Spatio-Temporal Hotspot Detection in Microsoft Azure

A Project Report Submitted in Partial Fulfillment of Requirements
for the Degree of

Bachelor of Technology

by

Rishabh Jain(2019CSB1286)
Vishawam Datta (2019CSB1305)

Department of Computer Science & Engineering

Indian Institute of Technology Ropar

Rupnagar 140001, India

May 2023

Abstract

The objective of this project was to work on the different aspects of periodic spatio-temporal hotspots in Microsoft Azure Services. Without millions of users using Azure APIs, it becomes necessary to track usage spikes and perform necessary configuration changes in the servers to handle large number of requests. We thus implemented algorithms to process usage data from Azure services to return periodic usage spikes.

In this report, we discuss the design and implementation of Spatio-temporal hotspot detection in Microsoft Azure Services.

Acknowledgements

We would like to acknowledge Dr. Venkata M.V. Gunturi, B.Tech project supervisor for his guidance and support. Without his invaluable guidance, support, encouragement and engagement none of this would've been possible. We would also like to thank Mr. Rakesh Rajeev who assisted and helped us throughout the project.

Honor Code

We certify that we have properly cited any material taken from other sources and have obtained permission for any copyrighted material included in this report. We take full responsibility for any code submitted as part of this project and the contents of this report.

Rishabh Jain (2019csb1286)
Vishawam Datta (2019csb1305)

Certificate

It is certified that the B. Tech. project "Periodic Spatio-Temporal Hotspot Detection in Microsoft Azure Traffic Data " has been done by Rishabh Jain(2019CSB1286), Vishawam Datta (2019CSB1305) under my supervision. This report has been submitted towards partial fulfillment of B. Tech. project requirements.

Viswanath Gunturi
Project Supervisor
Department of Computer Science & Engineering
Indian Institute of Technology Ropar
Rupnagar-140001

Contents

List of Figures

Chapter 1

Introduction

The objective of this project was to work on the different aspects of periodic spatio-temporal hotspots in Microsoft Azure Services. Without millions of users using Azure APIs, it becomes necessary to track usage spikes and perform necessary configuration changes in the servers to handle large number of requests. We thus implemented algorithms to process usage data from Azure services to return periodic usage spikes.

The first part of this report involves the implementation and experiments based on the Mono-Chromatic Spatio-Temporal Hotspot algorithm, proposed by Gunturi et al. [2022]. We add several modules to the same, modify the framework, and run experiments to verify its performance.

The second part involves design, implementation and experiments based on the novel Multi-Chromatic Spatio-Temporal Hotspot algorithm. We discuss the problem formulation, the design and implementation of the algorithm.

Chapter 2

Monochromatic Spatio-temporal Hotspot Detection

2.1 Problem Formulation

Given as input a collection of events and their corresponding occurrence time stamps and spatial locations, our task is to identify regions in space as well as time which exhibit high intensity of events which repeat over a period of time.

The basic input of this problem is an array with the following attributes: - (city name, total requests, timestamps, x_coord, y_coord). The aim is to return pairs of (start time, end time, x_coord , y_coord), for each significant and recurring periodic pattern at ((x,y) or a city) that repeats itself every day between (start time end time).

2.1.1 Algorithm Outline

We refer Gunturi et al. [2022] as the inspiration for our algorithm. The first step is to identify the possible spatial points to be considered together which may constitute a periodic pattern. It is important to come up with a definition of a neighborhood/closeness of a particular spatial data point to another one in order to narrow down our search space.

2.1.2 Terminologies

Gunturi et al. [2022] establishes closeness of spatial points as follows :-

1. **Star Neighborhood**: Given a location in space x and a radius parameter r, the star neighborhood of x is the set of all locations y such that the distance between x and y is less than equal to r.

2. **Spatial Connected Region (SCR)**: It is a collection of two or more star neighborhoods. To create these regions, each star neighborhood is treated as a separate node, connected together using edges. These edges are defined based on the distance between the convex hulls that surround each star connected component, and they're only created if the distance is less than a given threshold ϵ.

Now once the relevant sets of spatial data points are identified, i.e, all star neighborhoods and the spatial connected regions, these are considered together to find whether they are part of a pattern.

2.1.3 Proposed changes in framework

The inherent problem in considering such spatial data point sets together to find patterns is that we are defining closeness of two spatial points by only considering distance as a parameter. However we are not considering any network related closeness of the said spatial data point sets.

We require a network related parameter to define closeness of two spatial data points, connecting distinct data points by edges as given below-

Two spatial data points g1 and g2 are connected via an edge if both these conditions hold:

1. The geodetic distance between g1 and g2 is less than radius r.

2. The number of ISP providers common to both g1 and g2 is greater than a threshold θ.

Now once the spatial locations to be considered are identified, all possible temporal windows are iterated over along with all possible periodicity values to identify recurring patterns. Now the task reduces to: given the temporal window (start time , end time) , the location ((x,y) or area) and a time period, we need to classify whether this is a sufficiently strong pattern or not. For each such candidate pattern an interest measure, given in Gunturi et al. [2022], as well as a threshold is calculated. If the interest measure of this candidate pattern is above a threshold then the candidate pattern is accepted.

2.1.4 Threshold functions

The interest measure can be thought of as a value that represents how strong an identified pattern is. We refer the threshold measures from Gunturi et al. [2022]

Cold and Hot Regions : Hot region is the region which is considered to be the repeating pattern and the cold region is the hot region's complement.

A: Total number of events in request trace

c: The number of events in candidate the hot region

B: The expected number of events in that particular region.

The denominator is used to remove patterns which are not as strong w.r.t the entire dataset, i.e in case c = 100 but A = 100000000 , then the candidate pattern is not very strong , however if c = 100 and A = 100000 with the same b value, then the pattern is relatively stronger and hence the denominator value will increase the entire interest measure.

$$Hot\_RegLR = Log\left(\left(\frac{c}{B}\right)^{c} \Big/ \left(\frac{|A|-c}{|A|-B}\right)^{|A|-c}\right); if\ c > B \tag{2.1}$$

$$Cold\_RegLR = Log\left(\left(\frac{c'}{B}\right)^{c'} \Big/ \left(\frac{|A|-c'}{|A|-B}\right)^{|A|-c'}\right); if\ c' > B \tag{2.2}$$

$$PST\_HotspotLR = \frac{Hot\_RegLR}{max\{Cold\_RegLR, 1\}} \tag{2.3}$$

$$B = Vol.ofPattern \times ST\_Density \tag{2.4}$$

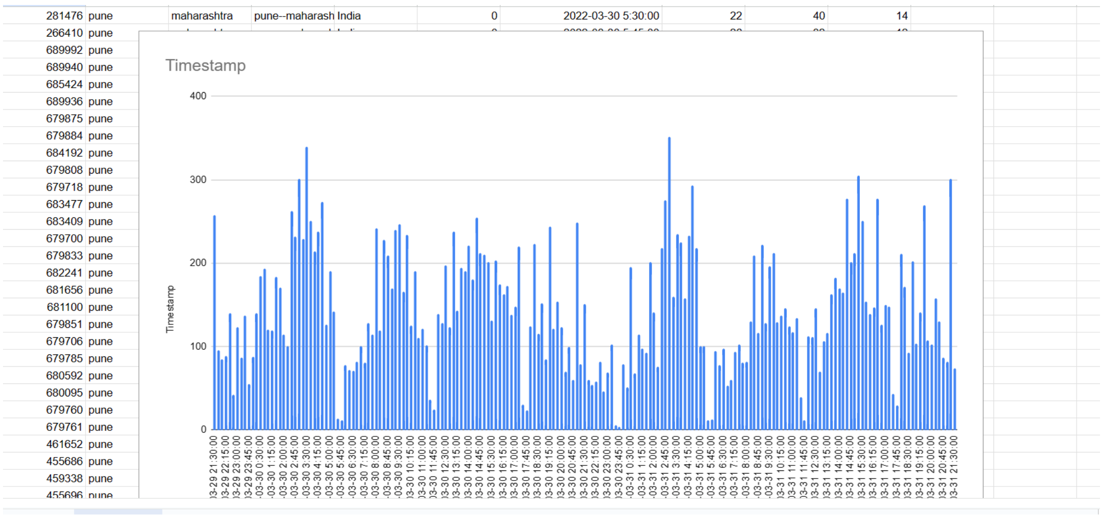
$$Vol.ofPattern = total\ slots \times length\ of\ hot/cold\ region \tag{2.5}$$

$$ST\_Density = \frac{|A|}{|G| \times T} \tag{2.6}$$

2.2 Dataset

Figure 2.1 gives a plot of a sample data we are considering. One can clearly observe periodic intervals of spikes in this dataset.

