Engineering Applications of Artificial Intelligence 29 (2014) 163–177

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Modeling of route planning system based on Q value-based dynamic programming with multi-agent reinforcement learning algorithms

Mortaza Zolfpour-Arokhlo [a,*], Ali Selamat [b], Siti Zaiton Mohd Hashim [b], Hossein Afkhami [c]

[a] Department of Computer Engineering, Sepidan Branch, Islamic Azad University, Sepidan, Iran
[b] Faculty of Computing, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia
[c] Department of Electronics and Power, Sepidan Branch, Islamic Azad University, Sepidan, Iran

ARTICLE INFO

Article history:
Received 29 April 2013
Received in revised form
1 December 2013
Accepted 5 January 2014
Available online 31 January 2014

Keywords:
Route planning system (RPS)
Multi-agent system (MAS)
Multi-agent reinforcement learning (MARL)
Q-learning
Traffic congestion

ABSTRACT

In this paper, a new model for a route planning system based on multi-agent reinforcement learning (MARL) algorithms is proposed. The combined Q-value based dynamic programming (QVDP) with Boltzmann distribution was used to solve vehicle delay′s problems by studying the weights of various components in road network environments such as weather, traffic, road safety, and fuel capacity to create a priority route plan for vehicles. The important part of the study was to use a multi-agent system (MAS) with learning abilities which in order to make decisions about routing vehicles between Malaysia′s cities. The evaluation was done using a number of case studies that focused on road networks in Malaysia. The results of these experiments indicated that the travel durations for the case studies predicted by existing approaches were between 0.00 and 12.33% off from the actual travel times by the proposed method. From the experiments, the results illustrate that the proposed approach is a unique contribution to the field of computational intelligence in the route planning system.

1. Introduction

Route planning systems (RPS) are one of several types of traffic information systems that offer routes that solve traffic problems. RPS provides optimum route solutions and traffic information (Ji et al., 2012) prior to a trip in order to help drivers arrive at their destination as quickly as possible. Route planning problems can be solved by determining the shortest paths using a model of the transportation network (Kosicek et al., 2012; Geisberger, 2011). The drivers of vehicles with different solutions available to them need quick updates when there are changes in road network conditions. Unfortunately, the task of efficiently routing vehicles in the route planning (Suzuki et al., 1995; Pellazar, 1998; Stephan and Yunhui, 2008) has not been emphasized enough in recent studies. Multi-agent systems (MAS) are a group of autonomous, interacting entities sharing a common environment, which they perceive with sensors and upon which they act with actuators (Shoham and Leyton-Brown, 2008; Vlassis, 2007). MASs are applied in a variety of areas, including robotics, distributed control, resource management, collaborative decision support systems (DSS), and data mining (Bakker et al., 2005; Riedmiller et al., 2000). They can be used as a natural way of operating on a system, or they may provide an alternative perspective for centralized systems. For instance, in robotic teams, controlling authority is naturally distributed between the robots. Reinforcement learning (RL), which allows (Tesauro et al., 2006; Lucian et al., 2010; Bakker and Kester, 2006) learning provides an environment for learning how to plan, what to do and how to map situations (Jie and Meng-yin, 2003) to actions, and how to maximize a numerical reward signal. In an RL, the learner is not told which actions to take, as is common in most forms of machine learning. Instead, the learner must discover through trial and error, which actions yield the most rewards. In the most interesting and challenging cases, actions affect not only the immediate rewards but also the next station or subsequent rewards. The characteristics of trial and error searches and delayed reward are two important distinguishing features of RL, which are defined not by characterizing learning methods, but by characterizing a learning problem. Any method that is suitable for problem solving is considered to be an RL method. An agent must be able to sense the state of the environment, and be able to take actions that affect the environment. The agent must also have goals related to the state of the environment. In other words, an RL agent learns by interacting with its dynamic environment. The agent perceives the state of the environment and takes actions that cause the environment to transit into a new state. A scalar reward signal evaluates the quality of each transition, and the agent must maximize the cumulative rewards along the course of interaction. RL system feedback reveals if an activity was beneficial and if it meets the objectives of a learning system by maximizing expected rewards over a period of time (Shoham and Leyton-Brown, 2008;

[*] Corresponding author.
E-mail address: zolfpour@gmail.com (M. Zolfpour-Arokhlo).

0952-1976/$ - see front matter © 2014 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.engappai.2014.01.001

Busoniu et al., 2005). The reward (RL feedback) is less informative than it is in supervised learning, where the agent is given the correct actions to perform (Bussoniu et al., 2010). Unfortunately, information regarding correct actions is not always available. RL feedback, however, is more informative than un-supervised learning feedback, where no explicit comments are made regarding performance. Well-understood, provably convergent algorithms are available for solving single-agent RL tasks. MARL faces challenges that are different from the challenges faced by single-agent RL such as convergence, high dimensionality, and multiple equilibria. In route planning processes (Schultes, 2008; Wedde et al., 2007), route planning should take into account traveler responses and even use these responses as its guiding strategy, and realize that a traveler's route choice behavior will be affected by the guidance control decision-making of route planning. On the other hand, the results of a traveler's choice of routes will determine network conditions and react to the guidance control decision-making of route planning (Dong et al., 2007; Yu et al., 2006). Reduced vehicle delays could be achieved by examining several conditions that affect transportation network studying the weight of transport environmental conditions (Tu, 2008; Zegeye, 2010). These conditions include: weather, traffic information, road safety (Camiel, 2011), accidents, seasonal cyclical effects (such as time-of-day, day-of-the-week and month) and cultural factors, population characteristics, traffic management (Tampere et al., 2008; Isabel et al., 2009; Almejalli et al., 2009; Balaji et al., 2007; Chang-Qing and Zhao-Sheng, 2007) and traffic mix. Regarding these variables can be used to provide a priority trip plan to vehicles for drivers. Increasingly, information agent-based route planning (Gehrke and Wojtusiak, 2008) and transportation system (Chowdhury and Sadek, 2003) applications are being developed. The goal of this study is to enable multiple agents to learn suitable behaviors in a dynamic environment using an RL that could create cooperative (Lauer and Riedmiller, 2000; Wilson et al., 2010) behaviors between the agents that had no prior-knowledge. The physical accessibility of traffic networks provided selective negotiation and communication between agents and simplified the transmission of information. MAS was suited to solve problems in complex systems because it could quickly generate routes to meet the real-time requirements of those systems (Shi et al., 2008). Our survey review indicated that MAS methods and techniques were applied in RPSs (Flinsenberg, 2004; Delling, 2009) including dynamic routing, modeling, simulation, congestion control and management, traffic control, decision support, communication, and collaboration. The main challenge faced by RPS was directing vehicles to their destination in a dynamic real-time RTN situation while reducing travel time and enabling the efficient use of the available capacity on the RTN (Khanjary and Hashemi, 2012). To avoid congestion and to facilitate travel, drivers need traffic direction services. These services lead to more efficient traffic flows on all transport networks, resulting in reduced pollution and lower fuel consumption. Generally, these problems are solved using a fast path planning method and it seems to be an effective approach to improve RPS. For example, RPS is used in the following areas: traffic control, traffic engineering, air traffic control (Volf et al., 2011), trip planning, RTN, traffic congestion, traffic light control, traffic simulation, traffic management, urban traffic, traffic information (Ji et al., 2012; Khanjary and Hashemi, 2012) and traffic coordination (Arel et al., 2010). The main thrust of using RPS in these situations is to compare the shortest path algorithms with Dijkstra's algorithm. Currently, there are various successful algorithms for shortest path problems that can be used to find the optimal and the shortest path in the RPS. The research will contribute to

- Modeling a new route planning system based on QVDP with the MARL algorithm.
- Using the ability of learning models to propose several route alternatives.
- Predicting the road and environmental conditions for vehicle trip planning.
- Providing high quality travel planning service to the vehicle drivers.

