International Journal of Computer Science and Applications,
©Technomathematics Research Foundation
Vol. 11, No. 2, pp. 73 – 92, 2014

SPATIAL TEMPORAL DATABASE MODEL FOR DETECTION AND CIRCUMVENTION OF TRAFFIC CONGESTION IN URBAN TRANSPORTATION NETWORK: GAME THEORETIC APPROACH

SEEMA PUROHIT

Dept of Mathematics, University of Mumbai, Kirti M. Doongursee College and NMITD
Mumbai, Maharashtra, 400028, India
supurohit@gmail.com
http://nmitd.net.in/

SHRUTI MANTRI

Department of Computer Science, University of Mumbai, UDCS and Kirti M. Doongursee College,
Mumbai, Maharashtra 400028, India
shrutimantri@gmail.com
http: //www.kirticollege.org

The urban transportation network is made up of vehicles in motion, dynamic static and moving obstacles. These obstacles block the path of the moving vehicles thereby causing traffic jam. Thus it is important to model the network explicitly and to describe the movement of the vehicle and obstacles relative to the transportation network. In this study the researchers have addressed the : (i) problem of the transportation network as a non-deterministic, non-cooperative n – person game and (ii) the collision free geometric path is computed by using a 2 player non-deterministic game which detects the dynamic obstacles in the transportation network and computes all the possible geometric paths and selects the optimal path for the vehicle to travel in a 2-D and 3-D workspace (iii)To resolve the traffic congestion at a road merge, cooperative game theory model is designed and developed using the concepts of Time-Divisions Multiple Access(TDMA) data slot that propagates through the a transportation channel. The 2-D spatial-temporal database model is implemented using game theoretic approach for facilitating disaster management response activity. User defined data types are designed for representing the static and moving network positions of the vehicles and the obstacles. In the transportation network entities such as hospitals, emergency vehicles, and obstacles corresponds to these data types. The roads in the network are also identified as simple or dual and bidirectional or unidirectional routes. These user defined data types are created using an object oriented programming language such as JAVA and then incorporated in DBMS.

Keywords: Data type; obstacle; network; game theory; spatial-temporal database; ROSE algebra

1. Introduction

1.1. Urban transportation network

A transportation network is made up of moving vehicles navigating from origin to destination to complete a trip. But in a dynamic environment the moving vehicles come across different types of obstacles. In the current research the transportation network is classified into two states of transition: (i) Stationary State and (ii) Moving State. Stationary State is a state of traffic, when the velocity of the each vehicle in the transportation network is V=0, i.e. a state when traffic congestion has occurred. Moving state is a state when the vehicle is in motion with a velocity V where $0 < V \leq$ Max Speed. As the vehicle traverses its path in a network it may have to change its trajectory to avoid known and unknown obstacles in its path way. In the current study, the obstacles in a transportation network can be classified as dynamic obstacles and static obstacles: (i) The physical structure that do not change such as public and private building, lakes, roads, water logging are all static obstacles (ii) The pedestrians, vehicle on roads where each vehicle driver trying to obstruct the path of the other vehicle driver are all dynamic moving obstacles. The static and dynamic obstacles with unknown distribution blocking the path of the moving vehicle result in traffic congestion. During a disaster it is very important for the emergency officers to reach to the disaster site as fast as possible to reduce the impact of the disaster thereby facilitating the disaster management activities.

1.2. Spatial database system

Spatial Database System is a database system that offers spatial data types in its data model and query language. It supports spatial data types in its implementation, providing spatial indexing and efficient algorithms for spatial queries. Spatial Database systems represent real world features such as city, river, and route. These features are represented by points, lines and regions in the spatial database system. Two different geometric approaches can be used to define the real world features: (i) Simplicial complexes (ii) Realm. In the current study the researchers have used realm geometric types to define the dynamic features of the real in a Spatial Database System

1.3. ROSE (Robust Spatial Extension) algebra

ROSE Algebra is a system of realm-based spatial data types made up of objects composed from realm elements. Realm is finite set of points and line segments defined over a grid such that:(i) Each point or end point of a segment is a grid point, (ii) Each end point of a segment is also a point of the realm, (iii) No realm point lies within a segment; (iv)Any two distinct segments do neither intersect nor overlap. It represents spatial types of the form points, lines and regions. ROSE Algebra also define complex structure of objects such as area inside another area, edge inside an area, vertex inside a region, disjoint areas, disjoint edges, disjoint vertex. ROSE algebra also offers operations for manipulating points, lines and regions. Realms data types are enumerations in C++ and operations are user-defined functions. These data types and operations are incorporated in SECONDO (Database Systems for moving objects) [Gütting *et al* (2006)]

1.4. Geometric path

Geometric Path is a path generated by using geometrical methods used for computing a collision or obstacle free path connecting the initial and final point in 2D and 3D environments [Bonchi *et al* (2009)]. Geometrical methods give the necessary geometrical descriptions for the physical objects in space such as obstacles, path followed by the obstacles. Using the geometrical descriptions it is possible to describe the obstacles in terms of arbitrary polygons and motion angles. This representation forms the base for Realm based ROSE Algebra.

1.5. Game theory for urban transportation

Game theory is the formal study of conflict and cooperation. Game theoretic concepts apply whenever the actions of several players are interdependent. In an urban transportation network private vehicle drivers are assumed to act selfishly and spontaneously to minimize their own travel cost/time. While doing this they act as an obstacle for the emergency vehicle drivers, thereby blocking the path of the emergency vehicle and delaying the disaster response activity. In the current research, researchers have attempted to study the pattern and behavior of the vehicles drivers in a transportation network to detect the traffic congestion and resolve the traffic congestion to speed up the time taken by the emergency vehicles to reach to the disaster site.

