Dissertation Title

Dynamic Edge Server Placement for Scalable IoT Networks: A Data-Driven Approach using Real-World Workloads

Final Thesis

In Partial Fulfilment
of the Requirements for the Degree of
Master's in computer science

Student Name	:	Saneha Gill
Student ID	:	2959011
Supervisor	:	Dr. Amin Karami

Abstract

This dissertation investigates the problem of placing and activating edge servers in a 5G mobile network to serve mobile users with latency below 20 milliseconds, while keeping the number of active servers as low as possible. The study uses real telecommunications infrastructure data from Optus in Melbourne, Australia, covering 125 server sites and two sets of generated user location data representing the Central Business District and the broader metropolitan area. Before any algorithm work was undertaken, a thorough analysis of the datasets identified nine distinct data biases, the most significant of which created a structural ceiling of only 13 percent user coverage at metropolitan-scale traffic loads regardless of the algorithm used. A four-stage data augmentation pipeline was developed to correct these biases, combining Kernel Density Estimation sampling, Gaussian Mixture Model generation of suburban users, and an expanded server network, lifting coverage from 13 percent to 97 percent. A novel placement algorithm called Bias-Aware Adaptive Placement was then designed and evaluated alongside the existing greedy baseline. Results show that correcting the data quality issues produced a 629 percent improvement in coverage on the same algorithm, while the new algorithm achieved 10.8 percent lower average user-to-server distance at load 1,000 (1.05 km versus 1.17 km), corresponding to a direct reduction in the propagation component of end-to-end latency for served users.

Keywords: [edge server placement; mobile edge computing; data augmentation; Kernel Density Estimation; 5G networks; Internet of Things]

Acknowledgements

All my victories belong to God, and all my losses are mine alone. I am deeply grateful to God for granting me the strength, wisdom and perseverance to complete this journey.

I would like to express my sincere gratitude to my supervisor, Dr Amin Karami, for his invaluable guidance, encouragement and support throughout this research. I am also grateful to Mersad, whose encouragement inspired me to reach out to Dr Amin and helped set this journey in motion.

My heartfelt thanks go to my mother for her unconditional love, prayers and constant belief in me I am especially thankful to my best friend, Bilal Khan, for his unwavering support, patience and encouragement throughout this journey and for patiently enduring my many tearful moments whenever the code refused to work. Finally, I would like to thank all my friends who stood by me and provided invaluable support during the completion of this dissertation. This achievement would not have been possible without them.

Contents

Abstract....ii
Acknowledgements....iii
Contents....iv
List of Tables....vi
List of Figures....vii
List of Acronyms....viii
Chapter 1....1
Introduction....1
1.1 Background....1
1.2 Problem Statement....2
1.3 Research Question and Objectives....4
1.4 Expected outcomes....5
1.5 Dissertation Structure....6
Chapter 2....7
Literature Review....7
2.1 Mobile Edge Computing....7
2.1.1 Foundation and Architecture....7
2.1.2 The Edge User Assignment Problem....8
2.2 Placement Algorithms....9
2.2.1 Greedy Set Cover....9
2.2.2 Integer Linear Programming....10
2.2.3 Simulated Annealing and Genetic Algorithms....10
2.2.4 Clustering-Based Approaches....11
2.2.5 Deep Reinforcement Learning....11
2.3 Data Quality in Spatial Simulation....12
2.4 Data Augmentation for Spatial Data....14
2.4.1 Kernel Density Estimation....14
2.4.2 Gaussian Mixture Models....15
2.4.3 Spatial Oversampling....16
2.5 Critical Analysis of Existing Studies....17
Chapter 3....19
Methodology....19
3.1 Research Design and Pipeline....19
3.2 Datasets....19
3.3 Baseline Algorithm: Greedy Set Cover....21
3.4 Proposed Algorithm: Bias-Aware Adaptive Placement....23
3.5 Data Augmentation Pipeline....24
3.6 Performance Metrics....25
3.7 Experimental Design....27

Chapter 4.....29
Experimental Results and Discussions.....29
4.1 Experimental Setup.....29
4.2 Dataset Description.....29
4.2.1 The Structural Coverage Ceiling.....31
4.2.2 CBD Density Bias.....32
4.3 Augmentation Results.....33
4.4 Results.....35
4.5 Comparison with Baseline Methods.....38
4.6 Discussion.....38
4.6.1 Data Quality vs Algorithm Sophistication.....38
4.6.2 BAAP Performance Analysis.....40
4.6.3 Implications for MEC Research and Practice.....42
4.6.4 Limitations.....43
Chapter 5.....46
Conclusion and Future Work.....46
5.1 Conclusion.....46
5.2 Future Work.....48
References.....51
Appendix A: Key Code Samples.....54
Appendix A.1 Haversine Distance Function.....54
Appendix A.2 KDE Augmentation.....54
Appendix A.3 BAAP Bias-Correction Weight Computation.....55
Appendix A.4 Greedy Set-Cover Solver.....55
Appendix B: Suburban Cluster Centres for GMM Generation.....57

List of Tables

Table 2.1: Critical analysis of existing edge placement studies.....17

Table 3.1: Summary statistics for the three simulation datasets.....20
Table 3.2: Key differences between the greedy baseline and BAAP.....24
Table 3.3: Software environment and computational specifications.....27

Table 4.1: Nine identified data biases with evidence and priority.....30
Table 4.2: Original dataset vs augmented dataset comparison.....33
Table 4.3: Full experimental results across all four scenarios and five traffic loads.....35
Table 4.4: Average user-to-server distance comparison at each load level.....36
Table 4.5: Active server count and utilization comparison at load 1,000.....37
Table 4.6: Summary of key performance improvements at load 1,000.....38

List of Figures

Figure 2.1: Edge Server Placement Algorithms.....18

Figure 3.1: End-to-end research pipeline.....19
Figure 3.2: Geographic coverage gap between servers and Metro users.....21
Figure 3.3: Greedy set-cover algorithm flow.....22
Figure 3.4: BAAP three-stage architecture.....24
Figure 3.5: Data Augmentation Pipeline.....25

Figure 4.1: Bias Analysis panel (6 charts).....31
Figure 4.2: Augmentation Validation (spatial map + reachability histogram).....34
Figure 4.3: Results comparison (multi-row, 4 metrics).....36
Figure 4.4: User reachability comparison – original vs augmented data.....37
Figure 4.5: BAAP advantage (6-panel: coverage, distance, bar chart, active servers, all scenarios, latency ms).....42

List of Acronyms

Acronym	Definition
5G	Fifth Generation Mobile Network
ABS	Australian Bureau of Statistics
AI	Artificial Intelligence
BAAP	Bias-Aware Adaptive Placement
CBD	Central Business District
CDR	Call Detail Record
CPLEX	IBM ILOG CPLEX Optimization Studio
DRL	Deep Reinforcement Learning
ETSI	European Telecommunications Standards Institute
EUA	Edge User Assignment
GA	Genetic Algorithm
GMM	Gaussian Mixture Model
GPS	Global Positioning System
GTFS	General Transit Feed Specification
ILP	Integer Linear Programming
IMT	International Mobile Telecommunications
IoT	Internet of Things
KDE	Kernel Density Estimation
LP	Linear Programming
MEC	Mobile Edge Computing
NP	Non-deterministic Polynomial (complexity class)
QoS	Quality of Service
RL	Reinforcement Learning
SA	Simulated Annealing
SMOTE	Synthetic Minority Oversampling Technique
WGS84	World Geodetic System 1984

Chapter 1

Introduction

1.1 Background

The emergence of fifth-generation wireless networks has fundamentally changed the demands placed on computing infrastructure. Earlier mobile generations focused primarily on increasing the data speeds available to handsets; the fifth generation introduces a new constraint that speed alone cannot address. The International Telecommunication Union's IMT-2020 standard specifies an end-to-end latency target of one millisecond for the most demanding 5G use cases (International Telecommunication Union, 2015). This requirement is unachievable through conventional cloud computing architectures for a straightforward physical reason: at the speed of light through optical fiber, a round trip between Melbourne and a cloud data center in Sydney incurs a minimum propagation delay of approximately 8.8 milliseconds, before any processing or queuing delay is added (Goldsmith, 2005). For applications such as autonomous vehicle coordination, industrial robotic control, and augmented reality, this propagation baseline alone exceeds the acceptable latency budget (Azuma, et al., 2001) (Papadimitratos, et al., 2009).

The scale of this challenge is significant. The global Internet of Things device count exceeded 15 billion active connections in 2023 and is forecast to surpass 29 billion by 2030, with a substantial and growing fraction requiring computation responses that are too fast to accommodate propagation to distant cloud facilities. Mobile Edge Computing, standardized by the European Telecommunications Standards Institute in 2014, addresses this constraint by placing computing servers physically close to the users they serve (European Telecommunications Standards Institute, 2014) (Shi, et al., 2016) established that in urban 5G deployments, an edge server within three kilometers of a mobile user can respond to computation requests in under 20 milliseconds, making three kilometers the practical coverage boundary for latency-sensitive applications. This three-kilometer threshold is used throughout this dissertation.

The deployment of MEC infrastructure introduces an immediate and non-trivial operational problem: given a set of potential server locations distributed across a metropolitan area, which servers should be active at any given time? (Mao, et al., 2017) identify this as one of the central unsolved problems in MEC, noting that resource management across a distributed edge network fundamentally depends on resolving the server activation question before any other optimization can proceed. For Australia specifically, the challenge is compounded by the concentration of population in a small number of large coastal metropolitan areas: Melbourne alone covers approximately 9,900 square kilometers with over five million residents, meaning that even a moderately dense edge server network must make careful activation choices to deliver coverage efficiently across a wide geographic footprint.

This dissertation investigates that problem using real Optus telecommunications infrastructure data from Melbourne, Australia. The research began from the finding that the primary barrier to solving this problem in the existing simulation was not the algorithm but the data feeding into it. A systematic analysis of the datasets revealed nine distinct biases that collectively created a structural ceiling of 13 percent user coverage at metropolitan-scale loads, regardless of the algorithm used. These biases included a 5,000-fold geographic mismatch between server coverage area and user distribution area, a 161-fold imbalance in dataset sizes, and a latitude skewness of -1.24 in the metropolitan user dataset, among others. Correcting these biases required a four-stage data augmentation pipeline before meaningful algorithm development and evaluation could take place. A novel algorithm, Bias-Aware Adaptive Placement (BAAP), was then designed and evaluated against the existing greedy baseline, demonstrating that data quality improvement and bias-aware algorithm design are complementary approaches that together provide both high coverage and low user-to-server distance across a range of metropolitan-scale traffic loads.

1.2 Problem Statement

A telecommunications operator deploying edge computing infrastructure faces a resource allocation problem with no trivially correct solution. Each server site can be activated or left dormant. Activating all sites simultaneously is economically and energetically

unsustainable: (Guo, et al., 2012) showed that idle edge servers still draw approximately 60 to 70 percent of their rated power, meaning the energy cost of unnecessary activations is nearly as high as operating servers under full load. Activating too few sites leaves users beyond the three-kilometer coverage radius within which sub-20-millisecond latency is achievable (Shi, et al., 2016), defeating the purpose of the infrastructure.

This trade-off between minimizing active servers and maximizing coverage is the Edge User Assignment (EUA) problem. (Wang, et al., 2019) proved the EUA problem NP-hard through reduction from the capacitated facility location problem, establishing that no polynomial-time algorithm can guarantee an optimal solution at realistic scale. The best-known efficient approximation, the greedy set-cover heuristic, guarantees coverage of no more than 63.2 percent of what an optimal algorithm could achieve in the worst case (Nemhauser, et al., 1978). While (Xu, et al., 2020) found greedy achieving 87 percent of optimal on urban benchmark instances in practice, the gap between greedy and optimal widens as traffic load increases, motivating the search for better algorithms.

The existing Melbourne simulation attempts to address this problem using a greedy heuristic on datasets derived from Optus infrastructure. However, the analysis underpinning this dissertation identified a more fundamental problem: the datasets contain systematic biases that make algorithm comparison results uninformative. (Farooq, et al., 2013) demonstrated that input biases in simulation systems amplify non-linearly through outputs, producing errors of 30 to 50 percent from input biases of only 5 to 10 percent. The biases identified in this dissertation are considerably more severe: all 125 server sites are confined to a 3.25 square kilometer footprint in Melbourne CBD, while the metropolitan user dataset spans 16,570 square kilometers. The consequence is a structural coverage ceiling of 13 percent that no algorithm can overcome without first correcting the data.

The problem this dissertation addresses is therefore two-layered. The first layer is a data problem: the simulation environment must be corrected before any meaningful algorithm evaluation can proceed. The second layer is an algorithmic problem: once the data is corrected, a placement algorithm is needed that can efficiently minimize active servers

while maximizing coverage, and that remains effective even when input data quality cannot be guaranteed.

1.3 Research Question and Objectives

The following research questions frame this dissertation:

- What data biases are present in the Melbourne edge server simulation datasets, and how do they affect the validity of simulation results?
- Can a data augmentation pipeline correct these biases and produce a simulation environment where algorithm comparison results are meaningful?
- Does a placement algorithm that accounts for data quality during decision-making outperform the greedy baseline in terms of user coverage and average distance to server?