We use synthetic dataset generated by us, as well as a dataset provided by Microsoft for testing the algorithms.



281476	pune	maharashtra	pune--maharash	India	0	2022-03-30 5:30:00	22	40	14
266410	pune								
689992	pune								
689940	pune								
685424	pune								
689936	pune								
679875	pune								
679884	pune								
684192	pune								
679808	pune								
679718	pune								
683477	pune								
683409	pune								
679700	pune								
679833	pune								
682241	pune								
681656	pune								
681100	pune								
679851	pune								
679706	pune								
679785	pune								
680592	pune								
680095	pune								
679760	pune								
679761	pune								
461652	pune								
455686	pune								
459338	pune								
455696	pune								

Figure 2.1: Dataset Sample

2.3 Spatio-Temporal Hotspot Enumeration

Gunturi et al. [2022] develop an algorithm to enumerate the candidate hotspots from a give request trace from a city.

We summarize the algorithm in Algorithm 1. We implemented this algorithm with Java, PostgreSQL and PostGIS. We devised the overlap removal and cluster selection portion of this algorithm.

Algorithm 1 Periodic Spatio-Temporal Hotspot Miner

Input A set of clusters **C** and their corresponding request traces A; time boundaries for enumeration $[T_s, T_e]$; LLR threshold τ ; fixed periodicity p

Output: Periodic Spatio-Temporal Hotspots

```
1:  loop for each cluster $c_i \in \mathbf{C}$
2:      loop for each window $[t_s, t_e]$ inside $[T_s, T_e]$
3:          Initialize: candidate PST-Hotspot pattern $P$
4:          Temporal window($P$) $\leftarrow$ time window $[t_s, t_e]$
5:          Compute PST-Hotspot-LR of $P$ by Eq. 2.3
6:          if PST-Hotspot-LR($P$) $\geq \tau$ then
7:              Add $P$ to the candidate hotspots list PSTH$_{cand}$
8:          end if
9:      end loop
10: end loop
11: PSTH$_{cand}$ $\leftarrow$ Remove overlapping candidates hotspots using Algo. 3
```

2.4 Strongly Connected Cities

Before feeding the request traces in Algorithm 1, we need to consider the phenomenon of correlation of data originating from different cities.

To address this problem, we enumerate subsets of cities which form a strongly connected subgraph in our definition of an edge between two cities. We devised Algorithm2 for the same.

2.5 Overlap removal

After a list of spatio-temporal hotspots are enumerated, there will be several predictions which have a significant spatio-temporal intersection. We thus need to eliminate such overlapping detections.

We thus implement an SQL query, implemented and tested using PostGRESQL and PostGIS to detect pairs of hotspots which have a interesect above a certain threshold. Such hotspots will be 3 dimensional objects, containing a spatial footprint (Figure 2.2a), and a temporal extension (Figure 2.2b). Algorithm 3 implements the same.

Algorithm 2 SAG-StrongComps

```
    Input: Subgraph $SAG$, boolean marker array $NodesSubG[\,]$,
    $totalEdges$, mask index $start\_i$, strongness threshold $\theta$
    Initialize: strongly connected component storage $STORE = [\,]$
 1: $n \leftarrow SAG.size()$
 2: if $\frac{totalEdges}{n(n-1)/2} > \theta$ then
 3:     Add $SAG$ to $STORE$
 4: end if
 5: loop for $i$ in [start_i, .... $n$]
 6:     $NodesSubG[i] = 0$
 7:     Initialize $edgesToRemove = 0$
 8:     loop for each neighbour $j$ of $i$ in $SAG$:
 9:         if $NodesSubG[j] == 1$ then
10:             $edgesToRemove + \leftarrow 1$
11:         end if
12:     end loop
13:     $remainingEdges \leftarrow totalEdges - edgesToRemove$
14:     SAG-StrongComps($SAG, NodesSubG, remainingEdges, i+1$)
15: end loop
```

2.6 Synthetic Data Experiments

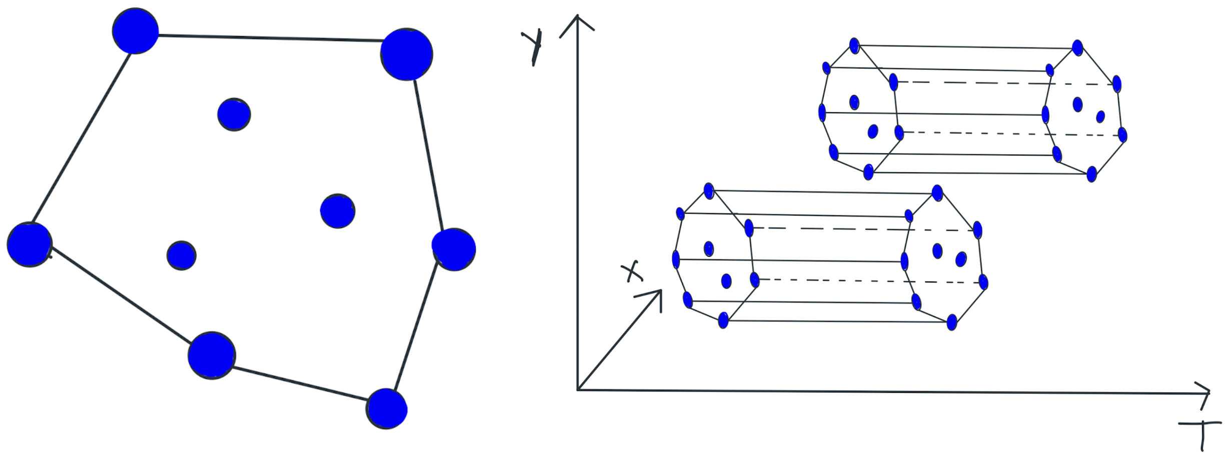
To test the performance of the algorithm and to test the sensitivity of the thresholding functions. For the same, we generated synthetic data while following the distribution in the original dataset.

We varied parameters such as pattern length, pattern height, baseline density, persistent patter density and composition of cities, and then plot F1-score and mean average precision (mAP) scores.

Algorithm 3 Spatio-temporal Overlap Detection

Input: Table of clusters *clusters*, 3D IoU threshold *threshold*
Returns: Table of pair of clusters with 3D IoU above threshold

```
select
    sub.cluster_1,
    sub.cluster_2
from
(
    select
        c1.id as cluster_1,
        c2.id as cluster_2,
        c1.geom_2d as c1_geom,
        c2.geom_2d as c2_geom,
        c1.t1 as c1t1,
        c1.t2 as c1t2,
        c2.t1 as c2t1,
        c2.t2 as c2t2
    from
        clusters as c1
    join clusters as c2 on
        ST_3DIntersects(c1.geom_3d, c2.geom_3d)
    where c1.id<c2.id
) as sub
where
threshold(ST_Area(ST_Intersection(sub.c1_geom, sub.c2_geom)),
          ST_Area(ST_Union(sub.c1_geom, sub.c2_geom)),
          sub.c1t1, sub.c1t2, sub.c2t1, sub.c2t2, threshold);
```



(a) 2D Convex Hull (ST_ConvexHull)   (b) 3D Convex Hull (ST_3DConvexHull)

Figure 2.2: PostGIS geometries. (b) extends (a) in T dimension

We classify city categories as follows based on the request types:

1. Periodic Cities (PDC): Cities having a periodic hotspot. The height of the hotspot in such traces is called periodic height, which is distributed over the baseline distribution or non-periodic height(noise).

2. Persistent cities (PTC): Cities having a constant high number of requests but no hotspot. Height of requests in these cities is called persistent height.

3. Non-periodic cities (NPC): Cities having low number of requests without periodic patterns. The height of the hotspot in such traces is called periodic height. The height of the hotspot in such traces is called non-periodic height.

Granularity of time axis is in time slots of 15 mins.

2.6.1 Computing IoU and F-1 score.

In order to study the performance of our algorith, we need to perform IoU threhsolding and compute the F-1 scores with the ground truth. Refer algorithm 4, which computes the same.

Algorithm 4 IoU Thresholding

Input: List of detected patterns DETECTED, List of injected patterns INJECTED, IoU threshold

Output: list D of filtered detections, F-1 score

```
 1: Initialize list D of filtered detections
 2: while DETECTED not empty do:
 3:     max_element = pick element with maximum LLR from DETECTED
 4:     Remove max_element from DETECTED, add it to D
 5:     Remove all elements from DETECTED which have iou > THRESH with
        max_element
 6: end while
 7: loop for each I in INJECTED:
 8:     pick element from D with max IoU and IoU > THRESH.
 9: end loop
10: falsePositive = all those elements in D which were unmapped
11: truePositive = all those elements in INJECTED with were mapped to an
    element in D
12: falseNegative = all those elements in INJECTED with were unmapped
13: return D, F-1 score
```

2.6.2 Varying Pattern Length and Height for 1 hotspot

In this experiment, we insert a single hotspot in our synthetic dataset and vary the parameters pattern length over different dataset configurations.

