Max-Min Ant System applied to the Police Patrol Routing Problem

Joachim Tan Yi Ming

Abstract

Amidst escalating crime rates and finite resource constraints, optimising police patrolling strategies emerges as an imperative need. This paper explores similar challenges to the Police Patrol Routing Problem (PPRP) and their solutions. Following that, the PPRP and its existing solutions are explored. The paper identifies gaps in the current solutions and proposes approaching the problem using a combination of Max-Min Ant System (MMAS) and Agent-Based Modelling (ABM). This report details the design, implementation and evaluation of the proposed approach applied to a simulated environment that closely resembles real-world conditions. Key findings show that whilst the performance of the proposed approach is relatively good, the approach struggles to maintain consistent results across all hotspots.

I certify that all material in this dissertation which is not my own work has been identified.

Signature: 

Contents

1 Introduction

The enduring worry over crime remains a top priority for the public [5]. It is unsurprising, given the well-documented links between crime rates, how people perceive crime, and its impact on overall life satisfaction [21]. As indicated by the Crime Survey for England and Wales (CSEW) report released by the Office for National Statistics (ONS), the United Kingdom encountered around 8.4 million offences up to June 2023 [52]. Police agencies have recently been given additional funding of £843 million with the task of improving productivity [72]. While governments devote substantial resources to tackle crime, it becomes clear that the remedy does not solely lie in bolstering police numbers but rather in improving police effectiveness [54].

An effective approach for crime prevention and emergency response in urban areas has been police patrolling [77]. This effectiveness has led to the development and use of many different strategies such as random patrols, problem-oriented policing, community policing, and hotspot policing [77][10]. The concept of community policing revolves around fostering partnerships and building relationships with the public to deter crime. This goal is often pursued by assigning officers to specific areas for extended durations [22]. Hotspot policing on the other hand involves strategically allocating law enforcement resources to locations with elevated levels of criminal activity, allowing law enforcement agencies to have a greater impact [77]. This has been widely recognised for its ability to deliver notable decreases in crime and disorder [9]. While effective, the hotspot policing approach requires an element of randomness and consistency to sustain its effectiveness [70]. When the perception of police presence diminishes or disappears among the public, the observed reduction in crime appears to diminish as well. The current information provided to ground personnel only covers the areas at risk and where patrols should be directed, leaving the planning and creation of patrol routes in the hands of ground personnel [71]. These limitations introduce the risk of diminished perceived police presence and inefficient resource utilisation. Hence, it is imperative that a solution for police patrols that avoids repetition while maintaining a consistently high level of police presence be created.

This emphasises the growing importance of improving police resource utilisation efficiency [72]. However, the development and testing of crime prevention strategies pose challenges due to their costly, time-consuming, and difficult-to-empirically-test nature. Agent-Based Modelling (ABM) offers a promising avenue to address these obstacles [40]. ABM is a method used for simulating complex systems, where individual agents and their interactions within an environment are depicted. In ABM, agents possess autonomy, characterised by unique traits and decision-making capabilities. These agents have the capacity to represent diverse entities, such as police patrol units in this context [33].

The decisions of these agents determine the patrol route and will need to be guided by an algorithm to achieve a police patrol solution that avoids repetition while maintaining a consistently high level of police presence. Various adaptations of the Ant Colony Optimisation (ACO) algorithm have been applied to similar problems, such as the Traveling Salesman Problem (TSP), aiming to find the shortest route to visit multiple locations; the Vehicle Routing Problem (VRP), focused on optimising routes for efficient goods delivery with multiple vehicles; and the Dynamic Vehicle Routing Problem (DVRP), dealing with real-time optimisation of vehicle routes [28][79][48].

Max-Min Ant System (MMAS), a variant of ACO, which is based on the collective foraging behaviour of ants and used for solving optimisation problems shows promise [14]. In the natural world, ants deposit chemical pheromones while foraging for food, marking their path [29]. Other ants detect these pheromones and tend to favour paths with higher concentrations of pheromones. As time passes, the pheromones gradually diminish, increasing the probability that paths frequented by more ants will be chosen by future ants. Through this cyclic process, ants gradually converge upon the most efficient route to the food source. In the context of optimisation problems, MMAS mimics this behaviour. It involves the iterative exploration of potential solutions and the reinforcement of better solutions through dynamically depositing 'pheromones' based on the quality of the solution found. Over time, the algorithm will converge towards the best solution among a set of possibilities.

Therefore, this report explores the use of MMAS to address the limitations of police patrol routing, a problem commonly known as the Police Patrol Routing Problem (PPRP) [67]. The study focuses on examining the quality of solutions from applying an implementation of MMAS integrated with ABM to the PPRP [23]. In this approach, a fleet of police patrol cars operates autonomously, coordinating

within a shared environment to enhance resource allocation efficiency and boost the perceived police presence.

1.1 Project Contributions

This report explores the feasibility and limitations of using ABM coupled with MMAS to aid agents in making decisions to solve PPRP in simulated environments.

The following report describes the implementation of MMAS integrated with ABM applied to the PPRP, within a simulated environment. The report is structured into distinct sections for clarity and coherence. Firstly, a literature review will be conducted to explore similar problems and existing approaches to the problem. Following this, a specification will outline the requirements for this project and define the criteria for success. The algorithm design will be described and substantiated in detail in the design section, showcasing the project's approach. Subsequently, the development section will showcase techniques used in the project, experimental designs and problems encountered, along with the corresponding solutions created. The testing section will demonstrate the implementation of MMAS integrated with ABM applied to the PPRP using a simulated environment, followed by an evaluation of the implementation performance. The critical assessment will offer a comparative examination in relation to the initial specification. Within the critical assessment section, the project's success and constraints will be evaluated, alongside details of any potential future work.

2 Literature Review

2.1 Hotspot Policing

A study conducted by Christophe et al reveals that crime is not uniformly or randomly spread across space [75]. Instead, it tends to cluster in specific areas of elevated crime activity. Similar to geological features, these areas of heightened crime activity are referred to as crime hotspots.

Amongst the various types of police patrolling strategies available, the hotspot policing strategy emerges as the most effective approach for addressing crime hotspots [9]. Hotspot policing is the strategic distribution of law enforcement resources to areas exhibiting heightened levels of criminal activity, thereby enabling law enforcement agencies to exert a more significant influence [78]. The Minneapolis hotspot patrol experiment, led by Weisburd et al, was groundbreaking in presenting compelling evidence on the effectiveness of policing in recognised crime hotspots [65]. It demonstrated that doubling the number of patrols in these areas led to reductions in total calls for service ranging from 6-13%. The experiment concluded that significant increases in police patrol presence can indeed result in modest decreases in crime and more notable reductions in disorder within high-crime locations. This aspect has garnered widespread recognition for its effectiveness in achieving significant reductions in crime and disorder [9]. Although effective, the hotspot policing strategy requires a degree of randomness and consistency to maintain its efficacy [70]. When the public's perception of police presence wanes or vanishes, the observed reduction in crime appears to weaken.

2.2 Identifying Crime Hotspots

Crime hotspots can be identified using methods such as the Spatial and Temporal Analysis of Crime (STAC), Geographic Boundary Thematic Mapping (GBTM) and K-Means algorithm for clustering [41][45]. STAC identifies areas of heightened crime density by initially establishing a grid structure, which can be triangular or rectangular, within the specified area boundary [62]. At each grid intersection, STAC positions overlapping circles with a radius of 1.414 times the specified search radius. Circles encompassing at least two data points are recorded, along with their coordinates and point counts. The top 25 circles with the highest number of data points are then selected. Because the circles overlap, data points can fall within the boundary of multiple circles. In the event of this happening, the circles involved will be combined. This process will continue iteratively until no overlapping circles remain because STAC identifies hotspots without using predefined boundaries like police administrative boundaries. It requires minimal parameters and is compatible with most Geographical Information

System (GIS) applications [16]. However, STAC has a tendency to generate larger clusters, potentially inaccurately reflecting areas of elevated crime, as it combines overlapping circles as part of its identification process [62].

On the other hand, GBTM identifies crime hotspots by delineating boundaries, such as census areas, police beats, or a grid, and shading each section according to the density of crime within it [45]. It establishes a threshold for crime levels, classifying areas that exceed this threshold as hotspots. GBTM allows users to promptly identify areas with high crime rates and facilitates deeper analysis of the issue by focusing on those specific areas [16]. However, due to the diverse sizes and shapes of geographical boundaries, thematic shading can sometimes mislead map readers in identifying the presence of the highest crime concentrations [31]. Consequently, this method may not effectively uncover patterns both across and within geographical divisions [16].

Lastly, the K-Means algorithm is a straightforward and widely employed clustering technique that categorises data into K clusters by measuring the proximity of each data point to the cluster centroid [3]. Initially, it randomly selects k data points from the dataset as the initial cluster centroids. Then, each data point is assigned to the nearest centroid based on a distance metric. Following that, the centroids of the clusters are recalculated by taking the mean of all data points assigned to each cluster. This process iterates until convergence, where either the centroids no longer change significantly or a maximum number of iterations is reached. It efficiently handles large datasets with minimal computational complexity, scaling linearly as the number of data points increases [56]. Utilising K-Means enables the configuration of the cluster count and is effective in clustering large datasets [43]. The clusters derived from K-Means can then be utilised to find hotspot regions in the map [41].

2.3 Agent Based Modelling

After crime hotspots have been identified, the police patrol strategies will have to be evaluated in a simulated environment. ABM stands out as a technique used to simulate complex systems by representing individual agents and their interactions within an environment [34]. ABM can be implemented using the object-oriented paradigm as it provides modularity useful for simulation [15]. Agents, representing police patrol units, are instantiated as objects in ABM [24]. Each agent possesses attributes such as speed, age, and location, which can vary depending on the specific context of the simulation. These attributes can be assigned during instantiation or later as needed. These agents can then be applied in an environment and will be guided by behaviours.

A study by Bosse et al introduced an Agent-Based Modelling (ABM) approach to evaluating different crime prevention strategies [7]. In the study, the environment remained constant whilst the behaviour of these agents varied across the different strategies under investigation. This method allowed for a fair comparison of the effectiveness of each strategy.

Employing ABM enables the management of individual-level attributes, such as the travel speed of each agent or their decision-making algorithms [46]. With this approach, simulated environments are created and populated with agents, strategies can then be developed, tested, and refined prior to real-world testing [33]. ABMs enable researchers to conduct repeated experiments under consistent conditions, with selected variations, making them widely applicable for testing important theories in criminology [4][7][40]. By accommodating the complex and dynamic nature of policing, ABMs possess significant potential as an approach to evaluating different crime prevention strategies.

2.4 Police Patrol Routing Problem

The PPRP presents the challenge of formulating an efficient police patrol strategy that optimally deploys available patrol units while maximising police presence to serve as a strong deterrent against criminal activities [27]. In the PPRP, a fleet of patrol vehicles dispatched from police stations must navigate efficiently through a geographically dispersed set of hotspots, all while adhering to a range of constraints. When using ABM, each patrol unit operates autonomously within the simulated environment, relying on guidance from an algorithm to effectively patrol the identified crime hotspots

[46]. Various algorithms, such as Genetic Algorithms (GA) and ACO have been proposed to address the PPRP and direct patrol units accordingly.

2.5 Ant Colony Optimisation

ACO is a metaheuristic designed for tackling complex optimisation challenges, inspired by the ant pheromone-based communication methods [30]. In ACO, a group of simple agents, or "ants", are randomly placed in the environment. Each ant explores the environment guided by pheromone levels and heuristic information. As the ants explore the environment, they deposit pheromones on paths travelled, which guides future ants. To prevent pheromone buildup, an evaporation process is applied to reduce pheromone levels. Over successive iterations, the algorithm converges towards optimal solutions by reinforcing successful paths and ignoring less successful ones. A paper by Eiben et al demonstrates the importance of customising algorithm parameter settings to attain exceptional performance with ACO algorithms in problem-solving scenarios [32]. These parameter values significantly influence the algorithm's ability to discover near-optimal solutions and do so efficiently. While the focus of the paper is on Evolutionary Algorithms (EA), its insights are also applicable to ACO, given their shared heuristic search-based approach.

MMAS is a variant of the ACO algorithm, characterised by its unique pheromone mechanism [68]. While sharing similarities with ACO, MMAS adopts a stricter approach where only the best solutions are allowed to deposit pheromones during trail updates. Additionally, MMAS introduces explicit constraints on the maximum and minimum trail strengths, denoted as tau max and tau min respectively. These bounds serve to mitigate premature convergence, thereby ensuring a more robust search process. These mechanisms facilitate the exploitation of optimal solutions by gradually intensifying the pheromone trail on favourable solutions, attracting ants to traverse them more frequently. In contrast, less desirable solutions undergo a reduction in pheromone intensity as they evaporate [66]. A study by Stützle et al showcases the advantages of employing MMAS in addressing both the Traveling Salesman Problem (TSP) and the Quadratic Assignment Problem (QAP) [68]. This holds potential significance for resolving the PPRP, as the mechanism prevents premature convergence of the search. This allows us to capitalise on the optimal solutions to generate alternative solutions, potentially resulting in less predictable patrols while still upholding a high perceived police presence.