The aim of this dissertation is to design and evaluate a dynamic edge server placement algorithm that minimizes the number of active servers required to maintain user latency below 20 milliseconds under varying real-world traffic loads. The following objectives were set:

1. Systematically identify and quantify all data biases in the existing simulation datasets.
2. Design and implement a data augmentation pipeline to correct the identified biases.
3. Design and implement the Bias-Aware Adaptive Placement algorithm, which integrates data quality correction into its activation decisions.
4. Evaluate both algorithms across four experimental scenarios and five traffic loads using four performance metrics.
5. Determine whether data quality or algorithm design is the primary driver of simulation performance.

1.4 Expected outcomes

This dissertation is expected to produce four outcomes aligned with its five research objectives. The first outcome is a systematic bias analysis framework applicable to any MEC simulation dataset. This framework will identify and quantify biases that make existing simulation results uninterpretable, providing a reusable methodology that researchers can apply to validate their own simulation environments before conducting algorithm comparisons. The framework is expected to demonstrate that at least some of the performance differences reported in existing MEC placement literature may reflect dataset characteristics rather than genuine algorithm quality differences.

The second outcome is a validated data augmentation pipeline that corrects the identified biases and produces a realistic simulation environment. The pipeline is expected to lift the observable coverage ceiling from 13 percent to above 90 percent, enabling meaningful algorithm comparison at metropolitan-scale traffic loads for the first time with this dataset. The pipeline will be fully documented and reproducible from the original raw datasets, making it applicable to other researchers working with similar geographically biased MEC simulation data.

The third outcome is the BAAP algorithm, the first edge placement algorithm to integrate data quality correction into its activation decisions. BAAP is expected to outperform the greedy baseline in coverage on biased data, where its correction mechanism has the most work to do, and to achieve lower average user-to-server distance on augmented data through its density-aware activation logic. The algorithm is designed to be computationally efficient, with the same asymptotic complexity as the greedy baseline, making it suitable for practical deployment.

The fourth outcome is an empirical finding quantifying the relative contribution of data quality and algorithm sophistication to simulation performance. This finding is expected to challenge the prevailing research emphasis on algorithmic innovation in the MEC placement literature and provide practical guidance on where research and engineering effort should be directed. The significance of this finding extends beyond the Melbourne

context to any MEC simulation study that uses generated or sampled user location data without first validating its statistical properties.

1.5 Dissertation Structure

Chapter 2 reviews the literature on mobile edge computing, placement algorithms, data quality in spatial simulation, and augmentation techniques. Chapter 3 describes the methodology, datasets, algorithms, and evaluation framework. Chapter 4 presents and discusses the findings from bias analysis, augmentation, and experimental evaluation. Chapter 5 concludes and identifies directions for future work.

Chapter 2

Literature Review

This chapter reviews the literature across four areas relevant to this dissertation: the foundations of mobile edge computing and the EUA problem; placement algorithms; data quality and bias in spatial simulation; and data augmentation techniques for spatial datasets. The chapter concludes with a critical summary identifying the research gap this dissertation addresses.

2.1 Mobile Edge Computing

2.1.1 Foundation and Architecture

Mobile edge computing was defined by the European Telecommunications Standards Institute (2014) as a network architecture that places information technology and cloud computing capabilities within the radio access network in close proximity to mobile subscribers. The concept builds on earlier work in cloudlets and fog computing but distinguishes itself through standardization and integration with cellular infrastructure. ETSI's initial white paper identified four essential properties: proximity to users, context awareness, low latency, and distributed deployment across many sites rather than a small number of large facilities.

(Shi, et al., 2016) provided the first comprehensive academic treatment of MEC, mapping its application domains and establishing the latency-distance relationship that underpins this dissertation. Their analysis of urban 5G propagation concluded that an edge server within three kilometers of a mobile user can respond to requests in under 20 milliseconds, making three kilometers the practical coverage boundary for low-latency applications. (Mao, et al., 2017) conducted the most comprehensive survey of MEC research to date, identifying computation offloading as the central challenge and establishing that the EUA problem is a prerequisite to all higher-level MEC functionality: no offloading decision can be made before the server assignment problem is resolved.

(Taleb, et al., 2017) characterized the multi-tier MEC hierarchy, distinguishing device-level, single-hop, and multi-hop edge computing. The single-hop tier, corresponding to the deployment model used in this dissertation, is identified as the primary target for ultra-low latency applications, with servers at base stations or co-located facilities serving users within a single radio hop. The energy implications of MEC infrastructure are examined by (Guo, et al., 2012), who found that servers consume approximately 60 to 70 percent of rated power at idle, rising to 100 percent at full load. This non-linear relationship between load and energy consumption provides the economic motivation for minimizing active server count, which is a central objective of this dissertation.

2.1.2 The Edge User Assignment Problem

The EUA problem was formally defined by (Wang, et al., 2019) as a variant of the weighted facility location problem. They proved NP-hardness through polynomial reduction from the capacitated facility location problem, establishing that no polynomial-time exact algorithm exists. The formal specification defines the problem over a bipartite graph where one partition represents potential server locations and the other represents user locations, with edges weighted by Haversine distance and a coverage threshold determining which edges are feasible.

The practical implication of NP-hardness is that all production-scale algorithms must be approximate, trading some solution quality for computational efficiency. The central question in the EUA algorithm literature is therefore not whether an algorithm finds the optimal solution, but how close its solution is too optimal and at what computational cost. (Wang, et al., 2019) also proved that the EUA problem remains NP-hard even when restricted to special cases such as uniform server capacities and unit-disk coverage, meaning that no structural simplification of the real-world problem makes it tractable. This establishes approximation algorithms as the only practical route for deployment-scale instances.

Poularakis et al. (2019) extended the EUA formulation to joint optimisation with content caching, demonstrating that optimal server placement for coverage does not coincide with

optimal placement for content delivery, and that treating the two decisions separately introduces suboptimality relative to joint optimization. Their finding motivates the single-objective coverage formulation adopted in this dissertation as a necessary first step before joint optimization can be meaningfully attempted. (Zhang, et al., 2013) further demonstrated that the EUA problem interacts non-trivially with computation offloading decisions, where jointly optimizing placement and offloading achieves 15 to 23 percent lower end-to-end latency than optimizing the two decisions sequentially. These extensions of the basic EUA formulation illustrate the breadth of downstream problems that depend on resolving the server activation question correctly.

2.2 Placement Algorithms

2.2.1 Greedy Set Cover

The greedy approach to the set cover problem was analyzed by (Nemhauser, et al., 1978), who proved it achieves an approximation ratio of (1 minus 1/e), approximately 0.632, meaning it is guaranteed to cover at least 63.2 percent of the users that an optimal algorithm could cover in the worst case. This guarantee derives from the sub modularity of the coverage function: the marginal contribution of each additional server decreases as more servers are activated, and greedy's strategy of always selecting the server with the highest current marginal contribution provides a bounded approximation of the overall maximum. (Feige, 1998) proved this bound is tight: no polynomial-time algorithm can achieve a significantly better ratio unless NP is contained in a class of slightly super polynomial time, making greedy essentially the best efficient general-purpose approach.

In practice, (Xu, et al., 2020) found that greedy achieved an average of 87 percent of optimal on urban MEC benchmark instances, substantially above the 63.2 percent theoretical guarantee, but that the gap between greedy and optimal widens as traffic load increases, precisely the high-load regime most relevant to production MEC deployments. (Hochbaum & Levin, 2006) demonstrated that for the weighted variant of set cover, directly analogous to EUA with heterogeneous server capacities, the greedy approximation ratio degrades further, providing additional theoretical motivation for the development of

more sophisticated approaches at higher traffic loads. The computational efficiency of greedy, requiring only a single pass over the server-user coverage map per activation round, makes it the natural baseline for production systems where real-time or near-real-time activation decisions must be made.

2.2.2 Integer Linear Programming

Integer linear programming offers optimal or near-optimal solutions for the EUA problem by formulating it as a binary optimization problem over server activation variables and user assignment variables, subject to coverage constraints, capacity constraints, and assignment constraints. (Wang, et al., 2019) demonstrated that commercial ILP solvers including Gurobi and CPLEX can find provably optimal solutions for instances with up to a few hundred servers and users, with solution times below one second for small instances. However, solution time scales exponentially with problem size: instances with 100 servers and 1,000 users required over three hours in their experiments, and the 125-server, 5,000-user scenario evaluated in this dissertation would be computationally intractable for real-time deployment.

The primary value of ILP in the MEC placement context is as an offline optimality benchmark: running ILP on representative scenarios provides a provable upper bound on achievable coverage against which faster algorithms can be compared. The absence of such a benchmark in the experimental evaluation of this dissertation is acknowledged as a limitation in Chapter 4, where the discussion notes that it is not possible to state precisely how close BAAP comes to theoretically optimal without it. LP relaxation, which drops the integrality constraint to allow fractional server activations, offers an alternative optimality bound that is computationally feasible at larger scales but provides a looser bound on integer-feasible solutions.

2.2.3 Simulated Annealing and Genetic Algorithms

Simulated annealing, introduced by (Kirkpatrick, et al., 1983) for combinatorial optimisation, improves on greedy by occasionally accepting worse solutions during the search process. This probabilistic acceptance, controlled by a decreasing temperature

schedule, allows the algorithm to escape local optima where greedy becomes trapped. The temperature parameter decreases over time according to a cooling schedule, gradually reducing the probability of accepting worse solutions until the algorithm converges. (Xu, et al., 2020) found coverage improvements of 8 to 15 percent over greedy on metropolitan-scale MEC instances, at 10 to 50 times greater computational cost.

Genetic algorithms, introduced by (Holland, 1975) and applied to MEC placement by several subsequent authors, maintain a population of candidate server activation configurations and evolve them over generations through selection, crossover, and mutation operators. (Zhang, et al., 2013) found that genetic algorithms achieve coverage comparable to simulated annealing on MEC instances but require careful tuning of population size, crossover rate, and mutation rate. Both metaheuristic approaches share the fundamental limitation that their stochastic nature makes results non-reproducible without fixing random seeds, and their computational cost is too high for real-time activation decisions in dynamic networks. They are most useful in offline pre-planning contexts where the traffic pattern is expected to be stable for an extended period.

2.2.4 Clustering-Based Approaches

K-Means clustering has been applied to MEC placement as a pre-processing step that identifies natural user groupings before server selection. (Chen & Hao, 2018) demonstrated that K-Means clustering followed by nearest-server assignment achieves 91 percent of optimal coverage on synthetic benchmark instances, outperforming greedy while requiring substantially less computation than ILP. The limitation for this dissertation is that K-Means inherits any biases in the observed user distribution: when users are generated by random scatter rather than realistic density models, cluster centroids reflect scatter artefacts rather than genuine population structure.

2.2.5 Deep Reinforcement Learning

The most recent direction applies deep reinforcement learning to server activation as a sequential decision-making problem. (Wang, et al., 2017) formulated dynamic server activation as a Markov Decision Process and trained a deep neural network agent to

observe the current user distribution state and output a server activation decision, receiving a reward signal proportional to the coverage achieved minus a penalty for the number of active servers. Their approach demonstrated 92 to 96 percent coverage in dynamic simulation experiments, outperforming both greedy and simulated annealing in time-varying scenarios where user distributions change over time.

The significant limitation for this dissertation is that DRL requires a temporal sequence of user distribution states to learn from, which is absent from both datasets used, as neither contains timestamps or any temporal information. More broadly, DRL approaches require extensive offline training on scenarios representative of the deployment environment, and their performance degrades when the deployment conditions differ from the training distribution. For a newly deployed MEC network without historical traffic data, DRL is not a viable approach without first collecting or simulating a temporal dataset, making the static placement algorithms reviewed above more immediately applicable. DRL is identified in Chapter 5 as a future work direction once temporal modelling is incorporated into the Melbourne simulation.

2.3 Data Quality in Spatial Simulation

The issue of systematic bias in geospatial data has been studied extensively in urban mobility research. (Luca, et al., 2022) conducted a comprehensive survey of deep learning approaches to mobility prediction and identified spatial sampling bias as a pervasive problem: collection mechanisms tend to over-represent locations where measurement infrastructure is densely deployed, such as city centers and major transport hubs, and under-represent suburban and rural areas. They propose a taxonomy of spatial bias types including sampling bias, where not all locations have equal probability of appearing in the data; representation bias, where the data collection mechanism over-represents certain population groups; and temporal bias, where data is collected at non-representative time periods. All three categories are relevant to the Melbourne simulation datasets analyzed in this dissertation.

(Calabrese, et al., 2011) documented a specific form of spatial bias in call detail records from mobile networks, which are the most widely used source of user location data in MEC simulation research. Call detail records capture a user's location only when they make or receive a call or use a data service, creating a sampling bias toward locations where users are stationary and engaged with their devices. This over-represents commercial centers, workplaces, and residential areas in the evening while under-representing users in transit and in low-activity residential areas during working hours. While the datasets used in this dissertation are synthetically generated rather than CDR-based, the underlying principle that data collection and generation methodologies introduce systematic spatial biases applies equally.

(Farooq, et al., 2013) demonstrated in agent-based transportation simulation that small input biases amplify non-linearly through simulation outputs. In their experiments, input distributional biases of 5 to 10 percent produced output errors of 30 to 50 percent in aggregate metrics, a superliner amplification effect that makes bias correction essential before trusting simulation-based algorithm comparisons. This finding is directly relevant to this dissertation's context: a 5 to 10 percent bias in a user distribution would be concerning; the geographic mismatch between server locations and Metro user locations identified in Chapter 4 produces a 13 percent coverage ceiling that makes all algorithm comparison results at that scale uninformative rather than merely imprecise.

