Published by
NIGERIAN SOCIETY OF PHYSICAL SCIENCES
Available online @ https://asr.nsps.org.ng/index.php/asr

African Scientific Reports 4 (2025) 292

African Scientific Reports

Optimal distribution of vehicular traffic flow with Dijkstra's algorithm and Markov chains

L. I. Igbinosun*, N. R. Udoenoh

Department of Mathematics, University of Uyo, Uyo. Nigeria

Abstract

The optimization of traffic distribution is a critical problem in transportation networks, where efficient routing can reduce congestion, minimize travel time, and improve overall traffic flow. This work explores the combined application of Dijkstra's algorithm and Markov chains to model and optimize vehicular traffic distribution in road networks linking multiple cities. Dijkstra's algorithm is deployed to determine the shortest path between two nodes in a weighted graph, allowing for optimal, real-time routing of vehicles across a network based on minimal distance or congestion. In parallel, Markov chains are used to model the long-term probabilistic distribution of traffic across various routes, providing insights into steady-state traffic flows and congestion patterns. The combination of these two approaches addresses both immediate and long-term traffic management concerns. Our findings show that while Dijkstra's algorithm offers immediate routing solutions, Markov chains can model the long-term behavior of traffic, leading to a more efficient planning and decision-making.

DOI:10.46481/asr.2025.4.2.292

Keywords: Dijkstra's algorithm, Markov chains, Vehicular traffic flow, Traffic optimization

Article History :
Received: 10 March 2025
Received in revised form: 13 July 2025
Accepted for publication: 25 July 2025
Published: 25 August 2025

1. Introduction

The term 'Vehicular Traffic' mostly refer to the movement of vehicles on the road, a phenomena that is literally an integral part of daily life in modern society today. The constant need to commute goods and services or simply oneself from one point to another may pose several challenges to road users, one of the most worrisome being traffic congestion. Road traffic congestion is a multi-facet global problem. To clearly picture the scale of traffic associated problems, see Refs. [1, 2]. Thus, a host of policy and decision-makers worldwide constantly promote and engaged in efforts geared towards remedying or at best reducing the traffic congestion problem in their cities [3–5].

The problem has also attracted a huge and growing spectrum of scientific minds including experts from the social sciences, as it is not unconnected to other concerns like air pollution, greenhouse gas emission, noise pollution and so on [6]. It has also been linked to lower quality of life in the modern cities as it hinders access to prompt healthcare (especially in life threatening scenarios), sustainable employment or even basic education in some cases [7]. The endless list of issues that could possibly arise from traffic

*Corresponding author: Tel.: +234-708-653-5324.
Email address: luckyigbinosun@uniuyo.edu.ng (L. I. Igbinosun)

congestion has motivated and inspired numerous research articles professing viable solutions to the negative cascading effects of vehicular traffic congestion [8–10].

In this paper, we propose an unchartered but effective method or approach to handling traffic congestion. Our approach involves the combination of Dijkstra's algorithm from graph theory with the well known Markov chains transition matrix from probability theory to achieve optimal distribution of traffic within a sampled network of nodes (cities or towns). We deploy Dijkstra's algorithm to obtain the shortest (least costly) paths within a network of nodes having weighted edges. Vehicular traffic on analogous road networks can then be optimally distributed along the shortest or least congested routes between any two points in the network. The Markov chains then presents a probabilistic approach to predicting long-term traffic distribution patterns using transition matrix.

Traffic flow optimization is critical in urban planning and transportation management to ensure efficient vehicle movement, reduce congestion, and minimize travel times. This paper explores the integration of Dijkstra's algorithm, which is used for real-time routing decisions by finding the shortest paths between nodes in a network, and Markov chains, which model the probabilistic distribution of traffic over time.

Combining these two methodologies, the authors propose a system that handles both immediate and long-term traffic management. Previous studies like those by Seabe & Simelane [4] on road traffic optimization in urban settings emphasize the utility of algorithms like Dijkstra's in mitigating congestion issues. Dijkstra's algorithm is well-known in computer science for finding the shortest paths in a weighted graph. The algorithm calculates the least congested or shortest route between two points in a transportation network. The steps involve iterating through nodes and updating distances based on previously computed minimal paths.