Refer Figure 2.3 for the F-1 score plots.

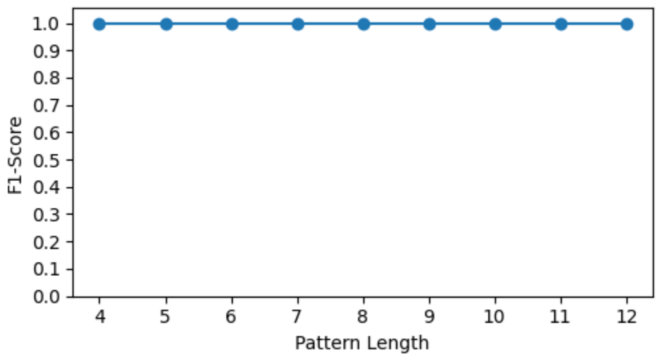
2.6.3 Varying Pattern Length and Height for 2 hotspots

In the following experiment, we insert a two hotspot in our synthetic dataset, and vary the parameters pattern length over different dataset configurations.

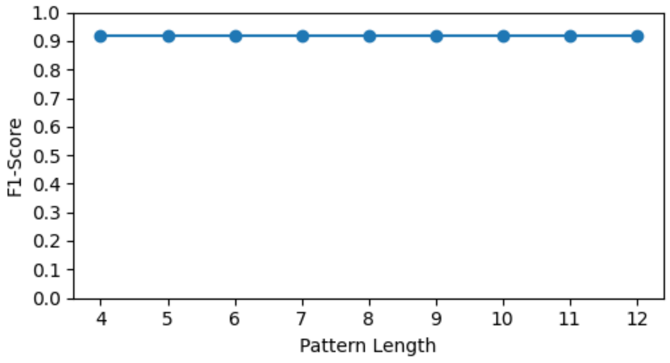
Refer Figure 2.5 for the F-1 score plots.

2.6.4 Computing mean average precision

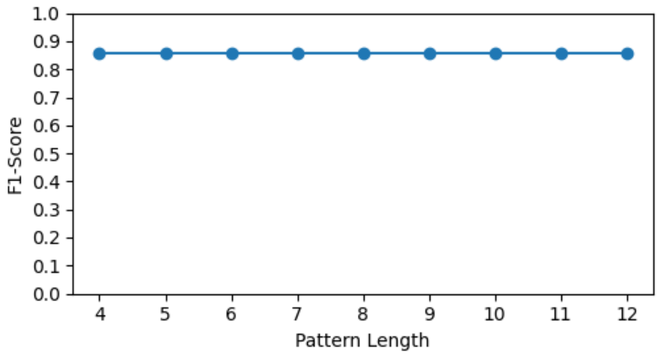
In the following experiment, compute the mean average precision over the experiments ran in the previous two sections. We vary the threshold measure over a range of values to generated a precision-recall curve. The area under this curve



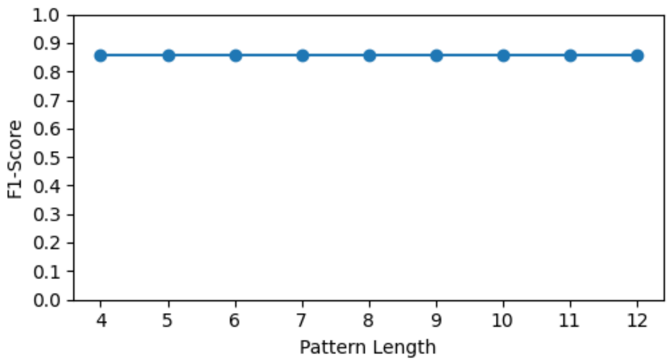
(a) PDC=10, PTC=5, NPC=5



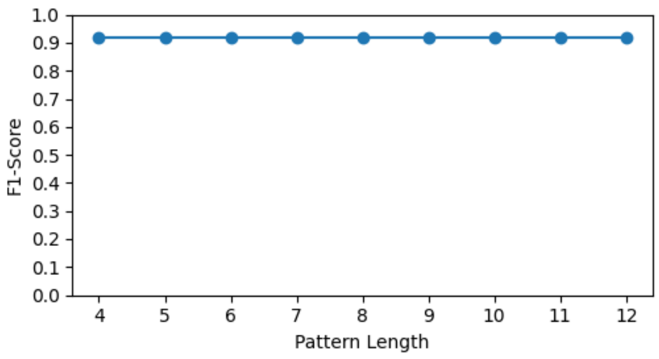
(b) PDC=7, PTC=5, NPC=8



(c) PDC=5, PTC=5, NPC=10

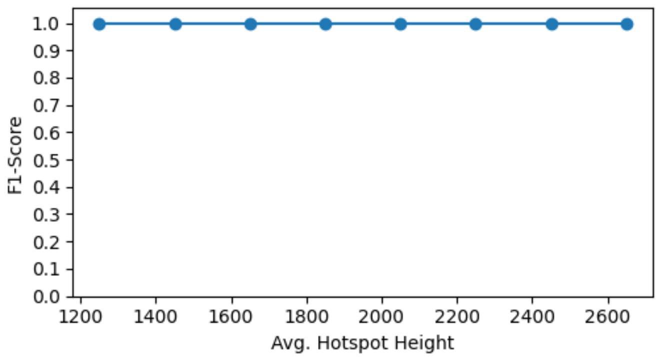


(d) PDC=5, PTC=8, NPC=7

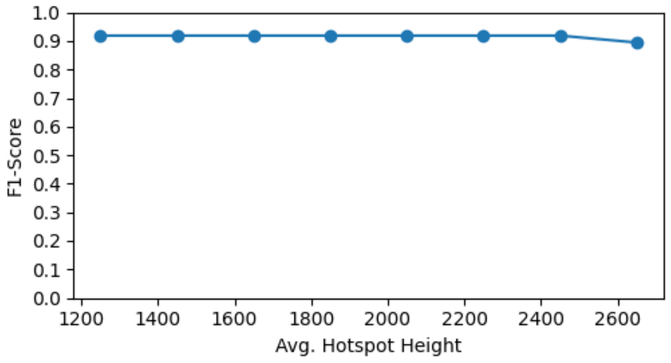


(e) PDC=7, PTC=8, NPC=5

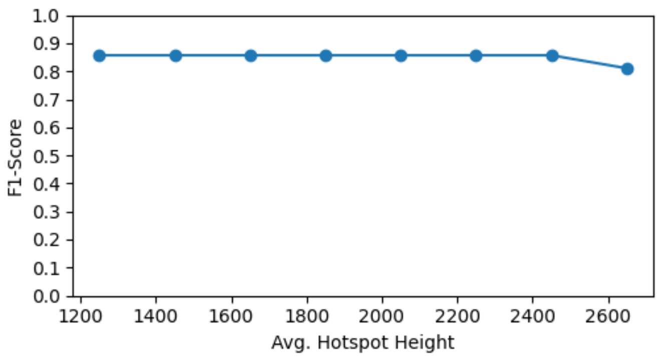
Figure 2.3: Varying Pattern length for one hotspot case for *pattern height* $= [1000, 1500]$, *persistent height* $= [800, 1100]$, *baseline distribution* $= [200, 300]$



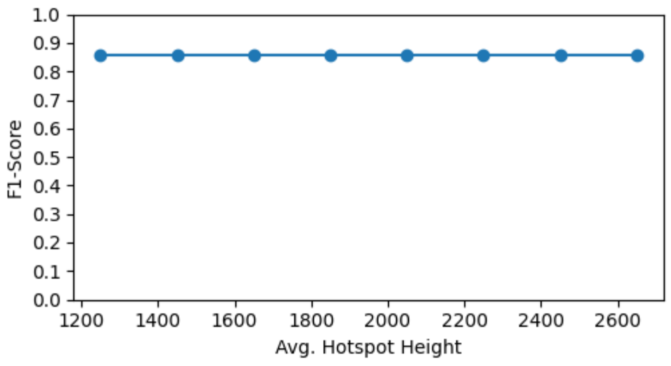
(a) PDC=10, PTC=5, NPC=5



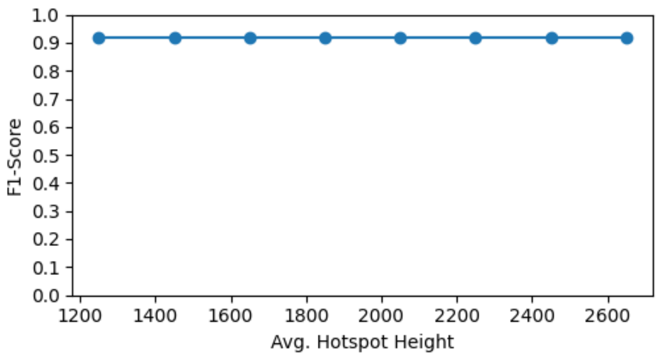
(b) PDC=7, PTC=5, NPC=8



(c) PDC=5, PTC=5, NPC=10



(d) PDC=5, PTC=8, NPC=7



(e) PDC=7, PTC=8, NPC=5

Figure 2.4: Varying avg. hotspot height for one hotspot case for *pattern length* = 8 *units*, *persistent height* = $[800, 1100]$, *baseline distribution* = $[200, 300]$

gave the mAP value.