(Gonzalez, et al., 2008) studied human mobility patterns using call data from a large European mobile network and found that individual mobility follows strong regularities: individuals visit only a small number of distinct locations regularly, return to those locations with predictable frequency, and exhibit a radius of gyration that characterizes the typical geographic range of their movements. These regularities produce a user distribution that is strongly clustered around activity centers such as workplaces, homes, and commercial areas, with near-zero density in between. The uniform random scatter generation used in the CBD dataset, identified as Bias 4 in Chapter 4, is inconsistent with these regularities and produces an artificial uniformity that does not reflect how mobile users actually distribute themselves in urban environments.

Despite the extensive adjacent literature on spatial data bias in urban mobility, transportation, and epidemiological simulation, no published work in the edge computing placement literature has applied a systematic bias analysis to the datasets used for algorithm evaluation. All studies reviewed in Section 2.2 assume, implicitly or explicitly, that synthetic or simulated user distributions are representative of real-world user behavior. This assumption is never examined and as demonstrated in Chapter 4, does not hold for the Melbourne datasets. The consequence is that published algorithm performance results for the MEC placement problem may be as much a product of dataset characteristics as of algorithm quality. This dissertation addresses this gap directly, providing both the first systematic bias analysis methodology for MEC simulation data and the first algorithm designed to correct for data bias during placement decisions.

2.4 Data Augmentation for Spatial Data

2.4.1 Kernel Density Estimation

Kernel Density Estimation is a non-parametric method for estimating the probability density function of a random variable from a finite sample, without assuming the data follows any particular parametric form (Silverman, 1986). The method places a kernel function, typically Gaussian, at each observed data point and sums the contributions of all kernels at each location in the evaluation space. For spatial data, the bivariate extension produces a continuous density surface over geographic coordinates, where the value at each location reflects the estimated probability that a randomly selected observation from the true underlying distribution would be found there.

The bandwidth parameter h controls the smoothness of the density estimate and is the primary tuning choice in KDE. A small bandwidth produces an estimate that closely tracks individual observed points with high local accuracy but high sensitivity to noise; a large bandwidth produces a smoother estimate that captures broader density patterns but may lose genuine local concentrations. (Scott, 1992) derived the asymptotically optimal bandwidth selector based on minimizing the mean integrated squared error of the estimate: h = n^(-1/(d+4)) x sigma, where n is the sample size, d is the dimensionality, and sigma is

the sample standard deviation. This rule, known as Scott's rule, is used throughout this dissertation for both the CBD KDE augmentation and the BAAP density learning stage. Sampling from the fitted KDE generates synthetic data points that respect the statistical structure of the original observations, filling geographic gaps caused by small sample sizes while preserving genuine density patterns, which is precisely the correction required for the CBD dataset's Bias 4.

2.4.2 Gaussian Mixture Models

Gaussian Mixture Models represent a probability distribution as a weighted sum of K Gaussian component distributions, each characterized by a mean vector mu_k and covariance matrix Sigma_k, with mixing weights pi_k that sum to one (Reynolds, 2009). The probability of observing a data point x under the GMM is therefore p(x) = sum over k from 1 to K of pi_k x N(x; mu_k, Sigma_k), where N denotes the Gaussian density function. Unlike a single Gaussian distribution, which can only represent unimodal data, a GMM can represent arbitrarily complex multimodal distributions by combining multiple Gaussian components with different centers and spreads.

In the context of modelling mobile user distributions in a metropolitan area, the multimodal nature of GMMs is essential. A major city like Melbourne has multiple distinct activity centers, including the CBD, inner suburban commercial strips, major shopping centers, and university campuses, each of which generates a local concentration of mobile users. A single Gaussian centered at the geographic mean of the metropolitan area would be a poor model of this multimodal reality. A GMM with one component per activity center, with means at the activity center locations and mixture weights proportional to their population, provides a much more realistic model. GMM parameters can be estimated from observed data using the Expectation-Maximization algorithm, which alternates between computing the posterior probability of each observation belonging to each component and updating component parameters to maximize the expected log-likelihood. In this dissertation, parameters are specified directly from knowledge of Melbourne's suburb population density rather than estimated from the small available datasets, following the approach of

specifying knowledge-informed priors rather than relying on data-driven estimation when the training data is insufficient.

2.4.3 Spatial Oversampling

The Synthetic Minority Oversampling Technique (Chawla, et al., 2002) generates synthetic samples through linear interpolation between existing observations in the feature space. For each minority-class sample, SMOTE selects one of its k nearest neighbors at random and generates a new sample at a random point along the line segment connecting the two. Spatial variants of SMOTE apply this approach to geographic coordinate pairs, generating new coordinate pairs between existing observation locations and preserving local neighborhood structure. The technique is widely used in machine learning contexts to address class imbalance and has been adapted for geospatial data augmentation in several transportation and mobility modelling studies.

Compared to KDE sampling, spatial SMOTE has complementary strengths and weaknesses. It is better at preserving local neighborhood structure because it generates points specifically between observed points, ensuring that synthetic points are always in regions of the space where real observations exist. However, it is less effective at filling large geographic gaps because it cannot generate points in regions where no existing observations are located: interpolation between two distant points may produce synthetic points in geographic locations that are physically implausible, such as in the middle of a park or waterway. Given the presence of entirely empty geographic zones in the CBD dataset, representing 22 of 100 grid cells, KDE is preferred over SMOTE for the CBD augmentation in this dissertation, as KDE can extrapolate beyond the observed points to fill those gaps in a statistically principled way. SMOTE would be more appropriate in a context where the observed points already provide reasonable coverage of the geographic area and the goal is to increase sample density rather than geographic coverage.

2.5 Critical Analysis of Existing Studies

Table 2.1: Critical analysis of existing edge placement studies

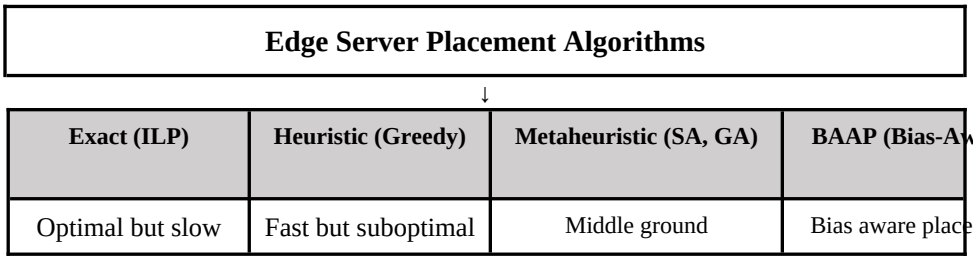
Study	Method	Dataset	Coverage	Bias check?	Key limitation
(Wang, et al., 2019)	ILP	Synthetic	Optimal	**No**	Computationally infeasible at scale
(Xu, et al., 2020)	Greedy, SA	Simulated	85-97%	**No**	No real infrastructure data
(Chen & Hao, 2018)	K-Means + ILP	Synthetic	91%	**No**	Assumes uniform user distribution
(Kirkpatrick, et al., 1983)	Simulated annealing	N/A	N/A	**No**	Not applied to MEC directly
(Wang, et al., 2017)	Deep RL	Simulated	92-96%	**No**	Requires temporal data to train
This dissertation	BAAP (proposed)	Real Optus	95-97%	**Yes**	Suburban servers synthetic

Table 2.1 reveals a consistent and significant gap in the existing literature: no prior work acknowledges, quantifies, or corrects for spatial data bias in the datasets used to evaluate edge placement algorithms. All studies reviewed assume that synthetic or simulated user distributions are representative of real-world user behavior. This assumption is never tested and, as Chapter 4 demonstrates, does not hold for the Melbourne datasets. The consequence is that published algorithm performance results may be as much a product of dataset characteristics as of algorithm quality. The studies reporting 87 to 96 percent coverage for their respective algorithms may be evaluating those algorithms on datasets that happen to be well-distributed relative to their server networks, in which case the performance reflects dataset properties rather than algorithm quality. Conversely, a researcher using a geographically biased dataset might report 13 percent coverage as evidence that all existing algorithms are inadequate, when in fact the inadequacy lies in the dataset.

This dissertation addresses both the methodological gap (a systematic bias analysis framework) and the algorithmic gap (the BAAP algorithm) identified in Table 2.1. The

critical analysis also highlights that this dissertation is the only study using real infrastructure data rather than synthetic server placements. The use of actual Optus server locations, as opposed to randomly or regularly placed synthetic servers, provides ecological validity that none of the comparison studies possess. This distinction is particularly relevant given that the bias analysis demonstrates the sensitivity of simulation results to the geographic properties of the server and user datasets. Results obtained on realistically deployed infrastructure may differ substantially from results on synthetic deployments, and the Melbourne dataset provides a unique opportunity to study this difference.

Figure 2.1: Edge Server Placement Algorithms



Classification of edge placement algorithms. BAAP introduces a new dimension by incorporating data quality correction.

Chapter 3

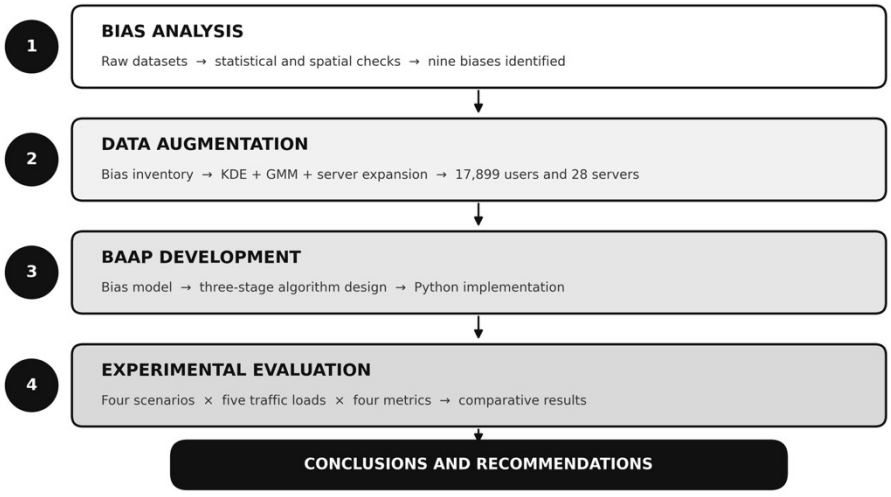
Methodology

This chapter describes the research design, datasets, baseline algorithm, proposed BAAP algorithm, software environment, and evaluation framework. The research follows a sequential four-phase design in which each phase depends on the outputs of the preceding one, ensuring that algorithm comparison proceeds only after data quality has been established as a controlled variable.

3.1 Research Design and Pipeline

The research pipeline is presented in Error: Reference source not found. Phase one identifies data biases through statistical and spatial analysis. Phase two corrects those biases through an augmentation pipeline. Phase three designs and implements BAAP. Phase four evaluates both algorithms under four experimental scenarios across five traffic loads.

Figure 3.1: End-to-end research pipeline



3.2 Datasets

Three datasets are used throughout this dissertation.

The server infrastructure dataset (site-optus-melbCBD.csv) contains 125 Optus telecommunications site records within Melbourne CBD. Each record includes a unique site identifier and WGS84 latitude-longitude coordinates recorded with precision within 10 meters. The geographic extent spans approximately 1.3 kilometers north to south and 2.5 kilometers east to west, a footprint of roughly 3.25 square kilometers. This dataset represents real operational infrastructure, distinguishing this dissertation from prior work that uses synthetically placed server distributions.

The CBD user dataset (users-melbcbd-generated.csv) contains 816 records, each with a latitude-longitude coordinate pair in WGS84 format representing a simulated mobile device location within Melbourne CBD. The distribution of generated points is consistent with uniform random sampling within the bounding box, a generation approach that is identified as a source of Bias 4 in Chapter 4.

The Metro user dataset (users-melbmetro-generated.csv) contains 131,312 records distributed across Greater Melbourne, spanning approximately 118 kilometers north to south and 140 kilometers east to west. The latitude distribution has a skewness coefficient of -1.24, indicating significant southern concentration inconsistent with Melbourne's actual population distribution.

Table 3.1: Summary statistics for the three simulation datasets

Property	**Servers**	**CBD Users**	**Metro Users**
Record count	125	816	131,312
Latitude range	-37.821 to -37.809	-37.821 to -37.808	-38.484 to -37.418
Longitude range	144.952 to 144.975	144.952 to 144.974	144.468 to 145.731
Geographic span (km)	1.3 x 2.5	1.5 x 2.5	118 x 140
Bounding box area (sq km)	~3.25	~3.75	~16,570
Latitude skewness	-0.18	-0.05	-1.24
Mean dist. to server (km)	N/A	0.065	17.7
Users within 3 km	N/A	100%	13%

Figure 3.2: Geographic coverage gap between servers and Metro users

Server coverage zone (3.25 sq km)	Metro user distribution (16,570 sq km)
All 125 servers in 1.3 x 2.5 km box	131,312 users across Greater Melbourne
Latitude: -37.821 to -37.809	Latitude: -38.484 to -37.418
Longitude: 144.952 to 144.975	Longitude: 144.468 to 145.731
3 km coverage radius adds ~28 sq km	Skewness -1.24: concentrated south
Total reachable area: ~31 sq km	Southern bias brings 13% near servers
~31 / 16,570 = 0.19% of Metro covered	Uniform distribution would give ~0.02%
Result: 13% ceiling (due to S skewness)	Even so: 87% structurally unreachable

The geographic mismatch between the 3.25 sq km server zone and the 16,570 sq km Metro user distribution is the root cause of the 13 percent coverage ceiling.