For real-time applications, Dijkstra's algorithm can dynamically route traffic based on current network conditions. Markov chains offer a probabilistic approach to understanding traffic distribution in the long term. The transition matrix in a Markov Chain represents the likelihood of moving from one state (intersection) to another. Over time, this model predicts steady-state distributions, which can inform traffic engineers about long-term congestion patterns. Studies such as Jallow [5] have shown that stochastic models like Markov chains are effective in modeling the dynamic nature of urban traffic systems.

The innovative aspect of this paper is the integration of Dijkstra's algorithm for immediate traffic routing and Markov chains for long-term distribution analysis. This combined approach ensures that the transportation network is optimized both in real-time and over extended periods. This dual strategy helps prevent congestion by adjusting paths based on traffic predictions, thus balancing vehicle flow across the network. As demonstrated in the provided illustrative example, the shortest path from City A to City D is calculated using Dijkstra's algorithm, while the long-term traffic distribution at different intersections is modeled using Markov chains. This method has potential applications in Intelligent Transportation Systems (ITS), where real-time data can be leveraged to enhance traffic management.

Previous research highlights several challenges in implementing traffic management strategies in urban areas. Igbinosun & Ezugwu [1] conducted traffic flow analysis at key intersections in Uyo, revealing high peak hour flows, which necessitate sophisticated traffic management systems. Additionally, Igbinosun & Izevbizua [2] explored control strategies for traffic management, emphasizing the relationship between vehicle speed and congestion. The authors identified poor road conditions, lack of coordination among agencies, and inadequate enforcement of traffic rules as major obstacles, consistent with findings in Nigeria's National Transport Policy 2010 [11].

Technological advancements are pivotal in solving urban traffic challenges. Intelligent Transportation Systems (ITS) and real-time data analysis are increasingly being employed to manage traffic more effectively, as discussed in several studies [12, 13]. These systems help optimize signal timings, provide alternative routes, and reduce congestion. By integrating predictive models like Markov chains, traffic management can decisively shift from reactive to proactive strategies, resulting in significantly better long-term planning. The combination of Dijkstra's algorithm and Markov chains provides a robust solution to both short-term and long-term traffic distribution challenges. This approach effectively balances real-time routing with probabilistic traffic modeling, leading to substantial reductions in congestion and marked improvements in travel efficiency, particularly in urban environments like Uyo. Future research must prioritize enhancing the real-time capabilities of these systems by incorporating dynamic data sources, such as vehicle GPS data, to further refine traffic predictions.

2. Methodology

Dijkstra's algorithm is quite suited for Single-Sourced Shortest Path (SSSP) problems with non-negatively weighted graphs. A graph is basically a set of nodes (vertices), connected or linked by edges. When the edges have the same weight (usually set to the value 1), the graph is termed unweighted otherwise weighted. In what follows, we outline the basic notions, definitions and other mathematical tools needed for this work.

2.1. Notions and steps in implementing Dijkstra's algorithm

The shortest path between any two nodes is the link-distance that has the shortest length. The source (or root) vertex is the starting node in the algorithm and designated with distance length 0.

Definition 2.1 (Digraph)**.** *Let $G = (V, E)$ be a digraph with weights $\omega : E \to \mathbb{R}$, assign edge to real-valued weights. If $\bar{e} = (u, v) \in E$, we may write $\omega(u, v)$ for $\omega(\bar{e})$ where $\bar{e} \in E$ and $\omega(\bar{e}) \geq 0$, implying non-negative edge weights.*