This paper consists of seven sections. Section 2 describes the meaning of RPS based on MARL and related works. This will be followed by a definition of the RPS based on the MARL problem (Section 3). Section 4 will present the MARL proposed for RPS. Section 5 discusses the experimental method used in this study. Section 6 presents the results, comparisons, and evaluations. Finally, the paper is concluded in the last section.

2. Related works

RPS based on multi-agent decisions is described as a tool in transportation planning (Dominik et al., 2011). The agent chooses among, competing vendors, distributes orders to customers, manages inventory and production, and determines price based on a perfect competitive behavior. The concept of an agent-based trip planning in transportation was studied by Yunhui and Stephan (2007) to reduce the negative effects of disruptive events. The RTN environment is highly dynamic, complex and has many constraints. In this study, a major effort was made to improve MAS solutions and algorithms to use in a simulated transportation environment (Kovalchuk, 2009). Consequently, each agent managed a specific function of the transportation environment network and shared information about other agents. Communication between agents was based on local information. As discussed above, the agents had their own problem solving capabilities and were able to interact with each other in order to reach a goal. Interaction could occur between agents (agent–agent interaction) and between agents and their environment. In the environmental sciences, MAS is often used as a platform (Jones, 2010; Robu et al., 2011) for space–time dynamics. The characteristics of the agents encountered also differed, ranging from simple reactive agents to more intelligent agents that show a limited capacity for reasoning and making decisions as noted by Ligtenberg (2006). RPS integrates technologies such as information technology, networks, communications, and computer science (Shimohara et al., 2005). A new approach in RPS is being studied that can save space and costs for cars at intersections or junctions (Vasirani and Ossowski, 2009). Chen and Cheng (2010) surveyed different applications of agent technology in traffic modeling and simulations. They emphasized the importance of agent technology for improvement of the RTN and for traffic system performance. Ruimin and Qixin (2003) conducted an extensive study of coordination controls and traffic flows that focused on agent-based technology. All of these studies used models based on agent flexibility to improve traffic management. The aim of this study was to propose a novel RPS approach based on MAS and MARL that could perform better than previous RTN methods such as those discussed (Adler et al., 2005; Wu et al., 2011a), in terms of accuracy and real-time performance. The results obtained from this approach were compared with Dijkstra's algorithm, which was used to find the best and optimal path between the origin and destination nodes in a directed transportation graph. Agents were defined as the specific functional performance of a set of inputs in Uppin (2010). The key problem faced by an agent was that of deciding which action it should perform in order to best satisfy its design objectives. Using suitable agent architectures facilitated agent decision making. Agent architecture is software architecture for decision-making systems that are embedded in an environment (Michael, 2009).

Kovalchuk (2009) attempted to find better MAS solutions and algorithms in simulated transportation environments, where each agent managed a specific function for the network and by sharing information with other agents. Agents communicated with other agents based on local information. Chang-Qing and Zhao-Sheng (2007) did their research in the field of intelligent traffic management by looking at trip planning and RTN based on MAS. Chen et al. (2009) investigated using a MAS approach in the field of traffic management and traffic detection. Rothkrantz (2009) studied a distributed routing model for a personal intelligent system. Shimohara et al. (2005) and Zou et al. (2007) investigated RPS. Seow et al. (2007) and Kiam and Lee (2010) developed a multi-agent taxi dispatch system in which collaborative taxi agents negotiated to minimize the total waiting time for their customers. Vasirani and Ossowski (2009) studied a market inspired approach for urban road traffic controls, which allowed cars to individually reserve space and time at a junction or an intersection. These studies attempted to use models based on agent flexibility; improved traffic management. Although these are important tasks for the efficient routing of vehicles, these studies did not focus on RTN availability. Transportation networks based on multi-agents for route selection were studied by Chuan-hua and Qing-song (2004). Transportation network control can be used by utility, road traffic and water distribution network (Vasan and Slobodan, 2010) management, which often requires a multi-agent control approach (Negenborn et al., 2008a). The proposed method and Dijkstra's algorithm have the ability to provide directions with regard to unspecified circumstances affecting transportation environments. The applications of the shortest path problem include vehicle routing in transportation systems and traffic routing in communication RPSs.

Therefore, using a MAS to help with trip planning and RTN problems (Dangelmaier et al., 2008) is an efficient approach leading to correct path selection and improvements in RPSs. Dynamic real-time vehicle RPS incorporates real-time transportation network information, enhancing reactions to changes in road conditions. This updates the vehicle's path (Ryan and Donald, 2008; Marcelo et al., 2010) in real time allowing the RPSs to provide more options to the driver of the vehicle en-route to their destination. Vehicle routing was studied by Changfeng et al. (2006), Watanabe and Lijiao (2011), Weyns et al. (2007), Barbucha (2011) and Claes et al. (2011). Vehicle routing planning systems take vehicles from trip origin to trip destination on an RTN (Schmitt and Jula, 2006). In order to improve route assignments in RPS, a Q-value based dynamic programming (QVDP) algorithm and Boltzmann distribution were used to minimize total travel time (Shanqing et al., 2012). The shortest path and the greedy strategy were calculated by using conventional algorithms such as Dijkstra algorithm and QVDP. In the RTN, a QVDP algorithm is used to find the optimal travel time from every origin node (intersection) to every destination node (Shanqing et al., 2012).

2.1. Multi-agent reinforcement learning (MARL)

One of the objectives of this study was to control traffic flow using an intelligent agent-learning model for dynamic route planning network nodes. Learning is essential for unknown environments, and it is useful for system construction as it is exposed to reality rather than trying to write it down. Learning modifies the decision mechanisms of an agent to improve performance, which was evaluated using some measures. Measures evaluate the environment as they appraise the quality of the actions performed by the agent, the amount of time taken to perform those actions, the amount of electricity consumed, and how comfortable the trip was. Performance measures also provide objective criteria for assessing the success of agent's behavior. The environment can include roads, other traffic, pedestrians, and customers. Agents interact with their environment through an actuator and sensors. Actuators include the steering mechanism, accelerators, brakes, signals, horns, and sensors, include cameras, speedometers and GPS (Jie and Meng-yin, 2003). The learning method depends on several factors including the type of performance and learning elements, available feedback, the type of the component to be improved, and its representation. A learning agent is composed of two fundamental parts, a learning element, and a performance element. The design of a learning element is dictated by what type of performance element is used, which functional component is to be learned, how that functional component is represented, and what kind of feedback is available. In Fig. 1, the conceptual framework based on MARL is presented. In this framework, each agent completes its assigned duties independently, but it must cooperate with other agents to achieve the overall goal. When agents are used as illustrated in this framework, the system is able to handle all the necessary calculations based on real-time traffic information to guide the vehicles to the nodes, and each node can suggest the next path to the vehicles. The methods for learning agents will be described in the following sections, and an experiment conducted using these methods to study how well they updated the knowledge of intelligent agents