Refer Figures 2.7 for 1 hotspot case. mAP was 92.3% @0.7 IoU. Figure 2.8 is for 2 hotspot cases respectively. mAP was 71.9% @0.7IoU.

2.6.5 Varying Persistent Height

Now, we aim to vary the parameter ST-Density in order to test the effectiveness of our threshold measure. In these experiments, we insert two hotspots in our synthetic dataset, and vary the parameters pattern length over different dataset configurations.

Refer Figures 2.7 and 2.8 for plots for 1 and 2 hotspot cases respectively. We vary the periodic height with 4 different avg. values, giving 4 different lines.

2.6.6 Periodic Baseline Density

In these experiments, the baseline is constructed to be a periodic positive sine wave with a time period of 3 hours, the hotspots inserted into this synthetic data are inserted on top of this periodic baseline. This is done so as to simulate the traffic of real data which doesn't throughout act as randomly distributed between two limits but has regular peaks and troughs.

In figure 2.10, we vary the persistent height on x-axis, for 4 values of periodic height, giving 4 different lines.

2.7 Results

2.7.1 Variation of Pattern Length and Height for 1 hotspot

We observe in Figure 2.3 that the F-1 score is high and nearly constant for all values of pattern length. Thus, the thresholding function is invariant to such variations in the data, and gives high performance.

2.7.2 Variation of Pattern Length and Height for 2 hotspots

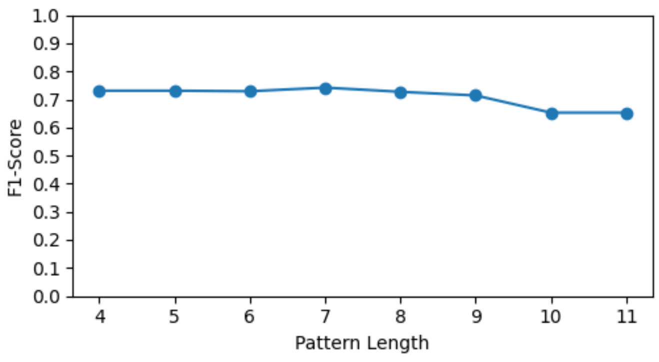
We observe in Figure 2.5 that the F-1 score slightly dipped for 2 hotspots. This is expected, and is mainly attributed to the merging of some adjacent hotspots by the algorithm. Again, the performance is nearly invariant to pattern length which is good.

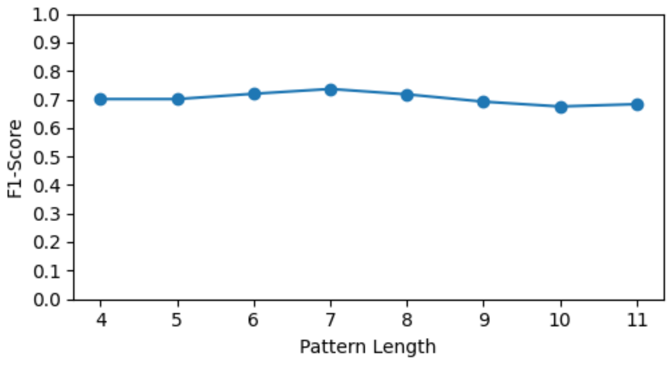
2.7.3 Varying Persistent Height

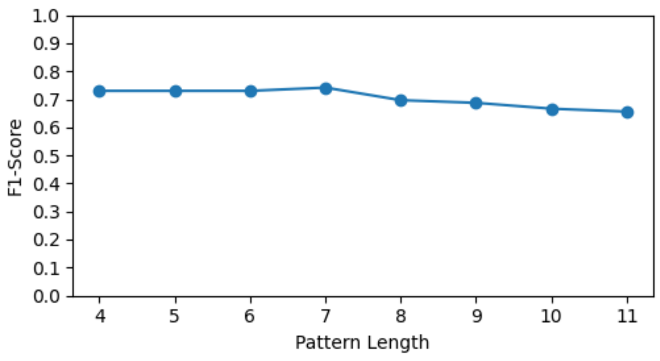
From figure 2.9 We observe that for the case of just 1 hotspot within the city, we see an extremely high and constant F1 score, signifying very high performance. For the case of 2 and 3 hotspots in a city we find that the F1 score increases when we increase the persistent heights in other cities , this happens because upon increasing these heights the hotspot patterns start getting detected separately because the expected number of connections in an area rises and so the interest measure of the hotspots taken separately exceeds the interest measure of 2 or more hotspots being taken together , and hence during the non max suppression process , the individually detected hotspots are taken as the more relevant ones.

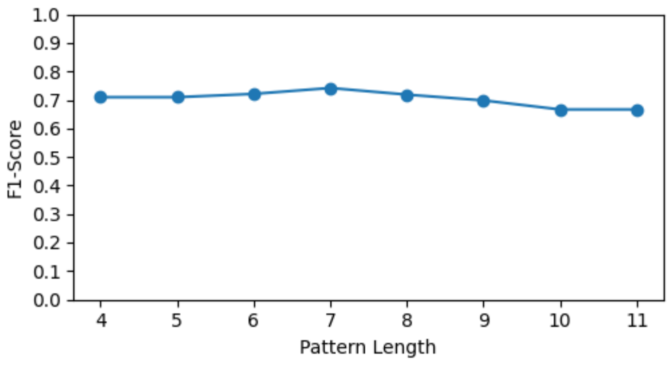
2.7.4 Periodic Baseline Density

The results are in Figure 2.10. The explanation is similar to that of the previous section. The only difference is that the data for these experiments contain a periodically varying baseline instead of a randomly distributed one (between 2 fixed upper and lower limits).


(a) PDC=10, PTC=5, NPC=5


(b) PDC=7, PTC=5, NPC=8


(c) PDC=5, PTC=5, NPC=10


(d) PDC=5, PTC=8, NPC=7

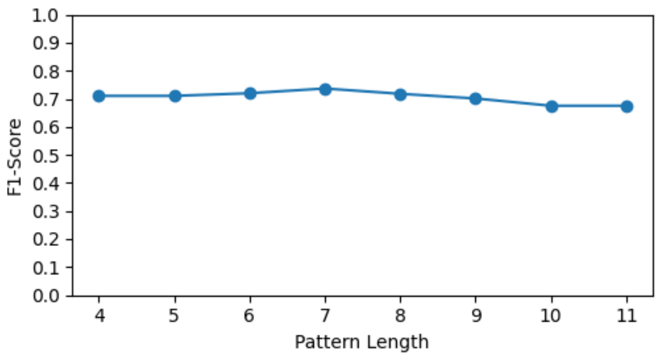
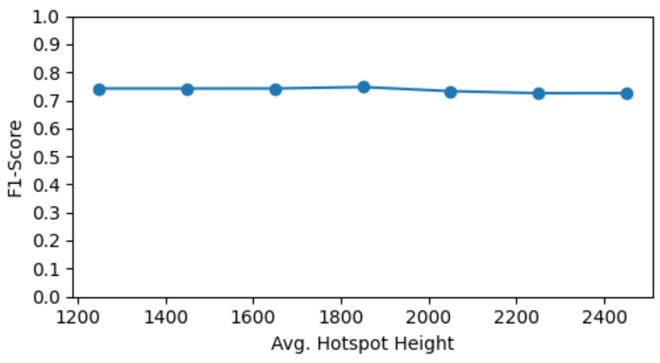
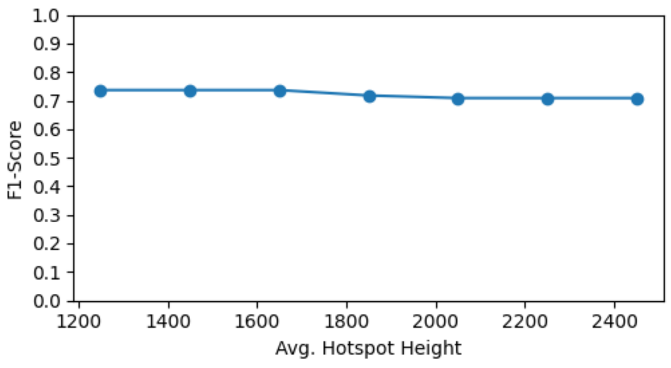