1.6. Obstacles and classification of obstacles

Obstacle in geometrical path refers to obstruction or blockage that arises in finding or traversing the path. Natural and created objects in geometric path problems are usually divided into dynamic and fixed objects called as obstacles. In many applications, physical obstacles like mountains, rivers, water logging, disaster sites etc. substantially affect the network analysis [Ziyad (2009)]. The obstacles in the current study can be classified as natural obstacles and created obstacles. Natural obstacles are those obstacles that cannot be moved and altered such as rocks, sea, stream, ledges and mountains. Created obstacles are those obstacles that are formed mainly due to environmental and geographic changes such as water logging formed due to improper land elevation. Pedestrian movement, public buildings, indoor areas, vehicles navigating through narrow lanes restricting the traffic and disaster affected sites can also be classified as created obstacles

2. Literature Review

Detailed study and analysis of the existing work was done in the current study is given as below:

Work Done	Approach/Methodology	Visibility Graph	Scope	Limitations
Cod-Clarans (2001)	Clusters objects into the same group with respect to the obstructed distance using a visibility graph	Visibility graph is pre-computed and materialized	The materialization and pre-computation of entire visibility graph is not feasible for large datasets as when new	1. The algorithm pre computes the graph and stores in the memory thereby

			obstacle are detected the entire visibility graph needs to be reconstructed	occupying memory space 2. As new obstacle is detected visibility graph needs to recomputed
Papadias *et al* (2003)	Spatial network made up of obstacles where obstacles are represented as areas where movement is prohibited, edges in spatial network denote the paths without obstacles.	The network is modeled as a graph and is stored as adjacency lists in the memory.	The network distance of two points is defined as the distance of the shortest path connecting them in a graph.	1. Only static obstacles are considered 2. Spatial queries only for static obstacles
De Berg *et al* (1997)	Path problem is treated as a computational geometry problem. Given a set O of non-overlapping obstacles in a 2D space, a starting point p_{start} and a destination point p_{end}, the goal is to find the shortest path from p_{start} to p_{end} which does not cross the interior of any obstacles O.	The network is modeled as a graph G and is stored as adjacency lists in the memory	Algorithm computes G, where G is made up of nodes that are vertices of all obstacles in O together with p_{start} and p_{end}	Shortest path is computed considering only static and predefined obstacles
[Lozano-Perez and Welsey (1979)	Computes the visibility graph by considering every possible node in G and check if line segment	Visibility Graph is pre stored in memory	The nodes of the graph G and static obstacles are pre stored in the memory	Graph G is pre computed and stored in memory along with

	connecting them intersects the interior of any obstacles.		thus occupying the memory space	the location of static obstacles
[Sharir and Schorr (1984)]	Rotational plane-sweep is performed for each graph node and find all the other nodes that are visible	Entire Visibility graph is stored in memory	Pre-materialization is not suitable for updating the obstacles as the obstacles are detected.	As new dynamic obstacles are detected updating the obstacles is difficult
[Wolfson in (1997, 1998, 1999)	Assume objects move in a network. Location attribute is given by Polyline.	Focuses on capturing the current motion of moving points and anticipated locations in the near future	Polyline has been derived from the network initially and movement is described as geometric terms	1. There was no relationship between moving objects and network 2. Requires expensive geometric computatio ns
Vazirgiannis and Wolfson (2001)	Considers modeling and querying moving objects in road network	Network is represented as blocks.	Each tuple describes geometry of the edge	Model is application specific
Güting and colleagues in (1999)	Developed a spatial temporal model that captures histories of movement	Moving objects and regions are represented using realms and rose algebra	Model is a route-oriented model	Moving obstacles points and regions discovery and predictions leading to traffic jams are not addressed

The work done so far has not addressed the problem of obstacles in transportation network as objects and entities do not freely move in a 2D space. The moving and static but dynamic obstacles can hamper the transportation network there by causing traffic congestion. Integrating realms into database system is difficult as updates of realms must be propagated to realm-based attributes values in objects. The aim and objective of the current research is to construct a object based spatial-temporal database model to represents spatial-temporal networks such as roads, highways, routes and dynamic static and moving obstacles in transportation network and computes spatial temporal queries for moving objects.

3. Mathematical Framework

The mathematical formulation of the model is as follows:

3.1. Workspace and environment

The transportation network (G) is represented as a Work Space W which is populated with static and dynamic obstacles called as "obstacle region space", which can be either 2 or 3 dimensional Euclidean space IR^2 or IR^3. Let q be any configuration in C-space, P: $x_{0,}\ x_1\ x_2 \ldots\ldots\ldots\ x_n$ be the path followed by the vehicle for the configuration q in any one of the environment and the closed set A (q) consists of points occupied by the vehicle for the configuration q in C. i.e. A (q) = $\{(x_i, y_i) / (x_i, y_i)\} \in P_{q,}$ a path in configuration q. The network is considered as an undirected graph g = (V, E) where each node in V corresponds to one intersections and edge set E represents a path.

(i) Stationary State Workspace and Environment

The stationary state work space W_s a subset of W, is made up of vehicles on roads.

(ii) Moving State Workspace

The moving state work space W_m, a subset of W, is made up of static and dynamic obstacles. The configuration subspace of W_m contains following different environment:

- Obstacles are static and known
- Obstacles are static and only large obstacles are known
- Obstacles are static and only small obstacles are known
- Obstacles are static and unknown
- Obstacles are moving and unknown

The configuration q in C-space represents static and moving objects relative to the transportation network such as: (i) static position of vehicles and obstacles (ii) Static regions (disaster area) (iii) Moving position of vehicles and obstacles and (iv)Moving regions (traffic jam, parts of network affected by disaster such as excess of water logging). There is thus a need to handle different spatial relationships such as (i) network-network space (ii) network space-space (iii) network-space.

