002

Avgeros Giannis 2022201704003

Tsiatsios George 2022201704024

Athens 2018

Contents

[Table of figures 2](#_Toc530788034)

[Introduction 3](#_Toc530788035)

[System Architecture 3](#_Toc530788036)

[System Deployment 3](#_Toc530788037)

[Detecting Trajectory Events 4](#_Toc530788038)

[Online Noise Reduction 4](#_Toc530788039)

[1. Out of order events 4](#_Toc530788040)

[2. High acceleration in small period of time 5](#_Toc530788041)

[3. Identical location has sent from the same vessel 5](#_Toc530788042)

[4. Ais messages with same timestamp for the same vessel 5](#_Toc530788043)

[Online Tracking of Moving Vessels 5](#_Toc530788044)

[Grid partitioning 5](#_Toc530788045)

[Trajectory Events and Complex events 6](#_Toc530788046)

[Empirical Evaluation 12](#_Toc530788047)

[Visualization of the detected events 12](#_Toc530788048)

[Running commands 13](#_Toc530788049)

[Running environment 14](#_Toc530788050)

[Pattern Time (window) 14](#_Toc530788051)

[References 15](#_Toc530788052)

# Table of figures

**No table of figures entries found.**

# Introduction

A system for online monitoring marine activity over streaming positions from numerous vessels sailing at sea will be presented in this assignment. The system detects important changes in the evolving trajectory of each vessel across time and after use these trajectory events to recognize complex suspicious events among vessels.

# System Architecture

The online monitoring system is a combination of the following systems: kafka and Apache Flink. Kafka is an open source distributed streaming platform which is used for stream processing and apache Flink is a real time distributed processing engine for stateful computations.

# System Deployment

More specifically this system detects trajectory and complex events from the ais messages. Ais messages are stored in a psql database. A python script has been written in order to fetch these messages from database and forward them at a kafka producer. For that reason ais messages are inserted at a kafka topic “DEMOCP” which is finally consumed form the kafka consumer.

The system consists of 2 different components. The 1st one is used to detect trajectory events and some complex events that refer to one vessel and the 2nd to detect complex event for more than one vessels based on the trajectories events that have already detected at the previous step. The 1st project “flinkcep” take as input what is written at topic DEMOCP (the ais messages fetched from database). The output of this project is forwarded at other kafka topics: DEMOCP2, DEMO\_CO, DEMO\_ACC,etc that are given as input at the kafka producer of the 2nd project “cep\_flinkcep”. The outcomes of both projects are written in txt files. Each row of the txt files contains information for the detected event.

* Vessel randevouz
* CoTravel vessel
* Fast Aprroach
* Illegal fishing

***AIS DATA***

Kafka

***Trajectory Events***

***Complex Events***

* Speed change
* Turn change
* Gap time in messages
* Remove Noise events

Write results at txt files

Kafka

Fkink

Fkink

# Detecting Trajectory Events

Trajectory Detection is the main module of the implementation since the Complex Event Recognition module is built on top of it.

## Online Noise Reduction

AIS dataset as a real world dataset contains data that are noise and as a result inadequate for monitoring. In order to have better and more accurate results in the monitoring system we should remove these noisy messages. In this implementation critical conditions have been added at the flinkcep patterns in order to check for noise data. These conditions are checked firstly in the patterns in order to improve the performance of the system. No inadequate data will be forwarded. We have added some cases of noise detection in our system. As noisy can be assumed:

### Out of order events

As it is known, transmission delays may frequently occur between the original message and its arrival. Successive positional messages from a single vessel may often arrive intermingled at a distorted order. In our case the ais messages are stored in database so we assume that there is no delay between the original message and when this message arrives on the system. If this system was in a real world at which we would have delays, we should take care of messages that arrived out of order. We decided to exclude these kinds of messages by adding condition that checks the timestamp of each arrived messages. If the timestamp of the ais message for a specific vessel is smaller than the timestamp of the previous ais message, then this ais message will be ignored.

### High acceleration in small period of time

As already said sometimes messages may arrive out of sequence and as a result a vessel seems to change a lot its speed in a small period. These high accelerations don’t take in mind as suspicious accelerations, and they considered noise. System detect suspicious acceleration when the speed change rapidly in a logical time period.

### Identical location has sent from the same vessel

This is an erroneous ais message as even though the vessel is in anchor its GPS sign will change a little bit. This ais message is noise and by removing it the system will have better performance and more accurate results. It isn’t implemented.