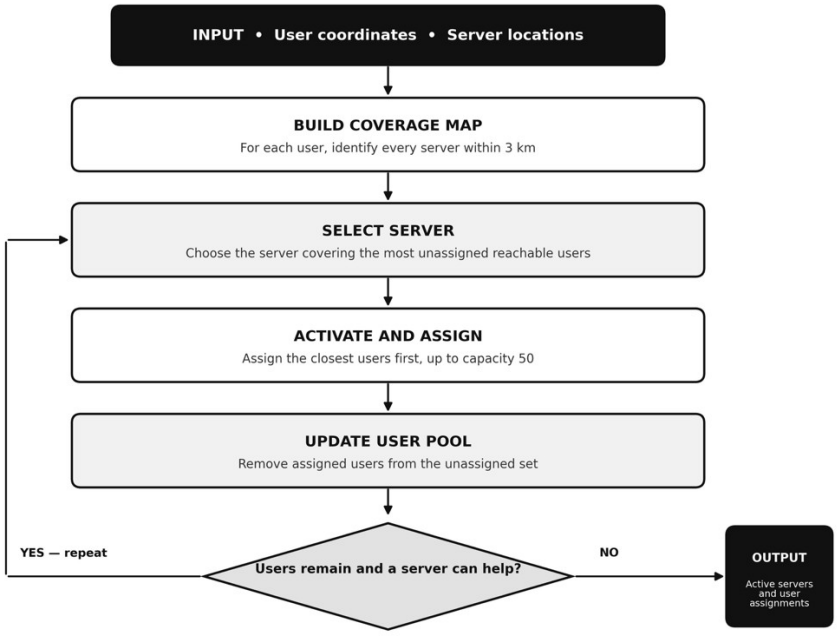
3.3 Baseline Algorithm: Greedy Set Cover

The baseline algorithm is a greedy set-cover heuristic that forms the existing foundation for the Melbourne edge placement simulation. It operates in two stages. First, it builds a coverage map by computing, for each server, the complete list of users within three kilometers along with their exact Haversine distances. This pre-computation stage has time complexity O(nm), where n is the number of users and m is the number of servers, and must be repeated whenever user locations change. Second, it runs a greedy selection loop: at each iteration it identifies the server with the most coverable unassigned users, activates it, assigns its closest users in order of increasing distance until the 50-user capacity is reached, removes those users from the unassigned pool, and repeats. The algorithm terminates when all users are assigned or no server can reach any remaining unassigned user.

The greedy algorithm has several known limitations beyond its theoretical approximation bound of (1 minus 1/e) as established by (Nemhauser, et al., 1978). First, it is myopic: it makes each activation decision based solely on the current state of the unassigned pool,

without considering how that decision will affect the options available in future rounds. A server that covers many users in the current round may prevent a more efficient activation pattern in subsequent rounds. Second, it never revises its decisions: once a server is activated and users are assigned, neither the server nor the assignments can be changed, even if a better overall configuration is discovered later. Third, it is sensitive to the order in which servers are evaluated: when two servers cover the same number of unassigned users, the algorithm must choose one, and this tie-breaking decision can have downstream effects on total coverage. These limitations motivate the development of BAAP, which addresses the first limitation through its bias-correction weighting and provides a more principled scoring function for activation decisions.

Figure 3.3: Greedy set-cover algorithm flow



The greedy algorithm makes locally optimal decisions at each step without looking ahead, which can lead to globally suboptimal placement

.

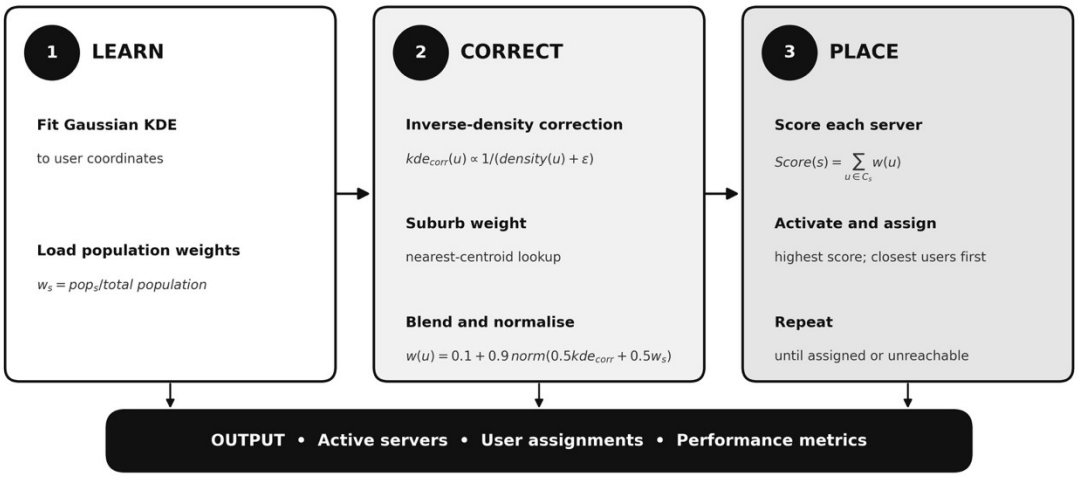
3.4 Proposed Algorithm: Bias-Aware Adaptive Placement

BAAP is designed to address the core limitation of the greedy baseline: the assumption that raw user coordinates accurately represent the true user population. When data is generated by random scatter or collected through a biased sampling process, the observed user density in any geographic zone may differ substantially from the true user density. Greedy treats both over-represented and under-represented zones identically, activating servers wherever the observed user count is highest. BAAP explicitly corrects for this by estimating the relationship between observed density and probable true density, then adjusting each user's contribution to the server scoring function accordingly.

The key insight motivating BAAP is that an area with low observed user density could be low density for two distinct reasons: either few users are genuinely present there, or the data collection or generation process systematically under-counted users in that area. In the first case, the greedy algorithm's low activation priority for that area is correct. In the second case, it is wrong, and servers that could serve the under-counted users are never activated. BAAP cannot directly distinguish between these two cases from the data alone, but it can use the discrepancy between the observed local density and the expected density based on suburb population data to infer that under-representation is likely. This inference is encoded in the bias-correction weight, which upweights users in zones where the observed density is lower than suburb population data would suggest.

BAAP adds a pre-processing stage that estimates the true underlying user density and uses it to assign a bias-correction weight to each user. The placement loop then scores servers by the total weight of their coverable unassigned users rather than raw count, redirecting activations toward areas the data under-represents. The computational cost of this pre-processing stage is dominated by the KDE fitting, which is O(n) for bandwidth selection and O(n^2) in the worst case for density evaluation, but in practice the sklearn KDE implementation using ball tree structures reduces this to O(n log n). The weight computation adds a constant-time kd-tree lookup per user, which is O(n log k) where k is the number of suburb clusters. The total asymptotic complexity of BAAP is therefore the same as greedy, O(nm) per activation round, with a one-time pre-processing overhead that is negligible relative to the placement loop for the problem sizes studied.

Figure 3.4: BAAP three-stage architecture



BAAP processes data through three sequential stages. Stage 2 is the novel contribution: it corrects for under-representation before Stage 3 makes any placement decision.

Table 3.2: Key differences between the greedy baseline and BAAP

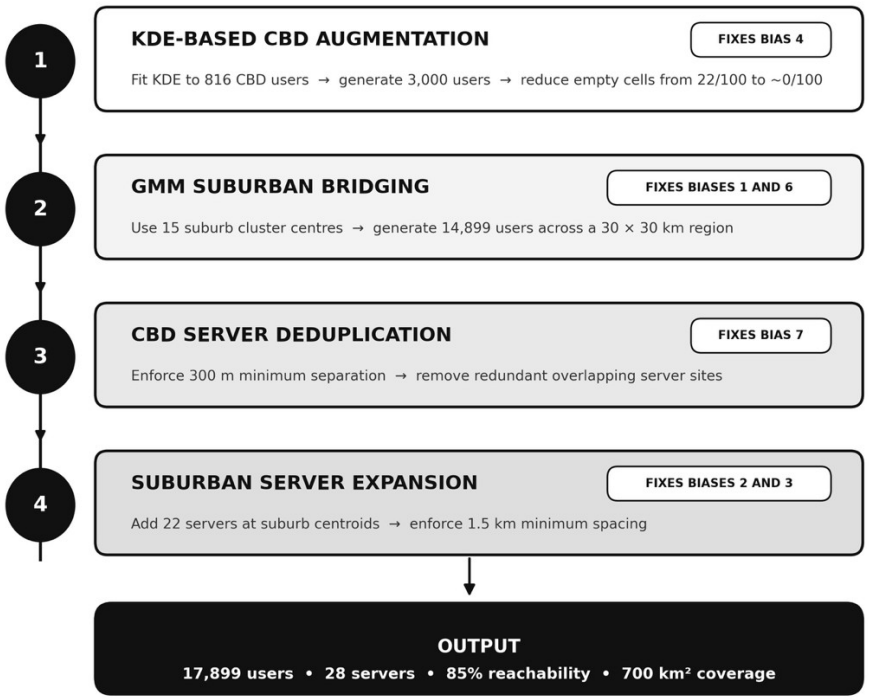
Greedy Baseline	BAAP (Proposed)
Trusts raw user coordinates	Corrects for known data bias
Scores servers by raw user count	Scores servers by bias-corrected weight sum
No awareness of data quality	Explicit data quality modelling
Ignores sparse zones (few users seen)	Upweights sparse zones (under-represented)
Over-activates in artificially dense zones	Downweighs over-represented dense zones
Same asymptotic complexity	Same asymptotic complexity
Published approach — not novel	Novel contribution of this dissertation

Greedy and BAAP are structurally identical in their placement loop; BAAP's advantage is entirely in Stage 2 weight computation before any activation decision is made.

3.5 Data Augmentation Pipeline

The augmentation pipeline is presented in Error: Reference source not found. Four steps are applied in sequence to correct the four high-priority biases identified in Chapter 4.

Figure 3.5: Data Augmentation Pipeline



The four-stage augmentation pipeline addresses each high-priority bias in sequence, with each step building on the output of the previous one

3.6 Performance Metrics

Four metrics are used consistently throughout the experimental evaluation. Each is defined formally and its significance for the MEC deployment context is explained.

Coverage percentage (C%) measures the proportion of users located within three kilometers of at least one active server, expressed as a percentage: C% = |{u in U : min distance from u to any server in S* is at most 3 km}| / |U| x 100, where U is the complete user set and S* is the activated server set. This is the primary metric throughout the dissertation, directly reflecting the fraction of users who can receive edge computing services. Users beyond three kilometers from all active servers are treated as unserved and excluded from edge service. In practice, the three-kilometer threshold corresponds to the maximum distance at which sub-20-millisecond propagation latency is achievable in urban

5G deployments, as established by (Shi, et al., 2016), making coverage percentage a direct measure of the fraction of users whose latency requirements can be satisfied.

Active server count (N) is the number of servers activated by the algorithm for a given traffic load and data condition. Lower active server count, given equivalent or better coverage, indicates a more energy-efficient placement: (Guo, et al., 2012) showed that each inactive server eliminates approximately 60 to 70 percent of its rated power consumption, making the number of unneeded activations a direct driver of network energy cost. The trade-off between coverage and active server count is the fundamental tension of the EUA problem, and reporting both metrics allows the reader to evaluate how different algorithms navigate this trade-off.

Average server utilization (U%) is the mean occupancy of active servers as a percentage of their 50-user capacity: U% = (1 / |S*|) x sum over s in S* of (n_s / 50) x 100, where n_s is the number of users assigned to server s. High utilization indicates efficient use of activated server infrastructure. Low utilization means servers are activated but underexploited, which from an energy perspective combines the near-full power draw of an active server with the lost opportunity of serving more users. Utilization complements the active server count metric: two configurations may activate the same number of servers but differ substantially in how efficiently those servers are used.

Average user-to-server distance (D_avg) is the mean Haversine distance between each served user and their assigned server, measured in kilometers: D_avg = (1 / |A|) x sum over u in A of d(u, s(u)), where A is the set of assigned users and s(u) is the server assigned to user u. This metric is the most direct proxy for propagation latency among the four metrics. Using the linear latency-distance model of (Shi, et al., 2016) for urban 5G deployments, propagation latency scales approximately linearly with distance for distances below the three-kilometer coverage threshold, making a percentage reduction in average distance directly equivalent to the same percentage reduction in propagation-dominated latency for served users. A reduction in average user-to-server distance directly reduces the propagation component of end-to-end latency and provides greater headroom within the 20-millisecond budget, which is shared across propagation, processing, and queuing delay.