The network G is made up of set of routes (R) and set of Junctions (J) between the routes. The routes can be bi-directional, indicating movement in both the direction. It is also necessary to distinguish positions on the two sides of route. The position of an object or entity on the route is defined by route measure and route location. Route measure defines the position of the object on the route from the starting point (distance from the origin of the route). Route location defines the position of the object or entity on the route depending upon the type of the route and side of the route. Junction between routes is at two distinct route measures in the network.
Let N be a set of all networks $\{N_1, N_2 \ldots N_k\}$ in a transportation scenario. The route R in the network N is defined by a route identifier (of the type *int*), length of the route (of the type *real*), geometry describing the geometry in plane (of the type line), kind indicating route type, flag for embedding routes into spaces.

Route R is defined as $R= \{(id, l, c, kind, start) \mid id \in int, l \in real, c \in line, kind \in \{simple, dual\}, start \in \{smaller, larger\}\}$.

Route is described by identifier id, a length l, a geometry c, kind k which indicates whether road is unidirectional or bidirectional and a flag start indicating whether a route is incident from or incident to.
Let R be a finite set of distinct routes. A route measure in R consists of identifier and a real number giving the position on that route.
Route Measure in Route $r_m(R) = \{(rid, d)\}$ $rid \in int$, $d \in real$, $\exists$ $(id, l, c, kind, start) \in R$ *such that* $0 \leq d \leq l\}$
Representing the connectivity at junctions is important for computing collision free shortest path. Connectivity code is defined using

1. Adjacency matrix $[A_{ij}]$ where

   $A_{ij} = 1$, if $\exists$ *a connection between i and j*,

   $= 0$, otherwise

2. Incidence Matrix $[I_{ij}]$ where

   $I_{ij} = 1$ if i is incident with x_j

   $= 0$, otherwise

A route location in R(Loc (R)) is either a route measure for a simple route or a route measure defined by a side value for a dual route.

$RLoc(R) = \{ rid, d, side) \mid (rid, d) \in r_m (R)$, side $\in$ Side, for $(rid, l, c, kind, start) \in R$: kind $=\{simple, dual\}\}$

The equality on network locations is defined as follows:

Let (r_1, d_1, s_1) and (r_2, s_2, d_2) be the network locations in N=(R,J) such that

$(r_1, d_1, s_1) =(r_2, s_2, d_2) : \leftrightarrow (r_1= r_2 \wedge\ d_1 = d_2 \wedge s_1= s_2) \vee (\exists ((r_1, d_1), (r_2, d_2), cc) \in J \wedge$ connects $((r_1, s_1), (r_2, s_2), cc) \wedge$ connects$((r_2, s_2), (r_1, s_1), cc))$

Obstacles on Route R in Network N is at position r_m (R) on that route

O (R) = *{ oid, type ∈ {static, moving})| oid ∈ int, type ∈ (static, moving) ∃ (rid, d) ∈* r_m *(R) ∀* r_m *(R) ∈ R, (rid, l, c, k,s) ∈ R such that* $0 \leq d \leq l$ *}.*

3.2. Data types

Defining a data type is to introduce a name for it and the set of possible values for it.
Let the Network be reprsented by data type $D_{network}$ = Network
Let N ={ N_1,….., N_k} be the set of network present in the database. Data type Geopoint and Geoline are defined as:
$D_{geopoint} = \{ (i, gp) \mid 1 \leq i \leq k \wedge gp \in Loc (N_i) \cup \{ \infty \}\}$
gp is a point in the Network i such that gp lies in N_i even though the position of gp is undefined which is represent by ∞
$D_{geoline} = \{ (i, gl) \mid 1 \leq i \leq k \wedge gl \in Reg (N_i) \}$
gl is a line in network i such that gl lies in a region in network N_i
Dmovpoint = $\{(i, gp_1, t_1, gp_2, t_2) \mid 1 \leq i \leq k \wedge [(gp_1, t_1) \wedge (gp_2, t_2) \in Loc (N_i) \cup \{ \infty \}\}$
Dmovline= $\{(i, gl_1, t_1, gl_2, t_2) \mid 1 \leq i \leq k \wedge [gl_1= (gp_1, gp_2, t_1) \wedge gl_2= (gp_1, gp_2, t_1)] \in Reg (N_i) \}$
Ostat = $\{(i, op) \mid 1 \leq i \leq k \wedge op \in Loc (N_i) \cup \{ \infty \}\}$
Omov= $\{(i, om, t_1, t_2) \mid 1 \leq i \leq k \wedge om \in Reg(Ni)\}$

3.3. Constructing the user defined data types

Network x int [rid] x real [r_m] x int [side] → geopoint
Network x int [rid] x → geoline
Network x int [rid] x real [r_m] x real [r_m] xint→geoline
Network x int [rid] x real [r_{m1}] x real [r_{m2}] x int [side] x instant [time1] → movpoint
Network x int [rid] x real [r_{m1}] x real [r_{m2}] x int [side] x instant [time1] x instant [time2] → Movline
Network x int [rid] x int [oid] x real [r_m] x int [side] → Ostat
Network x int [rid] x int [oid] x real [r_{m1}] x real [r_{m2}] x int [side] x instant [time1] x instant [time2] →Omov

3.4. Accessing the route information

Given a route identifier (i.e. a rid) we can access information belonging to that route.
Network x int [rid] → line→ Length
Network x int [rid] → Boolean→ Kind
Network x int [rid] → Boolean→ Startsmaller
Network x int [rid] x real [rm_1] x time [t_1] x time [t_2] x real [rm_2] → real → length

3.5. Accessing obstacles information

Network x int [rid] x int [oid] x real [rm_1] → real → location
Network x int [rid] x int [oid] →int→ type