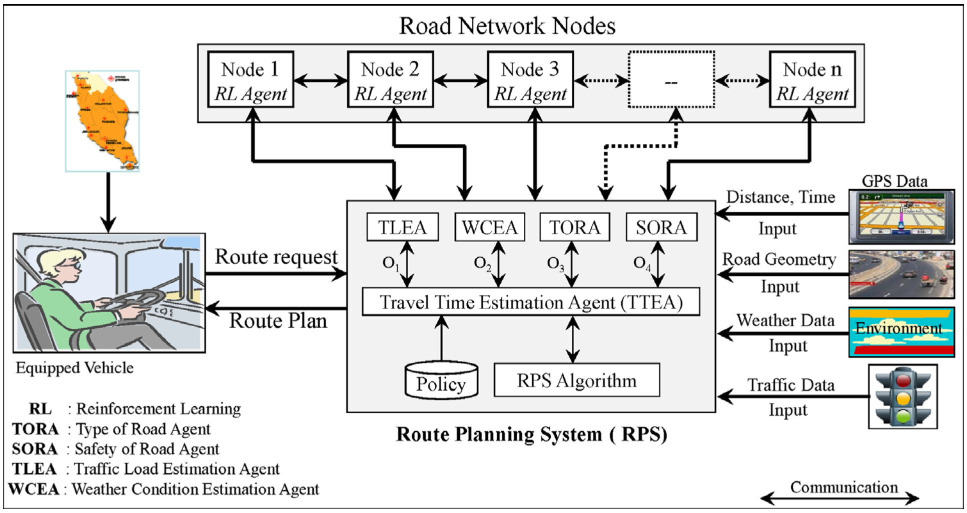

Fig. 1. The conceptual framework based on MARL.

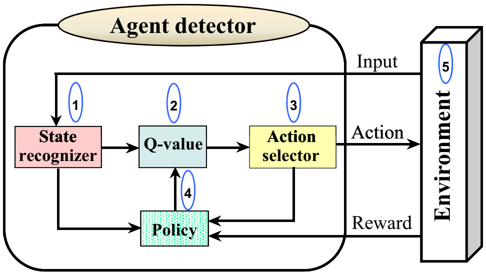

Fig. 2. RL agent/environment interaction.

will be discussed. The study assumed that the vehicles were equipped with the necessary technology to communicate with the agents. The routing preferences of the drivers were not taken into consideration. The position of each vehicle was determined using GPS and the road conditions were known to the driver of the vehicle. The duties of the agents are summarized as follows:

- Transferring data obtained from sensors.
- Handling route requests from vehicles.
- Calculating the best path.
- Sending path information to vehicle drivers.
- Receiving/sending information to other agents.
- Cooperating with other agents.

Other important features of the proposed framework include:

1. It is a distributed structure aimed at increasing the speed of computing in large-scale networks.
2. The system can learn. Therefore, its behavior can change under the influence of uncertain and dynamic environments.
3. In this system adequate stability can be achieved through cooperation and coordination between the various agents.

In many systems, including relatively simple systems, determining the exact behavior and the activity of a set of MAS is difficult. Accurate detection of agent behavior is needed to obtain information related to both external and internal environmental conditions, which is almost impossible in a dynamic environment. Adaptation and agent learning ability are important characteristics of MARL that help solve such problems. RL system feedback is a beneficial activity, and it is the objective of a learning system to maximize the expected reward over time (Shoham and Leyton-Brown, 2008; Busoniu et al., 2005). In this study, the term “state” refers to whatever information is available to the agent. The state is usually provided by some preprocessing system that is part of the environment. RL methods prescribe how an agent changes policies according to its experiences. As shown in Fig. 2, each agent detector consists of state recognizer [1], Q-value [2], an action selector [3], and policy [4]. A MARL generally has six basic components: an agent, an environment [5], a policy, a reward function, a value function, and a model of the environment. The RL components of the study are summarized in Table 1. Based on the above, MARL can be thought of consisting of six components (Ferdowsizadeh-Naeeni, 2004) as follows:

1. *Agent*: The learner that can interact with the environment via a set of sensors and actuators.
2. *Environment*: Everything that interacts with an agent, i.e. everything outside the agent.
3. *Policy*: A mapping from perceived states set “*S*” to the actions set “*A*”.
4. *Reward function*: A mapping from state-action pairs to a scalar number.
5. *Q-value*: The total amount of reward an agent can expect to accumulate over the future, starting from that state.
6. *Environment*: A model representing the environment that can be used for predicting its state transition before applying a particular action (Fig. 2).

Table 1
RL components of the study.

RL components	Component usage
1. Agent	Installed in each node
2. Environment	Dynamic transportation network
3. Policy	Routing policy from current to destination node
4. Reward	Real travel time
5. Q-value	Total rewards from current node to destination node
6. State (S_t)	The node that the vehicle passes through the network

In general, policy specifies the stimulus-response rules or associations and can determine what action should be taken at intervals. It also includes other components that serve to change and improve the policy. On the other hand, rewards stem from the goal of the RL task. It may be delivered to the agent with a delay. The reward function defines the goal of the RL agent. It maps the state of the environment to a single number. The agent's objective is to maximize the total number of rewards it receives. Value depends on the choice of an optimality model. The difference in the RL algorithm centers on how they approximate this value function specifies in terms of what is most beneficial in the long run. While rewards determine immediate desirability, value indicates the long-term desirability. Most of the methods discussed in this paper are centered on forming and improving approximate value functions. The model maps state-action pairs to probability distributions over the states. Consequently, it often requires a great deal of storage space. If there are “*S*” states and “*A*” actions, then a complete model will take up a space proportional to $S \times S \times A$. By contrast; the reward and value functions map states to real numbers and thus use only “*S*” amount of space.

2.2. RL agent/environment interaction

RL is a control strategy in which an agent, embedded in an environment, attempts to maximize total rewards in pursuit of a goal (Zolfpour-Arokhlo et al., 2011). RL entities and agent's interactions include everything except out the agent, which can be defined as the environment. In the RL framework, the agent makes its decisions as a function of a signal from the environment, which is called the “environment state”. In this interaction, the agent takes action with the objective of maximizing the expected rewards, which send new states to the agent. The agent at any time is in a particular state relative to the environment and can take one of a number of actions within that state to reach its goal. The agent observes starting with the initial state, then chooses an action to perform and then observes the new state to adjust its policy (Torrey, 2010; Paternina-Arboleda and Das, 2005). When the agent performs an action, it receives feedback in the form of a reward from the environment, which indicates if this is a good or bad action. The value of an action or being in any state can be defined using a Q-value (the Action-Value Function or Q-value), $Q^n(s,n)$ which indicates the expected return when starting from state (s), taking an action (a), and then following policy $\mu(n)$. Q-learning (Demircan et al., 2008) is model-free and is guaranteed to meet to the optimal policy in unmoving environments with a finite number of states. Its learning rate changes over time (Awad, 2011). Each time, S the agent is in a state of s_t, it takes an action a_t , and it observes the reward $R(s_t, a_t)$. Afterwards it moves to the next state, S_{t+1}. Q-learning is a standard technique for

MARL that is widely used in multi-agent environments (Akchurina, 2010) and it works by learning an action-value function (Q-value) (Chabrol et al., 2006).

3. RPS based on MARL problem definition

In this section, problem formulation, the proposed shortest path and MARL algorithms will be presented as follows.