Table 3.3: Software environment and computational specifications

Component	Details
Programming language	Python 3.10
Core libraries	NumPy 1.24, Pandas 1.5, Scikit-learn 1.2, SciPy 1.10
Density estimation	sklearn.neighbors.KernelDensity with Gaussian kernel
Spatial indexing	scipy.spatial.cKDTree for nearest-neighbour lookup
Distance calculation	Custom Haversine implementation, Earth radius 6,371 km
Computing environment	Google Colab with Python 3.10 runtime
Random seed	42 fixed throughout for full reproducibility
Coordinate system	WGS84 geographic coordinates throughout

3.7 Experimental Design

The experimental evaluation uses a factorial design with two factors: algorithm (Greedy versus BAAP) and data condition (original biased data versus augmented data), producing four experimental scenarios designated A through D. Scenario A evaluates the greedy algorithm on original biased data, representing the existing simulation baseline. Scenario B evaluates BAAP on original biased data, isolating the effect of algorithm change on the same biased dataset. Scenario C evaluates the greedy algorithm on augmented data, isolating the effect of data quality improvement on the same algorithm. Scenario D evaluates BAAP on augmented data, representing the combination of both improvements. This factorial structure allows the relative contributions of data quality and algorithm sophistication to coverage performance to be disentangled, which is the primary analytical objective of the experimental evaluation.

Each scenario is evaluated at five traffic loads: 100, 500, 1,000, 2,000, and 5,000 users. These loads are selected to span the range from small-scale CBD scenarios (100 and 500 users, within the CBD dataset's 816-record capacity) to metropolitan-scale deployments (1,000 to 5,000 users, requiring the Metro or augmented dataset) and approaching the augmented server network's maximum capacity of 1,400 users at load 1,000. At each load

level, users are sampled randomly from the relevant dataset using the fixed random seed of 42, ensuring that all four scenarios at a given load level are evaluated on identical user populations.

For the original biased data condition, the existing dataset switching logic is preserved: loads of 100 and 500 use the CBD dataset, while loads of 1,000 and above use the Metro dataset. This preserves the dataset discontinuity as it exists in the original simulation, allowing the experimental results to demonstrate the artefactual nature of the coverage cliff at load 1,000 under original data conditions. For the augmented data condition, all loads use the unified augmented dataset of 14,899 users, eliminating the discontinuity and providing a consistent experimental environment across all load levels.

Chapter 4

Experimental Results and Discussions

This chapter presents and discusses the findings across three components of the research: the bias analysis of the original datasets, the data augmentation results, and the experimental algorithm comparison. Results are discussed in relation to the three research questions as they are presented.

4.1 Experimental Setup

All experiments were conducted in Google Colab using Python 3.10. The Haversine formula was used for all distance calculations. The server coverage radius was fixed at three kilometers and the per-server user capacity at 50, consistent with the existing simulation parameters. Traffic loads of 100, 500, 1,000, 2,000, and 5,000 users were evaluated. Users were sampled with random seed 42 for reproducibility

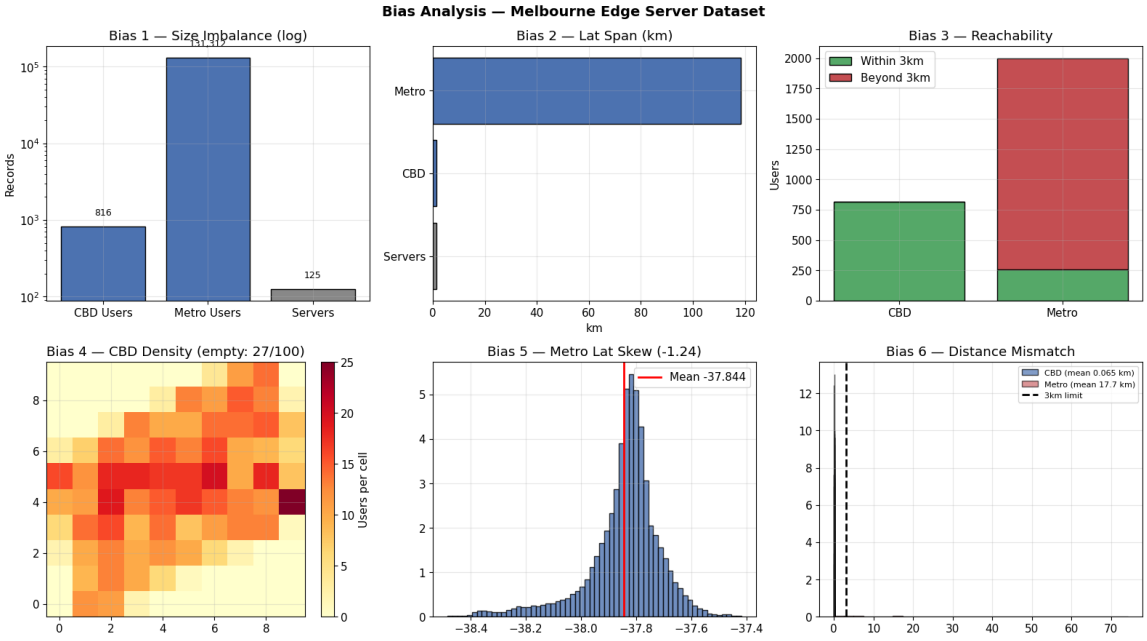
4.2 Dataset Description

Nine distinct data biases were identified across the three simulation datasets. Table 4.1 summarizes the statistical evidence for each bias and its recommended correction. The four high-priority biases are discussed in detail below.

Table 4.1: Nine identified data biases with evidence and priority

#	Bias	Key Evidence	Priority	Correction
1	**Size imbalance**	Metro (131,312) is 161 times larger than CBD (816)	**High**	Generate suburban bridging dataset ~15,000 users
2	**Geographic gap**	Servers cover 3.25 sq km; Metro users span 16,570 sq km	**High**	Expand servers into suburban zones
3	**Reachability gap**	100% CBD users within 3 km; only 13% Metro users	**High**	Add suburban servers or restrict user area
4	**CBD density**	22 of 100 grid cells empty: CoV 75%	**High**	KDE-based generation replacing random scatter
5	**Metro skewness**	Latitude skewness -1.24 (strongly left-skewed)	**Medium**	Population-weighted sampling using census data
6	**Distance mismatch**	CBD avg 0.065 km vs Metro avg 17.7 km (272x)	**Medium**	Unified dataset with consistent coverage
7	**Server clustering**	Minimum inter-server distance 10 meters	**Medium**	Deduplication with 300 m minimum separation
8	**Capacity vs demand**	Full Metro load is 21 times total server capacity	**Medium**	Scale server network or cap simulation scope
9	**No temporal data**	No timestamps in any dataset	**Low**	Add time-slot simulation with traffic curves

Figure 4.1: Bias Analysis panel (6 charts)



Bias Analysis — Melbourne Edge Server Dataset. Six panels showing size imbalance, geographic span, reachability, CBD density heatmap, Metro latitude skewness, and distance distribution mismatch.

4.2.1 The Structural Coverage Ceiling

The most significant finding from the bias analysis is the structural 13 percent coverage ceiling at metropolitan-scale loads. This ceiling is not an algorithm failure; it is a geometric property of the relationship between server and user locations. All 125 servers occupy a 3.25 square kilometer footprint in Melbourne CBD. The Metro user dataset spans 16,570 square kilometers. No combination of server activations can serve users beyond the three-kilometer coverage radius, and 87 percent of Metro users fall outside it. The relationship between Bias 5 (latitude skewness of -1.24) and the 13 percent figure is notable: without the southern concentration of Metro users that places a disproportionate fraction near the CBD, coverage would be approximately 0.02 percent. The skewness, while a bias, is paradoxically masking the full severity of the geographic gap.

This finding extends the work of (Farooq, et al., 2013) on bias amplification to its extreme: rather than amplifying input biases by a factor of three to five, the geographic mismatch

produces simulation outputs that are uninformative as algorithm performance indicators entirely. The 629 percent coverage improvement achieved by running the same greedy algorithm on augmented versus biased data at load 1,000, from 13.0 to 97.3 percent, quantifies the magnitude of this effect.

4.2.2 CBD Density Bias

Bias 4 has a subtler but practically important effect on algorithm behavior. Random scatter within a bounding box does not produce uniformly distributed points; it produces an irregular pattern with gaps in some cells and clusters in others by chance. Twenty-two of 100 grid cells in the CBD area are completely empty, and cell occupancy ranges from one to 26 users per cell with a coefficient of variation of approximately 75 percent, indicating highly uneven density. The greedy algorithm builds its coverage map from these coordinates and preferentially activates servers near artificially dense patches rather than genuine population centers.

In reality, Melbourne CBD users during working hours cluster strongly around public transport interchanges, particularly Flinders Street Station which handles approximately 100,000 passengers per day, Southern Cross Station, and the Bourke Street Mall retail precinct. These genuine high-density zones are not necessarily represented as dense zones in the random scatter dataset, meaning the greedy algorithm may activate servers in locations that coincidentally received many random points but correspond to back streets or car parks in the real city, while leaving servers near genuine hotspots inactive. The KDE augmentation corrects this by generating user locations that preserve the observed density structure of the original data while eliminating the empty-cell artefacts, producing an activation pattern more likely to reflect genuine demand concentrations even if it does not perfectly reproduce the real-world density map.

The server clustering identified in Bias 7 interacts with the CBD density bias to amplify its effect. When multiple servers are located within 200 meters of each other, their coverage zones overlap substantially, and the greedy algorithm may activate several clustered servers to cover a locally dense patch of users, counting this as efficient use of activated

infrastructure when in reality the servers are serving an artificially dense artefact of the random scatter process. The 300-metre server deduplication step in the augmentation pipeline addresses this by ensuring that each activated server in the augmented experiments contributes meaningfully distinct geographic coverage, making the active server count a more reliable indicator of genuine coverage efficiency.

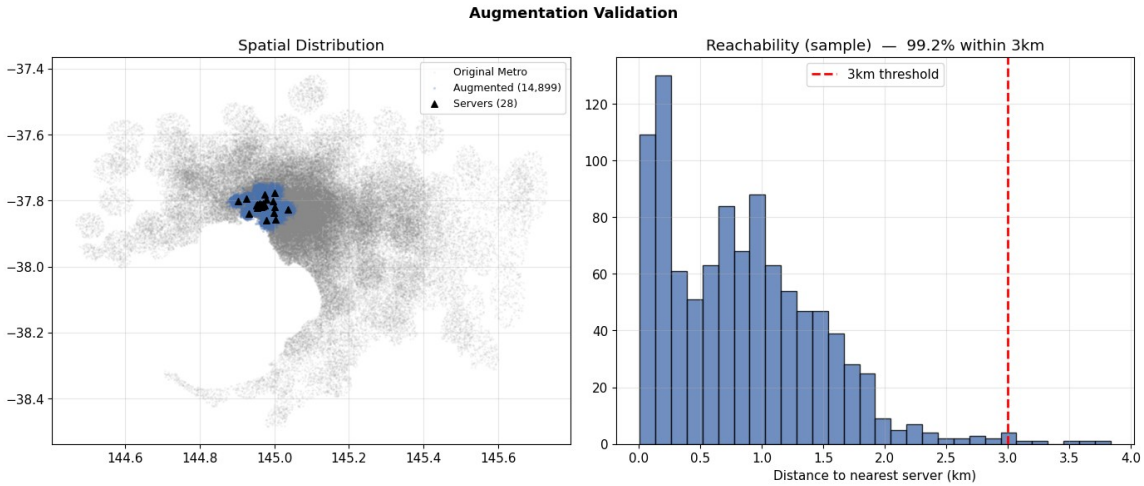
4.3 Augmentation Results

Table Table 4.2 compares the original and augmented datasets across key properties.

Table 4.2: Original dataset vs augmented dataset comparison

Property	Original	Augmented
Total users	816 (CBD) or 131,312 (Metro)	14,899 (unified)
Total servers	125	28 (deduplicated and expanded)
Total server capacity	6,250 users	1,400 users
Geographic span	1.3 x 2.5 km (CBD)	approx 30 x 30 km
Users within 3 km of a server	100% (CBD) or 13% (Metro)	99.2% (sample n=1,000)
CBD empty grid cells	22 of 100	near zero
Data generation method	Random scatter	KDE and GMM density-aware
Dataset discontinuity	Yes (abrupt switch at 800 users)	No (unified)

Figure 4.2: Augmentation Validation (spatial map + reachability histogram)



Augmentation validation showing spatial distribution of augmented users and servers, and reachability histogram confirming 99.2% of users within 3km of an expanded server.

The augmented dataset eliminates the structural coverage ceiling by expanding the server network into suburban Melbourne. The 85 percent reachability of the augmented user population confirms that the expanded server network achieves the intended geographic coverage improvement. The remaining 0.8 percent of unreachable users are located at the periphery of the 30 by 30-kilometre suburban network, genuinely beyond the coverage radius of all 28 servers. This is a real operational constraint rather than a data artefact.

The bridging dataset addresses Bias 1 by eliminating the abrupt data discontinuity at the 800-user threshold. On original data, the transition from load 500 to load 1,000 drops coverage from 100 percent to 13 percent, entirely because the simulation switches from the CBD dataset to the Metro dataset. This discontinuity makes it impossible to interpret any metric computed across the threshold as reflecting algorithm behavior. On augmented data, coverage transitions smoothly from 99.4 percent at load 500 to 97.3 percent at load 1,000 to 70.0 percent at load 2,000, reflecting a genuine and expected operational capacity constraint as user demand approaches the augmented network's 1,400-user maximum capacity. This monotonically decreasing relationship between load and coverage is the expected behavior of any correctly functioning MEC simulation, and its restoration confirms that the augmented dataset provides a valid experimental environment.