3.6. Interface to data types

To access the geopoint, geoline, movpoint, movline, ostatic and omoving there is a need of an interface to export these data values into relations, e.g. to find the information about the shortest path, network within the water logging etc Following operations are defined and declared:
Routes_info (rid: int, pos1: real, pos2: real, curve1: line)
Junction_info (rid1: int, jid: int, pos1: real, rid2: int, pos2: real, pos: point)
Sections_info (rid: int, secid: int, jid: int, pos1: real, pos2: real, sectm1: real, sectm2: real, curve1: line)
Proute (rid: int, pos1: real, pos2: real, curve: line, no: point)
Location (rid: int, pos1: real, t1: instant)
Route_Loc (rid: int, pos1: real, t1: instant, pos2: real, t2: instant)
Trajectory (rid: int, side: int, t1: instant, t2: instant, pos1: real, pos12: real)
Traverse (rid: int, nid: int, side: int, t1: instant, t2: instant, pos1: real, pos2: real, pos21: real, pos22: real)
Obstacle_location (Oid: int, rid: int, side: int, t1: instant, t2: instant, pos1: real, pos2: real, type: string)
Obstacle_position (Oid: itn, rid: int, side: int, t1: instant, t2: instant, pos1: real, pos2: real, type: string)

3.7. Game theoretic model

In the current study, the following games theory models have been designed and developed to resolve traffic congestion and avoid collision in a transportation network:
(i) Non-cooperative game between multi vehicles drivers on road to resolve traffic congestion
(ii) Cooperative game between multi vehicles on road at a road merges to avoid traffic congestion
(iii) Non-cooperative game between a vehicle driver as one player and nature as another player.

3.8. Problem definition

In the current research the researcher provides a spatial-temporal database model of the spatial network, locations and regions to represent a transportation network. The model represents network graph as routes, nodes of the graph and the position of moving vehicles on edges of the graph as junctions. The network is designed using user defined datatypes such network, geopoint, geoline, movpoint, movpoint, ostat and omov.

4. Methodology

4.1. Game theory model

4.1.1. Stationary state game theory model

A stationary state is made up of n-players: Traffic Authority as one central player and other vehicles drivers such as emergency vehicle driver and private and public transport vehicle drivers on road as other players.

The traffic authority is responsible for managing the traffic flow so as to implement an optimal assignment of paths so that the sum of all travel times or the total latency of the traffic network is minimized. By introducing rules and regulations into the existing traffic system, the traffic authority regulates the behavior of self-interested car drivers indirectly so that the quality of the whole traffic system is improved.

The game is formulated as a n person zero sum non cooperative game with Traffic Authority as one player and moving vehicles as other game players. Let A be the Traffic Authority. Let S be a set of n players, $\{S_1, S_2, Sn\}$. For each player S_i, a finite, nonempty set U^i be the action space for S_i. Each $u_i \in U_i$ is referred to as an action of S_i. For each player S_i, a function, L_i: $A \times U_1 \times U_2 \times \ldots\ldots\ldots \times U_n \rightarrow R \cup \{\infty\}$ is called as the cost function for S_i.

(i) Total Path Latency

Each path $p \in P$ is governed by a latency function l_p that describes the delay incurred by the traffic density on p. The path latency function l_p is a function of the traffic flow f_p on that path. Under the assumption that car drivers behave in a selfish manner, they will travel on the minimum-latency paths, given the perceived traffic congestion due to other road users. That is, the path latency function $l_p(f_p)$ is the function that car drivers want to minimize for their own benefits.

The total latency is measured by:

$$C(f) = \sum l_p (f_p \times f_p) \qquad (1)$$

(ii) Nash Equilibrium

In a transportation network each vehicle driver always chooses the best available path based on his belief about path congestion due to other agent action. In an environment in which each vehicle driver is aware of the situations facing all other drivers, Nash equilibrium is a combination of choices, one for each driver, from which no driver has an incentive to unilaterally move away. In a transportation network the traffic is set to be at Nash equilibrium when no drivers may lower his/her transportation cost/time by unilaterally changing the path.

4.1.2. Moving state same theory problem stationary

A moving state is made up of 2-players: Nature as one player and emergency vehicle driver as another player. As an emergency vehicle travels towards the disaster site, it may come across many obstacles dynamically introduced by nature, which may block its path. Thus the moving state problem is formulated as a 2 person zero sum non cooperative game with Nature as one player and moving emergency vehicles as another player.

Let Y denote nonempty finite set called the observation space, which is the set of all possible observations $y \in Y$. Let U be the nonempty set of vehicle drivers action space. Each $u \in U$ is referred to as an action. A nonempty set β is called as nature action space. Each $\beta \in \beta$ is referred to as a nature action. Thus the payoff L: $U \times \beta$, for all $\beta \in R \cup \{\infty\}$; where ∞ indicates the undefined position and does carry the usual meaning of Infinity. The optimal strategy is denoted by π^*. The set Y (β) for each $\beta \in \beta$ determines which nature actions are possible for each observation, $y \in Y$, where β(y) is defined by

$$\beta(y) = \{\beta \in \beta \mid y \in Y(\beta)\} \qquad (2)$$

The optimal strategy, π^* is given by

For each $y \in Y$

$$\pi^*(y) = \underset{u \in U}{\operatorname{argmin}} \left\{ \max_{\beta \in \beta(y)} \left\{ L(u, \beta) \right\} \right\} \qquad (3)$$

4.1.3. Game theory problem between multiple vehicles at a road merge

A merge is produced if there are n number of lanes turning into m number of lanes. In merging the drivers try to switch to the lanes where vehicles are moving fast on the lane or when a shorter traffic queue is observed. The road merge spot often encounters traffic congestion especially during peak hours. The game is formulated as an m person cooperative game between multiple vehicles on roads at a road merge. Let $M = \{M_1, M_2 \ldots M_m\}$ be the set of m players. For each player M_i, a finite, nonempty set W_i; be the action space and each $w_i \in W_i$ is referred to as an action of M_i. For each player M_i, a function, Li: $W_1 \times W_2 \times \ldots\ldots\ldots \times W_n \rightarrow R \cup \{\infty\}$ is called as the cost function for M_i.