2.6 Genetic Algorithms

Genetic Algorithms (GA), on the other hand, are optimisation algorithms influenced by the principles of natural selection and genetics [44]. In GAs, every problem presents a diverse range of potential solutions. These solutions undergo recombination and mutation processes, similar to biological genetics, resulting in the generation of new offspring. This iterative process repeats across multiple generations. Each member of the population is given a distinct fitness value, determined by its objective function. Individuals with higher fitness are more likely to mate and produce offspring. This mechanism fosters the emergence of fitter individuals or enhanced solutions in subsequent generations, persisting until the final destination criterion is achieved [47].

A study by Potvin et al, suggests that solutions derived from GAs rival those attained by the most efficient heuristics for the TSP [58]. Additionally, the research underscores the prospect of significant performance improvement via parallel implementation. Nevertheless, achieving satisfactory results with GAs often demands significant computational time, rendering them unsuitable for addressing exceedingly large problems like the 1,000,000-city problem. Given the resemblance between the TSP and the PPRP, GAs hold considerable promise for addressing the challenges posed by the PPRP. Subsequent paragraphs will explore the similarities.

2.7 Travelling Salesman Problem

The TSP is a widely recognised optimisation challenge, tasking a salesman with visiting a set of cities and returning to the starting point via the shortest possible route while visiting each city only once [42]. The primary objective revolves around identifying a solution that minimises the total distance

travelled while adhering to the constraint of visiting each city only once. This problem is often illustrated by using nodes to represent cities and edges to represent the distances between them [66].

The TSP and the PPRP are similar in their core optimisation objectives, revolving around the efficient traversal of designated locations. While the TSP aims to minimise overall distance travelled, the PPRP endeavours to enhance public safety through optimised police patrols. In the PPRP scenario, akin to the cities in the TSP, hotspots demand constant police presence. The PPRP entails creating patrol routes that are less predictable yet maintain a strong perceived police presence. Like the TSP, the PPRP entails finding solutions that balance various constraints and objectives. Recognising these similarities, leveraging the MMAS algorithm or GAs, proven effective in solving the TSP, holds promise in tackling the complexities of the PPRP [68][58].

2.8 Vehicle Routing Problem

The Vehicle Routing Problem (VRP) offers a broader perspective than the TSP and shares commonalities with both the TSP and the PPRP. Falling under the category of combinatorial problems, the VRP faces exponential growth in potential solutions as problem sizes expand [76]. In this scenario, a fleet of vehicles departs from a central depot to serve a geographically scattered group of customers efficiently, all while navigating various constraints [8]. The VRP has many variations, each introducing diverse problem attributes [76].

One notable variation is the Vehicle Routing Problem with Time Windows (VRPTW), which, similar to the VRP, adds complexity by incorporating time windows for servicing each customer. This additional constraint must be managed while optimising the total route distance. A paper by Bräysy et al suggests that the VRPTW can be effectively addressed by using GAs [11]. Another paper by Zhao et al suggests that addressing the VRPTW effectively can be achieved through a modified MMAS approach [80]. Given the similarities between the VRPTW and the PPRP, employing a modified MMAS or GA approach shows promise in addressing the latter.

2.9 Dynamic Vehicle Routing Problem

Another notable variation of the VRP is the Dynamic Vehicle Routing Problem (DVRP). In both problems, a geographically dispersed customer base is served by a fleet of vehicles while meeting predefined constraints. Where the DVRP differs is that it requires real-time adjustments to accommodate dynamic constraints, introducing increased complexity as a result. An example of a dynamic change would be shifting traffic conditions that a fleet will need to adapt its routes to accommodate and maintain optimal operations [74]. Bonilha et al discovered that employing a method integrating MMAS and change-related information effectively tackled the DVRP [6]. Their approach utilised change-related data to refine the best solution acquired from a prior environment, ensuring viability despite dynamic alterations. While their solution proved effective, Bonilha et al hinted at the possibility of investigating alternative, more efficient approaches for utilising change-related information.

2.10 Existing Approaches to Problem

The PPRP, as introduced in Section 2.4, shares similarities with the TSP, VRP, and DVRP. In these similar problems, nodes represent cities or customers, while edges signify the distances between them. Similarly, in the PPRP, nodes depict hotspots, and edges represent the distances between them. The primary aim of the PPRP is to efficiently patrol a set of geographically dispersed crime hotspots.

Approaches such as GAs and MMAS have proven effective in addressing the challenges of the TSP, VRP, and DVRP, indicating their promise for tackling the PPRP as well.

A study by Calvo et al proposed using the MMAS algorithm to address the PPRP problem [14]. The paper found that convergence to an optimal solution was often achieved within 50 iterations, highlighting the algorithm's effectiveness in optimising patrol routes. However, it is noteworthy that the implementation detailed in the paper results in a single route covering all crime hotspots, without accounting for scenarios where multiple police units may patrol simultaneously. Moreover, the paper lacks clarity regarding the methods employed for hotspot identification, and the evaluated metrics

were restricted to cost and time considerations. Utilising ABM would enable the simulation and consideration of scenarios involving multiple police units patrolling simultaneously.

In another study, Chen et al proposed using a combination of elements from ABM, ACO and Bayesian Strategy (BS), a hybrid strategy they referred to as the Bayesian Ant Patrolling Strategy (BAPS) [17]. Chen et al introduced the metric "global average idleness" to gauge success, indicating the duration a hotspot remains unattended by law enforcement. This performance is then contrasted with that of the conventional cycling algorithm. While effective, it is important to note that this strategy uses ACO and could get trapped in a local optima as mentioned in Ming et al's paper [50].

Alternatively, GA was proposed as an effective approach in a study by Reis et al [61]. The approach proposed used a multiagent simulation framework for devising patrol routes in hotspot policing. Although effective in reducing crime, it is important to acknowledge that this method does not prioritise finding the shortest patrol route and requires future crimes to be predicted.

Comparing both solutions derived from using GA and ACO, Ming et al found that ACO tends to become trapped in a local optima, while GA may offer a better solution [50]. However, it is noteworthy that the superior GA solution requires approximately 200 iterations to converge, whereas ACO typically converges in around 50 iterations. Given the limitations of both approaches, MMAS should be considered as it demonstrates the potential for rapid convergence, as evidenced in the study by Calvo et al, and is adept at avoiding local minima. [12][14].

3 Project Specification

The existing literature highlights the limitations of the current approaches, such as GA, ACO and Calvo et al's implementation of MMAS [14].

This project attempts to overcome the current limitations by using the integration of the MMAS algorithm with ABM to tackle the PPRP. The proposed algorithm should create police patrols that avoid repetition while maintaining a consistently high level of police presence. The effectiveness of the presented approach is assessed using a simulated environment that closely resembles real-world conditions. The core objective of this endeavour is to contribute research towards the PPRP, providing research for others to further improve upon for their own investigations and testing purposes.

3.1 Requirements

The environment is meant to simulate real-world conditions to test the project's proposed approach. Therefore, the project's environment creation requirements are:

- The necessary crime data should be sourced from data.police.uk [57].
- The crime data must undergo curation to safeguard individuals' identities and be filtered to exclude any data points lying beyond the project's designated geographical area of focus.
- The processed crime data should be used to pinpoint crime hotspots in order to establish an environment reflective of real-world conditions.
- The approach used to identify crime hotspots should be fast and allow flexible configuration of the number of hotspots identified. This is to allow for the problem environment size to be quickly reduced for parameter tuning.
- The patrol unit travel between crime hotspots should be simulated. This can be achieved either by calculating the shortest path between hotspots or by leveraging road network data to create a dynamic environment where patrol units can explore and determine the most optimal routes.
- The environment should be constructed as a graph comprising interconnected nodes and edges. This will allow the proposed approach to explore the environment.
- The environment should be saved in a format that enables rapid reconstruction, eliminating the need for repetitive creation.

MMAS, as discussed in the literature, demonstrates potential for solving the PPRP. This approach aims to leverage its capability to avoid local minima and create police patrols that avoid repetition while maintaining a consistently high level of police presence Therefore the project's implementation requirements are:

- The MMAS algorithm integrated with ABM implementation should control multiple patrol units asynchronously. The patrol units should be able to traverse and explore the environment efficiently while considering the environment's conditions such as pheromone level and crime hotspot "idle time".
- The algorithm implementation should possess the ability to configure the travel speed of patrol units and simulate its effect.
- The "global average idle time" metric introduced in Section 3.2 should be documented for evaluation purposes, assessing its effectiveness in addressing the PPRP.
- The standard deviation of idle time of each hotspot should be tracked for evaluation purposes, assessing the unpredictability of patrol routes created.
- The results from the implementation should be documented and saved in an appropriate format for evaluation.
- The algorithm implementation decision-making process should be transparent and comprehensible. This can be accomplished by documenting the data sources, detailing the data preprocessing steps, and outlining the development of the algorithm, which includes testing and validation processes.
- The parameter tuning process should be comprehensively documented, elucidating the range of values explored and the criteria employed for assessing performance.
- Any potential challenges the algorithm may face should be documented.
- The code should be well-commented so that it clarifies intricate calculations, formulas, or decision rules. This is essential for facilitating understanding and maintaining transparency throughout the decision-making process.

3.2 Evaluation Criteria

To thoroughly evaluate the project's success, the project will prioritise assessing "global average idle time" as the primary metric. To grasp the concept of "global average idle time", it is essential to define "idle time". In this context, "idle time" refers to the duration since a police patrol unit last visited a hotspot.

Crime hotspots can be represented as $\mathrm{CH} = \{\mathrm{ch}_1, \mathrm{ch}_2, \ldots, \mathrm{ch}_n\}$, where ch_i represents one hotspot and n represents the total number of crime hotspots. While the distances between crime hotspots can be represented as $D = \{d_{\mathrm{ch1ch2}}, d_{\mathrm{ch1ch3}}, d_{\mathrm{ch2ch4}}, d_{\mathrm{ch2ch3}}, d_{\mathrm{ch}_i\mathrm{ch}_i}\}$, where each ch_i within $d_{\mathrm{ch}_i\mathrm{ch}_i}$ corresponds to one of the two hotspots forming the pair.

The "idle time" of a crime hotspot ch_i at time t can be derived as:

$$\mathrm{IT}_{\mathrm{ch}_i} = t - t_{\mathrm{last\_visited(ch}_i)} \tag{1}$$

where $\mathrm{IT}_{\mathrm{ch}_i}$ represents the "idle time" of a crime hotspot ch_i and $t_{\mathrm{last\_visited(ch}_i)}$ represents the last time crime hotspot ch_i was visited by any police patrol unit.

The "average idle time" of hotspot ch_i is calculated as follows:

$$\mathrm{A\_IT}_{\mathrm{ch}_i} = \frac{\sum_{j=1}^{v}(t_j - t_{\mathrm{last\_visit\_before}}(t_j))}{v} \tag{2}$$

where $\text{A\_IT}_{\text{ch}_i}$ represents the "average idle time" of hotspot ch_i, v is the number of visits to hotspot ch_i, t_j represents the time of the j-th visit to hotspot ch_i, and $t_{\text{last\_visit\_before}}(t_j)$ represents the time of the last visit to hotspot ch_i before time t_j. If j is the first visit ($j = 1$), then $t_{\text{last\_visit\_before}}(t_1) = 0$.

The "global average idle time" of all hotspots is calculated as follows:

$$\text{Global\_Average\_Idle\_Time} = \frac{\sum_{i=1}^{n} \text{A\_IT}_{\text{ch}_i}}{n} \tag{3}$$

where $\text{A\_IT}_{\text{ch}_i}$ represents the average idle time of hotspot ch_i, n is the total number of hotspots, and Global_Average_Idle_Time denotes the global average idle time across all hotspots.

In assessing the "global average idle time" metric introduced in the project's design, the project will initially compare its results against a random walk baseline. Should the approach demonstrate efficacy, the project will proceed to conduct a comparative analysis by comparing the outcomes with those achieved through the integration of ACO and ABM, based on the BAPS implementation by Chen et al, as discussed in Section 2.10 [17]. However, due to time constraints inherent to the project, the Bayesian Strategy (BS) component of Chen et al's implementation will be omitted from this implementation. This comparative analysis allows for the assessment of the effectiveness of the solution on the "global average idle time" metric and its alignment with the project's goals. By integrating this metric and conducting a thorough comparative analysis against alternative approaches, our evaluation approach gains depth and breadth. This comprehensive evaluation process allows us to rigorously assess the solution's performance, thereby enhancing the credibility of the project's findings.

In order to assess the unpredictability of patrol routes, the standard deviation of idle time of each hotspot should be tracked. This is because the hotspot policing strategy requires a degree of randomness and consistency to maintain its efficacy [70].

4 Design

4.1 Technology Choice

Python has been selected as the primary programming language for this project owing to its vast library support, which proves advantageous for numerous project requirements [59]. Furthermore, Python's support for object-oriented programming, as highlighted in Section 2.3, makes it particularly well-suited for implementing Agent-Based Modeling (ABM) and other aspects of the project [15].

Key libraries such as GeoPandas, Folium, Pandas, Networkx, Shapely and Matplotlib are among the many available to facilitate development tasks.

In this project, for the visualisation of crime coordinates on a map, Folium, a library designed for generating interactive maps to visualise data is used [63]. To process both crime coordinates and administrative boundary data, GeoPandas, specialising in geospatial data handling, is coupled with Shapely, enabling manipulation and analysis of geometric objects [35][64]. Additionally, Networkx a library designed for the creation and manipulation of complex networks is used to create the environment [26]. For data manipulation, Pandas, known for its data manipulation and data analysis capabilities will be used [53]. Lastly, CSV files are employed for data storage due to their simplicity, lightweight nature, and flexibility, allowing for swift data handling operations as required by the project [13].

