# Introduction (Motivation)

F(n) = g (n) + h(n) – A\* star algorithm function.

Covid-19 is a global crisis which by now is all too familiar even for our youngest. Country after country went into lockdown and economies came into standstill. While there were many arguments as to why this Virus resulted in a pandemic, it also brings us a very serious concern as to how ill-prepared we all were. From that one patient-zero to such a widespread infection it was simply so amazing to see how we humans are so helpless in front of infections such as these and that no power in the world could do anything but simply watch and try to mitigate it somehow. The lack of efficient technology for effective contact tracing even after having seen multiple endemics / epidemics in last couple of decades and reluctance of major world powers to still not do much in terms of contact tracing technologies was simply mind boggling. As was seen in the last few months, there was no vaccine which could have been developed and fast tracked in just couple of months. A sudden increase in infected people also highlighted how hospitals and healthcare facilities got overwhelmed and shortage of medical supplies crippled countries into lockdown.

This brings us to a very important conclusion that the very first step in combating this or any such virus has to be “isolation-with-care”. Meaning, by carefully monitoring our most vulnerable and by remote monitoring the possibly infected and isolating them, we can help communities continue with some normality and for this reason the only option is to have effective **contact-tracing algorithms** and implementation without much of privacy violations.

A close up of a map

Description automatically generatedThe image here was an example of the Geographic routing: Greedy forwarding approach explained in class which triggered this idea of contact tracing could be done in real-time. Coupled with the A\* algorithm a suitable solution can be derived to calculate and backtrack contacts.

# Technical Approach

The force of infection spread as per one of the listed papers is shown below. Based on this I am trying one of the efficient algorithms like an A\* algorithm.

A picture containing knife, bird

Description automatically generated

The idea here is to build a Social distancing fabric using certain algorithms like A\* and mock an gossip epidemic messaging to effectively help people isolate themselves and if medical treatment is required how the healthcare workers can reach out to them. The technical approach for the implementation of a suitable contact-tracking system architecture contains usage of cloud infrastructure like AWS/GCP etc (which have their suite of IOT components as well), usage of appropriate wireless technologies like BLE, AGPS and Wifi along with effective sensors at the endpoints.

The core of all this is an effective backtracking system like A\* which would heuristically assign weight to nodes (persons) using an appropriate **scoring** functionality and connectivity between nodes are only considered if they are within a 6-meter distance to each other. Scoring is purely based of 2 main parameters, the health vitals of the individual node and the distance of the node from the possibly node (if available). The scoring and path taken by nodes at given coordinates location starts to get recorded only when the “health-vitals” trigger them to do so. Snapshots of this is stored for a month and replayed as and when required. For example, if a person starts to show symptoms of a Covid Patient like increased heartbeats, coughing, increase in body temperature etc, sensors on skin patches enabled with heat sensing and piezoelectric material can send measurements to the persons phone, which then using the inbuilt GPS, concatenate that information and send it to the nearby health clinic and also to the cloud ML systems. The ML lambda fn would also receive the vitals along with the coordinates and timestamp and based on its reading assign the node as “red”, “orange” or “green”. If a node is in red, all the nearby nodes would be treated as in “orange” and the farthest from source be treated in “green” but still monitored. Green to orange or orange to red would depend on the individual’s vitals and distance from red initially. Red nodes are can be turned back to green by visiting the “Greenhouse” – a possible drive-in where the individual is tested and his or her device reset back to green.

The below snippet taken from one of the papers highlight how time period and range of the infected person plays a role for other nodes being treated as infected as well. Though the nodes are never stationary, and this model would fall short in real-time detection, but it would certainly be useful while backtracking the coordinates when the node A was infected. Here node E is just passerby and B and D are completely away.

A close up of text on a black background

Description automatically generated

Also, messages will be sent back to all these possibly infected nodes that they have been a near a covid patient at some point and hence need to quarantine themselves and also go to the designated healthcare facility to get tested if they feel they should do so.