4.2. Object-Oriented spatial temporal database model numbering

To represent the features of the transportation network, user defined data types are defined such as Network, Geopoint, Geoline, Movpoint, Movline, Ostatic and Omoving to represent the network, a position within the network and a region within a network , moving point, moving region, static obstacles and moving obstacles within a network. These data types are defined in java and then implemented using SQL

4.2.1. Network data types

Network is created as an abstract data type. Network is made up of routes, junctions and Sections. Routes, Junctions and Sections are the inherited from the base class Network. Adjancey matrix is prepared to list the data structure connection between the routes, junctions and sections within the network.

Routes and Junctions are represented by the following relation

Routes (rid: int, length: real, geometry: line, kind: Boolean, Startsmaller: Boolean, RoadLevel: int}

Junctions (jid: int, rid1: int, pos1: real, rid2: int, pos2: real, junctiontype: int)

Sections (secid: int, jid: int, rid: int, pos1: real, pos2: int, kind: Boolean, geometry: line)

To represent static locations and routes the data types Geopoint and Geoline are used which represents simple points and lines in the transportation network; which are defined by:

Geopoint {networkid: int; routeid: int; position: real, side: {up, down, none}}

Geoline {networkid: int; routeid: int; side :{ up, down, none}, position1: real; position2: real}

4.2.2. Data types to represent moving point and moving line

To represent data types moving point and moving line we need to represent temporal unit consisting of a time interval (time1, time2)

The data type movpoint represent a moving point in a transportation network. The structure of Movpoint is as follows:

Movpoint {networkid: int; routeid: int; side: {up, down, none}; time1: instant; time2: instant; position1: real; position2: real}

The data type Movline is a collection of records of Movpoint. The structure which a collection of records of Movpoint is as follows:

Movline {networkid: int; side: {up, down, none} time1: Instant; time2: Instant; position1: real; position2: real; position21: real; position22: real}

4.2.3. Obstacles data types

To represent static and moving obstacle we need to represent them of type geopoint and geoline.

The data type Ostat represents static obstacles. The structure of Ostat is as follows:

Ostat {Oid: int; Type: string; routeid: int; networkid: int; Position1: real; Side {up, down, none}; Time1: instant;Time2: instant}

The data type Omoving represents moving obstacles. The structure of Omoving as follows

Omoving {Oid: int; networkid: int; routeid: int; Side: {up, down, none}; Time1: instant; Time2: instant; Position1: real; Position2: real}

5. Implementation

The game theoretic object-oriented spatial-temporal database model is implemented for the transportation network of Mumbai D Ward to facilitate the response activity during disaster.

5.1. Geographical region under study

Greater Mumbai region has a total area of 4355 sq. km, consisting of 1273 sq. km urban area. The Urban region has a population of 17.7 million, out of the total population of 18.9 million. Mumbai is divided into 13 wards. Mumbai D Ward is selected as study area. The Ward covers an area of 6.63 square kilometers. The approximate population of the ward is 3,99,931 (as per 1991 census) with an additional day-time floating population of 1,00,000. D ward has 23 buildings that are old and are identified as dilapidated and dangerous for occupation. The ward is selected as it covers historical, religious and tourist’s centers.

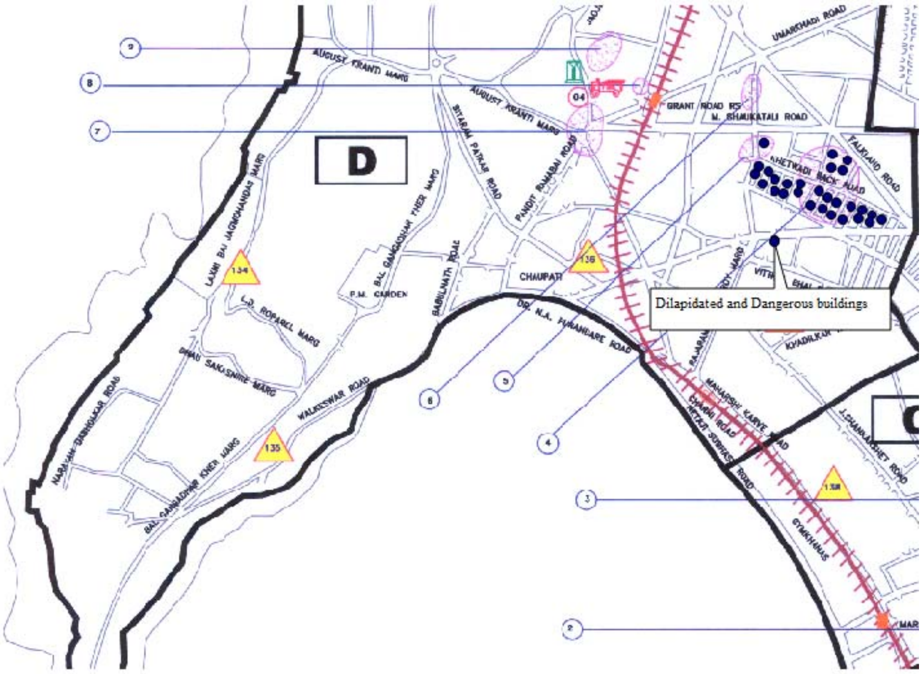

Fig 1. Dilapidated and dangerous buildings in Mumbai D Ward.