4.2 Environment Creation

Historical crime data from February 2021 to January 2024 will be used to pinpoint crime hotspots, accessing the necessary information from data.police.uk, a comprehensive repository of open data concerning crime and policing in England, Wales, and Northern Ireland [57]. This dataset, sourced directly from the police force, undergoes curation to ensure the removal of any personally identifiable information. Even though the data undergoes an initial format validation, the preprocessing stage will eliminate any entries marked as invalid, empty, or unusable to mitigate the risk of hidden irregularities.

Additionally, due to the dataset's nationwide coverage, a focused filtering process is necessary to narrow the scope to the project's specific area of interest, optimising computational efficiency without compromising analytical outcomes.

After filtering, it is expected that these crime occurrences will be spread across the designated area, spanning a significant geographic extent. To effectively identify concentrated crime areas, we will leverage the K-Means clustering algorithm, renowned for its efficiency in managing large datasets [41]. As previously introduced in Section 2.2, this algorithmic approach offers the advantage of handling extensive datasets with minimal computational complexity, scaling linearly with the number of data points [56]. Additionally, K-Means allows for flexible configuration of the cluster count, enhancing its adaptability to varying dataset sizes and structures [43]. Following that, leveraging insights from GBTM outlined in Section 2.2, a threshold for crime levels is established [45]. Areas with concentrated crime surpassing this threshold are designated as crime hotspots.

Once crime hotspots are identified, they should be translated into a graph, where each hotspot is depicted as a node, and the distances between them are depicted as edges. The shortest distances between crime hotspots can be derived using the Google Distance Matrix API [37]. Patrol units will commence their surveillance of crime hotspots from nearby police stations situated within the designated area of interest for the project [25]. Before representing the police stations as nodes on the graph and calculating distances to the crime hotspots, the addresses of these stations need to be converted into geographic coordinates. This conversion is achieved using the Google Geocoding API [36]. Subsequently, the distances between these stations and the crime hotspots are computed utilising the Google Distance Matrix API [37]. Once all essential nodes and edges are incorporated into the graph, the environment is considered prepared. Subsequently, the data utilised for graph construction, including crime clusters and the distances between them, is stored in a CSV file format. This enables quick reconstruction of the environment as necessary, ensuring consistency in its structure and properties.

In this simplified patrol unit travel process, patrol units are presumed to possess a thorough knowledge of the road network in the area of interest. Consequently, they consistently opt for the shortest path between crime hotspots which is derived using the Google Matrix API. The project currently lacks the necessary data to simulate the complete patrol unit travel process, which entails exploring the road network to find the best route to the next crime hotspot. However, this data can be acquired from Ordnance Survey (OS), a reputable provider of data on highways, roads, and waterways [69]. Regrettably, the project has been unable to purchase this data from OS due to budget constraints. Obtaining this data from OS could allow future studies to simulate the entire patrol unit travel process.

4.3 MMAS Algorithm Design

Following that, the MMAS algorithm will be applied to asynchronously guide multiple police patrol units through the environment, efficiently covering crime hotspots with the objective of reducing the "global average idle time" metric, as discussed in Section 3.2. The patrol units should be able to traverse and explore the environment efficiently while considering the environment's conditions such as pheromone level and crime hotspot "idle time" as outlined in Section 3.1.

Adhering to the MMAS principle, a user-defined number of ants are deployed in the implementation. Each ant constructs a solution, and subsequently, the best ant deposits pheromones on the edges it traversed. This process iterates until either the "global average idle time" converges or the maximum number of iterations is reached. Each ant constructs a solution by deploying a user-defined number of agents or police patrol units in this case, functioning as autonomous entities within the framework of agent-based modelling (ABM) introduced in Section 2.3.

Given its complexity, the implementation should be structured using classes to encapsulate related functionalities and enhance code modularity [60]. This approach would also be useful for implementing ABM as discussed in Section 2.3 [15]. Four classes can be used for this purpose: Agent, Node, Edge, and Environment. The Node class tracks attributes such as "idle time", number of visits from patrol units, and "average idle time" for each node in the graph created earlier. Meanwhile, the Edge class is responsible for tracking attributes such as pheromone level and distance between the origin and

destination nodes for each edge in the graph created earlier. The Agent class tracks each patrol unit's current location, travel time left and the nodes it has visited. On the other hand, the Environment class serves as the hub for setting up the environment using all the classes created. It holds all the algorithm settings, including parameters such as tau min and tau max, the number of ants, the number of agents, the maximum iterations for the algorithm, the maximum iterations for the agents, the minimum iterations agents need to explore the environment after visiting all crime hotspot nodes before convergence can occur, the pheromone evaporation rate, alpha, beta, Q, designation of police station nodes and patrol unit travel speed in metres per minute. Additionally, the implementation of the algorithm will reside in the Environment class.

During initialisation, the Environment class leverages the Node class to create all crime and police station nodes, while employing the Edge class to establish connections between these nodes by creating edges and setting the initial pheromone level to the tau max value defined by the user. Additionally, the specified number of agents is created using the Agent class during initialisation with the current node attribute set at random to one of the available police station nodes defined by the user. Once this is done, the agents are looped for the maximum iterations for the agents defined by the user. In each iteration, a nested for loop is used to iterate over all the agents to simulate the agent exploring the environment. When this happens, each agent will select the next node to visit if its travel time attribute which represents the amount of distance left to travel is 0. The selection of the next node is determined by a combination of factors: the idle time of the potential node, adjusted by the user-defined beta parameter, and the pheromone levels left by previous ants on the corresponding edge, adjusted by the user-defined alpha parameter. The probability of each node being the next node represented as $\mathrm{P_{nextnode}}$ is calculated as such:

$$P_{\text{nextnode}} = \frac{\mathrm{IT}_{\text{pnn}}^{\beta} \times \mathrm{PL}_{\text{pnn}}^{\alpha}}{\sum_{i=1}^{n} \left(\mathrm{IT}_{i}^{\beta} \times \mathrm{PL}_{i}^{\alpha} \right)} \tag{4}$$

Where $\mathrm{IT_{pnn}}$ represents the "idle time" of a potential next node *pnn*, $^{\beta}$ represents beta, $\mathrm{PL_{pnn}}$ represents pheromone levels on the corresponding edge of the potential next node *pnn*, $^{\alpha}$ denotes alpha and n denotes the total number of hotspots excluding the police station nodes. By taking these factors into account, the implementation can prioritise hotspots with low idle time and paths that have historically led to favourable outcomes, while also adjusting the weighting of each factor in the consideration process.

After selecting the next node, the agent updates its travel time attribute with the distance of the corresponding edge connected to the selected node. Additionally, the selected node is appended to the list of nodes visited by the agent. If the travel time attribute is not zero, the new travel time can be calculated and updated as follows: travel time = travel time - patrol unit travel speed. Here, the patrol unit travel speed, defined by the user in metres per minute, dictates the duration of each iteration, with each iteration representing a minute of simulated time. For this project, the patrol unit travel speed adheres to the speed limit within England, set at 800 m/min [73]. This setup aligns with the project requirements, allowing for the management of multiple patrol units asynchronously. These units traverse and explore the environment efficiently, taking into account factors such as pheromone levels and crime hotspot "idle time" as specified. Upon reaching a travel time of zero once more, before the selection of the subsequent node, the idle time of the current node is reset to 1 to indicate the patrol unit's arrival. This reset ensures that no node will have a probability of 0 during the node selection process, as the idle time starts at 1 instead of 0.

After ensuring that all crime hotspots have been visited at least once and meeting the minimum iterations required for agents to explore the algorithm post-visiting all crime hotspots, tracking of the "global average idle time" commences, alongside the initiation of convergence monitoring. A minimum iteration period is established to facilitate the stabilisation of the "global average idle time" metric value. The minimum iteration period is user-defined and used to set the number of minimum iterations before checking for convergence. This period accounts for a transient phase where the global idleness initially tends to be low as mentioned in Chen et al's paper [17]. For this project, the convergence criteria are user-defined and set to 5 consecutive iterations without any improvement in the "global average idle time". If convergence is achieved before reaching the maximum agent

iterations defined, the loop will terminate. This loop iterates for the number of ants specified by the user, with each iteration representing the movement of an ant in the MMAS. Additionally, this loop is nested within another loop that iterates for the maximum number of algorithm iterations defined by the user. After each iteration, the best ant deposits Q pheromones defined by the user on the edges visited by its agents, while the pheromones on all edges undergo evaporation according to the user-defined evaporation rate, maintaining the tau min and tau max values specified by the user. This approach helps mitigate premature convergence, thereby ensuring a more robust search process. It facilitates the exploration of optimal solutions, as discussed in Section 2.5, enhancing the algorithm's effectiveness in finding high-quality solutions. The implementation assesses convergence after each iteration, utilising the current best "global average idle time" as a reference point. The convergence criteria is user-defined and set to 5 consecutive iterations without any improvement in the "global average idle time". If convergence is achieved before reaching the maximum algorithm iterations defined, the loop will terminate. By checking for convergence and terminating early, unnecessary iterations are avoided, resulting in reduced processing time and saved computational resources. A diagram depicting the algorithm is provided in Figure 1 for clarity.

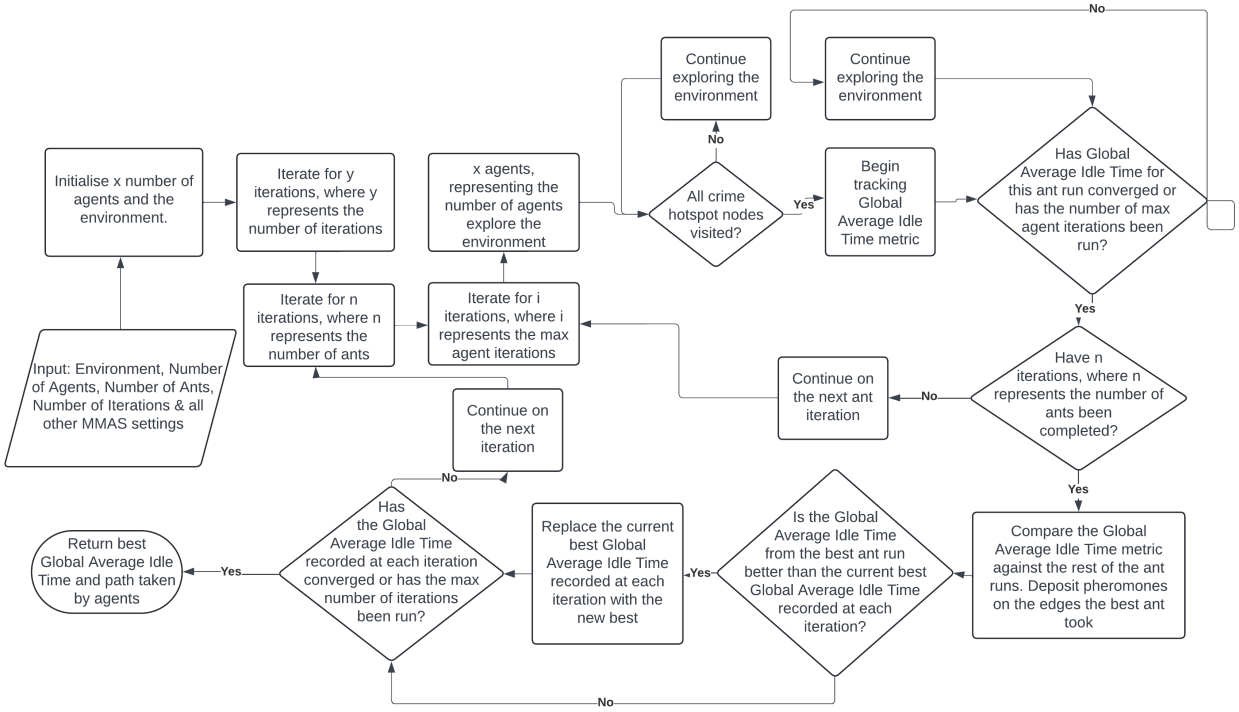

Figure 1: Algorithm Diagram

5 Development

5.1 Environment Creation

This subsection delineates the process of crafting a simulation environment mirroring real-world conditions within the City of London. As mentioned in the design section, Python (version 3.11.4) serves as the primary programming language for this project. Various Python libraries, including Geopandas, Folium, Pandas, Networkx, and Matplotlib, are employed to support development tasks.

The initial step involves acquiring crime data from data.police.uk, provided in CSV format. Subsequently, filtering is conducted to isolate files relevant to the City of London. Only the latitude, longitude, and crime type headers are extracted from these filtered files for each data point. This selection ensures that the extracted data cannot be used to identify any individuals, aligning with the project's requirements. Each data point is assigned a severity score based on the type of crime committed, as outlined in Figure 5. One challenge encountered during the filtering process is the

presence of irregularities likely stemming from human error during file creation by its provider. The irregularities surfaced upon plotting the coordinates of all data points on a scatter plot, as depicted in Figure 2. To support these observations, the coordinates were further visualised using Folium, as introduced in the design section and illustrated in Figure 3. These data points outside the area of interest (The City of London) could affect the result of the project and will need to be removed. Initially, the z-score for each data point was computed to identify and remove outliers. However, it was observed that employing the z-score method might not be the most suitable approach for spatial data. This method measures the deviation of a data point from the mean in terms of standard deviations, which may not precisely capture spatial outliers, as illustrated in Figure 4 [2]. Instead, the outliers of interest are those determined by spatial context.