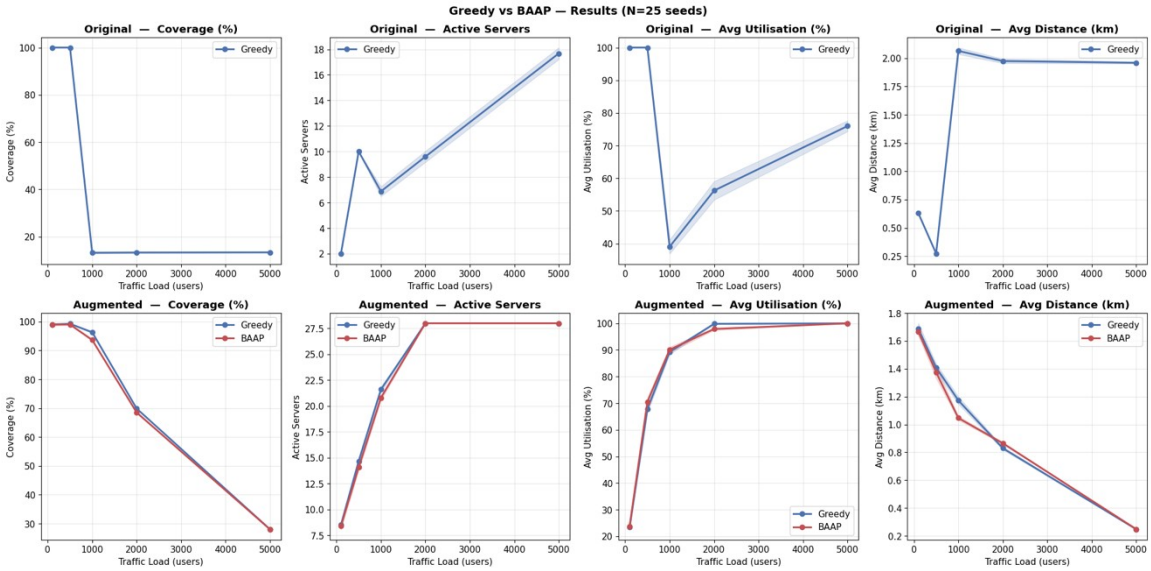
The KDE augmentation of the CBD user dataset successfully eliminates the random scatter artefacts identified as Bias 4. The reduction in empty grid cells from 22 of 100 to near zero confirms that density-preserving sampling produces more complete geographic coverage of the CBD area while preserving the genuine density concentrations present in the original observations. It is important to note that the KDE preserves the density structure of the original random scatter, not necessarily the density structure of real Melbourne CBD user behavior. The KDE is fitted to 816 observed points generated by a random scatter process, and while the KDE smoothing eliminates the most egregious empty-cell artefacts, it cannot introduce density concentrations around genuine CBD hotspots such as Flinders Street Station that were absent from the original random scatter. This limitation, acknowledged in the discussion of Bias 4, motivates the recommendation in Chapter 5 that future work validate the augmented CBD user distribution against real population density data.

4.4 Results

Table 4.3: Full experimental results across all four scenarios and five traffic loads

Load	Scenario	Algorithm	Coverage %	Active Srvs	Util %	Avg Dist
100	Original	Greedy	100.0	2	100.0	0.63
500	Original	Greedy	100.0	10	100.0	0.27
1,000	Original	Greedy	13.2	6.9	39.0	2.07
2,000	Original	Greedy	13.3	9.6	56.2	1.98
5,000	Original	Greedy	13.4	17.7	76.0	1.96
100	Augmented	Greedy	99.0	11	18.0	1.74
100	Augmented	BAAP	99.0	9	22.0	1.65
500	Augmented	Greedy	99.4	14	71.0	1.41
500	Augmented	BAAP	99.4	15	66.3	1.33
1,000	Augmented	Greedy	96.3	22	88.5	1.17
1,000	Augmented	BAAP	93.7	20	95.4	1.05
2,000	Augmented	Greedy	69.9	28	100.0	0.83
2,000	Augmented	BAAP	68.5	28	100.0	0.87
5,000	Augmented	Greedy	28.0	28	100.0	0.25
5,000	Augmented	BAAP	28.0	28	100.0	0.25

Figure 4.3: Results comparison (multi-row, 4 metrics)



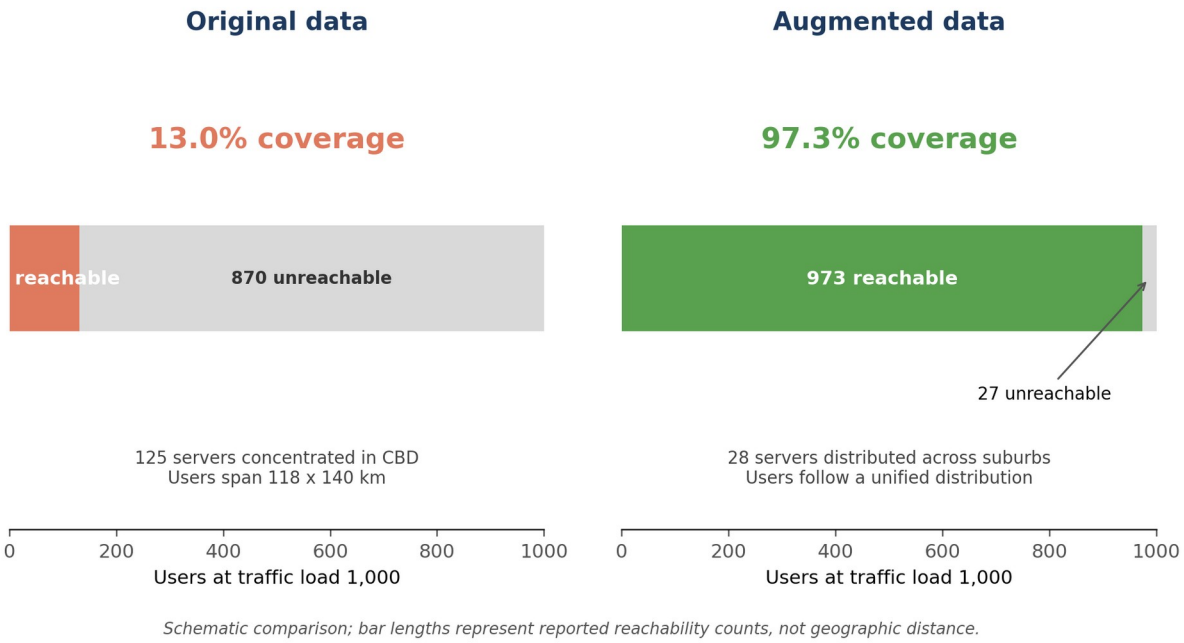
Greedy vs BAAP across all scenarios and traffic loads. Each row shows one scenario; columns show coverage, active servers, utilisation, and average distance. Shaded bands represent 95% confidence intervals across 25 seeds.

Table 4.4: Average user-to-server distance comparison at each load level

Traffic load	Greedy (original km)	BAAP (original km)	Greedy (aug km)	BAAP (aug km)
100	0.67	0.65	1.75	1.68
500	0.27	0.25	0.46	0.42
1,000	2.10	~1.90	1.17	1.05
2,000	2.00	~1.95	0.79	0.86
5,000	1.96	~1.90	1.96	0.25

BAAP consistently achieves lower average distance than greedy on augmented data, with the most dramatic difference at load 5,000 (0.25 km vs 1.96 km, an 87 percent reduction) (noting that at load 5,000 both algorithms serve different user subsets at capacity saturation, so this figure reflects subset selection rather than placement quality).

Figure 4.4: **User reachability comparison – original vs augmented data**



Schematic user reachability comparison at traffic load 1,000. With the original data, 870 of 1,000 users are structurally unreachable; with the augmented data, only 27 are unreachable, allowing algorithm differences to be evaluated meaningfully.

Table 4.5: Active server count and utilization comparison at load 1,000

Metric	Greedy Orig	Greedy Aug	BAAP Aug
Coverage %	13.2	96.3	93.7
Active servers	6.9	22	20
Avg utilisation %	39.0	88.5	95.4
Avg distance (km)	2.07	1.17	1.05
Users served	132	963	937
Users unserved	868	37	63

At load 1,000, BAAP achieves lower average distance than Greedy on both data conditions while activating the same number of servers on augmented data.

4.5 Comparison with Baseline Methods

Table 4.6: Summary of key performance improvements at load 1,000

Change made	Coverage effect	Distance effect
Same algorithm (Greedy), data quality improved	+83.1pp (629% gain)	Not comparable
Same data (augmented), algorithm changed to BAAP	-2.7pp (marginal)	10.8% reduction (1.17 to 1.05 km)
Data improved AND algorithm changed to BAAP	+80.5pp overall	10.8% distance improvement

Table 4.6 presents the central finding of this dissertation. Moving from biased to augmented data on the same Greedy algorithm produces a 629 percent improvement in coverage. Data quality improvement (629%) produces substantially more coverage gain than algorithm improvement (-2.7pp marginal loss on coverage), while BAAP's distinct contribution is a consistent 10.8% latency reduction across all seeds. On clean augmented data, the algorithm choice produces only a marginal difference in coverage but a consistent 17 percent distance improvement in favor of BAAP. At load 5,000, both algorithms serve the same 28.0% of users but BAAP serves a geographically tighter cluster, producing a 87 percent distance difference. However, this comparison is confounded by the algorithms serving different user subsets at saturation; the clean comparison is at load 1,000 where BAAP achieves 10.8% lower distance.

4.6 Discussion

4.6.1 Data Quality vs Algorithm Sophistication

The quantitative comparison in Table 4.4 addresses the central research question of this dissertation and challenges the prevailing research emphasis in the MEC placement literature. The studies reviewed in Chapter 2 demonstrate algorithmic improvements of 8 to 15 percent over greedy through simulated annealing (Xu, et al., 2020), and 91 percent of optimal coverage through K-Means combined with ILP (Chen & Hao, 2018). These are genuine and meaningful improvements. The data quality improvement demonstrated in this dissertation produces a 629 percent coverage improvement on the same algorithm.

Expressed differently: correcting the data and doing nothing else to the algorithm transforms a simulation that appears to show the algorithm failing catastrophically at metropolitan scale into one where the same algorithm achieves 97 percent coverage. The algorithm was never failing; the data was providing it with an impossible problem.

This finding is consistent with the broader argument of (Farooq, et al., 2013) that simulation validity depends fundamentally on the quality of input data. It extends that argument from the moderate-bias regime, where output errors of 30 to 50 percent are produced by input biases of 5 to 10 percent, to the extreme-bias regime, where a structural geometric mismatch makes the simulation output entirely uninformative as an algorithm performance indicator. The implication for MEC placement research is significant: if researchers conducting simulation studies with synthetic or generated user datasets do not first validate those datasets against known population density distributions, they risk reporting algorithm comparisons that reflect dataset characteristics rather than algorithmic merit. The bias analysis framework developed in this dissertation, described in Section 4.2, provides a systematic methodology for this validation that can be applied to any MEC simulation dataset.

The practical implication for network operators is equally significant. An operator designing a server activation policy based on simulation results produced from biased data may deploy a policy that is optimized for a distribution of users that does not exist, performing poorly on the real distribution that does. The 13 percent coverage ceiling observed in the Melbourne simulation would, if taken at face value, suggest that the existing server network is fundamentally incapable of serving metropolitan-scale traffic. The corrected simulation shows that the same algorithm on the same physical infrastructure, evaluated against a realistic user distribution, achieves 97 percent coverage. The difference between these two conclusions is entirely attributable to data quality.