5.2. Preparation of spatial data

Using the information of Adjacency Matrix and Incidence Matrix the coverage data structure is prepared. The coverage model incorporates the topological relationships into the structure of feature data. Using the adjacency matrix and incidence matrix spatial-temporal database is developed. The following relations are designed and developed to represent the vehicles on roads and the traffic jams:

MumbairoadsD (rid: int, name: string, routeid: int, length: real, geometry: line, dual: Boolean, major: Boolean, roadLevel: int)
Mumbaijunction (jid1: int, routeid: int, pos1: real, rid2: int, pos2: real, junctiontype: int)
Vehicle (license: string, trip: movpoint)
Emergencyv (vno: int, trip: movpoint)
Hospitals (name: string, nno: int, loc: geopoint)
Speed_limit (limit: int, stretch: geoline)
Traffic_jam (no: int, area: movline, no_of_vehicles: int)

5.3. Algorithm to determine the traffic congestion

1: Determine the traffic flow = traffic density * mean velocity, where traffic density = traffic flow/mean velocity.
2: Lane changing is determined as (p + V (1 + e)) = 0, where p represents the density, *V* is the velocity and *e* represents the value the lane-changing effect parameter where it is presented by the subtraction of two traffic queue length in two lanes where merging is concerned.

3: The traffic density from the fluid-dynamic algorithm is applied to *p* in the kinematic wave theory as follows:

(*Traffic flow* / *mean velocity*) + *V* (1+*e*) = 0

5.4. Game theoretic model to resolve traffic congestion

The transportation network is treated as a game, having cities, highways and drivers. The game is represented as a network graph. Each edge in the directed graph represents a road and is assigned a travel time. Multiple different vehicles drivers such as emergency vehicles, private vehicles, and public transport travel from a start point to a destination point. Travelling through this network is cost-intensive for a driver. For these reasons, the total travel time of a player sums up to his negative payoff. Travel time is measured in minutes. Each player's strategy is a path on the graph representing a driver's route.

(1) Identify the game player
Let A be the traffic central authority as one game player and the vehicles S be a set of n players, $S_1, S_2, \ldots, S_n$.

(2) Determine the action space for each player
Let β be the action space of traffic central authority A and U^i be the action space for each vehicles S_i. Each $u^i \in U^i$ is referred to as an action of each vehicle S_i and $\beta^i \in \beta$ as an action of traffic central authority A.

(3) Determine the vehicles obstructing the path
Determine the vehicles M^{obs}_1, $M^{obs}_{2\ldots}M^{obs}_n$ of S_i obstructing the path and causing network congestion.
LIST Ostat.oid from Ostat Left join Emergencyv on Ostat.routeid=Emergencyv. Routeid

(4) Modelling network traffic using game theory
The travel time on network routes increases proportionally to the number of drivers using the route. The route edges are therefore congestion-sensitive. Suppose the emergency responders need to travel from p_{start} to p_{end}, the disaster site. Let the number of vehicles travelling from p_{start} to p_{end} during peak hours be 1000. There are 3 possible strategies for the vehicles drivers to choose:

Case I: Distribution 0-1
Case II: Distribution 0.5-0.5
Case III: Distribution 0-1(75%) and 0.5-0.5(25%)
To determine the traffic density following queries were designed:
LET location= geopoint (MumbairoadsD, 1, 140, up)

LET density = LIST hour (atperiods (v.trip, 10), location)) as hour From Vehicles As v where atperiods (v.trip, 10) passes location;
LIST hour, count (*) as no_vehicles from density group by hour

(5) Nash Equilibrium
In the current study, the distribution 0-1(75%) and 0.5-0.5(25%) is the optimal strategy for transportation network

5.5. Game theoretic solution for moving state

As the emergency responder vehicle travels towards the disaster site it may come across the dynamic obstacles introduced by nature. A non-cooperative game between emergency vehicle as one payer and nature as another player is designed and developed.

(1) Determine the static and moving obstacles obstructing the path of vehicle
LET O = LIST Ostat.oid from Ostat JOIN on Emergencyv where Ostat.routeid= emergency.routeid;

LET M = LIST Omoving.oid from Omoving JOIN on Emergencyv where Omoving.routeid= Emergency.routeid;

(2) Selection of optimal path
Find the suboptimal path with pay – offs using

2 a) find pay – off, by constructing the cost function L: U ∈ β → IR ∪ { ∞ }, such that if the moving vehicle chooses the action u ε U then the nature chooses $\beta_j \in \beta$ (u). Express L as the matrix of the order |U |∈ |β|

Calculation of pay – off for each sub – optimal path P
{
Sum = 0
For each ordered pair (ui, β_j)
{
Computer payoff p_i, sum = sum + p_i,
Such that the sum depends on the vehicle drivers own action and its type, and the action of nature along that path
}
}.

2 b) *Path Selection Algorithm (q^{Start}, q^{End,} a suboptimal path)*

1 *{*
2 *Select a suboptimal P_{op}*
3 *Follow suboptimal P_{op}*
4 *If Obstacle β detected then go to Step 6*
Else
If goal reached then Store P, compute its
Pay – off, travel time, distance, no of re-Planning and deviations
5 *If next trial go to step 7 else stop*
6 *Plan a new path from its current position to the goal position by a distance transform algorithm and go to Step 2*
7 *Follow the path with low no of re-planning P_{low}*
8 *}*

5.6. Cooperative game theoretic solution to avoid traffic congestion at a road merge

In the current study we show that traffic congestion can be prevented with the help of these cooperating vehicles and urban landscape objects. The cooperating game theory model helps the drivers drive more intelligently – or rather more cooperatively – with the aim of preventing congestion.

To design and implement the concept of self regulating traffic the concept of Time-Division Multiple Access (TDMA) is used.

(1) Scenario of a road merge: With reference to Fig 2, at time t1 the vehicle on the right lane V2 (slot2) needs to change lane. This vehicle has 3 different strategies to do so:

Case I: The vehicle V2 breaks and waits for an opening on the left lane. The vehicle in slot 3 would not be affected, but this would cause the vehicle in slot 2 to be left behind its slot, to run into new slots that would potentially appear behind it, and to cause traffic congestion.