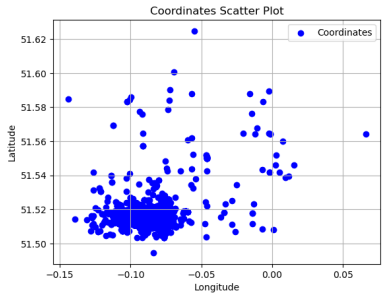

Figure 2: Scatter Plot of Crime Outliers

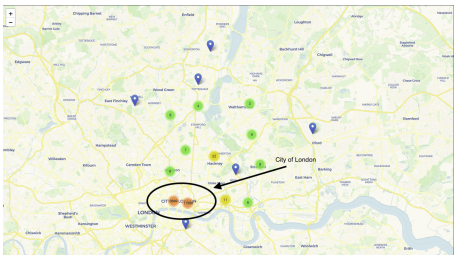

Figure 3: Visualisation of Crime Outliers

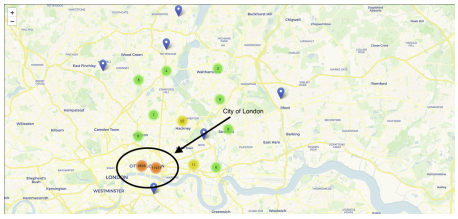

Figure 4: Folium visualisation after using z-score to remove outliers

Crime Type	Score
Violence and sexual offences	5
Possession of weapons	4
Robbery	4
Burglary	3
Theft from the person	3
Criminal damage and arson	3
Drugs	2
Public order	2
Vehicle crime	2
Bicycle theft	2
Shoplifting	1
Other theft	1
Anti-social behaviour	1
Other crime	1

Figure 5: Crime Severity Scores

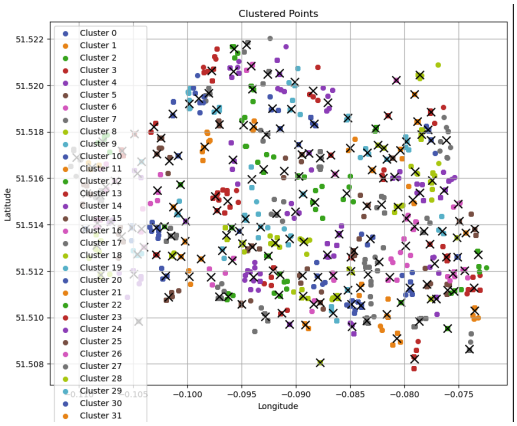

Figure 6: Scatter Plot of 200 clusters from K-Means

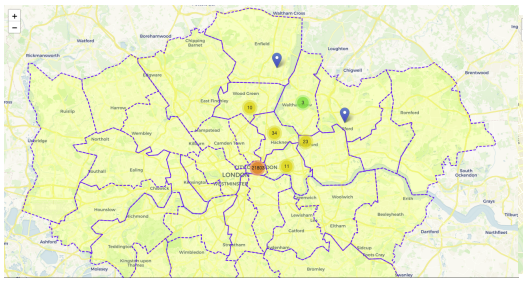

Figure 7: London Ward Borders added to Folium map

Figure 8: City of London Border with outliers removed

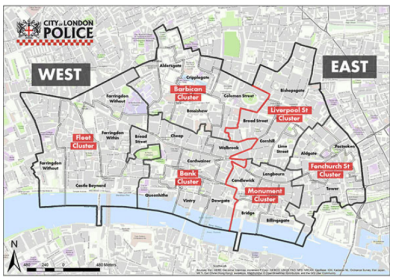

Figure 9: City of London Border [20]

Hence, an alternative approach for outlier removal was adopted. The project integrated London Ward Border GeoJSON data provided by Stuart K. Grange into the map generated using Folium, as depicted in Figure 7 [39]. Geopandas, introduced in the Design section, facilitated the reading

of the GeoJSON data. Subsequently, the borders underwent filtering to isolate the City of London border exclusively. This refined border delineation served as the basis for excluding all outliers lying beyond its confines, illustrated in Figure 8. Since the data is not sourced directly from a government entity, validation of the City of London border was performed by cross-referencing it with the outline provided on the City of London Police's website, depicted in Figure 9 [20].

The processed dataset comprising of 20,624 data points with 650 unique coordinates is utilised to identify crime hotspots by employing the K-Means clustering algorithm introduced in Section 2.2. This algorithm is imported from the "sklearn.cluster" module, facilitates the identification of these hotspots by grouping similar data points based on their coordinates. Subsequently, the K-Means algorithm is employed to identify 200 clusters, as depicted in Figure 6. Each cluster is characterised by attributes such as "data", which stores individual data points within the cluster, "total severity," calculated by summing the severity of all data points in the cluster, and "centroid coordinates", computed as the mean of all coordinates within the cluster. The "centroid coordinates" are represented as "X" in the scatter plot.

To identify crime hotspots, the project has established a threshold of 100 points. Clusters exceeding this threshold in their "total severity" value are classified as crime hotspots. The chosen threshold of 100 points is based on the analysis of the crime data spanning a three-year period (February 2021 to January 2024). This duration yields an average severity level of approximately 33 points per year. For instance, if all crimes within a cluster have a severity score of 5, it would take around 6 crimes per year to reach a total severity level of 30, which would not be considered significant. However, defining a crime hotspot remains a complex task, and the threshold value may vary depending on the context. As such, the threshold value is selected based on pragmatic considerations due to a lack of comprehensive research on the topic. It remains dynamic and subject to adjustments based on future research and analysis. After identifying the crime hotspots, they are saved as CSV files. This allows for the reconstruction of the same crime hotspot clusters, ensuring consistency in the environment across different tests.

Following that, the crime hotspot clusters are grouped into batches of 10, with any remainder forming a separate batch. These batches are then utilised to calculate the shortest distance between each crime hotspot cluster's "centroid coordinates" using the Google Distance Matrix API [37]. As mentioned in Section 4.2, in this simplified patrol unit travel process, it is assumed that patrol units have a comprehensive understanding of the road network in the area of interest, allowing them to consistently select the shortest path between crime hotspots. The resulting distances are saved to CSV files to mitigate excessive usage of the API, which has request limitations for free use [38]. Additionally, the API imposes constraints such as a maximum of 25 origins or destinations per request and a maximum of 100 elements per server-side request. To adhere to these limits, the crime hotspot clusters were grouped into batches of 10.

Once the shortest distances between crime hotspot clusters have been determined, the addresses of two police stations located within the City of London are geocoded using the Google Geocoding API to obtain their geographical coordinates [18][36]. Subsequently, these coordinates are employed to compute the distance between the police stations and each crime hotspot cluster utilising the Google Distance Matrix API referenced earlier.

After obtaining both crime hotspot clusters and their distances, Networkx, a Python library introduced in Section 4.1, is employed to construct a graph consisting of interconnected nodes and edges. Each crime hotspot and police station is represented as a node, while the distances between them are depicted as edges within the graph. Before being incorporated as edge weights, the distances are converted from kilometres to metres. Once all essential nodes and edges are incorporated into the graph, the environment is considered prepared.

5.2 Algorithm Creation

This section documents the implementation of the algorithm design outlined in Section 4.3. After establishing the environment, the algorithm has to be implemented. The initial step would be to create the four classes introduced in Section 4.3, Environment, Agent, Node and Edge. One of the initial hurdles encountered was navigating the NetworkX graph with the agents. While NetworkX

offers numerous functions for graph creation and manipulation, implementing custom functionality can be challenging. Tasks like resetting the environment and introducing pheromones on edges and tracking "idle time" on nodes are particularly complex to implement with NetworkX. To overcome this, instead of implementing classes to encapsulate NetworkX functionality, classes are used to recreate the environment from the NetworkX graph to allow for easier implementation of custom functionality. This is implemented using Node & Edge classes. The Node class is created with several attributes such as "node_id", "idle_time", "num_visits" and "all_idle_time" to track the idle time across all visits. The Edge class is created with attributes such as "edge_id", "pheromone" and "distance" representing the distance between two nodes. The environment is recreated from the NetworkX graph using the Node & Edge classes by instantiating a Node object for each node in the graph and instantiating an Edge object for each edge in the graph. The "edge_id" attribute is assigned a tuple "(origin node, destination node)" to identify the nodes the edge belongs to. Once this is done the environment is recreated.

To explore the new environment with agents, an Agent class is created. The Agent class is created with several attributes such as "agent_id" to identify different agents, "travel_time" to track the amount of distance left for the agent to arrive at the next node and if the agent is still travelling, "current_node" to track the current node of the agent and "path" to track the nodes visited by the agent. At initialisation, the "current_node" is set to a random police station node to simulate the process of a real world police patrol where police patrols start from police stations [25]. To explore the environment, a method "select_next_node" is created. The method guides the agent by calculating the $P_{nextnode}$ and using it to choose the next node for the agent to visit. However, during the search through neighbours of the current node using edges, inconsistencies were observed in the number of neighbours, resulting in some nodes lacking neighbours or having fewer neighbours than expected. Upon further inspection it was found that NetworkX by default creates only one edge between two nodes [51]. Any attempt to add a new edge between two nodes that already have an edge connecting them, will cause the existing edge to be replaced with the new one. As a result, all edges created from the NetworkX graph are only linked one way from origin node to destination node. To overcome this limitation, when searching for neighbouring nodes using the edges, the current node is assumed to be either the destination node or origin node in the "edge_id" tuple rather than just the origin node. Once an "edge_id" is found to contain the current node in its tuple, the other node is considered its neighbouring node and the $P_{nextnode}$ of the node is calculated. This is possible because the edges are undirected.

To simulate a node being visited, the Node class features a method named "visit" which updates the attributes "all_idle_time" with the "idle_time" since the last visit and the "num_visits" attribute which tracks the number of visits the node has had from patrol units. However, an initial period of low idle time was observed before stabilisation, consistent with Chen et al's findings regarding a transitional phase characterised by reduced global idleness, as discussed in Section 4.3 [17]. To address this issue, a mechanism was developed to regulate the tracking of "idle_time" and "num_visits". The "idle_time" and "num_visits" will only be tracked when all the nodes have been visited at least once. This period ensures the stabilisation of the "global average idle time" value before tracking begins, acknowledging the transitional phase characterised by low global idleness.

Once the Agent, Node and Edge classes are created, the Environment class can be created. The Environment class acts as a "controller" class, orchestrating the interactions between the various classes. The Environment class has all the attributes needed for MMAS such as Q, Alpha, Beta, evaporation_rate, tau_min, tau_max and number of ants. In addition, it also has attributes such as number of agents, graph assigned the NetworkX graph, max iterations, agent_max_iterations, agent_min_iterations, convergence_threshold and travel_speed. The Environment class is used to instantiate the Agent class according to the number of agents specified and create the environment using the NetworkX graph by instantiating the Node and Edge classes. The Environment class is also where the MMAS algorithm is implemented using the methods of the Agent, Node and Edge classes. One challenge encountered involved saving the paths taken by each agent in a CSV file. Initially, all agent paths were intended to be stored in a single CSV file cell. However, this approach led to file corruption when attempting to open it due to the cell character limit [49]. To address this issue, a method was

developed to split the agent path data into individual cells for each agent, ensuring successful storage without file corruption.

Once all the classes are created and the algorithm is implemented in accordance to Figure 1 and the design in Section 4.3, it is considered complete. The algorithm is then capable of traversing the created environment.

5.3 Tuning of Parameters

The implementation of the MMAS algorithm integrated with ABM involves several parameters essential for its functioning. These include the graph representing the environment, the number of ants, maximum iterations, evaporation rate, alpha, beta, Q value, agent maximum iterations, agent minimum iteration, tau min, tau max, patrol unit travel speed, police station nodes, and convergence threshold. Tuning the parameters such as the number of ants, evaporation rate, alpha, beta, Q, tau min, and tau max is crucial to optimise the algorithm's performance across various scenarios with different numbers of agents as seen in Figure 10. This optimisation process was facilitated using the Grid Search method, which automates parameter tuning. Instead of manually experimenting with diverse parameter combinations, Grid Search systematically explores a predefined set of parameter values, effectively creating a grid of potential configurations. This approach simplifies the optimisation process by automatically testing each combination, thus saving time and effort. By evaluating the model's performance across the grid, Grid Search identifies the optimal parameter combination that enhances the model's effectiveness [1]. Considering the constraints of computational resources and time available for computation, parameter tuning is conducted on a subset of the environment. This subset is generated using the same procedures outlined in Section 5.1 for environment creation. However, instead of identifying 200 clusters with K-Means, only 20 clusters are identified and used to create the environment. Tuning the parameters on this reduced subset not only diminishes the computational resources demanded but also minimises the time required for parameter adjustment. The values tested for each parameter using grid search are displayed in Figure 10. The evaporation rate controls the decay of pheromone trails over time, striking a balance between exploration and exploitation. Alpha and beta dictate the relative importance of pheromone levels and heuristic information in decision-making, with alpha prioritising pheromones and beta focusing on heuristic information. Q determines the amount of pheromone deposited by ants, affecting trail intensity. Tau min establishes a minimum pheromone level on edges, ensuring essential information retention in less-travelled paths. Meanwhile, tau max imposes an upper limit on pheromone levels, preventing premature convergence and encouraging adaptability and exploration [55]. Each combination undergoes computation five times, and the average of these five results is compared against the results from all other combinations to determine the best combination. This iterative process is conducted across various scenarios with different numbers of agents, leveraging the current number of available police officers assigned to patrol the City of London. With 21 officers currently assigned, the project explores scenarios where the number of agents ranges from half to four times this count, as depicted in Figure 11 [19].

num_ants	evaporation_rate	alpha	beta	Q	tau_min	tau_max
10	0.1	1.0	1.0	0.1	0.1	1.0
30	0.3	2.0	2.0	0.5	0.5	5.0
50	0.5			1.0	1.0	10.0
70	0.7			1.5		
90	0.9			2.0		

Figure 10: Parameter Grid for Grid Search

Num of Agents
10
21
60
80

Figure 11: Number of Agents

Parameter	Value
max_iterations	100
agent_max_iterations	10000
agent_min_iteration	1000
travel_speed	800
convergence_threshold	5

Figure 12: Constant Parameters

The constant parameter values utilised during this process are detailed in Figure 12. The agent minimum iteration parameter fosters exploration and ensures the stabilisation of the "global average idle time", acknowledging the transitional phase mentioned in Section 4.3. Meanwhile, the agent maximum iteration ensures computational efficiency by restricting the number of steps agents can take in the environment. Travel speed is set according to the speed limit within England, capped at

800 m/min [73]. The convergence threshold, set to 5 consecutive iterations without any improvement in the "global average idle time," ensures computational efficiency. Likewise, the max iteration parameter enhances computational efficiency by limiting the algorithm's runtime. The optimal MMAS parameter combination discovered for each scenario with varying numbers of agents is detailed in Figure 13.