### Ais messages with same timestamp for the same vessel

This is also an erroneous ais message. Having the same timestamp the velocity of the vessel can’t be calculated and the message cant be used from the system effectively. It isn’t implemented.

## Online Tracking of Moving Vessels

### Grid partitioning

In order to use the coordinates of a vessel and check how far is from an area or how far is from other vessels *geohash*, a grid portioning method has been used. Geohash is a geocoding system based on a hierarchical spatial data structure which subdivides space into buckets of grid (github.com/davidmoten/geo).

Each cell is labeled using a geohash which is of user-definable precision:

● High precision geohash have a long string length and represent cells that cover only a small area.

● Low precision geohash have a short string length and represent cells that each cover a large area.

GeoHash, can have a choice of precision between 1 and 12. As a consequence of the gradual precision degradation, nearby places will often present similar prefixes. The longer a shared prefix is, the closer the two places are. In the current implementation the below cell dimensions have been used.

|  |  |
| --- | --- |
| **Precision** | **Cell dimension** |
| 4 | 39,1 km x 19,5 km |
| 5 | 4,9 km x 4,9 km |
| 6 | 1,2 km x 60,4m |
| 7 | 159,2m x 152.4m |

### Trajectory Events and Complex events

At this phase system deduce instantaneousevents by examining the trace of each vessel alone. This system consumes a stream of AIS tracking messages from vessels and continuously detects important features that characterize their movement.

A sample ais message is:

*"lat":2.541122,"lon":3.90484,"mmsi":14,"status":7,"speed":30,"turn":,"heading":36,"course":13,1, "t": 1443650402*

Trajectory detection makes a stream of critical events like: acceleration\_start, acceleration\_end that will be used from the complex event recognition system. Each such event is accompanied by the coordinates and some other characteristics of each corresponding vessel. At each class of trajectory events there is a message serializer class which serialize as a stream the output of the detected event. These streams will be used form the 2nd complex event recognition system or will be written at txt files if the suspicious event has been detected and there is need to procced at the 2nd monitoring system.

At this phase the module is based on the output of Trajectory Detection Module, and its goal is to detect events which are more complex than the movement ais events.

Below are described the events we have implemented.

#### Suspicious Acceleration and fast approach

##### Trajectory event

First system detects vessels which has high acceleration at the open sea, this is a trajectory event. Acceleration is the temporal rate of change in velocity (speed) of a vessel. Suspicious accelerated considered each vessel that has accelerated more than 25%, more specifically speed has changed 20 KNOTS at less than 80 secs. [1]

a=Δυ/Δt🡺0.25=20/Δt🡪Δt=80 sec

In order to check if the vessel is not near a port the coordinates of the ports of Brittany have been used (the dataset has to do with vessels near the area of France). A csv file is read and after is checked if the geohash of the vessel equals to the geohash of the ports. We have used precision 6, results in a grid 1,2 km x 60,4m. We want to verify that vessel is far away form a port for tagging its acceleration as suspicious. High acceleration near ports is normal. This pattern characterized by the critical events “acceleration\_start” – “speed\_change”-“acceleration\_end”.

##### Complex event

The next step is to check if the corresponding vessel changes its speed at open sea (not near ports), the new speed is above 20 knots in a small time period, and there is at least one other nearby vessel toward. In other words, fast Approach is a dangerous situation may arise when a vessel is rapidly moving towards some other vessel(s). We used again the geohash grid with precision 6 (1,2 km x 60,4m) to check if the geohash of the vessel equals to the geohash of other vessels. At this point will be checked all the vessels that have sent ais messages until that moment. This precision used as we would like to inform on time the other vessels that they are in dangerous.

At that point the system will check just the events that have been characterized by the tag “acceleration\_start” – “speed\_change” - “acceleration\_end” in order to detect the fast approach event.

At the end a txt file will be created and will contained all the vessels that fast forward to other vessels as long as the vessels that are in danger.

The format of the messages is:

*MMSI , AccelerationStart , AccelerationEnd , , GeoHash , t\_start, t\_end, lat , lon*

For that event we used just the 1st project , there was no need to forward the results to the 2nd project as all the logic of fast forward events included at one pattern.

The formalization for the above event is:

* “acceleration\_start”
* speed\_change":

Vessel has: acceleration>25%, Δu>20 knots, Δt>20 sec, not near any port of Brittany (grid: 1,2 km x 60,4m))

* “acceleration\_end”:

For all “acceleration\_start” check if

near other vessel (grid: 1,2 km x 60,4m)) - check all the vessels that have sent ais message until that moment

WHAT ABOUT SHIP POU EXEI MEINEI NULL???

