Journal of Universal Computer Science, vol. 30, no. 4 (2024), 449-461
submitted: 30/8/2023, accepted: 16/11/2023, appeared: 28/4/2024 CC BY-ND 4.0

Reactive Traffic Congestion Control by Using a Hierarchical Graph

Sahar Idwan
(Department of Computer Science and Applications, The Hashemite University, Zarqa, Jordan
https://orcid.org/0000-0002-3602-4325, sahar@hu.edu.jo)

Junaid Ahmed Zubairi
(Department of Computer and Information Sciences, State University of New York at Fredonia, Fredonia, NY, USA
https://orcid.org/0000-0001-8588-1115, Junaid.Zubairi@fredonia.edu)

Syed Ali Haider
(Department of Computer and Information Sciences, State University of New York at Fredonia, Fredonia, NY, USA
https://orcid.org/0000-0003-3582-2971, haider@fredonia.edu)

Wael Etaiwi
(Princess Sumaya University for Technology, Amman, Jordan
https://orcid.org/0000-0003-0421-7971, w.etaiwi@psut.edu.jo)

Abstract: Traffic management is one of the major factors in growth strategy formulation in urban centers across the globe. The increasing population and, therefore, the increase in the number of vehicles on roads in urban centers cause congested traffic patterns. These patterns typically emerge on intersections in busy city roads at various times during the day, especially during peak hours. A direct consequence of congestion is the increase in commute time and pollution. This paper presents a hierarchical graph-based congestion control ($HGCC$) method. Congestion values are set and evaluated as a two-level hierarchical graph. The least congestion path algorithm (LCP) is integrated with the $HGCC$ to compute the optimal route between source and destination. The experimental results for a Manhattan-like grid network, together with the paired-sample t-test, show that the proposed method is efficient in achieving good congestion-avoiding routes.

Keywords: Traffic Management System,smart city, Traffic congestion, Hierarchical graph, Least congestion path
Categories: J.6, J.7
DOI: 10.3897/jucs.111879

1 Introduction

Road traffic congestion is considered to be a huge problem in major cities all around the world. Direct consequences of congestion are additional fuel consumption and increased travel times. Congestion in metropolitan areas can be classified into two groups, i.e., 1) recurring group and 2) non-recurring group. Typically, poorly designed infrastructure, population to infrastructure ratio, traffic light management, public works, and the weather could lead to huge traffic jams [Afrin and Yodo, 2020]. Various statistics show that the

annual cost of congestion runs about $160 billion primarily because of the extra fuel used by idling vehicles [Schrank et al., 2015]. This, in return, affects the productivity of the city and, eventually, the country. In the USA, New York City happens to be the most congested city, with an annual cost of congestion of about $11.0 billion [Schrank et al., 2015].

Traffic congestion is a universal problem that challenges researchers, transportation professionals, and policymakers. Route generation is one of the approaches that can be used to efficiently compute the optimal route between two nodes. Computing the optimal route depends on various factors such as segment length, speed limit, and number of cars, among others. These factors play a predominant role in initiating congestion on roads.

The congestion value (CV) for each road segment has been computed at a specific time as proposed in equation 1. It depends on the number of cars in the segment. The CV value is assigned to each road segment in the urban metropolitan to represent the traffic load.

$$CV = \frac{(n/l)}{\bar{v}/L} \tag{1}$$

where n is the number of cars, l is the segment length, $\bar{v}$ is the speed and L is the speed limit on the road segment.

The process of computing the route for the trip on a map with a large number of roads represented as a large flat graph is tedious and complicated. It is, therefore, imperative that a multi-level hierarchical graph is used instead to accelerate the computation process. A set of fragments represents intersections and road segments in such a graph.

The concept of the hierarchical graph has been used in various applications such as the pursuer-evader problem [Idwan, 2005]. The main aim of this paper is to design and develop $HGCC$, which assists the driver in choosing the route with the least congestion without any need to change the existing road infrastructure as far as possible. Moreover, a least congestion path (LCP) algorithm is proposed to detect congested vertices in the hierarchical graph and to generate an optimal route to minimize the possibility of drivers encountering congestion.

The main contributions of this paper are summarized as follows:

1. A novel Hierarchical Graph congestion control ($HGCC$) method based on partitioning a large flat graph into a set of fragments to control traffic congestion in the Manhattan-grid network;
2. The two-level graph is incorporated with the least congestion path (LCP) algorithm, which considers the road segments and the congestion value in the intersection vertices to determine the optimal route with the minimum computed congestion value (CV);
3. The proposed method is tested by running a set of experiments on Manhattan's actual map to emphasize the impact of applying the $HGCC$ on road real-world traffic.