(e) PDC=7, PTC=8, NPC=5

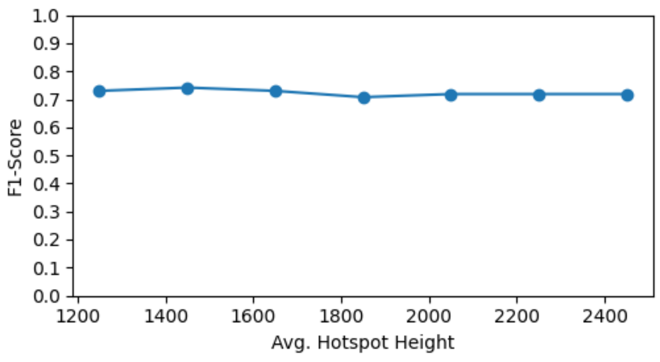
Figure 2.5: Varying pattern length for two hotspot case for *pattern height* $= [1000, 1500]$, *persistent height* $= [800, 1100]$, *baseline distribution* $= [200, 300]$



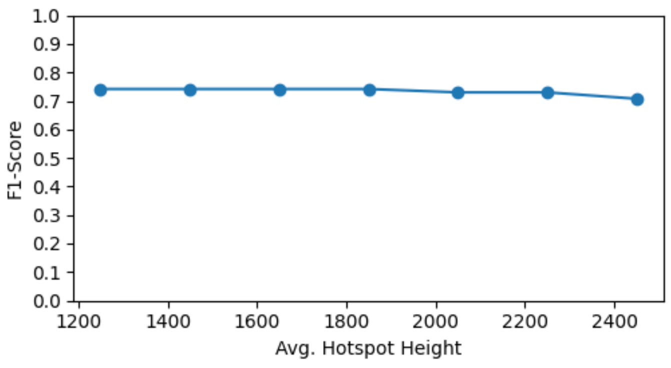
(a) PDC=10, PTC=5, NPC=5



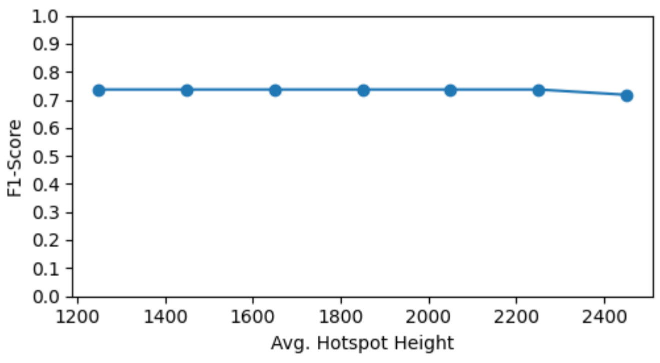
(b) PDC=7, PTC=5, NPC=8



(c) PDC=5, PTC=5, NPC=10



(d) PDC=5, PTC=8, NPC=7



(e) PDC=7, PTC=8, NPC=5

Figure 2.6: Varying pattern height for two hotspot case for *pattern length* $= 8$, persistent=[800, 1100] *baseline distribution* $= [200, 300]$

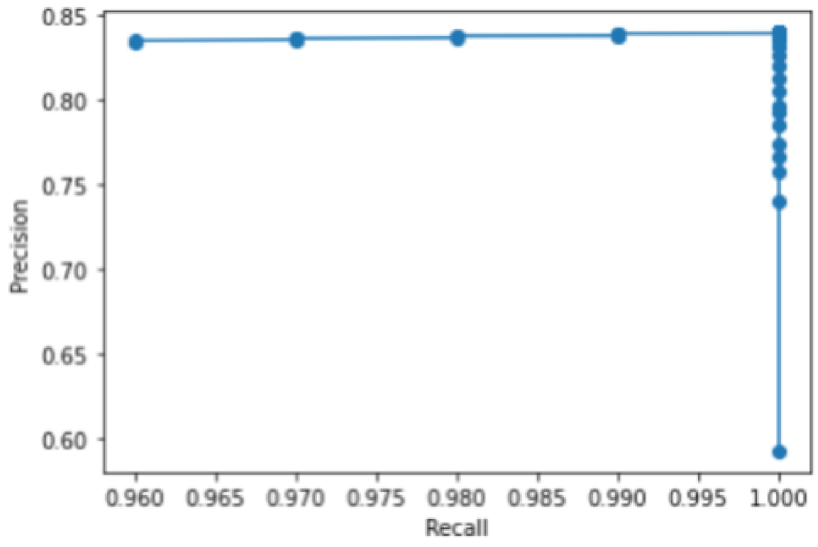


Figure 2.7: 1 hotspot mAP@0.7 IoU

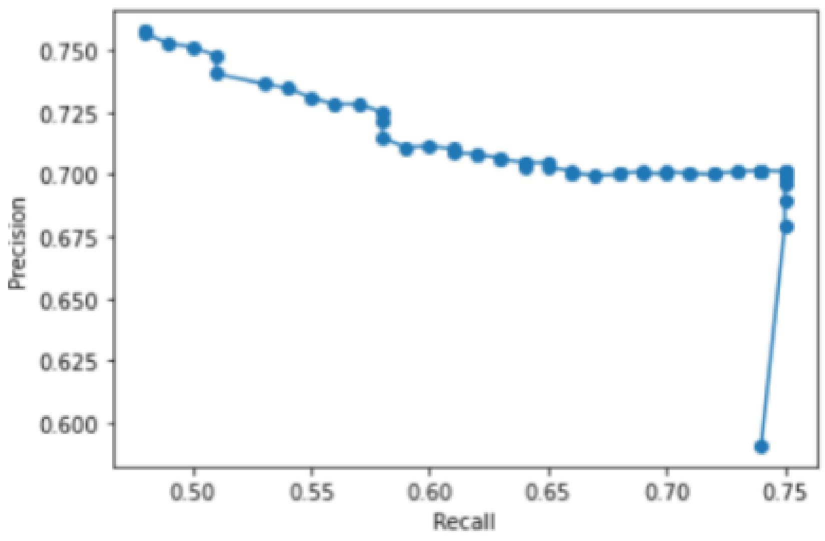
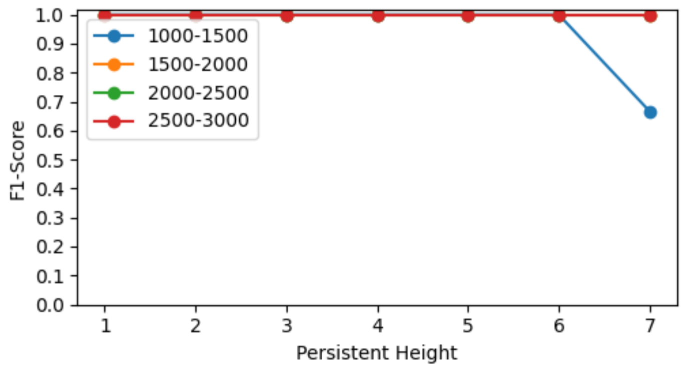
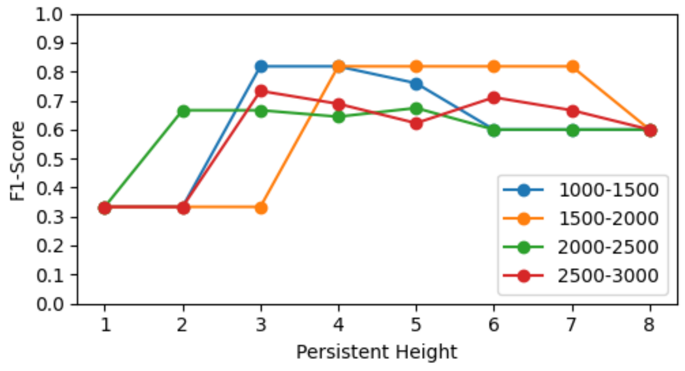