#### Gaps in communication

##### Trajectory event

System detects vessels which has communication gaps on sending ais messages, this is a trajectory event. A gap can be considered an absence of ais messages form a vessel for a specific time period. In our approach this time is 600 secs. [1]. So, if the system won’t receive an ais message for a vessel between 600 secs then this vessel tagged as suspicious. This pattern characterized by the critical events “gap\_start” – “gap\_end”.

At the end, all the detected events will be inserted into a kafka topic and will be given as input stream at the 2nd online monitoring system “cep\_flincep” to detect complex events.

The format of the messages which kafka producer will receive is:

*MMSI,GapStart,GapEnd,GeoHash*

The formalization for the above event is:

* “gap\_start”

Δt>600 sec (600 sec with no ais message)

* “gap\_end”

#### Suspicious fishing activity

##### Trajectory event

First system detects vessels which has sharp changes in their headings. As sharp can be considered Δθ>60, this is a trajectory event. Vessels normally don’t have big changes in heading while they are moving [1]

The next trajectory event that is detected is gaps in the communication. A gap can be considered an absence of ais messages form a vessel for a specific time period. In our approach this time is 600 secs. [1]. So, if the system won’t receive an ais message for a vessel between 600 secs then this vessel tagged as suspicious. This pattern characterized by the critical events “gap\_start” – “gap\_end”.

##### Complex event

The next step is to check if the vessel continues the zing-zang route, means continuously changes in the heading. The combination of continuous changes of the heading with the gap in communication can be considered as a sign alert for illegal fishing. The captain checks the area and after closes its GPS in order to do the illegal fishing. At that point the system will check just the events that have been characterized by the tag “gap\_start” – “gap\_end” in order to detect some more changes in heading of the vessels. These extra changes in heading will detect that this vessel may does illegal fishing and the port authority has to take care about it.

De tha mporousame na baloume Kleene star sto apo panw pattern?/?

At the end a txt file will be created and will contained all the vessels that may do illegal fishing.

The format of the messages is:

*Mmsi,gapStart,gapStartLot,gapStartLat,gapEnd,gapEndLot,gapEndLat,geoHash*

For that event we used just the 1st project , there was no need to forward the results to the 2nd project as all the logic of fast forward events included at one pattern.

The formalization for the above event is:

* “start”
* “gap\_start”

Δθ>60 (change in heading)

* “gap\_end”

Δt>600 sec (600 sec with no ais message)

* “change in heading again”

Δθ>60 (change in heading)

#### CoTravel for 2 vessels

##### Trajectory event

Na mpei to check me mmsi prwto , einai sth noise periptwsh

First system detects some trajectories events. Noise events like ais messages from the same vessel will be removed. These messages are unusable in this case as we would like to find 2 different vessels that cotravel in the sea.

The pattern, checks that the ais messages form 2 different vessels are between a time period of 60 secs. With that way pattern will detect events that refer at the same time. Next, will be checked if the ships are on route and not paused. The speed of the vessel should be bigger than 1KNOT (1 KNOT is the minimum speed of a vessel that isn’t in anchor). [1] At the end, the geohash of the 2 vessels will be checked in order to see how close are they. The precision of geohash is 7 (grid: 159,2m x 152.4m ).

This pattern characterized by the critical events “vessel\_1” – “vessel\_2”.

At the end, all the detected events will be inserted into a kafka topic and will be given as input stream at the 2nd online monitoring system “cep\_flincep” to detect comlplex events.

The format of the messages which kafka producer will receive is:

*MMSI\_1,Lon\_1,lat\_1,MMSI\_2,Lon\_2,lat\_2,time*

Gt exeis valeis 2 if sto serializer????

The formalization for the above event is:

* “vessel\_1”

Vessel1 speed>1KNOT ,vessel2 speed>1KNOT, vessel1!=vessel2, vessels near each other (grid: 159,2m x 152.4m)

* “vessel\_2”

#### Vessel rendevouz

##### Complex event

At this phase the second module “cep\_flinkcep” cosnumes a kafka topic which contains the trajectory events -gaps in communications- that detected from the 1st module “flikcep”.

??????na poume gia to window twn 600 secs

At the end a txt file will be created and will contained all the vessels that may had rendevouz.