Num of Agents	num_ants	evaporation_rate	alpha	beta	Q	tau_min	tau_max
10	30	0.3	2.0	2.0	0.1	0.1	10.0
21	90	0.3	2.0	2.0	0.1	0.1	10.0
60	70	0.1	2.0	2.0	0.1	0.1	10.0
80	10	0.1	1.0	1.0	0.1	0.1	1.0

Figure 13: Best MMAS Parameters for Different Number of Agents tuned on Environment Subset

Num of Agents	num_ants	evaporation_rate	alpha	beta	Q
10	70	0.5	2.0	2.0	2.0
21	90	0.9	1.0	1.0	0.5
60	10	0.1	1.0	1.0	0.1
80	10	0.1	1.0	1.0	0.1

Figure 14: Best ACO Parameters for Different Numbers of Agents tuned on Environment Subset

The variations observed in the best parameters across different scenarios with the same problem size indicate the sensitivity of the algorithm's performance to slight variations in parameter settings. This sensitivity suggests that even for the same problem size, the optimal configuration of parameters may vary based on the amount of agents patrolling. It underscores the importance of thorough parameter tuning and experimentation to achieve optimal performance across different scenarios.

The parameter grid described in Figure 10, excluding tau min and tau max, is employed to fine-tune the ACO integrated with ABM implementation within the 20-cluster environment, as introduced in Section 3.2. Furthermore, the constant parameters outlined in Table 4 are incorporated for consistency. This setup acts as a benchmark for evaluating the performance of the MMAS integrated with ABM, as presented in the project. The optimal parameter combinations identified for different scenarios with varying numbers of agents are elaborated upon in Figure 14.

6 Project Evaluation

Agents	MMAS	ACO	Random Walk
10	62.05	63.29	100.77
21	29.89	31.47	48.68
60	11.39	11.24	17.62
80	9.98	9.02	13.47

Figure 15: Global average idle time from the average of 5 simulation using ACO, MMAS & Random Walk (Value in minutes)

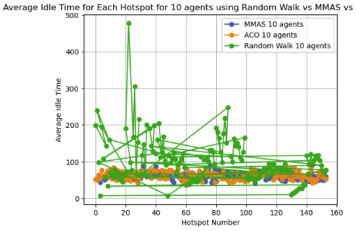

Figure 16: MMAS vs ACO vs Random Walk Average Idle Time of each hotspot with 10 agents

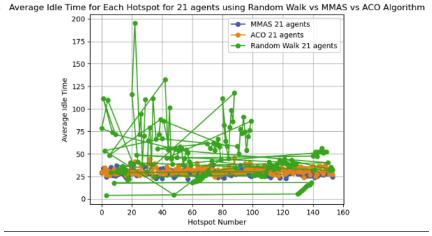

Figure 17: MMAS vs ACO vs Random Walk Average Idle Time of each hotspot with 21 agents

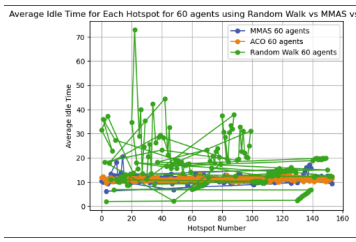

Figure 18: MMAS vs ACO vs Random Walk Average Idle Time of each hotspot with 60 agents

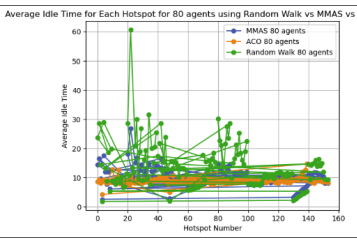

Figure 19: MMAS vs ACO vs Random Walk Average Idle Time of each hotspot with 80 agents

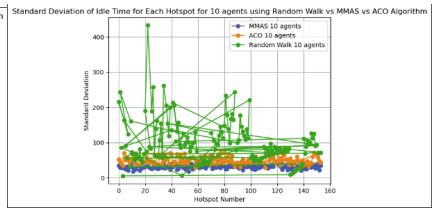

Figure 20: MMAS vs ACO vs Random Walk Standard Deviation of each hotspot idle time with 10 agents

The optimal parameters identified for various numbers of agents fine-tuned on a subset of the environment, are subsequently implemented across the entire environment. The "global average idle time"

resulting from averaging five runs for each scenario and algorithm is shown in Figure 15. It is noticed that the results obtained from using the Random Walk algorithm (in green) is much worse compared to ACO and MMAS. This disparity is further highlighted in Figures 16 to 19, where the "average idle time" across each hotspot, maintained using the Random Walk algorithm, consistently ranks highest among all algorithms across various numbers of patrolling agents. Similarly, Figures 20 to 23 demonstrate that the standard deviation of idle time across each hotspot is also highest when employing the Random Walk algorithm (in green), underscoring its unpredictability. While maintaining some degree of patrol randomness is essential for maintaining the crime reduction effect in hotspot patrolling, the observed high "average idle time" across hotspots is suboptimal [70].

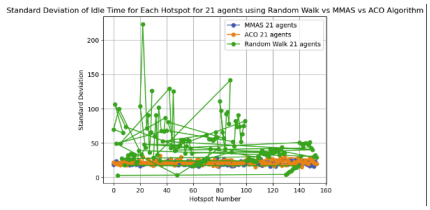

Figure 21: MMAS vs ACO vs Random Walk Standard Deviation of each hotspot idle time with 21 agents

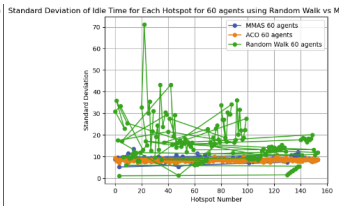

Figure 22: MMAS vs ACO vs Random Walk Standard Deviation of each hotspot idle time with 60 agents

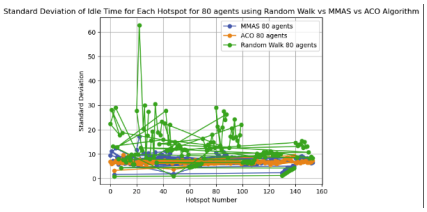

Figure 23: MMAS vs ACO vs Random Walk Standard Deviation of each hotspot idle time with 80 agents

Given that using MMAS has achieved better results compared to the Random Walk algorithm, the MMAS should be compared to ACO as mentioned in Section 3.2. The results shown in Figure 15 indicate that the MMAS performs better than the ACO in maintaining a low "global average idle time" with 10 & 21 agents. However, the ACO performs slightly better with 60 & 80 agents. It is noted that the results from both algorithms only differ by an average of 1.21 minutes. This is not significant enough to impact crime rates given that in 2022-2023 there were 7908 crimes committed within the City of London [57]. Divided equally, this would equate to approximately one crime per hour.

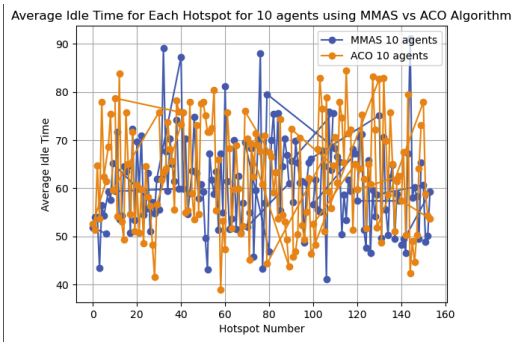

Figure 24: MMAS vs ACO Average Idle Time of each hotspot with 10 agents

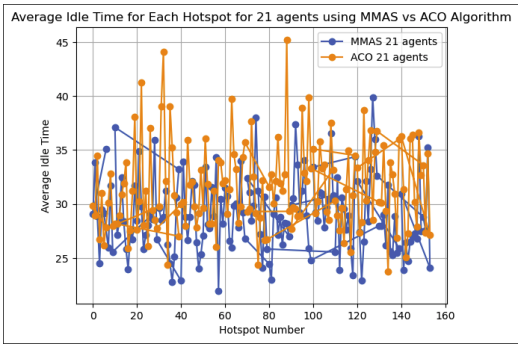

Figure 25: MMAS vs ACO Average Idle Time of each hotspot with 21 agents

Considerable variability in the average idle time across hotspots for both MMAS (in blue) and ACO (in orange) is observed with 10 & 21 agents patrolling, as evidenced in Figure 24 & 25. Both algorithms show significant overlap, indicating that performances are quite similar across the various hotspots. This could indicate an issue of overextended patrol units and a need for more patrol agents to achieve a consistent low average idle time across all hotspots. The standard deviation for both algorithms is seen in Figure 26 & 27. The ACO algorithm shows generally higher variability in idle times across most hotspots compared to the MMAS algorithm. The variability for both algorithms is quite high, suggesting that the idle times for hotspots can fluctuate significantly under both algorithms. While the average idle times are similar between the two algorithms, the ACO algorithm has more variability in its performance as seen by the higher standard deviation. This means that while the

average times may be similar, the ACO algorithm is less consistent. In this case, the ACO would be preferred over the MMAS as its randomness aligns more closely with the objective.

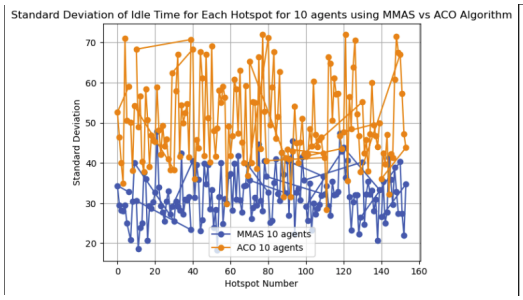

Figure 26: MMAS vs ACO Standard Deviation of each hotspot idle time with 10 agents

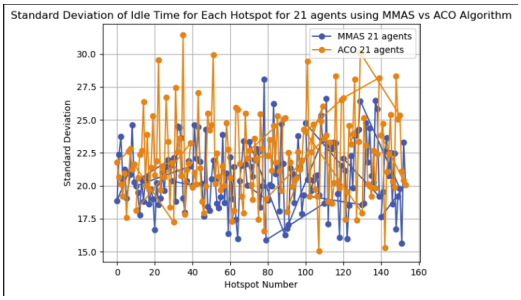

Figure 27: MMAS vs ACO Standard Deviation of each hotspot idle time with 21 agents

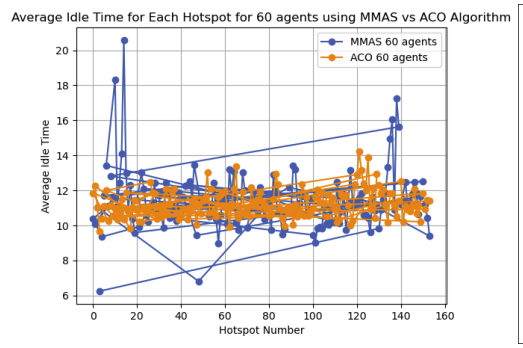

Figure 28: MMAS vs ACO Average Idle Time of each hotspot with 60 agents

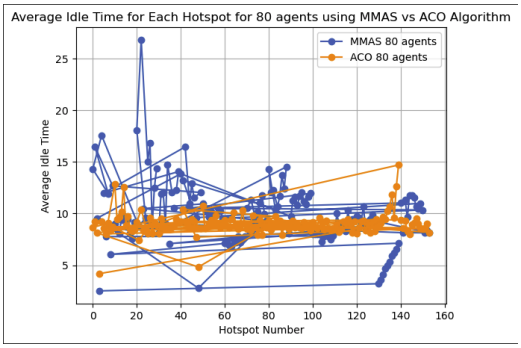

Figure 29: MMAS vs ACO Average Idle Time of each hotspot with 80 agents

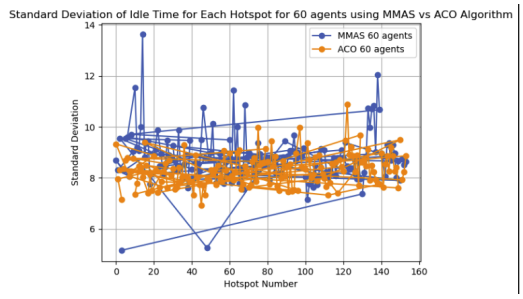

Figure 30: MMAS vs ACO Standard Deviation of each hotspot idle time with 60 agents