Definition 2.2 (Path length)**.** *Suppose the graph G has n edges with vertices v_i, $i = 0, 1, 2, ..., n$. If a path $p = \langle v_0, v_1, v_2, ...v_n \rangle$ contains all vertices of the graph G, then the length of the path $len(p)$, is the sum of all the edge weights that constitutes the path.*

$$len(p) = \sum_{i=1}^{n} \omega(v_{i-1}, v_i).$$

Dijkstra's algorithm defines a set of rules to determine the shortest path from s (a unique source vertex) to every vertex $v \in V$. Being a single-sourced problem, s is in V but different from all $v \in V$ in $G(V, E)$. The link distance connecting any two vertices u, v say, denoted by $\delta(u, v)$ is the shortest path length if there exist a path between vertices u and v. If a vertex u say, can be found within the shortest path from the s to v , then we have;

$$\delta(s, u) \leq \delta(s, v). \tag{1}$$

$$\delta(s, v) \leq d[v], \tag{2}$$

where $d[v]$ is the length of an already known path. At the root vertex s, we have;

$$d(s, s) = d[s] = 0, \tag{3}$$

while all other vertices are given the value $d[v] = d(s, v) = \infty \ \forall v \in V$. The vertices are then processed one after the other by the algorithm. When a particular vertex v say, is being processed, it estimate is validated and the shortest distance $d[v] = \delta(s, v)$. New paths are found and $d[v]$ is updated $\forall v \in Adj[u]$ (where necessary) once the vertex u has been processed. An updated vertex is termed relaxed (as the process is called relaxation).The moment every vertex has been processed by the algorithm, we claim;

$$d[v] = \delta(s, v) \ \ \forall v. \tag{4}$$

To establish a new paths through a processed vertex u, all vertex v in outgoing adjacencies ($v \in Adj[u]$) are examined and a new path found from s from u. This new path is the sum of the path from s to u and the new edge. Rexalation thus entail updating $d[v]$ to the length of new path if that length (from s to v) is below $d[v]$. Thus, if

$$d[v] > d[u] + \omega(u, v),$$

$$\implies d(s, v) > d(s, u) + \omega(u, v),$$

we then relax the edge (u, v) by setting

$$d(s, v) = d(s, u) + \omega(u, v), \tag{5}$$

as the new estimate and the predecessor $pred[v] = u$ determines the shortest path.

Lemma 2.1 (Correctness of Dijkstra's algorithm)**.**

Dijkstra's algorithm ends with

$$d(s, u) = \delta(s, u) \ \ \forall u \in V. \tag{6}$$

Proof. To begin, Dijkstra's algorithm works by updating a subset $S \subseteq V$, initiated as empty (i.e. $S = \emptyset$) and the procedure gradually populates S by selecting vertices from $V \setminus S$ and adding to S. To see that $d[u] = \delta(s, u)$ at the end of the algorithm, it suffices to show that when u is added to S (or removed from $V \setminus S$: a changeable priority queue Q say, with $(u, d(s, u))$ for each vertex $u \in V$), $d[u] = \delta(s, u)$. We may thus proceed by induction on the first $n \in \mathbb{N}$ vertices removed from Q. The base case $k = 1$ clearly holds, as the root vertex s is the first removed from Q and $d(s, s) = d[s] = \delta(s, s) = 0$. Assuming it holds for a certain $k \in \mathbb{N}$, we proceed to check for $(k+1)^{th}$ vertex. Consider the k^*th vertex v^* say, removed from Q. Suppose there is some shortest path σ from the source to v^* having $\omega(\sigma) = \delta(s, v^*)$ and let (x, y) be the first edge in σ with the vertex y not part of the first k vertices (possibly $y = v^*$). When the vertex x is removed from Q, then $d(s, x) = \delta(s, x)$ by induction, thus;

$$\begin{aligned} d(s, y) &\leq \delta(s, x) + \omega(x, y) \\ &= \delta(s, y) \text{ subpaths of shortest paths} \\ &\leq \delta(s, v^*) \text{ non-negative weights} \\ &\leq d(s, v^*) \text{ relaxation} \\ &\leq d(s, y) \ v^* \text{ is minimum } d(s, v^*) \text{ in } Q \end{aligned}$$

$$\implies d(s, v^*) = \delta(s, v^*) \text{ as required, thus completing the proof.}$$

□

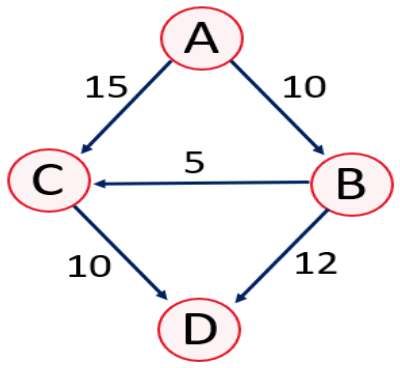

Figure 1: The weighted graph of connected cities.