Case II: The vehicle V2 keeps its speed and changes to the left lane, not keeping the safety distance to the vehicle in slot 3 behind it. This would likely cause the vehicle in slot 3 to do an emergency break, potentially running into slot 4 and thus resulting in congestion.

(2) Determine the traffic jam at road merge
As shown in Fig. 2 two traffic jams tend to merge at the junction of two routes or/and a combination.
Define
J1= Select no_of_vehciles from traffic_jam where traffic_jam.no=1
J2= Select no_of_vehciles from traffic_jam where traffic_jam.no=2
J3=J1+J2

In order to avoid the traffic jam at a road merges; the vehicle V2 (Fig. 2b) needs to cooperate with other vehicles to avoid traffic congestion. Vehicle V2 communicates with the vehicle in slot 3 to attempt to coordinate the lane change.

As a result, the vehicle in slot 3 would slightly slow down and vehicle V2 would slightly accelerates and vehicle V3 needs to coordinate with vehicle V4 running behind it to avoid collision.

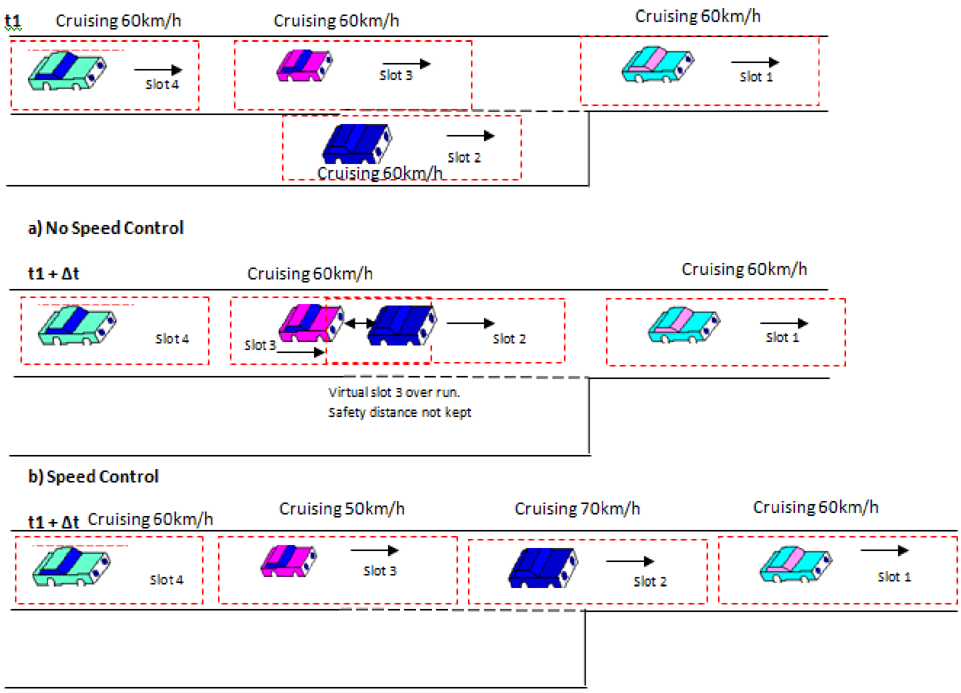

Fig 2. Merging lanes a) without and b) with vehicle cooperation.

5.7. Spatial temporal queries

To facilitate the disaster management response activities during a disaster it is important to detect and evade the traffic congestion and compute the shortest path taken by the emergency vehicles to reach the disaster site. User defined functions are designed to implement the spatial-temporal queries such as shortest distance, determine traffic congestion, current location of traffic jam, when and where did traffic jam appear and disappear etc. The spatial-temporal queries have been designed using user defined functions, inbuilt functions. The structure of the queries is as follows:

(1) To find the location of the emergency vehicle 101 at 10 am?

LET 10am= LIST Movpoint.routeid; Movpoint.t1, Movpoint.pos1 from Movpoint left join emergencyvehicle on emergencyv.routid=Movpoint.routeid where emergency.vno= 101 and Movpoint.time1=10

Return the road name

LIST r.name from mumbairoadsD where emergencyv.routeid=Movpoint.routeid

(2) To find where was the emergency vehicle 101 between 9.00 am and 11.00?

LIST Movpoint.position1, Movpoint.position2, Movpoint.routeid from Movpoint left join emergencyv on emergencyv.routeid= Movpoint.rid where Movpoint.t1 >= 9 AND Movpoint.t2 < =11 AND emergency.vno=101

LIST Mumbairoads.name from Mumbai.roads where Mumbairoads.routeid=1

(3) To find which emergency vehicle is closest to road id=101 and moving towards that direction?
LET road122= point ('Dward', 122);
LIST name, distance (current (trip), road122) As dist from emergencyv where current (der (distance (trip, road122))) <0 order by dist

(4) To order roads by their distance between hospitals
LIST mumbairoads.routeid, length (mumbairaods.routeid)/ (count (*) +1) As dist from mumbairoads where geopoint.rid>=1 AND mumbairoads.rid<=2 group by mumbairoads.rid order by dist

(5) To find which traffic jam exists currently and at which location?
LIST traffic_jam.no, traffic_jam.area, movline.nid, movline.rid, movline.side, movline.t1, movlinet2, movline.position1, movline.position2, movline.position21, movline.pos22 from movline As m, traffic_jam As h left join traffic_jam on traffic_jam.area=movpoint.nouteid

(6) To find when and where did traffic jam 101 appear and disappear?
LIST traffic_jam.area, traffic_jam.no, movline.position1, movline.position2, Movline.position21, movline.position22 from movline As m1, traffic_jam As h right join traffic_jam on traffic_jam.area=Movpoint.nid AND traffic_jam.no=101