# Analysis of Datasets available on Kaggle and other sources.

Analysis of these sample datasets are already available online. Reanalyzing them to retrofit into our application is required and hence they have been mentioned here briefly. We are considering 4 major parameters for our testing here – **patient heartbeat, body temperature, respiratory sound (cough), and AGPS coordinates**. These parameters are assumed to come from sensors available on a given node.

1. Heartbeat Samples: Here I have utilized the sample heartbeat data and randomly picked records from the available dataset. Records are used to mock the real-world heartbeat ECG record format.

### Respiratory Sound : Utilizing respiratory sound to diagnose COPD (Chronic Obstructive Pulmonary Disease), LRTI (Lower Respiratory Tract Infection) or URTI (Upper Respiratory Tract Infection) is normally done in a controlled environment like an ICU within a hospital but here we assume that the sensor we have created (the skin patch) has a quality piezoelectric material which only takes sounds from the patient’s body and disregards outside noise.

1. Body temperature: This is a random value between 95 degrees to 106 degrees and couples together the heartbeat vitals.
2. Coordinates: Coordinates are taken real-time from a sample android application I have built coupled with sample heartbeats and body temperature to mock the sensor response.

# Implementation Details

The design here is self-explanatory and utilizing AWS cloud infrastructure for implementation. The CovidGrapher over here gets its score from the Scoring ML function, which on basis of every real-time data received checks the database for additional information and finally creates a score for the current device. It also updates the scores of all the correlated devices. CloudWatch here helps in scheduling repeated ML predictions and updating the node information. It also helps in logging. If at given point a high score is received, the CovidGrapher replays the coordinates of the current device in last 14 days and reassigns scores to all the nearby nodes.

A close up of a map

Description automatically generated

On the left side I have mentioned the sensors which could be any device which can help collect health records of a person’s body. The health vitals (body temperature, respiratory sound, heartbeats etc.) can help calculate a score for a given node/person and with the help of coordinates the person has been throughout the last 14 days till the day he/she becomes positive can be captured onto a database. Here I have used DynamoDB since it is easier to setup and scalable NoSQL on AWS. The data model is based off a primary key (partition key + sort key) which helps with a natural index. Partitioning helps in better lookups and device hashes can be constructed to match a specific region/bucket.

Following are components being used in the system architecture:

1. IOT Sensor’s + Android App: The IOT sensor can be skin patch with various components to monitor the health vitals. The sensor needs to be small battery power or solar powered to let the Android app send the health vitals.
2. Greenhouse provisioned health clinics / Drive In’s: The health clinics should have wireless capability to read the vitals of the person’s body without a cellphone access. The future could be well RFID based sensors and sensors read at regular intervals at various checkpoints. Realtime coordinates when a person visits a clinic are obviously non-relevant and historical data should be already captured through the phone coordinates before.
3. AWS Action’s: AWS Actions are set to forward specific records to specific endpoints. Heavier files are streamed over to the S3 bucket with a s3 key and the key can be generated beforehand by the device.
4. Covid Grapher (Lambda Fn): This would be used to backtrack the device and based on the score received and route traversed the device could be send a appropriate warning message. “Red” –Covid+ and needs to testing, “Orange”: Possible Covid+ and needs to quarantine, “Green”: Possible Covid- (follow social distancing). These messages are sent back to the device.
5. Scoring ML (Lambda FN): Function to classify the readings and assign scores to devices.

## A close up of a map Description automatically generatedCovid19 Health Coordinator App (App)

I have implemented a sample android application to help send the coordinates to the AWS IOT platform using a MQTT messaging. The coordinates are generated and sent to the subscriber as shown below and forwarded to Lambda function on click of button, which makes it simple self-voluntary process unless the user gives explicit permission for collecting records automatically.