The remainder of the paper is organized as follows: The methodology is presented in Section 3, while the experimental setup, results, and findings are discussed in Section 4. Finally, Section 5 draws the conclusion and future works.

2 Literature Review

Traffic congestion is a widespread problem in metropolitan cities where the number of vehicles is increasing more rapidly compared to the capacity of the road infrastructure

[Strickl et al., 1995]. Enhancing the traffic capacity in big cities to reduce traffic congestion is done using traffic management systems (TMS). A core component in a TMS is the information that is gathered from the traffic lights, sensors, and vehicles. This information can be used in various ways and shared among vehicles or via a traffic management center to identify and manage traffic risks more efficiently [Djahel et al., 2015]. Various studies [Djahel et al., 2015, de Souza et al., 2017] highlight the challenges and limitations in the current TMSs and suggest solutions to increase the proficiency of these systems by detecting or avoiding congestion. In this section, we review and summarize recent works for computing the optimal route to avoid congestion. Researchers have investigated the impact of various techniques that were deployed in vehicles using smartphones to establish the optimal route [Shang et al., 2013, Lee et al., 2014, Shang et al., 2015, Zubairi et al., 2022]. Also, avoiding traffic congestion by observing real-time traffic data in a large area to enhance road traffic streaming was discussed in [Costea et al., 2014, Omar, 2015]. Costea et. al. [Costea et al., 2014], used General Packet Radio Service (GPRS) technology with a large number of sensors to collect and manage traffic movement without any human involvement. While Omar [Omar, 2015] developed a multi-agent system by utilizing the Radio Frequency Identification (RFID) and Wireless sensor network (WSN) to collect the current traffic data to monitor traffic flows. Additionally, handling traffic information promptly and accurately is a major task in developing a smart traffic management system (STMS). STMS [Rizwan et al., 2016] is modeled by deploying sensors at strategic locations on roads to provide current traffic information or an alternate path to the drivers during rush hours or during accidents that lead to reduced traffic density. The proposed system is costly because of the heavy capital expenditures involved. Maniccam [Maniccam, 2006] presented an approach to avoid congestion in a two-dimensional space by designing an adaptive and decentralized routing algorithm combined with congestion-avoiding traffic rules to compute the best route that should be selected by the vehicle. The traffic congestion problem has been studied in [Kala and Warwick, 2014] by considering factors such as traffic density and traffic lights for a city transportation infrastructure. They proposed a search algorithm using A^* to compute the routing algorithm from the source node to the destination node to minimize congestion. Their algorithm considers the role of the traffic lights and the scenario where the vehicles were allowed to transcend in a single lane.

Multiple schemes of Vehicular Ad-hoc Networks (VANET) have been used to handle the congestion problem on roads. For instance, optimal path calculation by utilizing Vehicular Ad-hoc NETworks (VANETs) to collect real-time significant information related to the roads was proposed in [Toulni et al., 2014]. The analyzed information is represented in a graph, and the optimal route is estimated using the Dijkstra algorithm. In other works, the VANET model is used to detect and avoid congestion in a road network without relying on the information generated by the vehicles [El-Sayed et al., 2017]. Pan et al. determined the congestion level in each lane based on the lightweight infrastructure-based histogram model, which initiates a re-routing plan for two different kinds of congestion. Moreover, the re-routing approach by exchanging the vehicle's information via VANET has been investigated in [Pan et al., 2017]. The authors introduced a Distributed Vehicular Traffic Re-Routing (DIVERT) system, which enhances the rerouting process in many real-time scenarios. DIVERT minimizes the use of the CPU and the network on the server. However, the system may generate incorrect or obsolete re-routing information since each vehicle computes the alternate route locally [Pan et al., 2017]. Also, Shahi et al. [singh Shahi et al., 2022] focused on determining congestion area and generating a re-routed path by using the MGRM mechanism, which depends on the vehicle congestion density function and path weight calculation. Utilizing

the MGRM enables the vehicle to choose the optimal route with the shortest driving time. Since the MGRM mechanism is based on the calculated vehicle congestion index, there is a possibility that this information may be useful for adjacent vehicles. Based on the state-of-the-art, it is shown that computing the optimal route has been implemented in large areas with complex infrastructure, which may not be suitable for resolving the congestion on the city grid. A hierarchical route-planning algorithm that specifically addresses all possible paths based on the needs of wheelchair users was introduced in [Yang and Mackworth, 2007]. A coarse path was computed on a higher level without considering the landscape variations. A threshold value was utilized in the algorithm to eliminate the unnecessary paths. The complexity and the running time of the algorithm were not based on real-time data or an actual road map. By substituting shortcut edges for unnecessary nodes, the contraction hierarchy (CH) technique [Geisberger et al., 2012] is employed to compress the road network. The proposed method can be used with mobile applications with limited resources and storage. The primary objective of the proposed approach is to reduce the computation time and the space needed to find the shortest path. Since the cost of each road in the proposed model corresponds to its length, network modifications have either involved adding new roads, removing existing ones, or altering the weight of the current roads. Compression of road networks may result in losing valuable information, which could help find better paths. Another approach to improve the effectiveness of route planning by employing the edge hierarchies was presented in [Hespe and Sanders, 2019]. They introduced a rank value, which is assigned to each edge in order to build a hierarchy of edges. A sequence of relax and stall operations has been implemented to generate the optimal path route. The computation process of the shortest path between two nodes is based only on the distance value between them; therefore, it does not consider congestion resolution, which is the primary goal of our work.