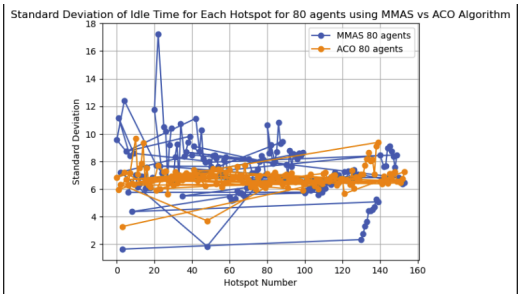

Figure 31: MMAS vs ACO Standard Deviation of each hotspot idle time with 80 agents

As mentioned previously, there could be an issue of overextension of patrol units and a need for more patrol agents to achieve a consistently low average idle time across hotspots. In Figures 28 and 29, fluctuations in the average idle time across different hotspots are observed when using 60 & 80 agents for both algorithms. The variability seems to be relatively similar for both algorithms, yet MMAS exhibits more pronounced and frequent spikes, suggesting greater inconsistency. This pattern holds for the standard deviation as well, where both algorithms exhibit fluctuations in standard deviation with MMAS exhibiting sharper and more frequent spikes. It is observed that increasing the number of agents from 10,21,60 and 80 leads to lower average idle times and lower standard deviation, implying better consistency for both algorithms. Although both algorithms have a similar range of average idle times across all hotspots, MMAS's tendency towards frequent sharps spikes introduces

inconsistency in average idle times across hotspots that could make it less than ideal for real-world use. It is also observed that both algorithms have a similar range of standard deviation of idle times across all hotspots with MMAS having more frequent and more pronounced spikes that indicate a randomness that is inconsistent across hotspots. These findings suggest that ACO with consistent standard deviation and average idle time would potentially yield better results when used for police patrols.

7 Critical Assessment

The creation of the environment began with the use of crime data from data.police.uk, complemented by Google's Distance Matrix API for determining distances between crime hotspots. Before the data was used, the data was curated, removing any identifying information and excluding data outside the defined area of interest. K-Means clustering was then employed for hotspot identification, chosen for its efficiency and adaptability in determining the desired number of clusters. The resulting dataset facilitated the creation of a graph representing the environment, saved in CSV format for efficient reconstruction. Consequently, all prerequisites for environment creation were successfully met.

The specified requirements called for the implementation of the MMAS algorithm integrated with ABM to manage multiple agents asynchronously, exploring an environment that mirrors real-world conditions. This objective has been effectively accomplished. Leveraging the inherent foraging behaviour of ants, the MMAS algorithm efficiently navigated the simulated environment. ABM complemented MMAS, enabling the simulation of autonomous police patrol units navigating the environment while considering crucial factors like edge pheromone levels and crime hotspot "idle time". The inclusion of distance data facilitated the addition of travel speed into the algorithm, allowing for realistic exploration simulations. The tracking of "global average idle time" as the primary metric, stored in CSV format, facilitated the evaluation of the proposed solution's effectiveness in addressing the PPRP. Consequently, all implementation requirements have been met.

One of the primary challenges encountered during this project was obtaining the distances between each crime hotspot using the Google Distance Matrix API. The API poses constraints on the number of coordinates permitted in each API call and the total number of calls allowed within the free tier. To circumvent these limitations, a batching algorithm was devised. This algorithm grouped crime hotspot coordinates into batches of 10, aligning with usage restrictions. By doing so, it minimised the number of API calls required to fetch the necessary data and optimised computational efficiency.

In this work, the integration of the MMAS algorithm with ABM was introduced to tackle the PPRP. The main objective was to minimise the "global average idle time" while maintaining a level of patrol randomness. Its performance has been compared against the Random Walk algorithm and the ACO algorithm. The results indicate that it performs much better than the random walk algorithm and achieves similar results to the ACO algorithm. It is however noted that the proposed algorithm is less stable compared to ACO and has frequent sharp spikes in standard deviation and hotspot "average idle time".

7.1 Limitations

While the project has made significant progress towards meeting its defined objectives, there is the opportunity for further improvement with additional development time. It is important to recognise certain key limitations in the current project. Firstly, the current environment assumes that patrol units possess complete knowledge of the road network in the area of interest and always choose the shortest path between crime hotspots. Consequently, the simulation does not replicate the process of finding the shortest path between hotspots but relies on data obtained from the Google Distance Matrix API. This limits the algorithm's ability to explore the environment dynamically based on the paths it chooses, restricting it to identifying the most optimal sequence for visiting hotspots rather than adapting its exploration strategy dynamically. However, despite this limitation, the algorithm's results remain valuable as Google's Distance Matrix API provides accurate distance information between locations. Moreover, it enables patrol units to leverage their domain expertise in a real-world context.

Another constraint in the project is the absence of consideration for the severity levels associated with each crime hotspot during patrol planning. Each hotspot is treated equally without any distinction based on severity. However, in real-world scenarios, hotspots with higher severity levels should receive more frequent patrols due to their greater potential to negatively impact overall life satisfaction, as discussed in Section 1. However, addressing this limitation falls outside the scope of the current project.

When interpreting the results and assessing the project's suitability for real-world police patrolling scenarios, it is important to acknowledge these limitations. Future research and development endeavours should prioritise resolving these constraints to enhance the implementation's effectiveness and applicability.

7.2 Future Improvements

Implementing solutions to address the limitations discussed in the previous section would greatly improve the project's overall effectiveness and mitigate existing constraints. One critical area for improvement is the simulation of finding the shortest path between hotspots. This could be facilitated by acquiring data from Ordnance Survey (OS), which includes information on highways, roads, and waterways [69]. Using this data, an environment could be constructed for agents to explore, effectively simulating the process of finding the shortest path between hotspots. This improvement would allow the model to investigate various paths between hotspots and identify specific roads to traverse when travelling between hotspots. This contrasts with the current outcome of the order of hotspots to be visited, with the shortest path between each hotspot derived from the Google Distance Matrix API.

To address the lack of consideration for severity levels associated with each crime hotspot during patrol planning, a solution similar to the approach explored by Chen et al could be implemented. This involves introducing distinct rates where idle time increases corresponding to different severity levels of hotspots [17]. By implementing this approach, hotspots with higher idle time rates would receive heightened attention, as their higher idle time rates would cause their idle time to increase faster. Consequently, these hotspots would exhibit a greater likelihood of being selected as the next node to visit, as described by the equation for P_{nextnode} in Section 4.3.

Finally, additional research should be conducted to investigate the cause of frequent spikes in the "average idle time" and standard deviation when using MMAS.

8 Conclusion

In this report, the main objective of the project has been to explore the application of MMAS integrated with ABM to the PPRP. The literature review has extensively explored similar challenges with the PPRP, offering valuable insights into the methodologies used to address these challenges and introducing existing solutions for the PPRP. Through this analysis, gaps in current approaches are identified, highlighting the issues this project seeks to address. The effectiveness of the proposed algorithm is evaluated using the steps outlined in Section 3.2. By conducting a comparative analysis against the Random Walk algorithm and the ACO integrated with ABM, as implemented by Chen et al, the project gains valuable insights into the effectiveness of the proposed solution. [17].

In summary, this project has successfully applied the MMAS integrated with the ABM algorithm to the PPRP, contributing to advancing understanding of this approach's feasibility in tackling the problem. By creating an environment that closely mimics real-world conditions, simulating travel time and using actual distances between crime hotspots, the project offers valuable insights into police patrolling strategies. While not replicating real-world scenarios perfectly, it provides a realistic simulation that sheds light on effective patrolling approaches. The outcomes of this project open avenues for future research, allowing for further exploration and a deeper understanding of police patrolling strategies in simulated real-world conditions. Notably, the results demonstrate the limitations of the proposed algorithm compared with using ACO, suggesting that ACO could be a better solution to the PPRP.

References

[1] *3.2. Tuning the hyper-parameters of an estimator*. https://scikit-learn.org/stable/modules/grid_search.html. (accessed: 28.03.2024).

[2] Chittaranjan Andrade. "Z Scores, Standard Scores, and Composite Test Scores Explained". In: *Indian Journal of Psychological Medicine* 43.6 (2021), pp. 555–557. DOI: 10.1177/02537176211046525. (accessed: 22.03.2024).

[3] Preeti Arora, Shipra Varshney, et al. "Analysis of k-means and k-medoids algorithm for big data". In: *Procedia Computer Science* 78 (2016), pp. 507–512.

[4] Daniel Birks and Toby Davies. "Street Network Structure and Crime Risk: An Agent-Based Investigation of the Encounter and Enclosure Hypotheses". In: *Criminology* 55 (Nov. 2017), p. 900. DOI: 10.1111/1745-9125.12163. (accessed: 5.03.2024).

[5] Robin M. Blakely. "Community Issues in New York State: What's Important?" In: *Rural New York Minute, Cornell University* Issue Number 7 (July 2007). (accessed: 05.03.2024).

[6] Iaê S. Bonilha et al. "Ant Colony optimization with Heuristic Repair for the Dynamic Vehicle Routing Problem". In: *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. 2020, pp. 313–320. DOI: 10.1109/SSCI47803.2020.9308156. (accessed: 16.03.2024).

[7] Tibor Bosse and Charlotte Gerritsen. "Comparing Crime Prevention Strategies by Agent-Based Simulation". In: *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 02*. WI-IAT '09. USA: IEEE Computer Society, 2009, pp. 491–496. ISBN: 9780769538013. DOI: 10.1109/WI-IAT.2009.200. URL: https://doi.org/10.1109/WI-IAT.2009.200. (accessed: 16.03.2024).

[8] Kris Braekers, Katrien Ramaekers, and Inneke Van Nieuwenhuyse. "The vehicle routing problem: State of the art classification and review". In: *Computers Industrial Engineering* 99 (2016), pp. 300–313. ISSN: 0360-8352. DOI: https://doi.org/10.1016/j.cie.2015.12.007. URL: https://www.sciencedirect.com/science/article/pii/S0360835215004775. (accessed: 16.03.2024).

[9] A.A. Braga and D.L. Weisburd. "Does Hot Spots Policing Have Meaningful Impacts on Crime? Findings from An Alternative Approach to Estimating Effect Sizes from Place-Based Program Evaluations". In: *Journal of Quantitative Criminology* 38.1 (2022), pp. 1–22. DOI: 10.1007/s10940-020-09481-7. URL: https://doi.org/10.1007/s10940-020-09481-7. (accessed: 05.03.2024).

[10] Anthony A Braga. "The Effects of Hot Spots Policing on Crime". In: *The ANNALS of the American Academy of Political and Social Science* 578.1 (2001), pp. 104–125. (accessed: 05.03.2024).

[11] Olli Bräysy, Wout Dullaert, and Michel Gendreau. "Evolutionary Algorithms for the Vehicle Routing Problem with Time Windows". In: *Journal of Heuristics* 10 (2004), pp. 587–611. DOI: 10.1007/s10732-005-5431-6. URL: https://doi.org/10.1007/s10732-005-5431-6. (accessed: 17.03.2024).

[12] Felix Broberg and Emelie Eriksson. *Comparing MAX-MIN and Rank-based Ant Colony Optimization Algorithms for solving the University Course Timetabling Problem*. http://www.diva-portal.org/smash/get/diva2:1214470/FULLTEXT01.pdf. 2018. (accessed: 17.03.2024).

[13] ByteScout. *CSV Format: History, Advantages*. URL: https://bytescout.com/blog/csv-format-history-advantages.html. (accessed: 24.03.2024).

[14] Hiram Calvo et al. "Patrolling Routes Optimization Using Ant Colonies". In: *Pattern Recognition*. Ed. by Jesús Ariel Carrasco-Ochoa et al. Cham: Springer International Publishing, 2015, pp. 302–312. ISBN: 978-3-319-19264-2. (accessed: 16.03.2024).

[15] Christian JE Castle and Andrew T Crooks. "Principles and concepts of agent-based modelling for developing geospatial simulations". In: (2006). (accessed: 05.04.2024).

[16] Spencer Chainey, Luke Tompson, and Stefan Uhlig. "The Utility of Hotspot Mapping for Predicting Spatial Patterns of Crime". In: *Security Journal* 21 (2008), pp. 4–28. DOI: 10.1057/palgrave.sj.8350066. URL: https://doi.org/10.1057/palgrave.sj.8350066. (accessed: 15.03.2024).

[17] H. Chen, T. Cheng, and S. Wise. "Designing Daily Patrol Routes for Policing Based on Ant Colony Algorithm". In: *ISPRS Annals of Photogrammetry, Remote Sensing, and Spatial Information Sciences* II-4/W2 (2015), pp. 103–109. DOI: 10.5194/isprsannals-II-4-W2-103-2015. (accessed: 16.03.2024).

[18] City of London Police. *City of London Police - Community Policing*. URL: https://www.cityoflondon.police.uk/a/your-area/city-of-london/city-of-london/community-policing/?tab=Overview&yourlocalteam=nearest-police-stations. (accessed: 23.03.2024).