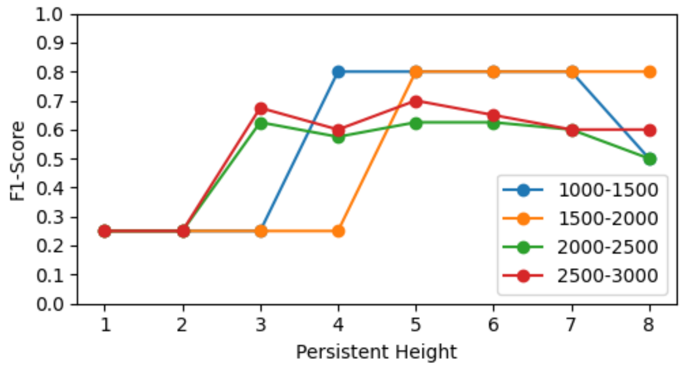
Figure 2.8: 2 hotspot mAP@0.7 IoU



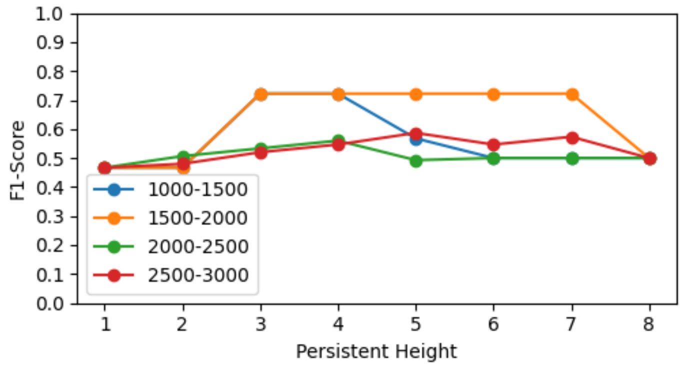
(a) 1 hotspot, pattern length=8, persistent cities=10



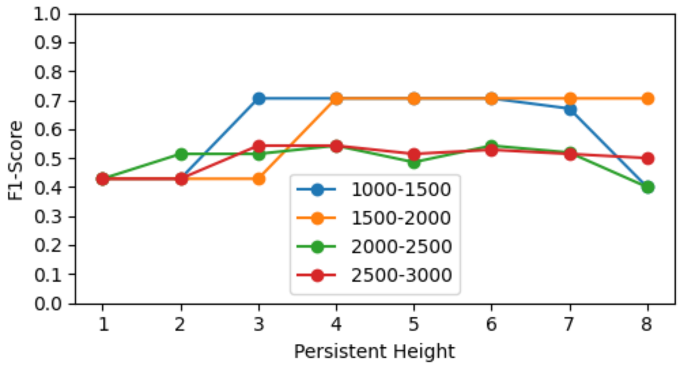
(b) 2 hotspots, pattern distance=3



(c) 2 hotspots, pattern distance=5

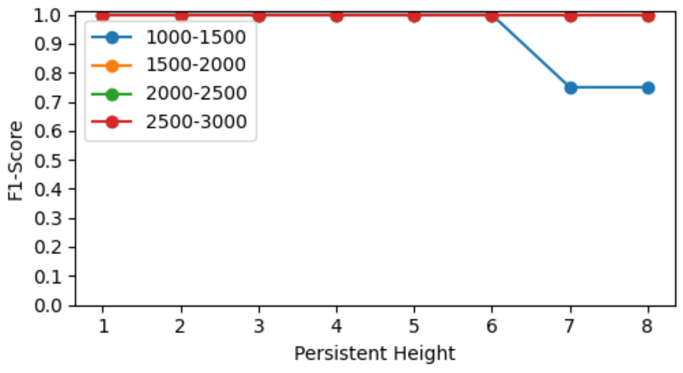


(d) 3 hotspots, pattern distance=3

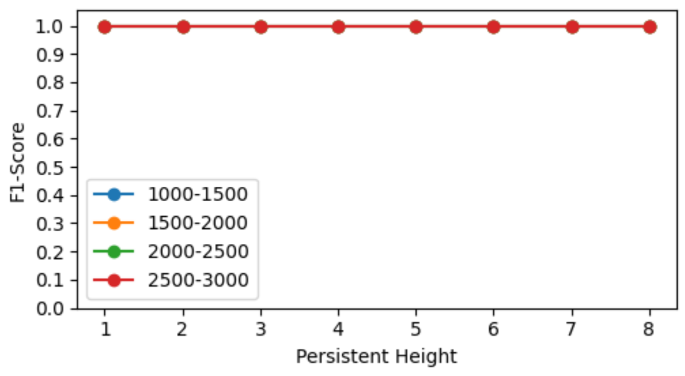


(e) 3 hotspots, pattern distance=5

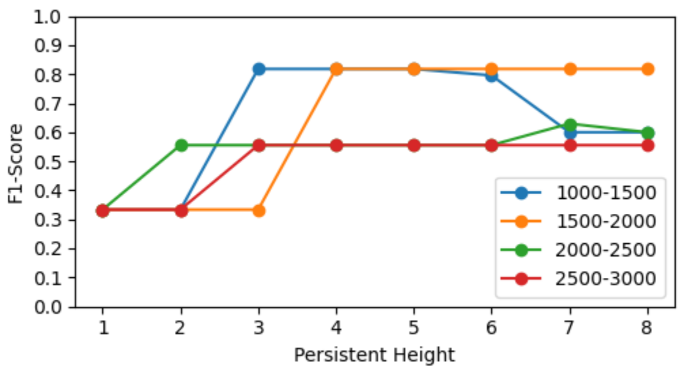
Figure 2.9: Plot of F-1 score, while increasing persistent pattern height from 1000 to 4000 in steps of 500. Periodic height is varied across the legend.



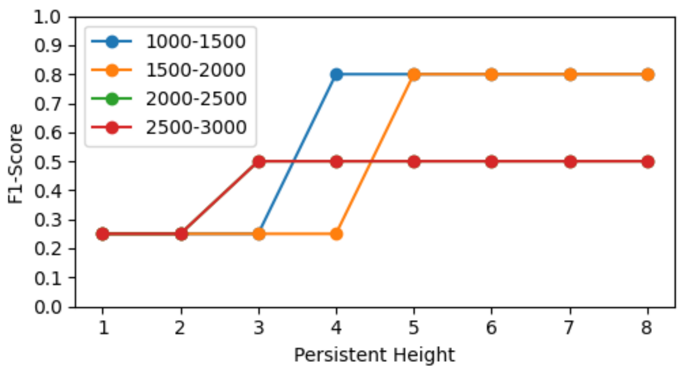
(a) 1 hotspot, pattern length=5



(b) 1 hotspot, pattern length=10



(c) 2 hotspots, pattern distance=3



(d) 2 hotspots, pattern distance=5

Figure 2.10: Periodic baseline density, while varying persistent height on x axis. Periodic height is varied across the legend.

Chapter 3

Multichromatic Spatio-temporal Hotspot Detection

3.1 Problem Statement

Till now, we have treated all requests generated from edges nodes identically. However, these requests are routed through different Autonomous System Numbers (ASNs). It is important to analyse the requests from different ASNs as well to have a better idea of the network requirements. We therefore introduce several modifications in the Monochromatic Spatio-Temporal Hotspot detection algorithm to introduce the Multichromatic Spatio-temporal Hotspot Detection algorithm.

"Multichromatic" here simply means that we assign "colors" to different request types.

3.2 Problem Framework

Given as input a collection of events and their corresponding occurrence time stamps and spatial locations, the task is to identify regions in space as well as time which exhibit high intensity of events which repeat over a period of time for each possible color.

"Color" in the above context signifies that now different events in the spatio-temporal framework have another attribute or "color" associated with them , this color can be associated with the ASN or the edgesite of that particular event or connection request.

The basic input of this problem is an array with the following attributes: - (city name, total requests, time stamp, x_coord, y_coord, ASN, edgeSite). The aim is to return pairs of (start time, end time, x_coord, y_coord, color k), which signify that there is a recurring periodic pattern of color type k (i.e events of a particular ASN or edge site) at ((x,y) or an area) that repeats itself every day between (start time and end time).