2.1 Distinction

Graphs are commonly used to simplify traffic management systems due to their ability to represent and formalize huge and complex data structures in a standard and formal way [de Souza et al., 2017]. Within the last decades, most of the proposed graph-based traffic management systems have been focused on finding the optimal route path in a large area by considering one or two factors of the road network such as distance, number of lanes, type of the street, traffic lights and others [Wilson and Boateng, 2018] and [Falek et al., 2021]. Also, those methods do not consider the congestion value and the road characteristics (i.e., number of lanes, maximum speed, total number of cars) together while computing the best path. Additionally, the graph-based shortest path algorithms in contributions compare the proposed approaches based on network setup, computational time, and storage requirements rather than studying the congestion, the properties of the roads, and the status of the roads' usage. In this work, a hierarchical graph-based model is proposed to find the lowest congestion path in road networks. The proposed method considers road distance, characteristics, and status to find the shortest path with the least congestion, as well as the computational time and the required storage. It is important to note that the proposed model utilizes Dijkstra's shortest path algorithm to find the lowest congestion path based on several factors such as road length, road speed limit, number of cars on the road, and the average car speed. The proposed hierarchical graph is built based on the node level, unlike the state-of-the-art technology, which is based on building extra short-cut edges. Moreover, a structured process of building the proposed hierarchical graph is applied to the original graph rather than relying on random decisions.

3 Methodology

The structure of $HGCC$ is presented in Figure 1. It contains two steps 1) Building the weighted hierarchical graph and 2) developing the least congestion path algorithm, which computes the optimal route between two specified vertices with the minimum congestion value. We elaborate on these two steps below.

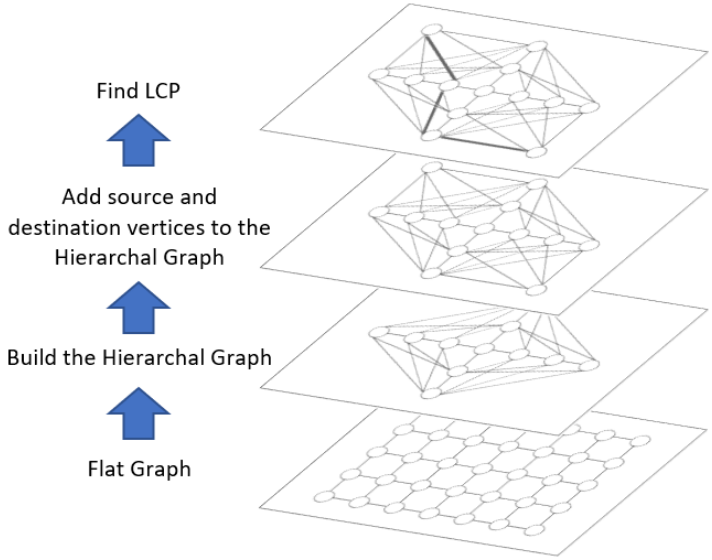

Figure 1: HGCC structure

3.1 Hierarchical Graph

The creation of the hierarchical graph involves pre-processing and then construction. In the first step, the original graph is divided into sub-graphs called fragments. These fragments are used in the second step to build a two-level graph – hierarchical graph.

The urban metropolitan area is represented by a weighted directed graph $G = (V, E, w)$, where V is the set of n vertices, E is the set of e edges, and $w(u, v) = CV$. Congestion value CV from equation 1 is computed and used to represent the weight of edges in the flat graph.

The weighted directed hierarchical graph is defined in definition 1.