[19] City of London Police. *Neighbourhood Policing*. https://www.cityoflondon.police.uk/police-forces/city-of-london-police/areas/city-of-london/about-us/about-us/neighbourhood-policing/. (accessed: 10.04.2024).

[20] City of London Police. *Sector Policing*. URL: https://www.cityoflondon.police.uk/sectorpolicing. (accessed: 23.03.2024).

[21] Mark A. Cohen. "The Effect of Crime on Life Satisfaction". In: *The Journal of Legal Studies* 37.S2 (June 2008). DOI: 10.1086/588220. URL: https://doi.org/10.1086/588220. (accessed: 05.03.2024).

[22] Office of Community Oriented Policing Services (COPS). *Community Policing Defined*. NCJ 226882. Apr. 2009. URL: https://www.ojp.gov/ncjrs/virtual-library/abstracts/community-policing-defined-0. (accessed: 05.03.2024).

[23] Andrew T. Crooks and Alison J. Heppenstall. "Introduction to Agent-Based Modelling". In: *Agent-Based Models of Geographical Systems*. Ed. by Alison J. Heppenstall et al. Dordrecht: Springer Netherlands, 2012, pp. 85–105. ISBN: 978-90-481-8927-4. DOI: 10.1007/978-90-481-8927-4_5. URL: https://doi.org/10.1007/978-90-481-8927-4_5. (accessed: 11.03.2024).

[24] Scott De Marchi and Scott E Page. "Agent-based models". In: *Annual Review of political science* 17 (2014), pp. 1–20. (accessed: 05.04.2024).

[25] *Deployment Base Data*. https://www.met.police.uk/foi-ai/metropolitan-police/d/april-2022/deployment-base/. (accessed: 19.03.2024).

[26] NetworkX developers. *NetworkX Documentation*. NetworkX. 2021. (accessed: 25.03.2024).

[27] Maite Dewinter et al. "Analysing the Police Patrol Routing Problem: A Review". In: *ISPRS International Journal of Geo-Information* 9.3 (2020). ISSN: 2220-9964. DOI: 10.3390/ijgi9030157. URL: https://www.mdpi.com/2220-9964/9/3/157. (accessed: 16.03.2024).

[28] M. Dorigo and L.M. Gambardella. "Ant colony system: a cooperative learning approach to the traveling salesman problem". In: *IEEE Transactions on Evolutionary Computation* 1.1 (1997), pp. 53–66. DOI: 10.1109/4235.585892. (accessed: 16.03.2024).

[29] Marco Dorigo, Mauro Birattari, and Thomas Stutzle. "Ant colony optimization". In: *IEEE Computational Intelligence Magazine* 1.4 (2006), pp. 28–39. DOI: 10.1109/MCI.2006.329691. (accessed: 16.03.2024).

[30] Marco Dorigo, Mauro Birattari, and Thomas Stutzle. "Ant colony optimization". In: *IEEE Computational Intelligence Magazine* 1.4 (2006), pp. 28–39. DOI: 10.1109/MCI.2006.329691. (accessed: 05.04.2024).

[31] John Eck et al. "Mapping Crime: Understanding Hot Spots". In: *Eck, J.E. and Chainey, S. and Cameron, J.G. and Leitner, M. and Wilson, R.E. (2005) Mapping crime: understanding hot spots. Research report. National Institute of Justice, US.* (Aug. 2005). (accessed: 15.03.2024).

[32] A.E. Eiben, R. Hinterding, and Z. Michalewicz. "Parameter control in evolutionary algorithms". In: *IEEE Transactions on Evolutionary Computation* 3.2 (1999), pp. 124–141. DOI: 10.1109/4235.771166. (accessed: 05.04.2024).

[33] Joshua M Epstein. "Agent-based computational models and generative social science". In: *Complexity* 4 (1999), pp. 41–60. DOI: 10.1002/(SICI)1099-0526(199905/06)4:5<41::AID-CPLX9>3.0.CO;2-F. URL: https://doi.org/10.1002/(SICI)1099-0526(199905/06)4:5%3C41::AID-CPLX9%3E3.0.CO;2-F. (accessed: 13.03.2024).

[34] Joshua M Epstein and Robert Axtell. *Growing artificial societies: social science from the bottom up*. Brookings Institution Press, 1996. (accessed: 5.03.2024).

[35] GeoPandas Development Team. *GeoPandas Documentation*. 2022. URL: https://geopandas.org/en/stable/. (accessed: 25.03.2024).

[36] Google Developers. *Google Geocoding API*. Accessed: 2024. URL: https://developers.google.com/maps/documentation/geocoding/overview. (accessed: 18.03.2024).

[37] Google Developers. *Google Maps Distance Matrix API Documentation*. https://developers.google.com/maps/documentation/distance-matrix/overview. 2022. (accessed: 18.03.2024).

[38] Google Developers. *Usage and Billing — Distance Matrix API — Google Developers*. 2022. URL: https://developers.google.com/maps/documentation/distance-matrix/usage-and-billing. (accessed: 22.03.2024).

[39] Stuart K. Grange. *London Ward Border Data*. https://skgrange.github.io/data.html. (accessed: 22.03.2024).

[40] Elizabeth Groff and Dan Birks. "Simulating Crime Prevention Strategies: A Look at the Possibilities". In: *Policing: A Journal of Policy and Practice* 2.2 (June 2008), pp. 175–184. ISSN: 1752-4512. DOI: 10.1093/police/pan020. eprint: https://academic.oup.com/policing/article-pdf/2/2/175/7287988/pan020.pdf. URL: https://doi.org/10.1093/police/pan020. (accessed: 13.03.2024).

[41] Gaurav Hajela, Meenu Chawla, and Akhtar Rasool. "A Clustering Based Hotspot Identification Approach For Crime Prediction". In: *Procedia Computer Science* 167 (2020). International Conference on Computational Intelligence and Data Science, pp. 1462–1470. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2020.03.357. URL: https://www.sciencedirect.com/science/article/pii/S1877050920308231. (accessed: 13.03.2024).

[42] Karla L Hoffman, Manfred Padberg, Giovanni Rinaldi, et al. "Traveling salesman problem". In: *Encyclopedia of operations research and management science* 1 (2013), pp. 1573–1578. (accessed: 16.03.2024).

[43] Zhexue Huang. "A fast clustering algorithm to cluster very large categorical data sets in data mining." In: *Dmkd* 3.8 (1997), pp. 34–39. (accessed: 16.03.2024).

[44] Annu Lambora, Kunal Gupta, and Kriti Chopra. "Genetic Algorithm- A Literature Review". In: *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*. 2019, pp. 380–384. DOI: 10.1109/COMITCon.2019.8862255. (accessed: 16.03.2024).

[45] Jason Leigh, Steven Dunnett, and Luke Jackson. "Predictive police patrolling to target hotspots and cover response demand". In: *Annals of Operations Research* 283 (2019), pp. 395–410. DOI: 10.1007/s10479-017-2528-x. URL: https://doi.org/10.1007/s10479-017-2528-x. (accessed: 15.03.2024).

[46] Michael Macy and Robert Willer. "From Factors to Actors: Computational Sociology and Agent-Based Modeling". In: *Annual Review of Sociology - ANNU REV SOCIOL* 28 (Aug. 2002), pp. 143–166. DOI: 10.1146/annurev.soc.28.110601.141117. (accessed: 5.03.2024).

[47] Tom V Mathew. "Genetic algorithm". In: *Report submitted at IIT Bombay* 53 (2012). URL: https://www.researchgate.net/publication/342020605_Genetic_Algorithm_Reviews_Implementations_and_Applications. (accessed: 17.03.2024).

[48] Michalis Mavrovouniotis and Shengxiang Yang. “Ant algorithms with immigrants schemes for the dynamic vehicle routing problem”. In: *Information Sciences* 294 (2015). Innovative Applications of Artificial Neural Networks in Engineering, pp. 456–477. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2014.10.002. URL: https://www.sciencedirect.com/science/article/pii/S0020025514009815. (accessed: 16.03.2024).

[49] Microsoft Corporation. *Excel Specifications and Limits*. Microsoft Support. 2022. URL: https://support.microsoft.com/en-us/office/excel-specifications-and-limits-1672b34d-7043-467e-8e27-269d656771c3. (accessed: 07.04.2024).

[50] Qiu Mingyue and Zhang Xueying. “Determining Accurate Patrol Routes Using Genetic Algorithm and Ant Colony”. In: *Automatic Control and Computer Sciences* 57 (2023), pp. 337–347. DOI: 10.3103/S0146411623040065. URL: https://doi.org/10.3103/S0146411623040065. (accessed: 16.03.2024).

[51] NetworkX Developers. *NetworkX Documentation: add_edge*. https://networkx.org/documentation/stable/reference/classes/generated/networkx.Graph.add_edge.html\#networkx.Graph.add_edge. (accessed: 07.04.2024).

[52] Office for National Statistics. *Crime in England and Wales: year ending June 2023*. 2023. URL: https://www.ons.gov.uk/peoplepopulationandcommunity/crimeandjustice/bulletins/crimeinenglandandwales/yearendingjune2023. (accessed: 05.03.2024).

[53] Pandas Development Team. *Pandas Documentation*. https://pandas.pydata.org/docs/. (accessed: 06.04.2024).

[54] J. Parra Domínguez, I.M. García Sánchez, and L. Rodríguez Domínguez. “Relationship between police efficiency and crime rate: a worldwide approach”. In: *European Journal of Law and Economics* 39 (2015), pp. 203–223. DOI: 10.1007/s10657-013-9398-8. URL: https://doi.org/10.1007/s10657-013-9398-8. (accessed: 05.03.2024).

[55] Paola Pellegrini, Daniela Favaretto, and Elena Moretti. “On MAX–MIN ant system’s parameters”. In: vol. 4150. Sept. 2006. ISBN: 978-3-540-38482-3. DOI: 10.1007/11839088_18. (accessed: 29.03.2024).

[56] DT Pham et al. “Data clustering using the bees algorithm”. In: (2007). URL: https://www.researchgate.net/profile/Sameh-Otri/publication/241767604_Data_Clustering_Using_the_Bees_Algorithm/links/0a85e5380fa6334e6b000000/Data-Clustering-Using-the-Bees-Algorithm.pdf. (accessed: 15.03.2024).

[57] Police UK. *Police UK Documentation*. URL: https://data.police.uk/docs/. (accessed: 18.03.2024).

[58] Jean-Yves Potvin. “Genetic algorithms for the traveling salesman problem”. In: *Annals of Operations Research* 63 (1996), pp. 337–370. DOI: 10.1007/BF02125403. URL: https://doi.org/10.1007/BF02125403. (accessed: 17.03.2024).

[59] Python Software Foundation. *Python 3.11.4 Release*. https://www.python.org/downloads/release/python-3114/. (accessed: 21.03.2024).

[60] Python Software Foundation. *Python Documentation: Classes*. 2022. URL: https://docs.python.org/3/tutorial/classes.html. (accessed: 24.03.2024).

[61] Danilo Reis et al. “GAPatrol: An Evolutionary Multiagent Approach for the Automatic Definition of Hotspots and Patrol Routes”. In: vol. 4140. Jan. 2006, pp. 118–127. ISBN: 978-3-540-45462-5. DOI: 10.1007/11874850_16. (accessed: 16.03.2024).

[62] Carolyn Rebecca Block Richard Block and Ned Levine. *Chapter 8: Hot Spot Analysis of Points II*. Online. Chicago, IL; Houston, TX: Loyola University, Illinois Criminal Justice Information Authority, Ned Levine & Associates, 1994. URL: https://nij.ojp.gov/sites/g/files/xyckuh171/files/media/document/CrimeStat%2520IV%2520Chapter%25208.pdf. (accessed: 05.04.2024).

[63] Rob Story. *Folium Documentation*. https://python-visualization.github.io/folium/latest/. (accessed: 24.03.2024).

[64] Sean Gillies and Shapely contributors. *Shapely Documentation*. https://shapely.readthedocs.io/en/stable/. (accessed: 06.04.2024).

[65] Lawrence W Sherman and David Weisburd. "General deterrent effects of police patrol in crime "hot spots": A randomized, controlled trial". In: *Justice Quarterly* 12.4 (1995), pp. 625–648. DOI: 10.1080/07418829500096221. (accessed: 15.03.2024).

[66] T. Stutzle and H. Hoos. "MAX-MIN Ant System and local search for the traveling salesman problem". In: *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC '97)*. 1997, pp. 309–314. DOI: 10.1109/ICEC.1997.592327. (accessed: 16.03.2024).

[67] Thomas Stützle and Holger H. Hoos. "MAX–MIN Ant System". In: *Future Generation Computer Systems* 16.8 (2000), pp. 889–914. ISSN: 0167-739X. DOI: 10.1016/S0167-739X(00)00043-1. URL: https://doi.org/10.1016/S0167-739X(00)00043-1. (accessed: 11.03.2024).

[68] Thomas Stützle and Holger H. Hoos. "MAX–MIN Ant System". In: *Future Generation Computer Systems* 16.8 (2000), pp. 889–914. ISSN: 0167-739X. DOI: https://doi.org/10.1016/S0167-739X(00)00043-1. URL: https://www.sciencedirect.com/science/article/pii/S0167739X00000431. (accessed: 16.03.2024).

[69] Ordnance Survey. *OS Detailed Path Network*. 2024. URL: https://www.ordnancesurvey.co.uk/products/os-detailed-path-network. (accessed: 19.03.2024).