2.2. Notions and steps in implementing Markov chains

A Markov chain is a stochastic model that describes a sequence of possible events where the probability of each event depends only on the current state (not the previous history).

Definition 2.3 (Markov Chains)**.** *Processes* $\{X_0, X_1, ...\}$ *where* X_t *is the state at time* t *such that* X_{t+1} *depends only on* X_t *and not on* $X_0, X_1, ...X_{t-1}$ *are called Markov chains.*

Definition 2.4 (Trajectory)**.** *Any particular set of values* $S = \{s_0, s_1, ...\}$ *mapped to* $\{X_0, X_1, ...\}$ *is a trajectory of the Markov chains.*

Apparently all Markov chains must satisfy the Markov property namely;

$$\mathbb{P}(X_{t+1} = s | X_t = s_t, X_{t-1} = s_{t-1}, ...X_0 = s_0) = \mathbb{P}(X_{t+1} = s | X_t = s_t) \; \forall t \in \mathbb{N}.$$

A special matrix used to describe Markov chains is known as the transition matrix $P = (P_{ij})$, where $(i, j)^{th}$ entry is the conditional probability that the system goes to j, given that it is currently in i (i.e. the probability of going from state i to state j). Thus, P contains all the conditional probabilities of the Markov chain.

$$P_{ij} = \mathbb{P}(X_{t+1} = j | X_t = i) \; \forall i, j \in S, t = 0, 1, 2, ...$$

The information captured in the transition matrix may also be represented in a transition diagram.

3. Results and discussion

We now take a closer look at how the two methods considered in this work are implemented for traffic distribution.

3.1. Dijkstra's algorithm for traffic distribution

Consider a network of roads connecting cities, where each road has a distance in kilometers, and we aim to find the shortest path from City A to City D. In Figure 1, nodes represent cities, and edges represent roads with weights (distances). To see how Dijkstra's algorithm obtains an optimal route;

Step 1: We begin at node A with a distance of 0. Other tentative distances are infinity (unknown for now).

Step 2: From A, the shortest known distances to B and C are $10km$ and $15km$, respectively.

Step 3: The node with the smallest distance (B) is chosen, and we check the routes from B. From B to D, the distance is $10km$ (from A to B) + $12km = 22km$. From B to C, the distance is 10 (from A to B) + $5km = 15km$. However, C already has a distance of 15, and so this doesn't change.

Step 4: Next, we visit C (tentative distance $15km$). From C to D, the distance is $15km + 10km = 25km$, which is greater than the current distance to D ($22km$), so we don't update.

Step 5: Finally, we visit D. The shortest path from A to D is $A \rightarrow B \rightarrow D$ with a distance of $22km$ as depicted in Figure 2. This represents an optimal route to distribute traffic based on the shortest path.

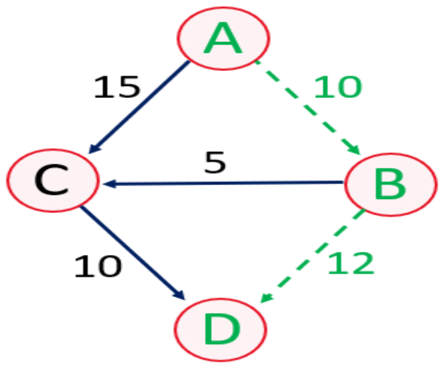

Figure 2: Green dashed path highlights the shortest path from city A to D.

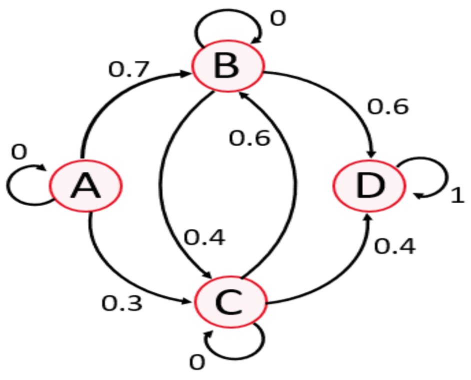

Figure 3: Transition diagram.