3.1. RPS problem formulation

The core of vehicle routing (Adler and Blue, 2002) in a RPS is to find the shortest path by collecting real-time traffic information. Real-time traffic information can be collected using GPS data, traffic cameras, traffic detectors, intelligent vehicle detectors and the speed of the vehicle. A road-directed graph, represented by $\overrightarrow{G}=(N,M)$, is a directed RPS that is based on an electronic map, a set of $N=\{1,2,\ldots,n\}$ nodes, and $A=\{1,2,\ldots m\}$, which are a set number of directed links. $S=\{s_{ij}|(i,j)\in A\}$ is a set of distance measures between nodes, $T=\{t_{ij}|(i,j)\in A\}$ consists of the sum of all travel time, and V is the maximum speed of the vehicles. Therefore, according to the trip origin (o) node and destination (d) node, this issue can be solved as an optimization problem (Zafar and Baig, 2012) based on the real-world RTN using the following equation:

$$R_{(o,d)}=\min\sum_{i=1}^{o}\sum_{j=1}^{d}\alpha * s_{ij}\begin{cases}\alpha=1 & \text{there is a link between } i,\ j \text{ and } (t_{ij}/V)\le s_{ij},\\ \alpha=0 & \text{otherwise}\end{cases}\qquad(1)$$

where:

- i and j are the current state movements into right and bottom side directions, respectively,
- $R_{(o,d)}$ is the shortest path from an origin node, "o" to a last node, "d",α
- is a binary digit (0 or 1), and
- links are independent of each other.

Real-time information can be acquired using installed agents, video cameras, GPS, and other devices. In this study, the traveling cost (time and distance) was provided by Google Maps. The new classification system used covered all public roads and extended to unclassified rural and urban roads. The main thrust of this approach was to make the classification more objective and consistent with specifying quantifiable parameters, such as traffic, population and spacing, to guide the selection of road classes. The system used in this study was a dynamic system where road classes could be periodically reviewed so that they could be adapted to reflect changes in traffic and function.

3.2. RPS algorithm (RPSA)

RPS algorithm (Algorithm 1) was created based on the TTEA information shown in Fig. 1. The inputs for this algorithm consisted of all the calculated route weights based on agent data generated for each route from origin to a destination. The proposed algorithm was based on nodes and route weights according to the reported agent data of each route. In this study, "Node weight" was defined as the shortest distance from each node to the last network node. If "n" was the last node number, then "n-1" and "k" were the two node numbers connected to Node "n" in the network. The RPSA procedure is presented in Algorithm 1, which uses the following steps:

- *Step* 1. The distance between node "n-1" and node "n" i.e., $R_{(n-1,n)}$ is equal to the weight of a node "n-1" i.e., W_{n-1}. Also the distance between nodes "n" and node "k", i.e., $R_{(k,n)}$ is equal to the weight of node "k" i.e., W_k. Therefore, $R_{(n-1,n)}=W_{n-1}$ and $R_{(k,n)}=W_k$.
- *Step* 2. Similarly, the procedure in Step 1 is continued to compute the weights of all the subsequent nodes until the first node weight is computed. Finally, the weights of the first node will be used as the minimum path distance between the first node to the last one in the network, and the equations can be defined as follows:

$$\begin{aligned}&R_{(n-1,n)}=W_{n-1},\quad R_{(k,n)}=W_k\ \text{s.t. } k,n\in\mathbb{N},\\&R_{(n-2,n-1)}=W_{n-2},\quad R_{(k-1,n-1)}=W_{k-1},\\&R_{(n-3,n-2)}=W_{n-3},\quad R_{(k-2,n-2)}=W_{k-2},\\&\ldots\ldots=\cdots\ ,,\qquad \ldots\ldots=\ldots\ldots\\&\ldots\ldots=\cdots\ ,,\qquad \ldots\ldots=\ldots\ldots\\&R_{(1,2)}=W_1,\quad R_{(2,3)}=W_2.\end{aligned}$$

Algorithm 1. RPS algorithm (RPSA).

```
INPUT:
G(N,M) be a directed graph with a set of "N" nodes and set of "M" directed links.
PARAMETERS:
R(o,d), a nonnegative number stands for the cost where "o" is the start node and "d" is the last one. i, j, k; loop index, G (1, i) is an array of vertexes source; G(2,i) is array of vertexes destination. G(3, i) is an array of link distance(or cost); CostArray(i) is an array of node costs(node weights) table. For each node and SP(i) is arrayed of the shortest path from the origin to final node.
OUPUT:
  CostArray(k) is a cost data table for all nodes. SP(k), a shortest path data table for a graph.
INITIALIZATION:
  // All nodes from last to first nodes are examined for the route's connected nodes. For each link do the operation in two steps as follows:
set CostArray[1···node-1]=999, CostArray(node)=0, SP (i)=0;
BEGIN
  for all nodes
    for j=first to last links // j is set to be the destination node of links.
      if (G(2 , j)=i ) // k is set to be the source node of links.
        cost=CostArray(i) + G(3 , j);
      end if
    end for
  end for

  for i=first to last links
    while (the origin (k) is the same in graph, G)
      if (G(3 , i)=CostArray(k) - CostArray(j))
        SP(k)=G(2 , i);
      else
        i=i + 1;
        k=G(1 , i);
      end if
    end while
  end for
END
```

3.3. MARL problem formulation

In this study, T: $S\times A\longrightarrow S$, was assumed to be a transition function, R: $S\times A\longrightarrow R$, was assumed to be a reward function and

$\mu : S \longrightarrow A$, became the learned policy, a state space represented by S, and A represented by an action space. In addition, μ^* was the learned optimal policy used to maximize the action-value for all states, and V^* was the Q-value of optimal policy. $Q^*(s_t, a_t)$ became the expected policy that provided the optimal performance in the state s_t. Problem of RL was modeled as a Markov decision process (MDP) (Lecce and Amato, 2008; Demircan et al., 2008) using the following variables:

- S is a finite set of possible states,
- A is a finite set of actions,
- r is a scalar reward,
- $P : S \times S \times A \longrightarrow [0, 1]$ is the transition function, where $P(s_t, s_{t+1}, a_t)$ gives the probability of arriving in state s_{t+1} when performs action a_t in state s_t, and
- $\pi : S \longrightarrow p(A), p(A)$ is the set of all probability measures on A.

For example, suppose $V(s_t)$ was the optimal Q-value,

$$V(s_t) = \max \sum_{a \in A(s)} \pi(s_t, a_t) Q(s_t, a_t), \quad \forall s_t \in S, \tag{2}$$

where

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma * \sum_{s_{t+1} \in S} p(s_{t+1} | s_t, a_t) V(s_{t+1}),$$
$$\forall s_t \in S, a_t \in A(s_t), \tag{3}$$

If α is the learning rate ($0 < = \alpha < = 1$), γ is the discount rate ($0 < = \gamma < = 1$), and $Q(s_t, a_t)$ is the value of action a_t executed by the agent. If $\gamma = 0$, the agent was concerned only with maximizing immediate rewards. In these cases, the objective was to learn how to choose action a_t to maximize only reward r_{t+1}. Generally, acting to maximize immediate rewards can reduce access to future rewards so that the sum of returns may be reduced. If $\gamma = 1$, the agent became a better predictor and the objective paid greater attention to future rewards. In this situation, the SARSA (State-Action-Reward-State-Action) algorithm (Shanqing et al., 2012) and the Q value of action $a_t \in A(s_t)$ were updated by following equations:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r(s_{t+1}, a_{t+1}) + \gamma * Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)],$$
$$\forall s_t \in S, a_t \in A(s_t). \tag{4}$$