If the health messages are received, the user can be shown a list of nearby health-centers he can visit and possibly on click of the “Send Health Data” do automated registration with a nearby clinic. Once the health center (greenhouse systems) records the reservation, they can capture the real time vitals of the patient as and when they arrive and move them to quarantine immediately. This is to ensure that +-patients don’t impact others since hospitals can be hot-zones too.

The screenshot below showcases the sample MQTT messages getting received from the phones onto the subscriber and the coordinates being sent are captured. Heavier files like respiratory sounds and other images are to be send to S3 bucket instead of storing them in a DynamoDB due to it’s space limitation and other factors.

### 

### Data model:

Note: DynamoDB does not require schema definition, and so there is no such thing as a "column". You can just add a new item with a new attribute.

|  |  |
| --- | --- |
| Table 1: covid\_device\_health\_info - This table stores data on health vitals. The partition key is device and coordinate at the specific timestamp and all the health vitals at that particular timestamp | aws dynamodb create-table \  --table-name covid\_device\_health\_info\  --attribute-definitions \  AttributeName=device\_id\_coords,AttributeType=S \  AttributeName=entry\_dtm,AttributeType=S \  --key-schema \  AttributeName=device\_id\_coords,KeyType=HASH \  AttributeName=entry\_dtm,KeyType=RANGE \ --provisioned-throughput \  ReadCapacityUnits=10,WriteCapacityUnits=5 |
| Table 2: covid\_device\_tracker - Add device health information. The distance from a specific hot spot coordinate is stored here | aws dynamodb create-table \  --table-name covid\_device\_tracker \  --attribute-definitions \  AttributeName=device\_id,AttributeType=S \  AttributeName=entry\_dtm,AttributeType=S \  --key-schema \  AttributeName=device\_id,KeyType=HASH \  AttributeName=entry\_dtm,KeyType=RANGE \ --provisioned-throughput \  ReadCapacityUnits=10,WriteCapacityUnits=5 |
| Table 3: covid\_coordinate\_tracker This table simply stores the coordinates and the timestamp it was captured and all the devices and in that specfic coordinates | aws dynamodb create-table \  --table-name covid\_coordinate\_tracker \  --attribute-definitions \  AttributeName=coordinate,AttributeType=S \  AttributeName=entry\_dtm,AttributeType=S \  --key-schema \  AttributeName=coordinate,KeyType=HASH \  AttributeName=entry\_dtm,KeyType=RANGE \ --provisioned-throughput \  ReadCapacityUnits=10,WriteCapacityUnits=5 |

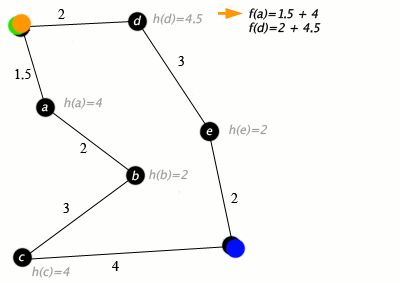
Sample Inputs:

|  |  |
| --- | --- |
| Table 1 | aws dynamodb put-item \ --table-name covid\_device\_health\_info \ --item \  '{"device\_id\_coords": {"S": "0\_32.9094801,-97.0080203"}, "entry\_dtm": {"S": "2020-05-12 10:00:00"}, "coordinates": {"S": "32.9094801,-97.0080203"}, "body\_temperature": {"N": "98.5"},"s3\_ecg\_key": {"S": "ecg/device\_0"},"s3\_respiratory\_key": {"S": "respiratory/device\_0"}, "heartbeats\_per\_cycle": {"S": "0.9856321811676024,0.7859195470809935,0.3591954112052917,0.09626436978578568,0.04310344904661178,0.1451149433851242,0.19683907926082608,0.17528735101222992,0.15660919249057767,0.15948276221752167,0.17241379618644712,0.1551724076271057,0.1465517282485962,0.16091954708099362,0.1709770113229752,0.17528735101222992,0.17241379618644712,0.19252873957157132,0.2155172377824784,0.2284482717514038,0.24856321513652796,0.27442529797554016,0.29166665673255926,0.31609195470809937,0.3433907926082611,0.3649425208568573,0.395114928483963,0.41091954708099365,0.4295977056026458,0.4511494338512421,0.4497126340866089,0.44683909416198725,0.4310344755649568,0.4181034564971924,0.4037356376647949,0.36925286054611206,0.3477011620998382,0.33764368295669556,0.30890804529190063,0.2787356376647949,0.27155172824859614,0.25143676996231074,0.2399425357580185,0.2327586263418198,0.22413793206214905,0.2385057508945465,0.2385057508945465,0.2327586263418198,0.2399425357580185,0.2370689660310745,0.2643678188323974,0.2902298867702484,0.29741379618644714,0.31896552443504333,0.3290229737758636,0.2959770262241364,0.25143676996231074,0.2284482717514038,0.22557471692562106,0.21839080750942233,0.21408045291900638,0.22557471692562106,0.2284482717514038,0.21839080750942233,0.2284482717514038,0.2212643623352051,0.2040229886770248,0.3477011620998382,0.7054597735404968,1.0,0.8663793206214904,0.4928160905838012,0.14798851311206815,0.0,0.0933908075094223,0.1537356376647949,0.12643678486347198,0.1149425283074379,0.1163793131709099,0.10057470947504045,0.09913793206214903,0.10057470947504045,0.09770114719867705,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0"}}' |
| Table 2 | aws dynamodb put-item \ --table-name covid\_device\_tracker \ --item \ '{  "device\_id": {  "S": "0"  },  "entry\_dtm": {  "S": "2020-05-12 10:00:00"  },  "current\_score": {  "N": "95"  },  "category": {  "S": "RED"  },  "referred\_health\_clinic": {  "L": [ { "S" : "<https://www.primaryhealth.com/coronavirus-covid-19>"}]  },  "current\_coordinates": {  "S": "32.9094801,-97.0080203"  }  }' |
| Table 3 | aws dynamodb put-item \ --table-name covid\_coordinate\_tracker \ --item \  '{  "coordinate": {  "S": "32.9094801,-97.0080203"  },  "entry\_dtm": {  "S": "2020-05-12 14:00:00"  },  "device\_list": {  "L": [  {  "M": {"device\_0": {"N": "10"},"heuristic\_weight": {"N":"3"}}  },  {  "M": { "device\_1": { "N": "15"},"heuristic\_weight": {"N":"2"}}  },  {  "M": {"device\_2": {"N": "15"},"heuristic\_weight": {"N":"4"}}  },  {  "M": {"device\_3": {"N": "15"},"heuristic\_weight": {"N":"14"}}  },  {  "M": {"device\_4": {"N": "13"},"heuristic\_weight": {"N":"10"}}  },  {  "M": {"device\_5": {"N": "14"},"heuristic\_weight": {"N":"7"}}  },  {  "M": {"device\_6": {"N": "7"},"heuristic\_weight": {"N":"12"}}  }  ]  }  }' |

###### Algorithm (A\* Algorithm – heuristic version of Dijkstra’s)

A straightforward Djiktra’s algorithm would take an approach of just the distance between nodes but here apart from weight and heuristic score is taken on the nodes as well.

A picture containing text, map, table, man

Description automatically generated

At each iteration of main loop A\* determines which path to extend. Since this path extension is based off a cost / distance + a heaursitic function that determines the cost of the cheapest path from n to the goal, this approach suits our requirement of effective contact tracing. The heuristic function hence gets tweaked by the score we received from the ML lambda function.

Here, let N be the number of individuals in a location and fi be the fraction that say were infected. Then total infected: fi \* N. If after testing say fraction rc is infected, then confirmed cases becomes rc \* fi \* N.