Definition 1 *HierG=(HierV, HierE, Hierw),* $HierV = \{Hn_1, Hn_2, ...Hn_k\}$ *, where* $|HierV| = H_k$. $HierE = \{He_1, He_2, ...He_t\}$ *, where* $|HierE| = H_t$ *and Hierw: weight of the hierarchical edges.*

For pre-processing, we use a spatial partitioning technique to group all nodes close to each other in one fragment. Each fragment contains an equal number of vertices. Some of these nodes, primarily the boundary nodes, are “moved up” to the top level to form

the vertices $HierV$ in the hierarchical graph ($HierG$), where $H_k < n$. Congestion value (C) is assigned for each $v \in HierV$ as given in equation 2. This is to avoid the congested intersection nodes. Two fragments are considered adjacent if they have at least one common boundary hierarchical node. For example, given a representation of the Manhattan map illustrated in Figure 2, we divided the graph into four fragments as shown in Figure 3. The graph shows that F1 and F2 are adjacent fragments.

$$C(v) = C_{in\_degree}(v) + C_{out\_degree}(v) \tag{2}$$

Where,

$$C_{in\_degree}(v) = \sum_{(u,v)\in E} w(u,v) \tag{3}$$

Also,

$$C_{out\_degree}(v) = \sum_{(v,u)\in E} w(v,u) \tag{4}$$

Next, we build the hierarchical graph $HierG$ as given in definition 1. The hierarchical vertices are defined during pre-processing. There are two kinds of edges in the graph 1) local and 2) hierarchical. An edge is identified as a local edge when the two end vertices of the edge belong to the same fragment. The hierarchical edge connects the hierarchical nodes together. The shortest distance between a pair of hierarchical nodes H_u and H_v is computed and is assigned as a weight for the hierarchical edge.

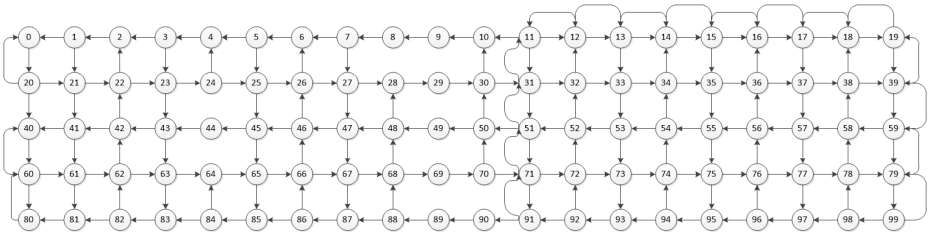

Figure 2: A representation of a sub-section of Manhattan map

The algorithm for building $HierG$ is given in Algorithm 1. Given that the original graph is directed and weighted, we start by inserting the number of fragments in the graph as listed in *[Line 1]*. Pre-processing is done in *[Line 2 and Line 3]*. It partitions the graph into a set of fragments and generates hierarchical nodes. Later, for each fragment, we determine a list of hierarchical nodes *[Lines 5 and 6]*. These hierarchical nodes are considered boundary nodes of this fragment. Our objective is to compute the weight of hierarchical edges $HierE$ by computing the shortest distance between a pair of hierarchical nodes u and v in the flat graph *[Line 7 and Line 8]*. Figure 4 demonstrates the hierarchical graph derived from the initial Manhattan graph depicted in Figure 2. The time complexity of building the hierarchical graph is $O(N_{frag} * |H_k|^2)$. In addition to this, the complexity of the Dijkstra algorithm is $O(|HierE| \times log(|HierV|)$.

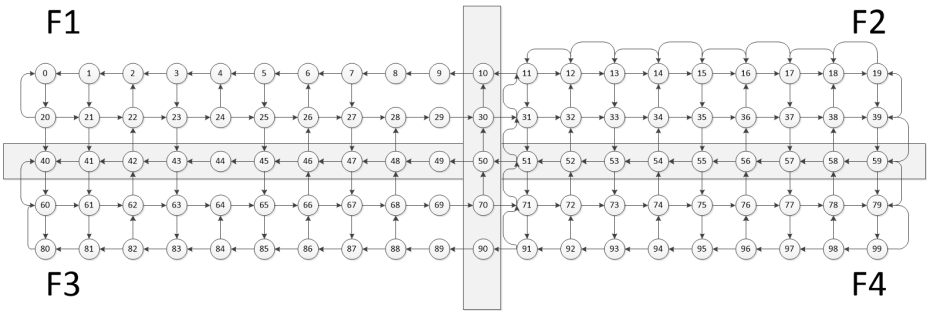

Figure 3: Manhattan map fragmentation. Highlighted vertices 10,30,90,... represent a set of hierarchical vertices (HierV)