[70] Bruce Taylor, Christopher Koper, and Daniel Woods. "A randomized controlled trial of different policing strategies at hot spots of violent crime". In: *Journal of Experimental Criminology* 7 (June 2011), pp. 149–181. DOI: 10.1007/s11292-010-9120-6. (accessed: 16.03.2024).

[71] The Rt Hon Kit Malthouse MP. *Forces given funding boost to increase roll out of Hotspot Policing*. Sept. 2021. URL: https://www.gov.uk/government/news/forces-given-funding-boost-to-increase-roll-out-of-hotspot-policing. (accessed: 05.03.2024).

[72] UK Government. *Police to get funding boost to cut crime and keep public safe*. https://www.gov.uk/government/news/police-to-get-funding-boost-to-cut-crime-and-keep-public-safe. 2023. (accessed: 12.03.2024).

[73] UK Government. *Speed limits*. URL: https://www.gov.uk/speed-limits. (accessed: 28.03.2024).

[74] Marlin W Ulmer et al. "Dynamic vehicle routing: Literature review and modeling framework". In: (2017). URL: https://www.researchgate.net/profile/Barrett-Thomas-2/publication/313421699_Dynamic_Vehicle_Routing_Literature_Review_and_Modeling_Framework/links/5b1fd72fa6fdcc69745cdb11/Dynamic-Vehicle-Routing-Literature-Review-and-Modeling-Framework.pdf. (accessed: 16.03.2024).

[75] Vandeviver, Christophe and Bernasco, Wim. *The geography of crime and crime control*. eng. Ed. by Vandeviver, Christophe and Bernasco, Wim. 2017. URL: %7Bhttp://doi.org/10.1016/j.apgeog.2017.08.012%7D. (accessed: 13.03.2024).

[76] Thibaut Vidal, Gilbert Laporte, and Piotr Matl. "A concise guide to existing and emerging vehicle routing problem variants". In: *European Journal of Operational Research* 286.2 (2020), pp. 401–416. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2019.10.010. URL: https://www.sciencedirect.com/science/article/pii/S0377221719308422. (accessed: 16.03.2024).

[77] David Weisburd and John E. Eck. "What Can Police Do to Reduce Crime, Disorder, and Fear?" In: *The ANNALS of the American Academy of Political and Social Science* 593.1 (2004), pp. 42–65. DOI: 10.1177/0002716203262548. URL: https://journals.sagepub.com/doi/full/10.1177/0002716203262548. (accessed: 05.03.2024).

[78] David Weisburd and Cody W Telep. "Hot Spots Policing: What We Know and What We Need to Know". In: *Journal of Contemporary Criminal Justice* 30.2 (2014), pp. 200–220. DOI: 10.1177/1043986214525083. URL: https://doi.org/10.1177/1043986214525083. (accessed: 05.04.2024).

[79] Bin Yu, Zhong-Zhen Yang, and Baozhen Yao. "An improved ant colony optimization for vehicle routing problem". In: *European Journal of Operational Research* 196.1 (2009), pp. 171–176. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2008.02.028. URL: https://www.sciencedirect.com/science/article/pii/S0377221708002373. (accessed: 16.03.2024).

[80] Gang Zhao et al. "A Modified Max-Min Ant System for Vehicle Routing Problems". In: (2008), pp. 1–4. DOI: 10.1109/WiCom.2008.1507. (accessed: 16.03.2024).

9 Acknowledgements

The author would like to thank Dr. Ayah Helal for guidance in this project.

10 Appendix

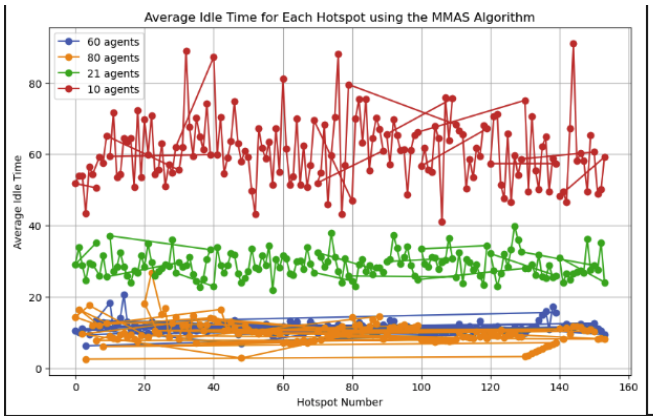

Figure 32: MMAS Average Idle Time of each hotspot with different number of agents

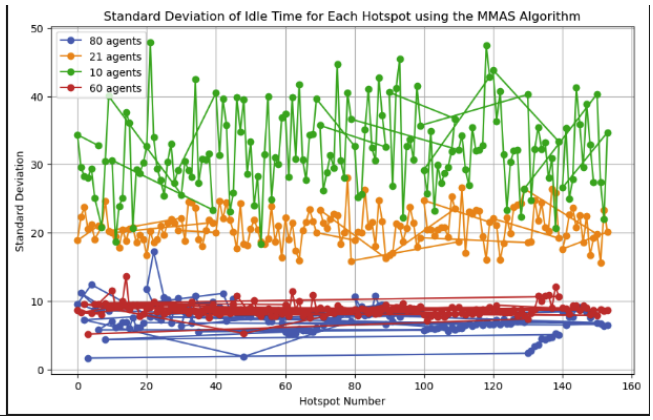

Figure 33: MMAS Standard Deviation of each hotspot idle time with different number of agents

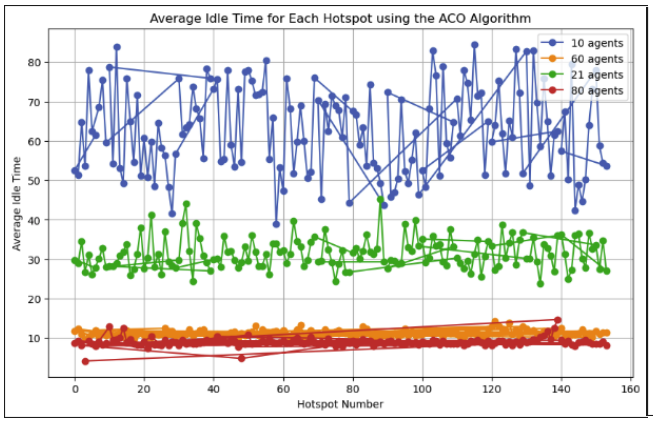

Figure 34: ACO Average Idle Time of each hotspot with different number of agents

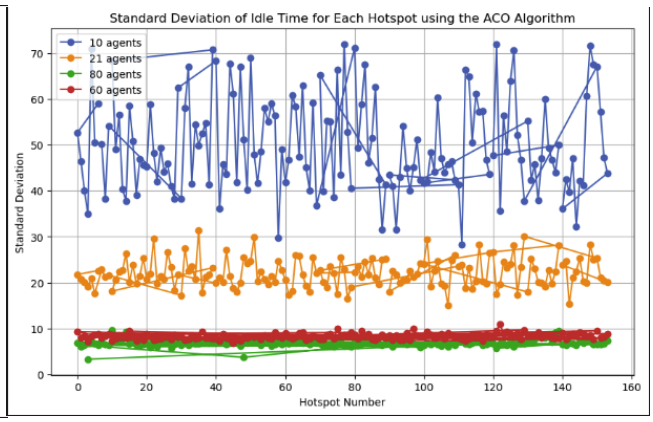

Figure 35: ACO Standard Deviation of each hotspot idle time with different number of agents

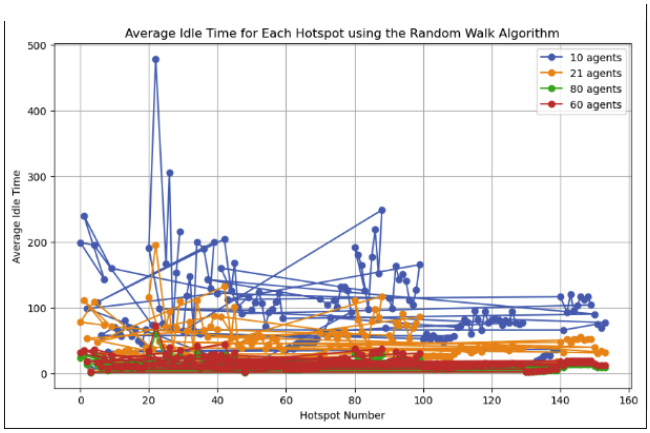

Figure 36: Random Walk Average Idle Time of each hotspot with different number of agents

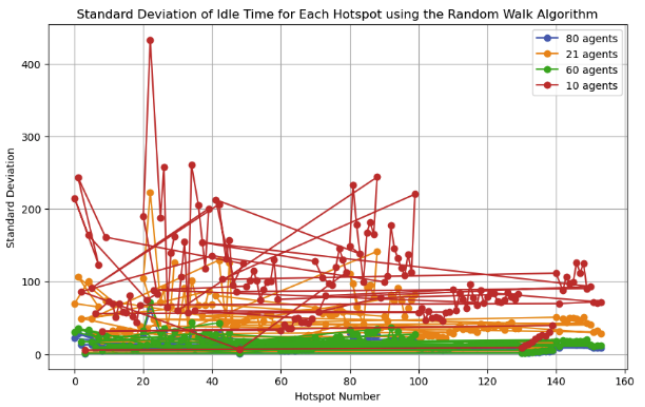

Figure 37: Random Walk Standard Deviation of each hotspot idle time with different number of agents

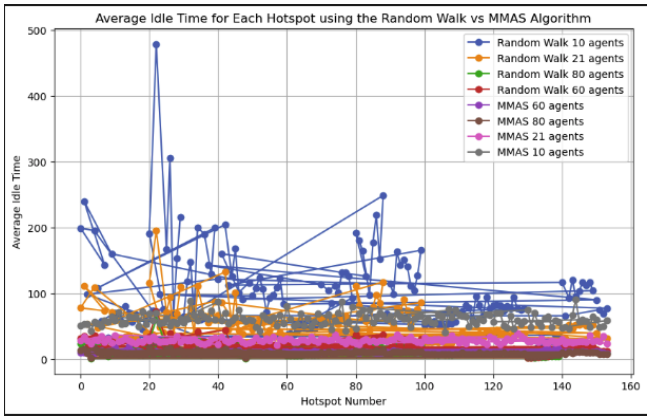

Figure 38: MMAS vs Random Walk Average Idle Time of each hotspot with different number of agents

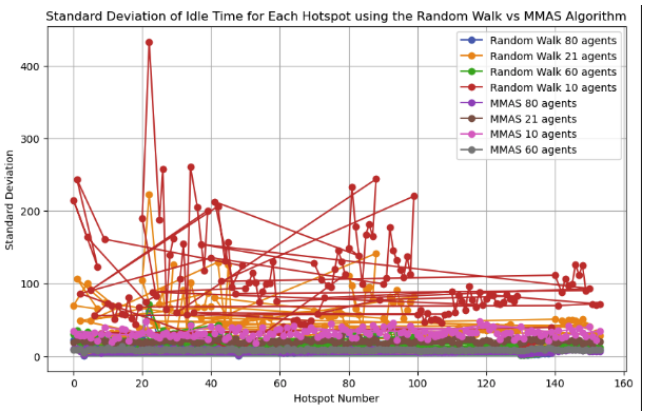

Figure 39: MMAS vs Random Walk Standard Deviation of each hotspot idle time with different number of agents

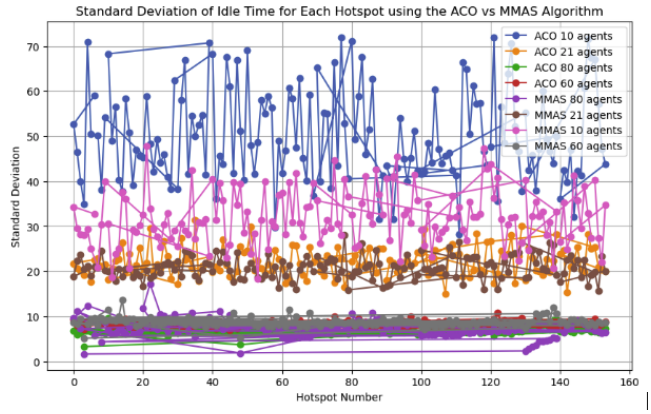

Figure 40: MMAS vs ACO Average Idle Time of each hotspot with different number of agents

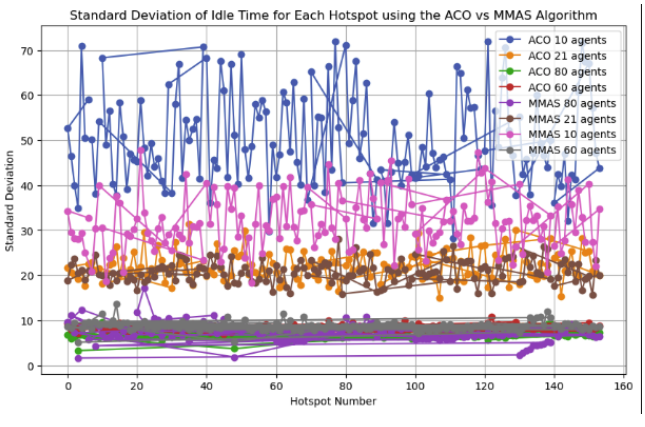

Figure 41: MMAS vs ACO Standard Deviation of each hotspot idle time with different number of agents