Therefore,

$$Q(s_t, a_t) = (1 - \alpha) Q(s_t, a_t) + \alpha[r(s_{t+1}, a_{t+1}) + \gamma * Q(s_{t+1}, a_{t+1})],$$
$$\forall s_t \in S, a_t \in A(s_t), \tag{5}$$

Fig. 4 shows a Q-value architecture. When an optimal policy is found, the Q-learning (Fig. 3) algorithm (Chang-Qing and Zhao-Sheng, 2007; Chen et al., 2009) can compute the Q-value using Eq. (5). The Q-value was the expected traveling time to destination d, when the vehicle bound with node i moved to its destination node j (Kuyer et al., 2008). In this study, the method that combined a Q-value based dynamic programming (QVDP) with the Boltzmann distribution (Shanqing et al., 2012) was based on the following equations:

$$Q_d^{(n)}(i,j) \longleftarrow r(i,j) + \sum_{k \in A(j)} P_d^{(n-1)}(j,k),$$
$$d \in D, \quad i \in I - d - B(d), \quad j \in A(i) \tag{6}$$

$$P_d^{(n)}(i,j) \longleftarrow \frac{e^{Q_d^{(n)}(i,j)/\tau}}{\sum_{k \in A(j)} e^{Q_d^{(n)}(i,j)/\tau}}, \quad \sum_{k \in A(j)} P_d^{(n)}(i,j) = 1, \tag{7}$$

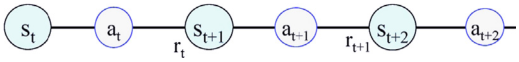

Fig. 3. Q-learning scheme.

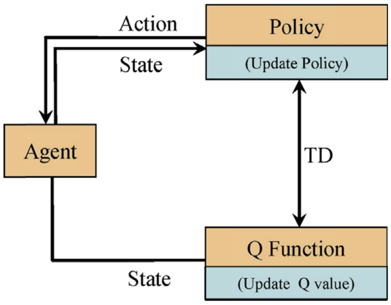

Fig. 4. Q-value architecture.

$$Q_d^{(n)}(d,j) = 0, \quad d \in D, \ j \in A(d), \tag{8}$$

$$Q_d^{(n)}(i,d) = r(i,d), \quad d \in D, \ i \in B(d), \tag{9}$$

$$P_d^{(n)}(d,j) = 0, \quad d \in D, \ j \in A(d) - d, \tag{10}$$

$$P_d^{(n)}(d,d) = 1, \quad d \in D. \tag{11}$$

Noted

- $i,j \in I$: set of suffixes of nodes,
- $d \in D$: set of suffixes of destinations,
- $r(i,j)$: traveling time from i to j,
- $A(i)$: set of suffixes of nodes moving directly from i,
- $B(i)$: set of suffixes of nodes moving directly to i,
- $Q_d^{(n)}(i,j) : Q_d(i,j)$ in the nth iteration, and
- $P_d^{(n)}(i,j)$: the probability that the vehicle bound for destination d moves to node j at node i in the nth iteration,
- τ: temperature parameter.

The goals of the combined Q-value DB with the Boltzmann distribution are as follows:

(a) To reduce traffic congestion.
(b) To adapt temperature parameter (τ) is to different traffic volumes.
(c) To change traveling time from i to j (r_{ij}).
(d) To prove a global optimal route and traveling time.

The aim of this study was to combine QVDP with the Boltzmann distribution to calculate the average traveling time so that the optimal and best routes could be found for different road traffic situations. In order to find the way to distribute the traffic volume effectively, it was crucial to select an optimal and reasonable temperature parameter for the QVDP with the Boltzmann distribution. When τ was large, the Boltzmann distribution was identical to the random policy, while $\tau = 0$ means that the shortest path is only available like greedy strategy (Eq. 7). In this study, the different road traffic situations were evaluated by looking at the different τ's. The equipped vehicle traveled from the origin node (state) to its destination and all the nodes (states) crossed by the vehicle, while it is traveled along its route, will have their Q-values updated. The Q-learning algorithm (Algorithm 2) process is as follows:

- *Step* 1. Observe the running of all equipped vehicles and record their travel times.
- *Step* 2. For each state (node), update the Q-value using Eq. (5).
- *Step* 3. If the state is traversed on route to the destination, go to Step 4, otherwise observe and record the travel time and track to the next state of vehicles and go to Step 2.
- *Step* 4. If Q-value = 0, record travel times for all vehicles.

Algorithm 2. Q-learning algorithm.

```
INPUT:
  $n$ number of states (nodes), $e$ number of experience routes (actions), $r(s_t, a_t)$ set of costs, $\tau$ the temperature parameter;
PARAMETERS:
  $\alpha$, learning rate $(0 \le \alpha \le 1)$ and $\gamma$, is discount rate $(0 \le \gamma \le 1)$. $R$, $P$, and $\tau$are a reward function, the probability of vehicle arriving in state $s_{t+1}$ and temperature parameter respectively;
OUTPUT:
  $P(s_t, i)$, $Q(s_t, a_t)$;
INITIALIZATION:
  // Initialize $s_t$=0, select initial learning rate $\alpha$ and $\gamma$, discount rate. Choose action $a_1$, $a_2$,…,$a_n$ based on probability $P$ with some exploration $s_t$. let $t$=0,
BEGIN
  while true do
  for $i$:=1 to $e$
    if $s_t$ is a coordinate state
    Execute action $a_1, a_2, \ldots, a_n$ in sate $s_t$;
    Observe reward $r(s_1, a_1), r(s_2, a_2), \ldots, r(s_n, a_n)$;
  else
    Transit a new state, $s_{t+1}$
    $s_t$:=$s_{t+1}$;
  end if
  for each Agent
        Take action $a_t$, Observe reward $r(s_t, a_t)$ and state $s_t$;
        Update Q-value as follows:
        $Q(s_t, a_t) = 1-\alpha)*Q(s_t, a_t) + \alpha[(r(s_t, a_t) + \gamma * V(s_{t+1})]$;
        Calculate $P(s_t, i)$ (using Boltzmann formula) as follows:
        $P(s_t, i) = (Q(s_t, a_t)/\tau)/[\sum_{k \in A(i)} Q(s_t, a_t)/\tau]$;
        Choose action according to $P(s_t, i)$formula;
        $s_t$:=$s_{t+1}$;
        Observe $s_t$;
      end for
    end for
  end while
END
```

4. MARL proposed for RPS

While traveling, several critical, real world factors are considered and measured, such as energy use, time, waste (of products), traffic safety and health, accessibility, and economy. The objectives of this system are

- To maximize RTN tools usage.
- To maximize the safety of drivers and other people.
- To maximize the productivity and reliability.
- To minimize air pollution and the vehicle usage.
- To minimize impacts on ecosystem and energy consumption.
- To minimize operating and travel time costs.