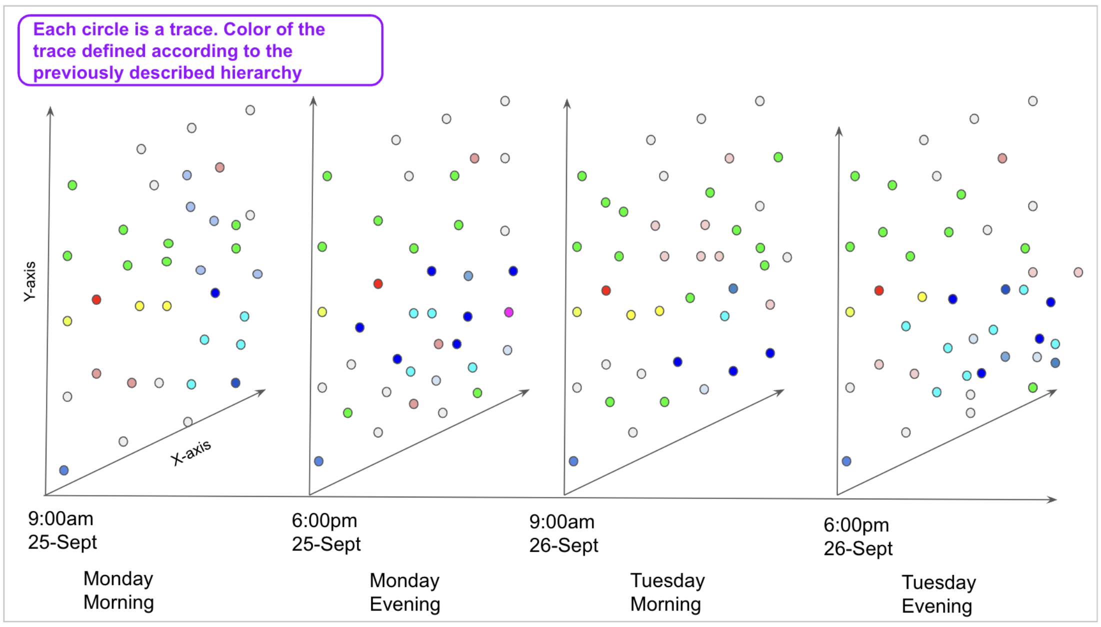


Figure 3.1: Request Colored according to ASNs

3.3 Algorithm Outline

The first step is to identify the possible spatial points for each color to be considered together which may constitute a periodic pattern. It is important to come up with a definition of a neighborhood/closeness of a particular spatial data point to another one in order to narrow down our search space.

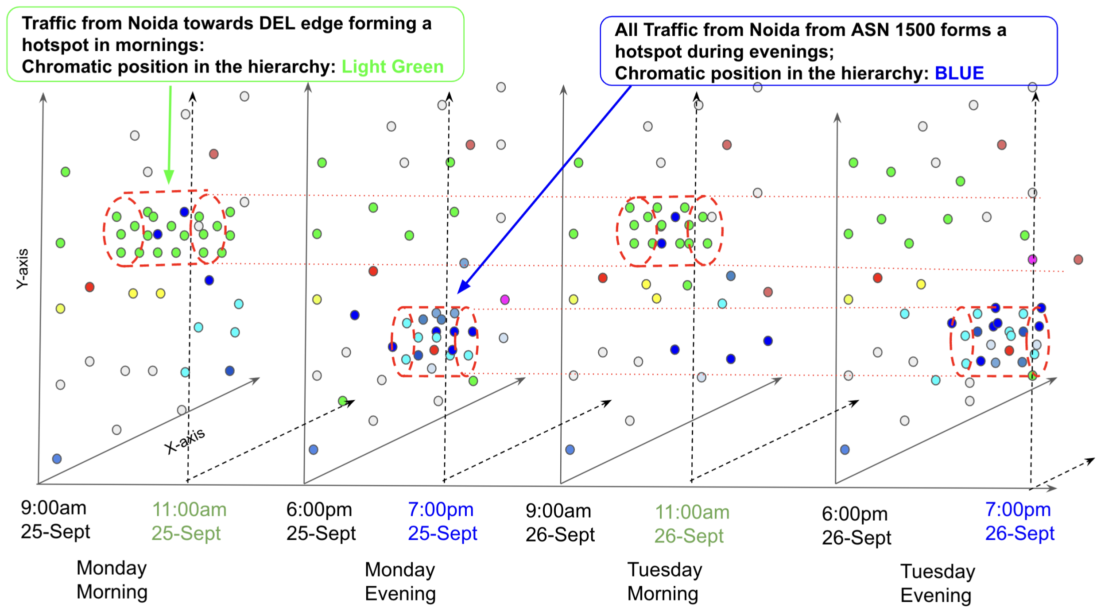

Traffic from Noida towards DEL edge forming a hotspot in mornings:
Chromatic position in the hierarchy: Light Green
All Traffic from Noida from ASN 1500 forms a hotspot during evenings;
Chromatic position in the hierarchy: BLUE
Y-axis
X-axis
9:00am 25-Sept
11:00am 25-Sept
6:00pm 25-Sept
7:00pm 25-Sept
9:00am 26-Sept
11:00am 26-Sept
6:00pm 26-Sept
7:00pm 26-Sept
Monday Morning
Monday Evening
Tuesday Morning
Tuesday Evening

Figure 3.2: Hot and Cold region in Multichromatic Framework

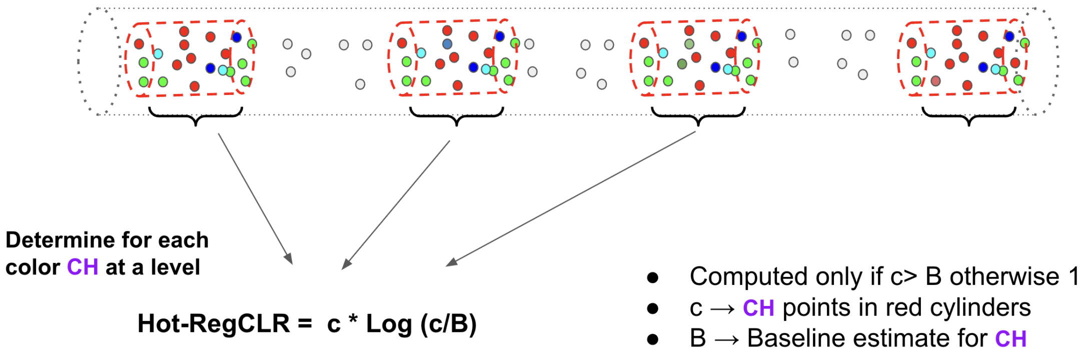

Determine for each color CH at a level
Hot-RegCLR = c * Log (c/B)
Computed only if c> B otherwise 1
c → CH points in red cylinders
B → Baseline estimate for CH

Figure 3.3: Hot region Multichromatic Framework

We require a network related parameter to define closeness of two spatial data points, connecting distinct data points by edges as given below-

Two spatial data points g1 and g2 are connected via an edge if both these conditions hold:

1. The geodetic distance between g1 and g2 is less than radius r.

2. The number of ISP providers common to both g1 and g2 is greater than a threshold θ.

Now once the spatial locations to be considered are identified (for a particular color), all possible temporal windows are iterated over along with all possible periodicity values to identify recurring patterns. Now the final task is that given the temporal window (start time , end time), the location ((x,y) or area) and a time period, we need to classify whether this is a sufficiently strong pattern or not.

For each such candidate pattern an interest measure is defined, this interest measure is new and different from the one defined in mono-chromatic hotspot algorithm, as well as a threshold is defined. Instead of fixing a threshold, we select the top k patterns from the enumerated patterns.

Algorithm 5 implements the logic for enumerating the patterns in a color heirarchy.

3.4 Color Heirarchy

The color heirarchy is an attempt to classify different request from edge sites. An ASN forms the parents for all the underlying routes through which the data arrived. Thus, after assigning a color to the ASN, we assign different shades of the color to its children, and so on to form a tree. Refer Figure 3.4.

3.5 Threshold Functions

In multichromatic hotspot problem, we simplify the hot (Eq. 3.1) and cold (Eq. 3.2) region threshold measures.