3.2. *Markov chains for traffic distribution*

Markov chains provide a way to model the probabilistic movement of traffic through a network, where the system transitions from one state (or intersection) to another based on a set of probabilities. Each intersection (or node) in the traffic network can be viewed as a state, and the roads between intersections represent the transition probabilities of vehicles moving from one intersection to another. In traffic distribution, Markov chains can model the probability of moving from one point in a network to another. One can also model the probabilities of vehicles moving between cities and determine where traffic will likely be distributed after many iterations (steady state). The following are basic steps for implementing Markov chains;

- Define system states (e.g., intersections or roads in a network)
- Construct a transition matrix, where each element P_{ij} represents the probability of moving from state i to state j
- Use the transition matrix to calculate the steady-state distribution, which gives the long-term probabilities of being at each state

Let's assume the following transition probabilities for vehicles moving between cities. From A, 70 percent of vehicles go to B and 30% go to C. From B, 60% of vehicles go to D and 40% go to C. From C, 40% of vehicles go to D and 60% go to B and from D, 100% of vehicles terminate here, the final destination (see Figure 3).

The transition matrix (P) for the Markov Chain looks like this:

$$P = \begin{pmatrix} 0 & 0.7 & 0.3 & 0 \\ 0 & 0 & 0.4 & 0.6 \\ 0 & 0.6 & 0 & 0.4 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{7}$$

where each row represents the probability of transitioning from city i to city j, with cities A, B, C, and D as states.

Let's say initially, we have 500 vehicles/hour starting at A. We want to determine how the traffic will distribute over time. From A, we have 70% of 500 vehicles implying 350 vehicles going to B. 30% of 500 vehicles, 150 vehicles go to C. From B 60% of 350

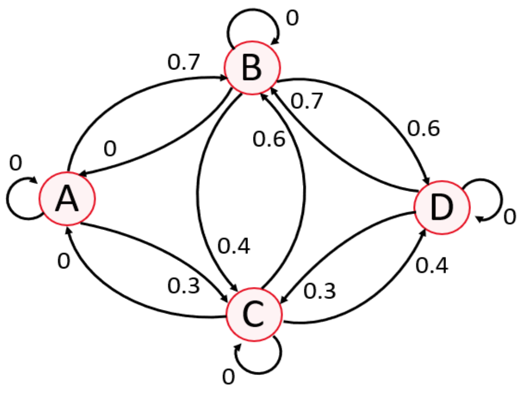

Figure 4: Transition diagram

vehicles, 210 vehicles go to D. 40% of 350 vehicles 140 vehicles go to C. From C 40% of 290 vehicles $(150 + 140)$ and 116 vehicles go to D. 60% of 290 vehicles 174 vehicles go to B (see Figure 4). The transition matrix is thus given as:

$$P = \begin{pmatrix} 0 & 0.7 & 0.3 & 0 \\ 0 & 0 & 0.4 & 0.6 \\ 0 & 0.6 & 0 & 0.4 \\ 0 & 0.7 & 0.3 & 0 \end{pmatrix}. \tag{8}$$

From

$$\pi = \pi P, \tag{9}$$

the steady state probabilities is $\pi_A \approx 0$, $\pi_B \approx 0$, $\pi_C \approx 0$ and $\pi_D \approx 1$. This result indicates that the system will always end up in state D. i.e. it is an absorbing state. All 500 vehicles terminates at node (city D). After multiple transitions (many iterations), the system reaches a steady-state distribution, where the total number of vehicles at each node stabilizes.

3.2.1. *Solving the steady-state probabilities*

Let π_A, π_B, π_C, and π_D be the long-term traffic distribution at cities A, B, C, and D respectively.