In this study, two types of agents were used to respond to various types of RTN services in the RPS. These agents were a control agent and an estimate agent. Agents negotiate and communicate with other agents, perform operations based on local available information, and pursue their local goals. If the software agent is suitably modeled, RPS can improve the speed and quality of work in RTN activities (Srinivasan et al., 2010). In this study, the MARL in RPS was used for modeling the RTN. We identified agents in the RPS to utilize a subset of managing and controlling elements of the RTN. The controlling elements helped the decision-making process by utilizing various agents for demand, supply, and information within the routing path. In the RPS, critical, real-world factors were considered and measured. Each of the factors was reported by an agent and they were defined as follows:

$$\mathbf{Ag} = \{Ag_1, Ag_2, Ag_3, \ldots., Ag_n\} \quad s.t.\ n \in \mathbb{N} \tag{12}$$

In this study, we used five agents Ag_1, Ag_2, Ag_3, Ag_4, and Ag_5, named TLEA (traffic load estimation agent), WCEA (weather condition estimation agent), TORA (type of road agent), SORA (safety of road agent), and TTEA (travel time estimation agent), respectively. The objectives of this system were to maximize the use of RTN tools, ensure the safety of the driver and others, improve productivity and reliability, as well as minimize air pollution, energy consumption, operating costs, travel cost, and travel time. Fig. 1 shows the use of integrated agents to support the RPS. We considered the following agent details related to the input and the results of this study:

(1) *WCEA*: The weather condition estimation agent (WCEA) predicted the weather conditions based on data from local weather stations or the internet. The data resulting from this agent's report will be used by TTEA of RPS.
(2) *TORA*: This agent reported on the type of road that will be avoided when there are alternatives. The type of road agent (TORA) supervised road status and delivered result to the TTEA so that the best route algorithm could be suggested.
(3) *SORA*: This agent reported and evaluated the safety rating of each route using the historical data saved in the system and other specific inputs including weather data, current traffic density status of the route and so on. The results found by this agent became the computed rate of routes safety that was used by the RPS algorithm (RPSA) for route planning in TTEA (travel time estimation agent).
(4) *TTEA*: The travel time estimation agent (TTEA) evaluated the route distance and the vehicle speed for each route by using the RTN map data saved in the system. It also used the specific input data including the time and distance of the route. The TTEA can be extrapolated from the traffic load estimation agent (TLEA) data or from direct measurements of travel times. The output of this agent was the calculation that was used by the RPSA. This agent is the decision-making brain of the proposed model.

The goals of the RPSA were as follows:

1. To receive both the travel origin and the destination (o, d) from the vehicle (Fig. 1).
2. To receive the trip plan for the vehicle,

$$R_{(o,d)} = \sum_{i=1}^{o} \sum_{j=1}^{d} R_{(i,j)}; \quad i,j \ge 1.$$

3. To receive required information via the environmental agents (TLEA, WCEA, TORA, SORA, and TTEA) for each route (Fig. 1).
4. To calculate the total route rate and the real cost (time) of each route, O_{TTEA}, and the actual time (ActTim).
5. To calculate a suitable trip plan for the vehicle,

$$R^*_{(o,d)} = \sum_{i=1}^{o} \sum_{j=1}^{d} R^*_{(i,j)}; \quad i,j \ge 1.$$

6. To propose a real trip plan for the vehicle.

The roles of the agents are to receive cost-effective information from environmental agents and calculate the real route rate based

on the other agent's information and the RPSA results. The RPSA uses the information from the agents. Data regarding the origin and destination of the vehicle, such as time and distance of route, the fuel consumption of the vehicle and other required data were entered and it enabled this agent to assess the best route. Additional information such as the weather forecast was also considered. According to the definition of TTEA, O_{TTEA} and the other agent outputs were defined as follows:
Summation of factors =

$$\sum_{i=1}^{n} \overline{O}_i \quad s.t.\ \overline{O}_i \in [0,1], \quad i \ and\ n \in \mathbb{N} \tag{13}$$

If we limit the number of factors to four, therefore $n=4$, the equation is

$$\sum_{i=1}^{4} \overline{O}_i = \overline{O}_{TLEA} + \overline{O}_{WCEA} + \overline{O}_{TORA} + \overline{O}_{SORA} \tag{14}$$

Subject to

$$ActTim_{(m)} = 60*Distance/\left(1-\sum_{i=1}^{4}\overline{O}_i\right)*AvgS\ pd \tag{15}$$

Assuming that $r = route$, $f_r(t) = \overline{ActTim}$ then

$$f_r(t) = 60*Distance/\left(1-\sum_{i=1}^{4}\overline{O}_i\right)*AvgS\ pd \tag{16}$$

The two following examples demonstrate the model described earlier.

Example 1. As shown in Table 3, the actual time at 9:00 h on Monday for the route IPOH(IP)⟶MELAKA(ML) can be described as follows:

$$\begin{aligned} & f_{IP\longrightarrow ML}(09:00) \\ & \quad = 60*Distance_{IP\rightarrow ML}/[(1-\textstyle\sum_{i=1}^{4}\overline{O}_{IP\rightarrow ML})*AvgSpd] \\ & \quad f_r(t) = 60\ \text{min}*346\ \text{km}/[(1–0.12)*93\ \text{km/h}] \\ & \quad \simeq\ = 254\ \text{min} \end{aligned}$$

Example 2. Regarding Eqs. (10)–(13), the actual time of IP⟶ML at different times during a Saturday as shown in Table 3 can be calculated as follows:

Table 3 shows the actual travel time from IP to ML at different times during the day. Variations in travel time $(\Delta(x_t))$ may be the result of traffic and weather conditions (Table 3). Between 08:00 h and 12:00 h, there is a slight decrease in travel time. However, 12:00–18:00 h signifies little increase in travel time. Finally, the traffic travel time decreases as the day ends and enters into the following morning. Time was defined as t, $y_{t'} = f(x_t + \Delta(x_t))$ such that $\Delta(x_t)$ is the variation of x_t.

The goals of TTEA were as follows:

(1) To receive the travel origin and destination (o,d) from the vehicle (Fig. 1).
(2) To find the primary path, $R_{(o,d)}$ (including the ideal times and distances for multiple routes), regarding the travel origin and destination,

$$R_{(o,d)} = \sum_{i=1}^{o}\sum_{j=1}^{d} R_{(i,j)} \quad s.t.\ i,j \geq 1.$$

(3) To adjust the trip plan to satisfy the travel objectives of the driver.
(4) To propose a path using information from Google Maps and the other agents (TLEA, WCEA, TORA, and SORA).
(5) To adapt route suggestions in response to updated weather, traffic, road geometry, and safety advisories.
(6) To receive the route rate and time costs from the agents.
(7) To compute the total route rate, O_{TTEA}.
(8) To propose path information and total route rate to the RPSA (route, actual time, and O_{TTEA}).
(9) To receive the optimal path (R^*) from RPSA and save it in the database,

$$R^*_{(o,d)} = \sum_{i=1}^{o}\sum_{j=1}^{d} R^*_{(i,j)} \quad s.t.\ i,j \geq 1.$$

(10) To report the most suitable trip plan to the vehicle.