Algorithm 5 Multichromatic Periodic Spatio-Temporal Hotspot Miner

Input A set of clusters **C** and their corresponding request traces A; a color-hierarchy H, time boundaries for enumeration $[T_s, T_e]$; LLR threshold τ ; fixed periodicity p

Output: Multichromatic Periodic Spatio-Temporal Hotspots MCSTH$_{cand}$ with color label k

```
1:  loop for each cluster $c_i \in \mathbf{C}$
2:      loop for each color $k$ in color hierarchy $H$
3:          loop for each window $[t_s, t_e]$ inside $[T_s, T_e]$
4:              Initialize $P_k$ candidate MCSTH-Hotspot pattern
5:              Isolate traces from A for cluster $c_i$ for color $k$ into $P_k$
6:              Temporal window($P_k$) $\leftarrow$ time window $[t_s, t_e]$
7:              Compute Chromatic-LR of $Q$ by Eq. 3.3
8:              if Chromatic-LR($P_k$) $\geq \tau$ then
9:                  Add $P_k$ to the candidate hotspots list MCSTH$_{cand}$
10:             end if
11:         end loop
12:     end loop
13: end loop
14: MCSTH$_{cand}$ $\leftarrow$ Remove overlapping candidates hotspots using Algo. 3
```

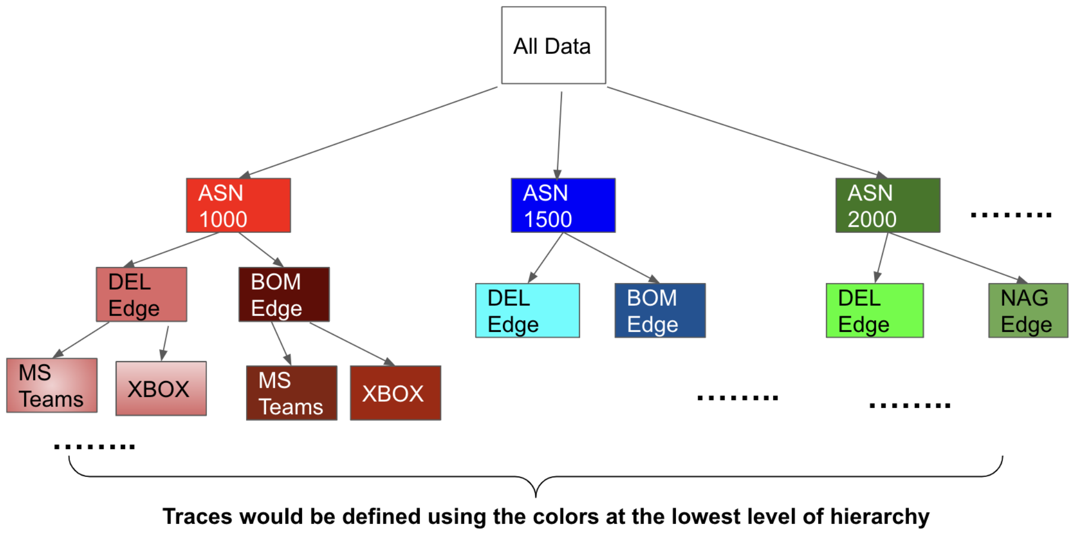


Figure 3.4: Heirarchy of Colors

The parameters c, c', B and B' are calculated as before.

$$Hot\_RegCLR = c * log\left(\frac{c}{B}\right); if\ c > B \tag{3.1}$$

$$Cold\_RegCLR = c' * log\left(\frac{c'}{B'}\right); if\ c' > B \tag{3.2}$$

$$ChromaticLR = \frac{Hot\_RegLR}{max\{Cold\_RegCLR, 1\}} \tag{3.3}$$

3.6 Experiments

3.6.1 Data Annotation

We annotated request traces from 30 cities in the real dataset. For each city, data length was 3 days. We annotated hotspots manually for these cities for our experiments.

3.6.2 Running the algorithm

We ran the developed algorithm on the real dataset provided by Microsoft to extract significant patterns. We used the request traces for Navi Mumbai and Kolkata for the same. In figure 3.5 and 3.6, we extracted the most significant pattern which is marked by a black line.

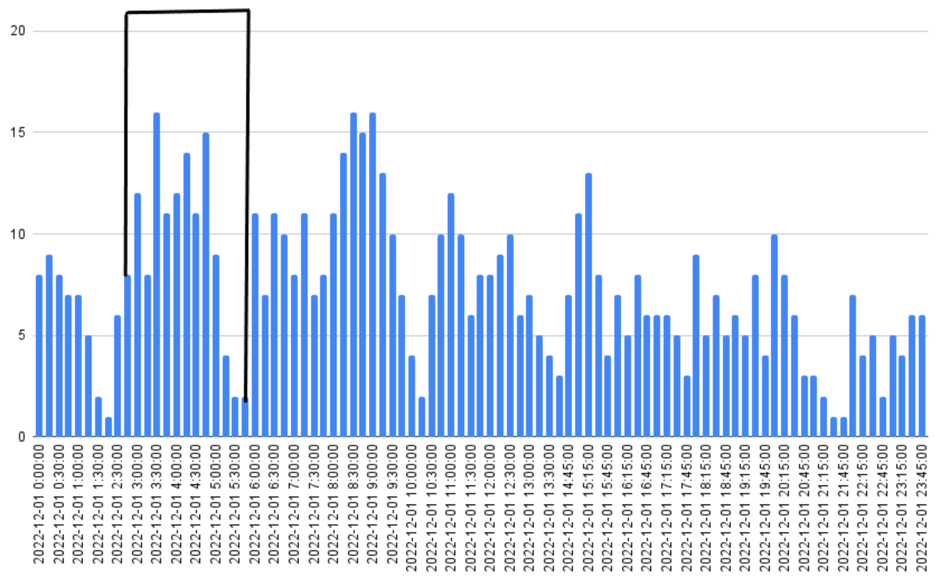


Figure 3.5: Significant Hotspot for Navi Mumbai

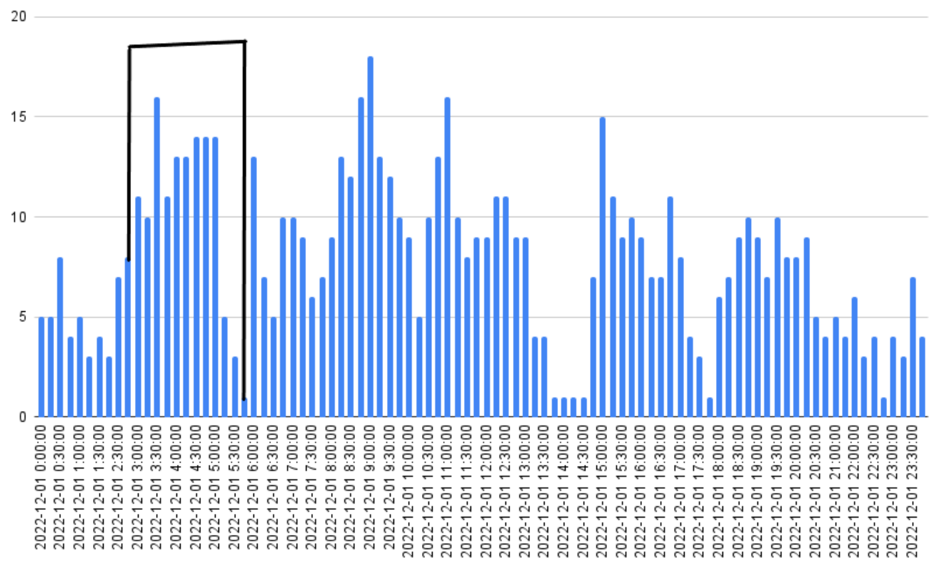


Figure 3.6: Significant Hotspot for Kolkata

Chapter 4

Conclusion

We thus implemented monochromatic as well as multi-chromatic hotspot detection algorithms as Microsoft Azure request traces. Deployment of such algorithms will help the company better configure it's network routers to handle such periodic usage spikes in the requests.

4.1 Work Division

4.1.1 Work done in CP302

In CP302, we implemented the monochromatic spatio-temporal hotspot detection algorithms by Gunturi et al. [2022]. We worked on the implementation of monochromatic hotspot detection Algorithm 1, subset enumeration algorithm 2 , and overlap removal algorithm 3.

4.1.2 Work done in CP303

In CP303 we worked on synthetic experiments and analysis, involving the testing of the monochromatic hotspot detection that we worked on in the previous semester. We also worked on data annotation of the real data obtained from Microsoft, i.e, manual identification of hotspots so as to analyze the algorithm's performance on the real dataset.

In this semester we also extended the monochromatic approach to formulate the multichromatic hotspot detection problem i.e, to identify usage hotspots corresponding to different ASNs and edge sites. We thus propose a modification in Algorithm 1, implemented and tested the Algorithm.

References

Venkata M. V. Gunturi, Rakesh Rajeev, Vipul Bondre, Aaditya Barnwal, Samir Jain, Ashank Anshuman, and Manish Gupta. A case study on periodic spatio- temporal hotspot detection in azure traffic data. In K. Selçuk Candan, Thang N. Dinh, My T. Thai, and Takashi Washio, editors, *IEEE International Conference on Data Mining Workshops, ICDM 2022 - Workshops, Orlando, FL, USA, November 28 - Dec. 1, 2022*, pages 1037–1044. IEEE, 2022. doi: 10.1109/ICDMW58026.2022.00135. URL `https://doi.org/10.1109/ICDMW58026.2022.00135`. 1, 2, 3, 4, 5, 27