(7) To find during which time did traffic jam 101 grow and shrink?
No= LIST traffic_jam.no_of_vehciles from traffic_jam where traffic_jam.no =101
T1= LIST mgline.t1 from mgline where mgline.nid =1
T2= LIST mgline.t2 from mgline where mgline.nid=1
P= T1-T2

(8) To find what time did vehicle 555 spend within traffic jam 101?
LIST Movpoint.networkid, Movpoint.routeid, Movpoint.position1, Difference (Movpoint.time1-movpoint.time2) from movpoint right join vehicle where vehicle.routeid= Movpoint.routeid

5.8. Multi-Agent navigation graph for computing the shortest path

To compute the navigation graphs all static obstacles and moving vehicles are represented as sites. The multi agent navigation graph is computed using the concepts of second order vornoi diagram To compute the navigation graph NG the work space W is partition into two subsets-the set of static obstacles W_o and the sets of agents W_S. The multi agent navigation graph is a union of the first order vornoi graph VG^1 (W) and a subset of second order vornoi graph VG^2 (W) contained inside 1st order vornoi region of each agent.

$$NG(W) = (V, E) \quad (4)$$

Where V= $\{v \mid v \varepsilon V^1 \cup (V^2 \cap \text{Vor}(w_i \mid W)) \text{ for all } w_i \varepsilon W_S\}$, E= $\{e \mid e \varepsilon E^1 \cup (E^2 \cap \text{Vor}(w_i \mid W)) \text{ for all } w_i \varepsilon WS\}$, $NG^1(W) = (V^1, E^1)$ and $NG^2(W) = (V^2, E^2)$
The NG (W) consists of vertices and edges from the 1st and the 2nd order vornoi graphs VG1 (W) and VG^2 (W).
User defined function fshortestpath () is designed to compute the shortest path from p_{start} to p_{end} private static void fShortestPath (String line, String [] parts, Graph graph, File problemFile, int counter)

6. Computational Model

6.1. Game Theoretic Approach

The Computational Model is developed using game theoretic approach and is tested for 30 test runs for in a dynamic environment. The test evaluates the efficiency and stability of the game theoretic model to identify the vacant slots to remove deadlock to set the stationary vehicle in motion to resolve network congestion. Each moving vehicle acts a dynamic obstacle for other vehicle. The dynamic moving obstacles are detected by the vehicle sensors as the vehicles travels from q^{start} to $q^{goal.}$ Based on the vehicle driver action space and the action space of the central authority the vacant slots are computed to resolve network congestion.

Once the cars are in motion the dynamic static obstacles are detected and optimal path is computed. On reaching the goal the optimal path P along with travel time, distance, number of replaning and deviations stored. If the vehicle driver has a requirement to traverses the same path in future the previously stored optimal path with less number of replaning can be used. The test reveals that in a dynamic environment the best path to the goal is not necessarily the shortest path. Taking a longer path can sometimes reduce the collision risk and speed up the mission.

6.2. Object Oriented Spatial Temporal Extension

The Spatial Temporal Extension of the model is able to represent a spatial embedded network in terms of routes, junctions. It distinguishes between simple and divided roads and describes connectivity at junctions. The new data types are integrated into a relational environment with suitable interface functions. The model is able to extend rose algebraic concepts to game theory to represent a transportation network

7. Conclusion

This paper tackles the problems of traffic assignment in a transportation network in a 2D and 3D environment. A dynamic environment is made up of static known obstacles, static unknown obstacles and dynamic moving obstacles. In this study the problem of network congestion is addressed by using concepts of game theory. In order to implement the game theory concepts in a transportation network a spatial temporal model is designed and developed. The spatial temporal model developed is a comprehensive model of abstract data types for representing moving entities in a dynamic environment. The user defined data types, functions and queries represent urban transportation traffic scenario in terms of a directed network graph. The model designed is able to incorporate game theoretic concepts for detection and evasion of traffic congestion.

References

Aljarboua Z. (2009): Geometric path planning for general Robot Manipulators, World Congress on Engineering and Computer Science, Vol II

Almeida V.T., Gütting R.H., Ding, Z. (2006): Modeling and querying moving objects in a network, The VLDB Journal, 15(2): pp. 165-190

Andersen E., Sud A (2008): Real-time Path Planning for Virtual Agents in Dynamic Environment, SIGGRAPH 08, ACM, 10.1145/1401132.1401206

Bonchi F., Potamias M., Castillo C.; Gionis A. (2009): Fast Shortest Path Distance Estimation in

Large Networks, CIKM 09, ACM, 978-1-60558-512-3/09/11

Berthold V. (2005): Congestion Games: Optimization in Competition, Proceedings of the thirty-seventh annual ACM symposium on Theory of computing, pp 67-73

Chen X., Li Y. (2006): Smooth Formation Navigation of Multiple Mobile Robots for Avoiding Moving Obstacle, International Journal of Control, Automation and Systems, vol. 4 no 4 pp. 466-479

David L. (2005): Micro-Foundation of Congestion and Pricing: A Game Theory Perspective, Transportation Research Part A 39, pp. 691-704

El-Zawawy M., El-Sharkawi M. (2009): Clustering with Obstacles in Spatial Databases, In Proc. 2001 IEEE Int. Symposium of Signal Processing and Information Technology (ISSPIT01), pp. 420-425, Cairo, Egypt

Faldini E., Yu J. (2004): A Game Theoretic approach to Traffic Flow Control, University of Leuven

Gütting R.H., Ridder T., Schneider M., (2008): Implementation of ROSE Algebra: Efficient Algorithms for Realm-Based Spatial Data Types, GU 293/1-2

Prikryl J.(2006) : Overview of distributed decision making for urban traffic control, IEEE Industrial Electronics, IECON 2006 - 32nd Annual Conference on, 4695 - 4700

Villalobos I. A. (2009): Urban Traffic Control Problem: A Game theory Approach, Proceedings of 47th IEEE Conference on Decision and Control, IEEE Explorer