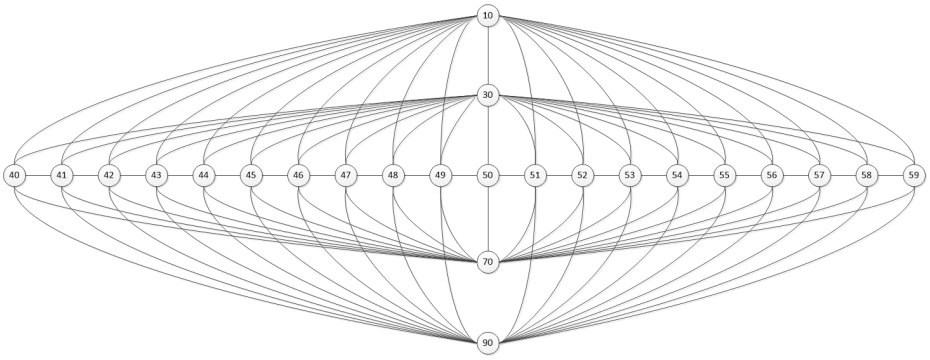

Figure 4: Derived Manhattan hierarchical graph on 2×2 *grid*

3.2 Computing the Least Congestion Path (LCP)

In this section, we explain how the least congested path from the source vertex to the destination vertex is computed. The process is shown in Algorithm 2. Our approach is rooted in the premise that the source and destination vertices are located at a significant distance from each other. We compute the least congested path (LCP) that avoids congested intersection nodes. Therefore, it is unnecessary to compute the LCP if the source vertex and the destination vertex are in the same fragment since they are close to each other. We identify the fragment number for the source and destination vertices *[Line 3 and 4]*. If the source vertex and the destination vertex are in different fragments, LCP between them is computed. LCP is computed by using $HierG$ if both source and destination are $HierV$ nodes. Alternatively, edges from the source and destination vertices to $HierV$ have to be added in $HierG$.

An edge is added from the source vertex s to each hierarchical vertex u that belongs to a fragment F_s. It is worth mentioning that $HierV(F_s)$ represents the list of hierarchical vertices (boundary nodes) that belong to the fragment that contains the source vertex as presented in *[Line 6]*. The weight of the edge is computed by using the length of the shortest path between the source vertex and the hierarchical vertex in the flat graph and the congestion value (C) of all hierarchical nodes in this shortest path as shown in *[Line 8*]. Similarly, an edge is added from each hierarchical vertex v that belongs to the fragment F_d to the destination vertex d. The weight of the edge is computed by using the length

Algorithm 1 Building Hierarchical Graph(G)

Input $G = (V, E)$
Output $HierG = (HierNodes, HierEdges, fragmentList)$

```
1: NoofFragments = Input(value)
2: fragmentList = SpatialPartitioning(G)
3: HierNodes = FormTopLevelVertices(V)
4: for each FragmentF ∈ fragmentList do
5:     for each u ∈ HierV(F) do
6:         for each v ∈ HierV(F) do
7:             Add(u, v) to HierG
8:             w(u, v) = ShortestPath(HierG, u, v)
9:         end for
10:    end for
11: end for
```

of the shortest path in the flat graph between the hierarchical vertex and the destination vertex and the congestion value (C) of all hierarchical nodes in this shortest path as presented *[Line 12]*. Finally, LCP is computed between source vertex s and destination vertex d in the $HierG$ *[Line 15]*. The time complexity of LCP is $O(|H_k|)$ in addition to the complexity of the Dijkstra algorithm, which is $O(|HierE(F)| \times log(|HierV(F)|)$. Figure 5 illustrates the graph.

Algorithm 2 Compute the Least Congestion Path LCP
LCP(HierNodes,HierEdges,fragmentList,s,d)

Require: s : *sourceNode*; d : *destinationNode*;

```
1: Add source vertex (s) to HierG
2: Add destination vertex (d) to HierG
3: F_s = DetermineFragementNo(s)
4: F_d = DetermineFragementNo(d)
5: if F_s ≠ F_d then
6:     for each u ∈ HierV(F_s) do
7:         Add(s, u) to HierG
8:         w(s, u) = ShortestPath(G, s, u) + C(u)
9:     end for
10:    for each v ∈ HierV(F_d) do
11:        Add(v, d) to HierG
12:        w(v, d) = ShortestPath(G, v, d) + C(v)
13:    end for
14: end if
15: return ShortestPath(HierG, s, d)
```