The format of the messages is:

*Vessel\_1, Vessel\_2 , Gap\_End\_1 , Gap\_End\_2, GeoHash*

The formalization for the above event is:

* “vessel\_1”
* “vessel\_2”

#### CoTravel vessel

##### Complex event

At this phase the second module “cep\_flinkcep” cosnumes a kafka topic which contains the trajectory events -coTravel for 2 vessels- that detected from the 1st module “flikcep”.

??????na poume gia to window twn 600 secs

At the end a txt file will be created and will contained all the vessels that coTravel.

The format of the messages is:

*Vessel\_1, Vessel\_2 , timestamp*

The formalization for the above event is:

* “msg\_1”
* “msg\_2”

# Empirical Evaluation

Changing the values on mobility tracking parameters eg: minimum speed, maximum acceleration etc we can see that the number of events that are detected changes dramatically. Doing some tests we conclude at the below results. We decided to use as parameters values these that were giving us more realistic results.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

# Visualization of the detected events

Except from the txt files that are created for each complex event (contains information about the vessles and that take part in a complex event) csv files are also created with the same information. These files are used in QGIS. The below pictures come from QGIS .

# Running commands

* ***$ flink\_1.6.2:bin/start-cluster.sh***
* start the cluster of flink
* Check that server is running on http://localhost:8081/#/overview
* ***$ kafka2.2:bin/zookeeper-server-start.sh config/zookeeper.properties***
* ***$ kafka2.2:bin/kafka-server-start.sh config/server.properties***
* Start the kafka and zookeeper servers
* ***$ kafka2.2:bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 partitions 1 --topic DEMOCP***
* ***$kafka2.2:bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic DEMOCP2***
* ***$kafka2.2:bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic DEMOCP\_CO***
* ***$kafka2.2:bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic DEMOCP\_ACC***
  + Create the topics on kafka
* ***sudo /home/cer/Downloads/flink-1.6.2/bin/flink run /home/cer/Desktop/cer\_2/flinkcep/cep\_flinkcep/target/flinkicu\_cep-1.0-jar-with-dependencies.jar --topic\_gap DEMOCP2 --bootstrap.servers localhost:9092 --zookeeper.connect localhost:2181 --topic\_co DEMOCP\_CO***
* ***sudo /home/cer/Downloads/flink-1.6.2/bin/flink run /home/cer/Desktop/cer\_2/flinkcep/flinkcep/target/flinkicu-1.0-jar-with-dependencies.jar --topic DEMOCP --bootstrap.servers localhost:9092 --zookeeper.connect localhost:2181 --out /home/cer/Desktop/out.txt --topic\_output\_acc DEMO\_ACC --topic\_output\_gap DEMOCP2 --topic\_output\_co DEMOCP\_CO***
* ***$ Kafka2.2:bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 topic DEMOCP***
* You can check the events that detected at each job at this url. http://localhost:8081/#/overview
* You can see what is send at each topic running the consumer of each topic. For example for topic DEMOCP, running the consumer you will see the ais messages
* ***./ais.py DEMOCP --topic\_output DEMOCP2***
* Run the python script in order to full fill the topic that contains all the ais messages and 1st module starts receiving ais messages
* Producer is inside the project at the path ‘flinkcep/producer’

Running the script suspicious events will start to be detected. All of these will be written at the desktop as .txt files. There will be a .txt file for each trajectory and suspicious event.

# Running environment

|  |  |
| --- | --- |
| OS | PRETTY\_NAME="Ubuntu 18.04.1 LTS"  VERSION\_ID="18.04" |
| Postgres | postgres=# SELECT version();  PostgreSQL 10.5 (Ubuntu 10.5-0ubuntu0.18.04) on x86\_64-pc-linux-gnu, compiled by gcc (Ubuntu 7.3.0-16ubuntu3) 7.3.0, 64-bit |
| QGIS | 2.18.0 |
| Python | Python 2.7.15 |
| Kafka | kafka\_2.11-1.0.0 |
| Flink | flink-1.6.2 |
| Scala | Scala code runner version 2.11.12 -- Copyright 2002-2017, LAMP/E |

## Pattern Time (window)

Na exhghsouem ta followedBy, followedByANy kok , gnk to selection strategy tou kathe pattern

TO DO:

AN EXOUME BALEI NA TSEKARE THN PERIOXH POU PSAREVEI, H AN OTAN PSAREVEI TI STATUS EXEI KAI NA VGALOUEM POSOSTA.

## References

[1] Online Event Recognition from Moving Vessel Trajectories Kostas Patroumpas · Elias Alevizos · Alexander Artikis · Marios Vodas · Nikos Pelekis · Yannis Theodoridis.