The steady-state solution satisfies $\pi P = \pi$, along with the normalization condition $\pi_A + \pi_B + \pi_C + \pi_D = 1$. Solving this system of equations, we get the steady-state probabilities (proportions of vehicles in each city over time). The solution gives: $\pi_A = 0$ (No vehicles stay in A), $\pi_B = 0.693$ ($\approx$ 69% of vehicles end up in B), $\pi_C = 0.071$ ($\approx$ 7% of vehicles end up in C) and $\pi_D = 0.236$ ($\approx$ 24% of vehicles reach D, the destination). If 500 vehicles/hour are initially entering the network at A, in the long run, the traffic will be distributed as: 0 vehicles/hour at A, 345 vehicles/hour at B, 35 vehicles/hour at C and 120 vehicles/hour at D.

3.3. *Combination of both Dijkstra and Markov*

An innovative coupling of both Dijkstra's algorithm for immediate route planning (shortest or most cost effective paths) and Markov chains to understand long-term traffic patterns and adapt the network (e.g., reroute some paths) may have significant impact on traffic flow and enhance optimal traffic distribution. This combination leads to a traffic management system that not only handles short-term demands but also accounts for long-term traffic flow stability. To explore the concept of optimal distribution of traffic using Dijkstra's algorithm and Markov chains, we consider a modification and expansion of the illustrative example earlier provided as follows;

Suppose there is a road network between four cities: A, B, C, and D. The goal is to distribute traffic (vehicles) optimally from City A to City D, using the shortest path. The number of vehicles traveling per hour is also provided. The road network is represented as a weighted graph, where the weights represent distances (in kilometers), and traffic conditions are such that more vehicles should be routed on the shortest, least congested path. Consider the network in Figure 5:

Step 1: Start at A with a distance of 0. The tentative distances to all other cities are infinity.

Step 2: From A, the shortest path to B is $10km$, and to $C$is $15km$. So, update the tentative distances to B (10) and C (15).

Step 3: Select the city with the smallest distance, which is B (distance = $10km$). From B, calculate the distance to D : $10km$ (A to B) + $12km$ (B to D) = $22km$. From B, calculate the distance to C : $10km$ (A to B) + $5km$ (B to C) = $15km$. This doesn't change the distance to C since it's already $15km$.

Step 4: Now, the tentative distance to D from A through B is $22km$, and to C, it's 15 km. Visit C next, but the total distance from A to D through C is $25km$ ($A \rightarrow C \rightarrow D$), which is longer than the $22km$ path through B.

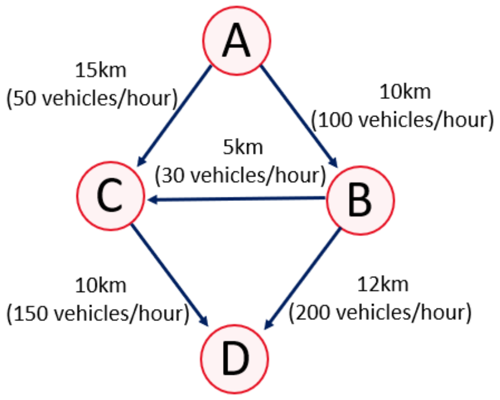

Figure 5: Transition diagram

Step 5: Finally, choose the shortest path from A to D, which is $A \rightarrow B \rightarrow D$ with a total distance of $22km$. Distribution of Vehicles Using Dijkstra's Path: Based on the shortest path ($A \rightarrow B \rightarrow D$), you can route 100 vehicles/hour from A to B, and then 200 vehicles/hour from B to D. This is the optimal route for minimizing travel distance, but there could be congestion if too many vehicles use this path. In our case, $A \rightarrow B \rightarrow D$ is the optimal path for the vehicles in the short term, handling 100 vehicles/hour from A to B and 200 vehicles/hour from B to D. Even if Dijkstra's shortest path is optimal at the moment, traffic may distribute itself differently over time, leading to a balanced distribution across all roads in the network. By considering both approaches, you can create a dynamic routing system where traffic is routed efficiently both in the short-term (using Dijkstra) and balanced over time to prevent congestion (using Markov chains).