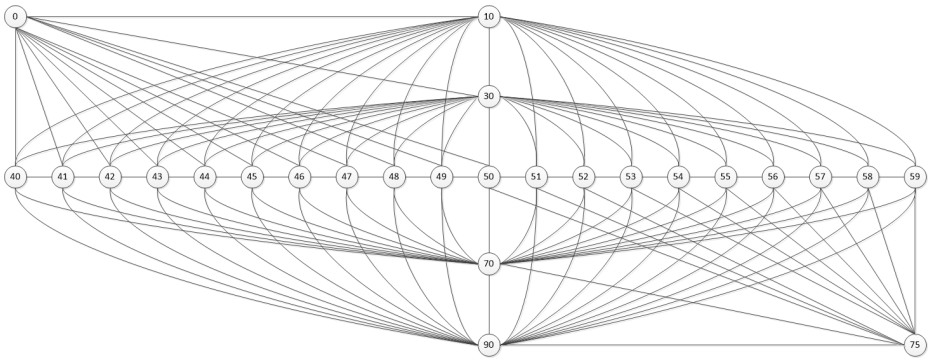

Figure 5: Manhattan hierarchical graph to compute LCP from vertex 0 to vertex 75

Stage	4 Fragments		8 Fragments	
	Vertices	Edges	Vertices	Edges
Flat graph	100	186	100	186
$HierG$	24	95	32	116
$HierG$ with source and destination	26	121	34	129

Table 1: The total number of vertices and edges in each stage of HGCC mentioned in figure 1

4 Experimental Results

In this section, we test the solution approach described in Section 3. We run a set of experiments on a PC with a dual-core 2.59 GHz CPU, 4 Logical processors, and 16 GB of RAM on a 64-bit Windows operating system. The whole structure is implemented in C language [Ritchie, 1993] to study the performance of the $HGCC$ method. Two data files are used to test the proposed $HGCC$ method. In both files, directed road segments are modeled on the western portion of midtown Manhattan, from 11^{th} Avenue to 7^{th} Avenue, between 23^{rd} Street and 42^{nd} Street (a 5×20 grid with a total of 100 intersections). The first file presents the lengths of the road segments that were randomly generated between 0.65 miles and 0.75 miles. The second file presents the actual road segment lengths of Manhattan streets and avenues. The actual distance of road segments in Manhattan varies slightly from block to block. The total number of vertices and edges in each stage of the $HGCC$ is shown in Table 1, which decreases in each ongoing step as illustrated in Figure 1. We list below examples 1, 2 and 3 with grid size 2×2 and 4×2 for computing the least congestion path between two vertices in $HierG$. The experimental results of hierarchical and flat graphs are listed in Tables 2 and 3.

Example 1. Given Manhattan map illustrated in Figure 2, and the spatial partitioning illustrated in Figure 3. The $HierG$ is illustrated in Figure 4, and the $HierG$ to find the LCP from vertex 0 to vertex 75 is illustrated in Figure 5.

Example 2. Given the $HierG$ illustrated in Figure 4, the LCP from source vertex number 18 to several selected other destination vertices is listed in table 2. The table lists

two values for each case: (1) The $CV(LCP)$ in a flat graph. (2) The $CV(LCP)$ using $HGCC$.

Example 3. Given the Manhattan map illustrated in Figure 2, its $HierG$, after partitioning it into a grid of size 4×2, is illustrated in Figure 6. The LCP from source vertex number 18 to several selected other destination vertices is listed in table 3. The table lists two values for each case: (1) The $CV(LCP)$ in a flat graph. (2) The $CV(LCP)$ using $HGCC$.

By running a set of experiments for a 2×2 grid on the Manhattan area road network as shown in Table 2, the LCP determined by the $HGCC$ is less (and often significantly less) than the LCP determined by the flat graph. It is noted that in some experiments, the LCP found on a flat graph is equal to the LCP found by using $HGCC$ because the source and destination nodes are in the same fragment. The results of running a set of experiments for 4×2 grid on the Manhattan area road network as presented in Table 3 achieve the same conclusion. Moreover, paired-sample t-tests [Helmert, 1876] were conducted to compare the means between the $CV(LCP)$ in the Flat graph and the $CV(LCP)$ in $HGCC$. A t-test is a statistical test that is employed to compare the means of two groups of experiments. It is commonly utilized in hypothesis testing to discover whether two groups differ from one another. The two-tailed P value is less than 0.0001. By conventional criteria, this difference is considered to be extremely statistically significant, with $t = 13.8207$ for 8 fragments and $t = 9.2836$ for 4 fragments. This concludes that the proposed $HGCC$ provides the LCP efficiently. It is worth mentioning that any changes in the congestion values will be reflected in the $HGCC$ method by rebuilding the $HierG$ and recomputing the new LCP at that time. However, detecting the changes in the congestion values on all roads is a heavy and time-consuming process.