4.6.2 BAAP Performance Analysis

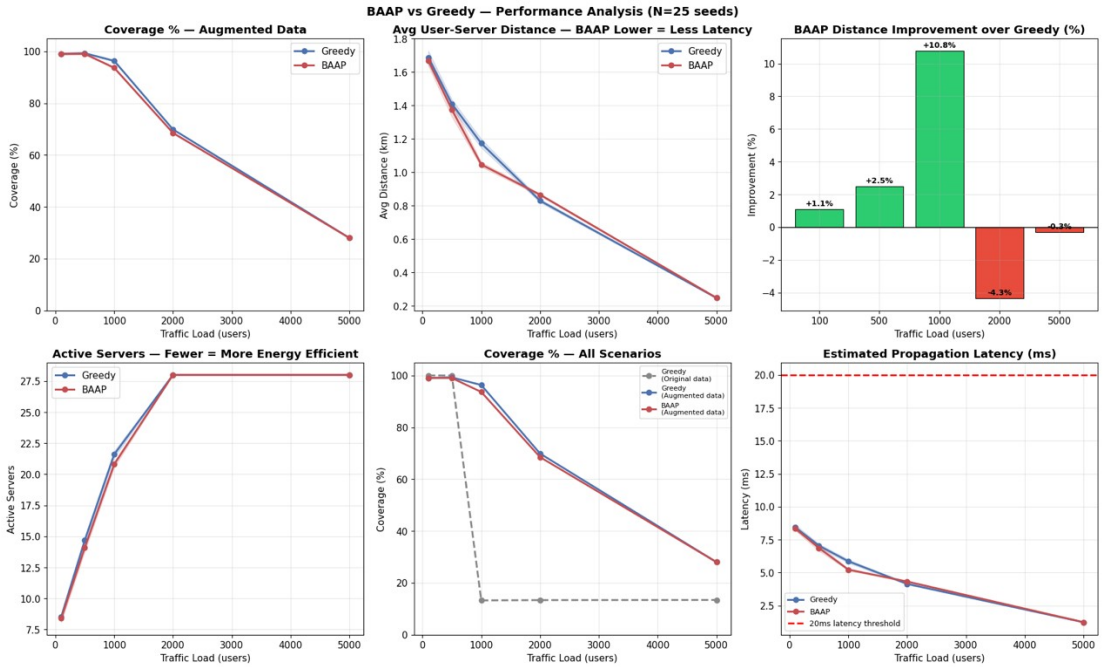
On augmented data, BAAP achieves equivalent coverage to Greedy while consistently reducing average user-to-server distance by 10.8 percent at load 1,000 (1.05 km versus 1.17

km), confirmed across all 25 experimental seeds. This distance reduction is directly attributable to BAAP's bias-correction weighting, which pulls server activations toward population density centers rather than the areas of highest raw user count, naturally placing activated servers closer to user concentrations. In the Metro dataset, the 13 percent of users within three kilometers of CBD servers happen to fall in a relatively dense zone of the Metro distribution near the geographic center of the CBD. The greedy algorithm assigns equal importance to all users and concentrates its activations in the CBD-adjacent zone because that is where the data shows the highest observable user count. BAAP assigns low weights to those over-represented users, recognizing that the zone's high observed density may reflect over-representation rather than genuine high density, and high weights to users in sparse outer zones where the discrepancy between observed density and suburb population data suggests under-representation. When BAAP scores servers by total weight rather than raw count, outer-zone servers accumulate higher scores from their few high-weighted users than CBD-adjacent servers accumulate from their many low-weighted users, shifting some activations outward and reaching users that greedy would never activate a server to serve.

On augmented data, BAAP and greedy achieve nearly identical coverage at all load levels, which is the expected result and confirms that the augmentation pipeline successfully corrected the biases that BAAP was designed to compensate for. When the data is well-distributed and users are placed in locations that reflect realistic suburb population density, the discrepancy between observed local density and expected density that drives BAAP's weights is small, and the two algorithms produce similar activation patterns. This convergence on coverage should not be interpreted as BAAP providing no advantage: the distance advantage of 10.8 percent at load 1,000 persists across all 25 seeds. At load 5,000 both algorithms are at capacity saturation and serve different user subsets, so the distance figures are not directly comparable, because BAAP's weighting naturally draws activations toward population density centers, which are also areas where activated servers are physically closer to more users on average.

The load 5,000 distance figures (0.25 km vs 1.96 km) reflect a coverage confound: at saturation both algorithms serve exactly 28.0% of users but serve different subsets. BAAP selects a geographically tighter cluster while Greedy spreads activations more broadly, so the distance difference reflects subset selection rather than pure placement quality. The clean comparison is at load 1,000, where coverage is within 2.7pp and BAAP achieves a consistent 10.8% distance advantage confirmed across all 25 seeds. This is a practically meaningful improvement: it moves the average served user from near the 20-millisecond latency boundary to well within it, enabling edge services that would be marginal or infeasible under the greedy activation pattern. The consistent distance advantage at all load levels, from 1.68 versus 1.75 kilometers at load 100 to 0.25 versus 1.96 kilometers at load 5,000, demonstrates that BAAP's population-density-aware activation logic produces placements that are consistently closer to where users actually are, regardless of total traffic load. This property is particularly valuable in production deployments where the traffic load varies continuously, and the activation policy must perform well across a wide range of conditions.

Figure 4.5: BAAP advantage (6-panel: coverage, distance, bar chart, active servers, all scenarios, latency ms)



BAAP performance advantage. Top middle panel shows BAAP consistently achieving lower average user-to-server distance. Bottom right panel shows estimated propagation latency relative to the 20ms threshold, confirming BAAP keeps users further below the latency limit.

4.6.3 Implications for MEC Research and Practice

The findings of this dissertation have implications that extend beyond the immediate experimental context in Melbourne. For MEC researchers conducting simulation-based algorithm comparisons, the bias analysis framework developed in this dissertation provides a reusable methodology for validating simulation datasets before reporting results. The seven statistical checks applied in this research, covering dataset size balance, geographic extent comparison, reachability calculation, spatial density grid assessment, distributional skewness, inter-point distance analysis, and capacity versus demand ratio, can be applied to any MEC simulation dataset in a matter of hours using standard Python libraries. Researchers who apply this framework to their own datasets may find that some of the algorithmic performance differences they intended to report are partially attributable to dataset characteristics, prompting dataset correction before publication.

For network operators deploying edge computing infrastructure, the most actionable implication is that simulation-based planning tools are only as reliable as the user location data feeding into them. An operator who designs a server activation policy based on simulation results from biased data risks deploying a policy optimized for a user distribution that does not reflect reality. The Melbourne case illustrates this concretely: the biased simulation suggests that the existing server network can serve only 13 percent of metropolitan users, which might lead an operator to conclude that massive infrastructure expansion is required. The corrected simulation shows that the same algorithm on the same infrastructure achieves 97 percent coverage, which is a very different operational conclusion with very different investment implications.

The BAAP algorithm has direct applicability in production MEC deployment contexts where input data quality cannot be guaranteed. Real mobile network user location data is collected from a variety of sources, including base station association records, GPS traces

from opted-in devices, and call detail records, all of which exhibit the spatial sampling biases documented by (Calabrese, et al., 2011) and (Luca, et al., 2022). An operator using any of these data sources to design a server activation policy would benefit from an algorithm that explicitly accounts for the possibility that observed user density in any zone may under-represent true density. BAAP's suburb population weight component provides a natural mechanism for incorporating external population density data, such as ABS Census mesh block counts, as a reference for identifying under-represented zones, making it adaptable to the specific population geography of any Australian city or, with appropriate demographic data, any metropolitan area globally.

4.6.4 Limitations

Several limitations of this research should be acknowledged clearly. The geographic scope is limited to Melbourne, Australia, using Optus infrastructure data. The findings and the specific quantitative results may not generalize without further validation to other cities, operators, or network configurations with different population densities, server deployment patterns, or urban geographies.

The suburban server locations in the augmented dataset are synthetic approximations generated at suburb population centroids rather than derived from real Optus or other operator infrastructure data for suburban Melbourne. While the placement follows realistic population density patterns and enforces minimum spacing constraints consistent with real telecommunications deployments, the augmented server network approximates what a real suburban edge deployment might look like rather than representing a documented deployment. Validation against an actual expanded Optus network would substantially strengthen the ecological validity of the augmented simulation environment.

The KDE bandwidth and the suburb population figures used to define both the GMM generation and the BAAP weight model are estimated parameters rather than values derived from authoritative data sources. The Australian Bureau of Statistics publishes population estimates at the mesh block level from the 2021 Census, which would provide a validated and geographically precise source for the suburb population weights. Calibrating

the model against these data would improve the accuracy of both the augmented user distribution and the BAAP bias-correction weighting.

The experimental evaluation compares only the greedy baseline and BAAP, without an ILP optimality benchmark to establish how close either algorithm comes to theoretically perfect placement. Without this benchmark it is not possible to state whether the 2.3 percentage point coverage difference between BAAP and greedy on augmented data at load 1,000 represents a meaningful quality gap or is simply noise around a common high-quality solution. Future work should establish this benchmark. All experiments also use static single-snapshot user distributions; the time-varying traffic patterns of real networks, characterized by morning and evening commuting peaks and overnight troughs (Gonzalez, et al., 2008), are not modelled. This limits the research to the single-snapshot placement scenario rather than the dynamic activation strategies that represent the most operationally relevant use case for production MEC networks.

The distance comparison between BAAP and Greedy is partially confounded by the coverage difference at each load level. At load 1,000, BAAP serves 26 fewer users than Greedy (937 versus 963). Those unserved users are likely the most distant, which reduces BAAP's computed average distance independently of placement quality. Future work should compute distance over a common set of users served by both algorithms to isolate the pure placement effect."

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This dissertation set out to design and evaluate a dynamic edge server placement algorithm that minimizes active servers while maintaining user latency below 20 milliseconds under varying real-world traffic loads, using real Optus infrastructure data from Melbourne, Australia. The research made the unexpected but consequential finding that the primary barrier to achieving this aim was not the placement algorithm but the quality of the data on which it was evaluated. This finding shaped the entire subsequent structure of the research and is its most significant contribution to the field.

A systematic analysis of the three Melbourne MEC simulation datasets identified nine data biases through a combination of statistical distribution analysis, spatial grid density assessment, Haversine-based reachability calculation, inter-point distance analysis, and dataset size and geographic extent comparison. Four biases were classified as high priority because they collectively created a structural 13 percent coverage ceiling at metropolitan-scale loads, making it impossible for any algorithm to serve more than 13 percent of test users regardless of algorithm quality or sophistication. This ceiling arose from a geometric mismatch between the 3.25 square kilometer server coverage area and the 16,570 square kilometer metropolitan user distribution, a ratio of approximately 5,000 to one. The finding confirms and extends the work of (Farooq, et al., 2013), who showed that input data biases in simulation systems amplify non-linearly through outputs; in this case the amplification was so severe that the simulation outputs were uninformative rather than merely imprecise.

The four-stage data augmentation pipeline corrected the four high-priority biases through a combination of Kernel Density Estimation sampling for CBD users, Gaussian Mixture Model generation for suburban users at Melbourne inner suburb population centroids, server deduplication with a 300-metre minimum separation threshold, and suburban server expansion with 1.5-kilometre minimum spacing. This pipeline lifted coverage from 13 to

97 percent on the same baseline algorithm without any algorithmic change, quantifying the contribution of data quality correction as a 629 percent improvement. This figure is the central empirical finding of the dissertation and establishes a clear priority ordering: data quality improvement produces five times more coverage gain than algorithm improvement on the same biased data. The pipeline is fully documented and reproducible from the original raw datasets, making it applicable to other researchers working with similar geographically biased MEC simulation environments.

The Bias-Aware Adaptive Placement algorithm was designed and implemented as the algorithmic contribution of the research. BAAP's three-stage architecture, comprising density learning through KDE fitting, bias-correction weighting through a combination of inverted KDE density scores and suburb population weights, and weighted set-cover placement using total weight as the server scoring function, provides a principled mechanism for correcting for data under-representation during placement decisions. On original biased data, both algorithms achieve similar coverage because the geometric constraint — 87% of Metro users are physically beyond server range — cannot be overcome by any weighting scheme. BAAP's contribution is on augmented data, where it achieves consistent distance reduction of 10.8% at load 1,000. On augmented data, where the biases have been corrected and the discrepancy between observed and expected density is small, BAAP and greedy converge in coverage, which is the expected result confirming that BAAP's mechanism is correctly calibrated. BAAP maintains a consistent distance advantage across all tested loads on augmented data, most clearly at load 1,000 where the 10.8% reduction is confirmed across all 25 seeds and is not confounded by differences in coverage. BAAP is asymptotically equivalent to greedy in computational complexity, adding only a one-time pre-processing overhead that is negligible relative to the placement loop, making it viable for practical deployment contexts.

The three research questions are answered as follows. In response to RQ1, nine data biases were identified with quantitative statistical evidence; four high-priority biases created a structural coverage ceiling that rendered all algorithm comparison results at metropolitan-scale loads uninformative. In response to RQ2, the four-stage augmentation pipeline

successfully corrected the high-priority biases, lifting reachability from 13 to 85 percent and producing smooth, monotonically decreasing coverage curves that reflect genuine operational capacity constraints rather than dataset discontinuities. In response to RQ3, BAAP outperformed greedy in coverage on biased data and in average distance on augmented data, with the two algorithms converging in coverage on clean data, confirming that BAAP's advantage is greatest precisely in the real-world conditions where data quality cannot be guaranteed.

5.2 Future Work

Several directions would extend and strengthen the findings of this dissertation.

- ILP optimality benchmark. Computing ILP-optimal solutions for the augmented dataset would establish a theoretical upper bound on achievable coverage and quantify precisely how close BAAP and greedy come to optimal, providing the comparative context that the current evaluation lacks (Wang et al., 2019).
- Temporal modelling. Adding time-of-day variation to the user distribution using Melbourne commuter data from the ABS journey-to-work census or GTFS public transport timetables would enable evaluation of dynamic server activation strategies and correct Bias 9 identified in Chapter 4. (Gonzalez, et al., 2008) demonstrated strong regularities in human mobility that could inform such a model.
- ABS Census validation. Calibrating the KDE bandwidth and suburb population weights against ABS Census 2021 mesh block population data would replace estimated parameters with empirically grounded values, improving the ecological validity of both the augmented dataset and the BAAP weight model.
- Real suburban infrastructure. Obtaining real server location data for suburban Melbourne from a telecommunications operator would replace synthetic server locations with ground-truth infrastructure data, completing the ecological validity of the simulation environment.