4. Conclusion

In a nutshell, understanding the shortest paths and travel times helps in;

- Traffic Management; allocating resources like traffic signals and road maintenance based on traffic flow
- Route Planning; providing drivers with the most efficient routes to minimize travel time
- Infrastructure Development: identifying critical roads that require expansion or improvement to handle traffic load

Urban planners and traffic engineers can leverage Dijkstra's algorithm to optimize traffic distribution, significantly reduce congestion, and greatly enhance overall mobility within our cities. This powerful algorithm offers a systematic and highly efficient approach to identifying the shortest paths in complex traffic networks. By representing intersections as nodes and roads as edges with specific travel times, Dijkstra's algorithm not only facilitates optimal traffic flow but also actively improves urban mobility and alleviates traffic bottlenecks and congestion. Its scalability allows it to be effectively applied to larger and more intricate networks, making it an indispensable tool in the advancement of modern traffic management systems.

Data availability

The current study does not involve any accessible datasets, however any pseudocode used and/or analyzed for the algorithm implemented during the study are available from the corresponding author on request.

Acknowledgment

The authors are sincerely grateful to the anonymous reviewers for their meticulous review and valuable comments that helped improve this work.

References

[1] L. I. Igbinosun & V. Ezugwu, "Traffic flow analysis of some selected road network in Uyo metropolis ", Journal of the Nigerian Association of Mathematical Physics **39** (2017) 165. https://www.ajol.info/index.php/jonamp/article/view/206355.
[2] L. I. Igbinosun & O. Izevbizua, "Some control strategies for road traffic flow in Nigeria", International Journal of Statistics and Applied Mathematics **5** (2020) 56. https://tinyurl.com/5enye58a.

[3] H. A. Nkwocha & B. D. Ilozor, "Traffic congestion reduction strategies in urban areas: a review of the current state-of-the-art", Journal of Traffic and Logistics Engineering **7** (2019) 234. https://doi.org/10.18178/jtle.7.5.234-242.

[4] H. A. Seabe & S. M. Simelane, "Optimization of road traffic flow: a case study of the city of Tshwane", The Journal of Urbanism **12** (2019) 27. https://doi.org/10.1080/17549175.2018.1505294.

[5] J. Jallow, "Exploring the factors affecting the efficiency of the existing traffic management system in the greater Banjul area ", Journal of Traffic and Transportation Engineering **7** (2020) 86. https://doi.org/10.1016/j.jtte.2019.07.003.

[6] A. I. Mansour & H. A. Aljamil, "Investigating the effect of traffic flow on pollution, noise for urban road network", IOP Conference Series: Earth and Environmental Science **961** (2022) 012067. https://doi.org/10.1088/1755-1315/961/1/012067.

[7] M. Ali, G. Ahsan & A. Hossain, "Traffic congestion and physical health of commuters: perspective of Dhaka city", Journal of Contemporary Studies in Epidemiology and Public Health **2** (2021) ep21002. https://doi.org/10.30935/jconseph/9365.

[8] Y. S. Huang, Y. S. Weng, W. Wu & B. Y. Chen, "Control strategies for solving the problem of traffic congestion", IET Intelligent Transport Systems **10** (2016) 642. https://doi.org/10.1049/iet-its.2016.0003.

[9] L. I. Igbinosun & S. E. Omosigho, "Traffic flow model at fixed control signals with discrete service time distribution", Croatian Operational Research Review **19** (2016) 32. https://doi.org/10.1088/1755-1315/961/1/012067.

[10] B. Singh & A. Gupta, "Recent trends in intelligent transportation systems: a review", Journal of transport literature **9** (2015) 30. https://doi.org/10.1590/2238-1031.jtl.v9n2a6.

[11] National Transport Policy, Federal Republic of Nigeria, 2010. [Online]. https://www.transportation.gov.ng/ovadoc/national-transport-policy-ntp.

[12] J. Zhang, F. Wang, K. Wang, W. Lin, X. Xu & C. Chen, "Data-driven intelligent transportation systems: a survey", IEEE Transactions on Intelligent Transportation Systems **12** (2011) 1624. https://doi.org/10.1109/tits.2011.2158001.

[13] J. Mahona, C. Mhilu, J. Kihedu & H. Bwire, "Factors contributing to traffic flow congestion in heterogenous traffic conditions", International Journal for Traffic and Transport Engineering **9** (2019) 238. https://doi.org/10.7708/ijtte.2019.9(2).09.