Source vertex	Source's Fragment Number	Destination vertex	Destination's Fragment Number	$CV(LCP)$ in Flat graph	$CV(LCP)$ in $HGCC$
18	2	2	1	0.825617	0.346062
18	2	20	1	0.848463	0.368908
18	2	26	1	0.401247	0.255395
18	2	12	2	0.166637	0.166637
18	2	32	2	0.199721	0.199721
18	2	37	2	0.076116	0.076116
18	2	61	3	0.701952	0.318818
18	2	67	3	0.373778	0.274302
18	2	85	3	0.466567	0.269909
18	2	71	4	0.351527	0.238785
18	2	93	4	0.257495	0.154930
18	2	98	4	0.206559	0.138878

Table 2: The LCPs in Manhattan map from vertex number 18 to a number of selected vertices on 2×2 grid with four fragments

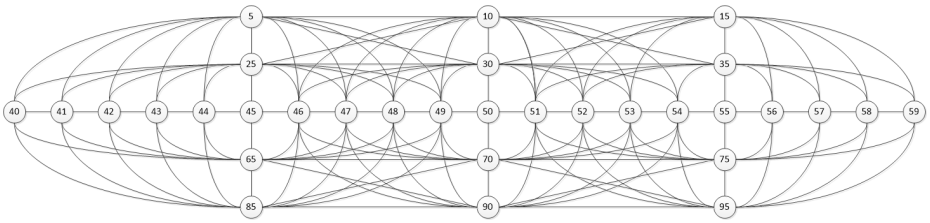

Figure 6: Manhattan hierarchical graph on 4×2 *grid with eight fragments*

Source vertex	Source's Fragment Number	Destination vertex	Destination's Fragment Number	$CV(LCP)$ in Flat graph	$CV(LCP)$ in $HGCC$
18	4	2	1	0.904259	0.526890
18	4	20	1	0.927105	0.549736
18	4	26	2	0.479890	0.255395
18	4	12	3	0.245280	0.166637
18	4	32	3	0.278363	0.199721
18	4	37	4	0.076116	0.076116
18	4	61	5	0.780594	0.414913
18	4	67	6	0.452421	0.274302
18	4	85	5	0.604281	0.269909
18	4	71	7	0.430169	0.238785
18	4	93	7	0.336138	0.168437
18	4	98	8	0.206559	0.138878

Table 3: The $LCPs$ *in Manhattan map from vertex number 18 to a number of selected vertices on* 4×2 *grid with eight fragments*

5 Conclusion

In this paper, we present a Hierarchical Graph Congestion Control ($HGCC$) method for implementing reactive congestion control in densely populated urban areas. A two-level hierarchical graph is generated by splitting a flat grid graph (Manhattan-grid) into fragments. The Least Congestion Path (LCP) algorithm is developed and implemented in the hierarchical graph. Several experiments have been conducted, and the results show the efficient computation of the least congested routes for commuters who can use this information to avoid congestion on the roads. We anticipate two avenues for future work. The first avenue is to generate an n-level hierarchical graph to reduce the working area, further reducing the compute time. The second involves utilizing other algorithms to compute the least congestion path, such as the A* algorithm. Moreover, a more general evaluation of arbitrary networks still remains open and might be a topic of future work.

Declarations

Declaration of Competing Interest None.

Availability of data and materials A link to the source code is available on request from the corresponding author.

References

[Afrin and Yodo, 2020] Afrin, T. and Yodo, N. (2020). A survey of road traffic congestion measures towards a sustainable and resilient transportation system. *Sustainability*, 12(11):4660.

[Costea et al., 2014] Costea, I. M., Nemtanu, F. C., Dumitrescu, C., Banu, C. V., and Banu, G. S. (2014). Monitoring system with applications in road transport. In *2014 IEEE 20th International Symposium for Design and Technology in Electronic Packaging (SIITME)*, Bucharest, Romania. IEEE.

[de Souza et al., 2017] de Souza, A. M., Brennand, C. A., Yokoyama, R. S., Donato, E. A., Madeira, E. R., and Villas, L. A. (2017). Traffic management systems: A classification, review, challenges, and future perspectives. *International Journal of Distributed Sensor Networks*, 13(4):155014771668361.

[Djahel et al., 2015] Djahel, S., Doolan, R., Muntean, G.-M., and Murphy, J. (2015). A communications-oriented perspective on traffic management systems for smart cities: Challenges and innovative approaches. *IEEE Communications Surveys & Tutorials*, 17(1):125–151.

[El-Sayed et al., 2017] El-Sayed, H., Thandavarayan, G., Sankar, S., and Mahmood, I. (2017). An infrastructure based congestion detection and avoidance scheme for VANETs. In *13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 1035–1040, Valencia, Spain. IEEE.

[Falek et al., 2021] Falek, A. M., Gallais, A., Pelsser, C., Julien, S., and Theoleyre, F. (2021). To re-route, or not to re-route: Impact of real-time re-routing in urban road networks. *Journal of Intelligent Transportation Systems*, 26(2):198–212.

[Geisberger et al., 2012] Geisberger, R., Sanders, P., Schultes, D., and Vetter, C. (2012). Exact routing in large road networks using contraction hierarchies. *Transportation Science*, 46(3):388–404.