5. Experimental results

This section summarizes the results obtained using MARL based on different temperatures (τ) for RPS in each node to acquire agent information about the status of the next path or routes status as illustrated in Table 2. One of the requirements of RPS is current information about the travel time of a vehicle on a continuous path. This information can be acquired by several detectors, such as magnet sensors, video cameras, GPS, global system for mobile communication (GSM), and other network traffic sensors on the transport route (Zi and Jian-Min, 2005). Other information requirements include RPS equipment hardware, software, and communication between a simulated model and a real traffic network using a routing data protocol. We applied the roles of the agents using MARL for RPS as follows:

(i) Transferring the trip plan information acquired through the sensors to the vehicle's RPS system and the trip plant vehicle,

$$R_{(o,d)} = \sum_{i=1}^{o}\sum_{j=1}^{d} R_{(i,j)}; \quad i,j \geq 1.$$

(ii) Receiving trip route requests from vehicles through the sensor.
(iii) Sending optimal path information via the sensors to other agents and vehicles,

$$R^*_{(o,d)} = \sum_{i=1}^{o}\sum_{j=1}^{d} R^*_{(i,j)}; \quad i,j \geq 1.$$

(iv) Receiving or sending information on the status of the route from other collaborative agents to alert the incoming vehicles.

Table 2 shows the output information of the agents used in this study. The parameters used in Table 2 are the distance between cities, travel time (m), the optimal weight of the computed total route (O_{TTEA}), and the actual travel time (m). These parameters were defined as follows:

- O_{TLEA}, O_{WCEA}, O_{TORA}, O_{SORA}, and O_{TTEA} were the outputs of TLEA, WCEA, TORA, SORA, and TTEA, respectively.
- TrvTim was the route travel time calculated per minute based on Google maps data from the internet.
- ActTim was the actual travel time computed using Eq. (16).
- Distance was the distance in kilometers (km) between Malaysian cities based on data from Google maps.

In Table 2, each agent output reported a specific route weight, which was represented by a number between 0 and 1. The specific route weight reflected the estimated status of each route, and with respect to Eqs. (14)–(16), O_{TTEA} indicated that the updated route weights for each route were computed from the results of other

Table 2
The rates of integrated agent outputs of each route.

Route	Dist. (km)	AvgSpd (km/h)	TrvTim (min)
GM⟷IP	165	61	162
GM⟷KL	267	67	238
GM⟷ML	406	72	338
GM⟷JB	596	80	445
GM⟷KN	342	71	290
GM⟷KT	286	63	273
KG⟷KJ	50	60	50
KG⟷ML	166	79	126
KG⟷JB	356	90	238
KJ⟷ML	124	75	99
KJ⟷JB	314	89	212
KB⟷IP	343	61	337
KB⟷GM	189	63	181
KB⟷JB	787	75	626
KB⟷KN	365	60	365
KB⟷ML	593	68	520
KB⟷KT	169	66	153
KB⟷KL	461	66	417
KT⟷KL	435	72	364
KT⟷KN	199	55	215
KT⟷IP	444	65	408
KN⟷KJ	268	78	205
KN⟷JB	360	75	289
KN⟷ML	255	69	221
KN⟷KL	253	86	177
ML⟷JB	211	91	139
IP⟷ML	346	93	224
IP⟷JB	541	95	342
IP⟷KN	405	82	296
IP⟷KL	202	91	134
IP⟷KG	217	89	146
KL⟷KG	38	62	37
KL⟷KJ	25	56	27
KL⟷ML	136	85	96
KL⟷JB	274	88	186

Dist. is the distance in kilometers (km) between the Malaysian cities based on Google maps data from the internet.
AvgSpd is the average speed of vehicle (per km/h) in this travel.
TrvTim is the route travel time (per minute) computed based on average speed of the vehicle (AvgSpd) and travel distance (Dist.).

Table 3
The actual time of IP ⟶ML at different times within a day.

Travel time hh:mm	Dist. (km)	AvgSpd (km/h)	TrvTim (min)	O_{TLEA}	O_{WCEA}	O_{TORA}	O_{SORA}	$f_r(t)$
00:00	346	93	224	0.00	0.01	0.01	0.00	228
03:00				0.10	0.01	0.01	0.00	254
06:00				0.19	0.03	0.01	0.00	290
09:00				0.12	0.04	0.01	0.00	269
12:00				0.07	0.04	0.01	0.00	254
15:00				0.08	0.07	0.01	0.00	266
18:00				0.19	0.07	0.01	0.00	306
21:00				0.10	0.09	0.01	0.00	279
24:00				0.04	0.02	0.01	0.00	240

O_{TLEA}, O_{WCEA}, O_{TORA}, O_{SORA} and O_{TTEA} are the outputs of TLEA, WCEA, TORA, SORA, and TTEA, respectively.
$T_{IP \longrightarrow ML}$ is the travel time (per minute) received from Google maps.
$f_r(t)$ is the actual travel time computed using Eq. (13).

Table 4
Travel time of IP⟶ML at different times within a week.

Weekday	00:00	03:00	06:00	09:00	12:00	15:00	18:00	21:00	24:00
Sat	228	240	233	254	248	257	248	257	228
Sun	225	235	235	257	269	257	260	257	237
Mon	228	254	290	269	254	266	306	279	240
Tue	225	248	290	263	248	257	290	279	237
Wed	225	240	298	272	251	263	276	269	237
Thu	230	257	298	290	251	266	306	279	243
Fri	235	254	298	279	263	279	306	286	245

received agent's outputs (Table 4). The *ActTim* column was computed using Eq. (16). Using RPSA to generate new route costs and O_{TTEA}, the optimal path between Ipoh (IP) and Johor Bahru (JB) was changed from IP→KL→ML to IP⟷KG→ML (Figs. 7 and 8).

5.1. Case 1

Fig. 5 shows the Malaysian roadway network graph comprising five routes and four cities (nodes): Klang (KG), Kajang (KJ), Melaka (ML) and JB. The optimal path from KG to JB in ideal route conditions is KG→KJ→JB, a distance of 364 km, which should take 262 min of trip time. However, Fig. 6 shows that KG to JB in real-time status is a distance of 356 km with the actual trip time of 282 min. In real-time status, RPSA uses the received agent information, such as traffic agent data, weather agent data and other agent information from the installed MARL by roads. Therefore, by using it to determine the new route information of each route (actual trip time) and O_{TTEA} information, the optimal path from KG to JB is KG→ML→JB. By using the proposed approach for calculating travel time for the suggested optimal path by an existing approach algorithm, the travel time for Fig. 5 will be 294 min, while in Fig. 6 it is 282 min. Therefore, there is a decrease of 12 min. Based on these results, the proposed approach is better than the existing approach.

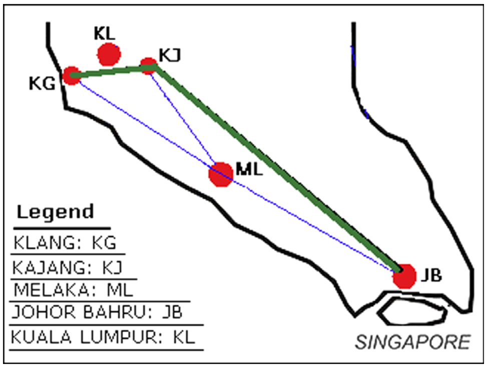

Fig. 5. The optimal path from KG to JB in ideal status per minute (min) in Case 1.

5.2. Case 2

Fig. 7 shows the Malaysian roadway network graph comprising 12 routes and 6 cities (nodes): IP, Kuala-Lumpur (KL), KG, KJ, KN, and ML. The optimal path from IP to ML in ideal route conditions is IP→KL→ML, a distance of 353 km that should take 230 min of trip time. However, Fig. 8 shows that IP to ML in real-time status is a distance of 383 km with the actual trip time of 321 min. In real-time status, RPSA uses the received agent information, such as traffic agent data, weather agent data and other agent information from the installed MARL by roads. Therefore, by using it to determine the new route information of each route (actual trip time) and O_{TTEA} information, the optimal path from IP to ML is IP→KG→ML. By using the proposed approach for calculating travel time for the suggested optimal path by an existing approach algorithm, the travel time for Fig. 7 will be 330 min, while in Fig. 8 it is 321 min. Therefore, there is a decrease of 9 min. Based on these results, the proposed approach is better than the existing approach.