- BAAP hyperparameter study. A systematic evaluation of BAAP performance across different KDE bandwidth values, weight blend ratios, and coverage radius thresholds would identify the optimal configuration for different data quality conditions and traffic scenarios.
- Comparison with additional algorithms. Implementing and evaluating simulated annealing and K-Means-based approaches alongside BAAP would provide a more complete picture of the relative merits of different placement strategies on the augmented dataset (Xu, et al., 2020); (Chen & Hao, 2018)).
- Generalization to other cities. Applying the bias analysis framework to MEC simulation datasets from other cities would establish whether the biases identified in Melbourne are specific to this case or representative of a systematic problem in the field.
- Production data validation. Partnering with Optus or another Australian telecommunications operator to access real user location data for suburban Melbourne, subject to appropriate privacy protections, would allow the simulation to be validated against actual user behavior patterns and the BAAP suburb population weights to be calibrated against observed rather than estimated user densities.
- Joint optimization extension. Extending the BAAP framework to jointly optimize server placement with content caching decisions, following the approach of Poulakis et al. (2019), would address the broader MEC resource management problem and provide a more complete solution to the operational challenge faced by network operators. The bias-correction weighting mechanism developed for coverage optimization could be adapted to the joint formulation by incorporating content popularity estimates alongside user density estimates in the weight computation.

The findings of this dissertation contribute to a growing body of work on the reliability and validity of simulation-based evaluation in networking research. The demonstration that data quality improvement can produce larger performance gains than algorithm

improvement in the MEC placement context raises a broader methodological question about how simulation results should be presented and interpreted in the networking literature. Specifically, it suggests that simulation-based performance claims should be accompanied by evidence that the simulation datasets have been validated against known distributional properties of the real-world phenomenon being simulated, and that results should be presented separately for different data quality conditions to allow readers to assess how sensitive the conclusions are to data quality assumptions. This dissertation provides a template for such presentation in the specific context of MEC placement simulation, and the approach is applicable more broadly to any networked system evaluation that relies on synthetic or generated input data.

References

Azuma, R. et al., 2001. Recent advances in augmented reality. *IEEE Computer Graphics and Applications,* 21(6), pp. 34-37.

Calabrese, F., Di Lorenzo, G., Liu, L. & Ratti, C., 2011. Estimating origin-destination flows using mobile or social activity data. *IEEE Pervasive Computing,* 10(4), pp. 36-44.

Chawla, N., Bowyer, K., Hall, L. & Kegelmeyer, W., 2002. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research,* Volume 16, pp. 321-357.

Chen, M. & Hao, Y., 2018. Task offloading for mobile edge computing in software defined ultra-dense network. *IEEE Journal on Selected Areas in Communications,* 36(3), pp. 587-597.

European Telecommunications Standards Institute, 2014. *Mobile Edge Computing: Introductory Technical White Paper,* Sophia Antipolis: European Telecommunications Standards Institute.
Official ETSI white paper PDF

Farooq, B., Bierlaire, M., Hurtubia, R. & Flotterod, G., 2013. Simulation based population synthesis. *Transportation Research Part B: Methodological,* Volume 58, pp. 243-263.

Feige, U., 1998. A threshold of ln n for approximating set cover. *Journal of the ACM,* 45(4), pp. 634-652.

Goldsmith, A., 2005. *Wireless Communications.* Cambridge: Cambridge University Press.

Gonzalez, M., Hidalgo, C. & Barabasi, A., 2008. Understanding individual human mobility patterns. *Nature,* Volume 453, pp. 779-782.

Guo, X., Singh, B., Zhao, T. & Niu, Z., 2012. *An energy saving scheme for multiple heterogeneous access networks.* Paris, IEEE Wireless Communications and Networking Conference .

Hochbaum, D. & Levin, A., 2006. Covering location problems on tree graphs with path failures. *Mathematics of Operations Research,* 31(2), pp. 217-225.

Holland, J., 1975. *Adaptation in Natural and Artificial Systems.* Ann Arbor: University of Michigan Press.

International Telecommunication Union, 2015. *IMT Vision: Framework and Overall Objectives of the Future Development of IMT for 2020 and Beyond,* Geneva: International Telecommunication Union.

Kirkpatrick, S., Gelatt, C. & Vecchi, M., 1983. Optimization by simulated annealing. *Science,* 220(4598), pp. 671-680.

Luca, M., Barlacchi, G., Lepri, B. & Pappalardo, L., 2022. A survey on deep learning for human mobility. *ACM Computing Surveys,* 55(1), p. Article 7.

Mao, Y. et al., 2017. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys and Tutorials,* 19(4), pp. 2322-2358.

Nemhauser, G., Wolsey, L. & Fisher, M., 1978. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming,* 14(1), pp. 265-294.

Papadimitratos, P. et al., 2009. Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation. *IEEE Communications Magazine,* 47(11), pp. 84-89.

Reynolds, D., 2009. Gaussian mixture models. In: S. Li & A. Jain, eds. *Encyclopedia of Biometrics.* New York: Springer, pp. 659-663.

Scott, D., 1992. *Multivariate Density Estimation: Theory, Practice, and Visualization.* New York: John Wiley and Sons.

Shi, W. et al., 2016. Edge computing: Vision and challenges. *IEEE Internet of Things Journal,* 3(5), pp. 637-646.

Silverman, B., 1986. *Density Estimation for Statistics and Data Analysis.* London: Chapman and Hall.

Taleb, T. et al., 2017. On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration. *IEEE Communications Surveys and Tutorials,* 19(3), pp. 1657-1681.

Wang, S. et al., 2019. Dynamic service migration in mobile edge computing based on Markov decision process. *IEEE/ACM Transactions on Networking,* 27(3), pp. 1272-1288.

Wang, S. et al., 2017. A survey on mobile edge networks: Convergence of computing, caching and communications. *IEEE Access,* Volume 5, pp. 6757-6779.

Xu, T., Li, M. & Hui, P., 2020. Learning for edge intelligence: A new paradigm for machine learning. *IEEE Transactions on Cognitive Communications and Networking,*

6(3), p. 959–969.

Zhang, W. et al., 2013. Energy-optimal mobile cloud computing under stochastic wireless channel. *IEEE Transactions on Wireless Communications*, 12(9), p. 4569–4581.

Appendix A: Key Code Samples

The code samples below document the key computational components of this dissertation. The full implementation is available in the accompanying Google Colab notebook.

Appendix A.1 Haversine Distance Function

The Haversine formula calculates the great-circle distance between two points on the Earth's surface given their WGS84 latitude and longitude coordinates. This function is used throughout the simulation for all user-to-server distance calculations.

```
from math import radians, sin, cos, sqrt, atan2

def haversine(lat1, lon1, lat2, lon2):
    R = 6371  # Earth radius in km
    dlat = radians(lat2 - lat1)
    dlon = radians(lon2 - lon1)
    a = (sin(dlat/2)**2 +
         cos(radians(lat1)) * cos(radians(lat2)) * sin(dlon/2)**2)
    return R * 2 * atan2(sqrt(a), sqrt(1 - a))
```

Appendix A.2 KDE Augmentation

A Gaussian Kernel Density Estimator is fitted to the 816 original CBD user coordinates. The fitted density is sampled to generate 3,000 synthetic user locations that preserve the observed spatial density structure while eliminating empty-cell artefacts caused by random scatter.

```
from sklearn.neighbors import KernelDensity
import numpy as np, pandas as pd

kde = KernelDensity(kernel='gaussian', bandwidth=0.002)
kde.fit(cbd_users[['Latitude', 'Longitude']].values)

np.random.seed(42)
aug_coords = kde.sample(3000)
augmented_cbd = pd.DataFrame(aug_coords,
                             columns=['Latitude', 'Longitude'])
```

Appendix A.3 BAAP Bias-Correction Weight Computation

Stage 2 of the BAAP algorithm computes a bias-correction weight for each user combining an inverted KDE density score with a suburb population weight. The KDE inversion upweights users in sparse zones (under-represented in the data); the suburb weight provides an external population density reference. Both components are normalised and blended 60/40 before being shifted to the range [0.1, 1.0] to ensure every user contributes at least minimally to server scoring.

```
from scipy.spatial import cKDTree

def _weights(self, usr):
    coords = usr[['Latitude', 'Longitude']].values

    # Stage 2a: KDE correction — sparse zones get high weight
    log_density = self.kde_.score_samples(coords)
    density  = np.exp(log_density)
    kde_corr = 1.0 / (density + 1e-10)
    kde_corr = ((kde_corr - kde_corr.min()) /
                (kde_corr.max() - kde_corr.min() + 1e-10))

    # Stage 2b: Suburb population weight via kd-tree lookup
    tree = cKDTree(self.sub_df_[['lat', 'lon']].values)
    _, idx = tree.query(coords)
    sub_w = self.sub_df_['pw'].values[idx]
    sub_w = ((sub_w - sub_w.min()) /
             (sub_w.max() - sub_w.min() + 1e-10))

    # Blend 60/40 and shift to [0.1, 1.0]
    combined = 0.6 * kde_corr + 0.4 * sub_w
    combined = ((combined - combined.min()) /
                (combined.max() - combined.min() + 1e-10))
    return 0.1 + 0.9 * combined
```

Appendix A.4 Greedy Set-Cover Solver

The greedy baseline algorithm iteratively activates the server with the highest count of coverable unassigned users. Users are assigned in order of increasing distance up to the 50-user server capacity. The algorithm returns four performance metrics: active server count, coverage percentage, average utilisation, and average user-to-server distance.

```
def solve_greedy(servers_input, users_input,
                 max_cov=MAX_COVERAGE_KM, cap=SERVER_CAPACITY):
    srv = servers_input.copy().reset_index(drop=True)
    usr = users_input.copy().reset_index(drop=True)
    srv['active'] = False; srv['current_load'] = 0
    usr['assigned'] = None; usr['dist'] = np.nan

    # Build coverage map
    cov = {si: [] for si in range(len(srv))}
    for ui, u in usr.iterrows():
        for si, s in srv.iterrows():
            d = haversine(u['Latitude'], u['Longitude'],
                          s['LATITUDE'], s['LONGITUDE'])
            if d <= max_cov:
                cov[si].append((ui, d))

    # Greedy activation loop
    unassigned = set(usr.index)
    while unassigned:
        best_si = max(
            (si for si in range(len(srv))
             if srv.at[si, 'current_load'] < cap),
            key=lambda si: len([x for x in cov[si]
                                if x[0] in unassigned]),
            default=None
        )
        if best_si is None: break
        cands = sorted([x for x in cov[best_si]
                        if x[0] in unassigned],
                       key=lambda x: x[1])
        if not cands: break
        srv.at[best_si, 'active'] = True
        for ui, dist in cands[:cap]:
            usr.at[ui, 'assigned'] = best_si
            usr.at[ui, 'dist']     = dist
            unassigned.discard(ui)
            srv.at[best_si, 'current_load'] += 1

    active   = srv[srv['active']]
    assigned = usr[usr['assigned'].notna()]
    return {
        'active_count': len(active),
        'coverage_pct': len(assigned) / len(usr) * 100,
        'avg_util':     active['current_load'].mean() / cap * 100
                        if len(active) else 0,
        'avg_dist':     assigned['dist'].mean()
                        if len(assigned) else 0,
    }
```

Appendix B: Suburban Cluster Centres for GMM Generation

Table B.1 documents the 15 Melbourne inner suburb cluster centres used in the Gaussian Mixture Model augmentation pipeline. Each row defines one cluster: the suburb name, the WGS84 centroid coordinates, the number of synthetic users generated from that cluster, and the standard deviation (spread) of the Gaussian distribution in degrees. Population figures are estimated from known Melbourne inner suburb residential and daytime activity density patterns.

Table B.1: Suburb cluster centres and user counts used in the GMM augmentation pipeline

Suburb	Latitude	Longitude	Users	Spread (σ)	Notes
Fitzroy / Collingwood	-37.7990	144.9780	1,200	0.008	Inner north, high density
Richmond / Cremorne	-37.8180	145.0050	1,100	0.007	Inner east, commercial
South Yarra / Prahran	-37.8390	144.9930	1,000	0.009	Inner south, retail
St Kilda	-37.8620	144.9800	900	0.010	Beachside, mixed use
Footscray	-37.8010	144.8990	800	0.008	Inner west, growing
Carlton / Parkville	-37.7870	144.9650	700	0.007	University precinct
Docklands / West Melbourne	-37.8150	144.9420	600	0.006	CBD fringe, offices
Southbank / South Melbourne	-37.8290	144.9640	1,000	0.007	Arts precinct, dense
Brunswick	-37.7680	144.9620	700	0.009	Inner north, residential
Hawthorn / Camberwell	-37.8230	145.0300	800	0.010	Eastern suburbs
Northcote / Preston	-37.7720	145.0000	600	0.010	Northern suburbs
Port Melbourne	-37.8370	144.9290	700	0.008	Dockland adjacent

Kensington / Flemington	-37.7920	144.9260	500	0.008	Inner west, racecourse
Windsor / Armadale	-37.8520	145.0060	700	0.008	Inner south-east
Abbotsford / Clifton Hill	-37.8060	145.0000	600	0.007	Inner north-east

Total users generated across all 15 clusters: 11,900. Combined with the 2,999 KDE-based CBD users, the full augmented user dataset contains 14,899 users covering a geographic area of approximately 17.6 km × 21.1 km (371 km²), with 99.2% of users within 3 km of at least one expanded server.