[Helmert, 1876] Helmert (1876). Die genauigkeit der formel von peters zur berechnung des wahrscheinlichen beobachtungsfehlers directer beobachtungen gleicher genauigkeit. *Astronomische Nachrichten*, 88(8-9):113–131.

[Hespe and Sanders, 2019] Hespe, D. and Sanders, P. (2019). More Hierarchy in Route Planning Using Edge Hierarchies. In Cacchiani, V. and Marchetti-Spaccamela, A., editors, *19th Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2019)*, volume 75 of *Open Access Series in Informatics (OASIcs)*, pages 10:1–10:14, Dagstuhl, Germany. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[Idwan, 2005] Idwan, S. (2005). *Algorithms for discrete geometric and graph theoretic pursuit problems*. Phd thesis, Colorado School of Mines, Golden, Co. Available at https://repository.mines.edu/bitstream/handle/11124/177734/Idwan_10797097.pdf?sequence=1.

[Kala and Warwick, 2014] Kala, R. and Warwick, K. (2014). Congestion avoidance in city traffic. *Journal of Advanced Transportation*, 49(4):581–595.

[Lee et al., 2014] Lee, S., Tewolde, G., and Kwon, J. (2014). Design and implementation of vehicle tracking system using GPS/GSM/GPRS technology and smartphone application. In *2014 IEEE World Forum on Internet of Things (WF-IoT)*, Seoul, Korea. IEEE.

[Maniccam, 2006] Maniccam, S. (2006). Adaptive decentralized congestion avoidance in two-dimensional traffic. *Physica A: Statistical Mechanics and its Applications*, 363(2):512–526.

[Omar, 2015] Omar, H. (2015). Intelligent traffic information system based on integration of internet of things and agent technology. *International Journal of Advanced Computer Science and Applications*, 6(2).

[Pan et al., 2017] Pan, J., Popa, I. S., and Borcea, C. (2017). DIVERT: A distributed vehicular traffic re-routing system for congestion avoidance. *IEEE Transactions on Mobile Computing*, 16(1):58–72.

[Ritchie, 1993] Ritchie, D. M. (1993). The development of the c language. *ACM SIGPLAN Notices*, 28(3):201–208.

[Rizwan et al., 2016] Rizwan, P., Suresh, K., and Babu, M. R. (2016). Real-time smart traffic management system for smart cities by using internet of things and big data. In *2016 International Conference on Emerging Technological Trends (ICETT)*, Kollam, India. IEEE.

[Schrank et al., 2015] Schrank, D., Eisele, B., Lomax, T., and Bak, J. (2015). 2015 urban mobility scorecard. https://trid.trb.org/view/1367337.

[Shang et al., 2015] Shang, S., Liu, J., Zheng, K., Lu, H., Pedersen, T. B., and Wen, J.-R. (2015). Planning unobstructed paths in traffic-aware spatial networks. *GeoInformatica*, 19(4):723–746.

[Shang et al., 2013] Shang, S., Lu, H., Pedersen, T. B., and Xie, X. (2013). Finding traffic-aware fastest paths in spatial networks. In *Advances in Spatial and Temporal Databases*, pages 128–145. Springer Berlin Heidelberg, Berlin, Heidelberg.

[singh Shahi et al., 2022] singh Shahi, G., Batth, R. S., and Egerton, S. (2022). MRGM: An adaptive mechanism for congestion control in smart vehicular network. *International Journal of Communication Networks and Information Security (IJCNIS)*, 12(2).

[Strickl et al., 1995] Strickl, S. G., , and Wayne, B. (1995). Congestion control and demand management.

[Toulni et al., 2014] Toulni, H., Nsiri, B., Boulmalf, M., Bakhouya, M., and Sadiki, T. (2014). An approach to avoid traffic congestion using VANET. In *2014 International Conference on Next Generation Networks and Services (NGNS)*, pages 154–159, Casablanca, Morocco. IEEE.

[Wilson and Boateng, 2018] Wilson, M. and Boateng, K. O. (2018). Congestion-aware routing (CAR):vehicular traffic routing based on real-time road occupancy estimates. *Ghana Journal of Science*, 59(0):5.

[Yang and Mackworth, 2007] Yang, S. and Mackworth, A. K. (2007). Hierarchical shortest pathfinding applied to route-planning for wheelchair users. In *Advances in Artificial Intelligence*, pages 539–550. Springer Berlin Heidelberg.

[Zubairi et al., 2022] Zubairi, J. A., Idwan, S., Haider, S. A., and Hurtgen, D. (2022). Smart city traffic management for reducing congestion. In *2022 IEEE 19th International Conference on Smart Communities: Improving Quality of Life Using ICT, IoT and AI (HONET)*, pages 225–230.