Now upto this point calculation has been with data available with healtcare providers but for an effective contact tracing voluntary public support is required. Let say a fe as the fraction of people voluntary enrolled in automated tracing fc as the number of individuals who reported Covid+. Then the total number of cases that can be traced would be fc \* fe \* rc \* fi \* N \* ac \* fe where ac is the average number of contacts per person in the period of time to who are at risk of being infected due to proximity with a sick individual.

# Results

### Grapher Heuristics:

Here we have a sample measurements of a A\* algorithm with 50000 nodes and 100000 edges. A\* here simply converged in 15 ms shows how effective the algorithm can be while tracing out possible path taken while a Virus is assumed to jump from Source to Target and which nodes/people it could have visited.

|  |
| --- |
| AStar.main -- Random Seed : 143019386577792  AStar.main -- SOURCE : DigraphNode{id=17556}  AStar.main -- TARGET : DigraphNode{id=39617}  AStar.main -- A\* in 15 ms  DigraphNode{id=17556}  DigraphNode{id=8492}  DigraphNode{id=48988}  DigraphNode{id=34290}  DigraphNode{id=25812}  DigraphNode{id=46059}  DigraphNode{id=29605}  DigraphNode{id=39201}  DigraphNode{id=37685}  DigraphNode{id=11774}  DigraphNode{id=34955}  DigraphNode{id=39617} |

1. Scoring function calculation:

A screenshot of a computer

Description automatically generatedThe ML Lambda function takes in parameter for respiratory sound waves, heartbeat and body temperature from the sensors. Since these parameters are not very easy to be calculated in real-time and an effective scoring function cannot be generated within a limited time, the results were mocked from ML. A significant contact would be if a node remained within 2 meters of an infected person and that lasted for 15 minutes or so. That increases the score for the node and hence impacts its category (Red, Orange or Green). Respiratory scoring could be 1-No Crackles or Wheeze, 2-Only Crackles, 3-Only Wheezes and 4-Both Crackles and Wheezes

This chart simply showcases the data distribution and what each category looks like. This model was taken from an already implemented solution online. The graph here is just for understanding how the sample dataset looks like.

Similarly, we can chart for the heartbeat cycles which we covered in lab 4.

# References:

* <https://arxiv.org/pdf/2004.14665.pdf> - COVID-19 AND CONTACT TRACING APPS: A REVIEW UNDER THE EUROPEAN LEGAL FRAMEWORK
  + Team of 7 people discuss how various strategies can be followed for contact tracing.
  + Explanation of technologies like BLE and how EphID can help preserve privacy.
  + Usage of DP-3T and its acceptance into frameworks currently now being built by Apple and Google themselves.
* <https://arxiv.org/pdf/2004.10762.pdf> - Contact Tracing: a game of big numbers in the time of COVID-19
  + One of the snippets shown in my paper is from this paper itself where the document shows how and what should trigger from a normally infected to a possibly infected node.
  + The paper also talks about estimating the number of individuals which can be traced successfully assuming people have opted for that service.
* <https://github.com/DP-3T/documents/blob/master/DP3T%20White%20Paper.pdf>
  + Paper on DP3T technologies and its implementation suggestion for “Decentralized privacy-preserving proximity tracing.
* <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1578276/> - An detail study of epidemic
* <https://www.youtube.com/watch?v=vywGgSrGODU> – Android one-meter location accuracy
* <http://www.piezo.ws/piezo_products/Piezo-Patch-Transducer/index.php>
* <https://blog.cloudera.com/an-architecture-for-secure-covid-19-contact-tracing/>
* <https://www.kaggle.com/vbookshelf/respiratory-sound-database> - sample respiratory sound database
* <https://eden.dei.uc.pt/~ruipedro/publications/Conferences/ICBHI2017a.pdf> - paper on the respiratory sound database collection.
* <https://www.kaggle.com/vbookshelf/respiratory-sound-database>
* [https://en.wikipedia.org/wiki/A\*\_search\_algorithm](https://en.wikipedia.org/wiki/A*_search_algorithm)