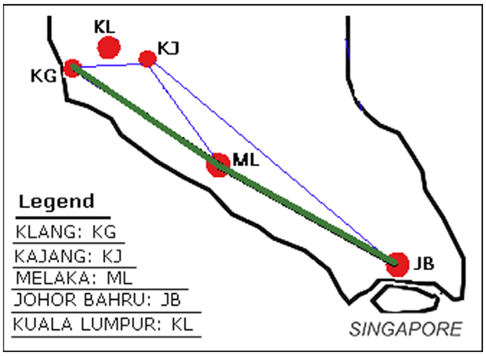

Fig. 6. The optimal path from KG to JB using MARL status per minute (min) in Case 1.

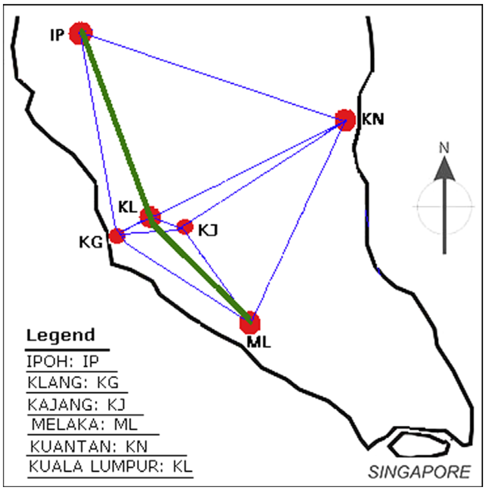

Fig. 7. The optimal path from IP to ML in ideal status per minute (min) in Case 2.

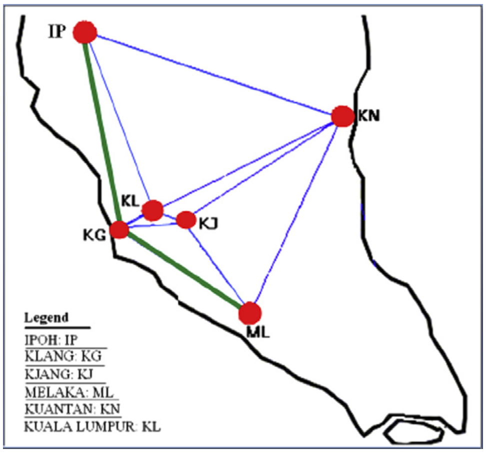

Fig. 8. The optimal path from IP to ML using MARL status per minute (min) in Case 2.

5.3. Case 3

Fig. 9 shows the Malaysian roadway network graph comprising 23 routes and 9 cities (nodes): Kota Bahru (KB), Kuala Terengganu (KT), Gua Musang (GM), IP, KL, KG, KJ, KN and ML. The optimal path from KB to ML in ideal route conditions is KB→GM→KL→ML, a distance of 593 km that should take 515 min of trip time. However, Fig. 10 shows that KB to ML in real-time status is a distance of 595 km with the actual trip time of 569 min. In real-time status, RPSA uses the received agent information, such as traffic agent data, weather agent data and other agent information from the installed MARL by roads. Therefore, by using it to determine the new route information of each route (actual trip time) and O_{TTEA} information, the optimal path from KB to ML is KB→GM→ML. By using the proposed approach for calculating travel time for the suggested optimal path by an existing approach algorithm, the travel time for Fig. 9 will be 638 min, while in Fig. 10, it is 569 min. Therefore, there is a decrease of 69 min. Based on these results, the proposed approach is better than the existing approach.

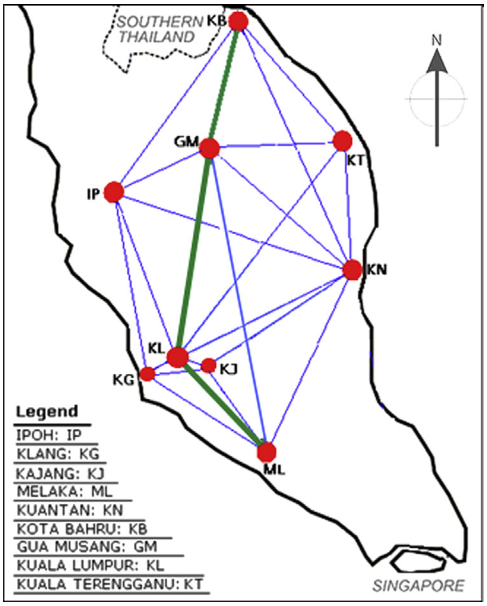

Fig. 9. The optimal path from KB to ML in ideal status per minute (min) Case 3.

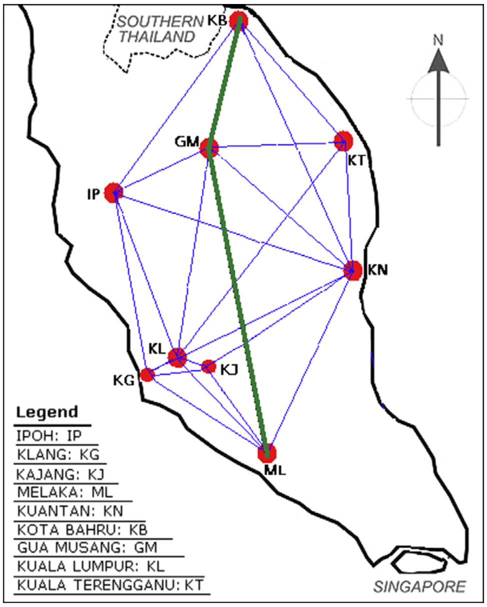

Fig. 10. The optimal path from KB to ML using MARL status per minute (min) in Case 3.

5.4. Case 4

Fig. 11 shows the Malaysian roadway network graph comprising three routes and three cities (nodes): KT, KN and JB. The optimal path from KT to JB in ideal route conditions is KT→JB directly, a distance of 554 km that should take 514 min of trip time. However, Fig. 12 shows that KT to JB in real-time status is a distance

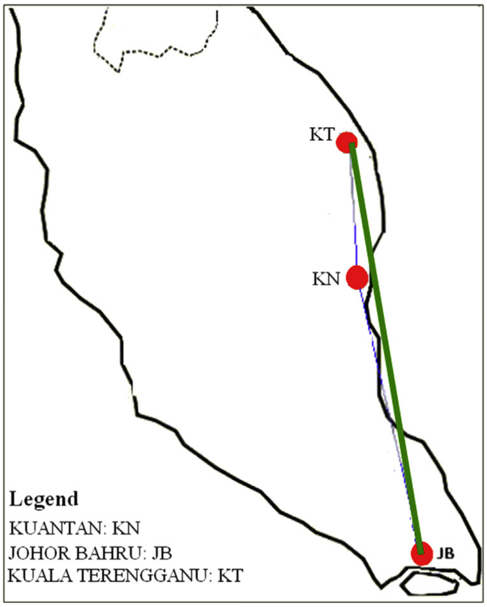

Fig. 11. The optimal path from KT to JB in ideal status per minute (min) in Case 4.

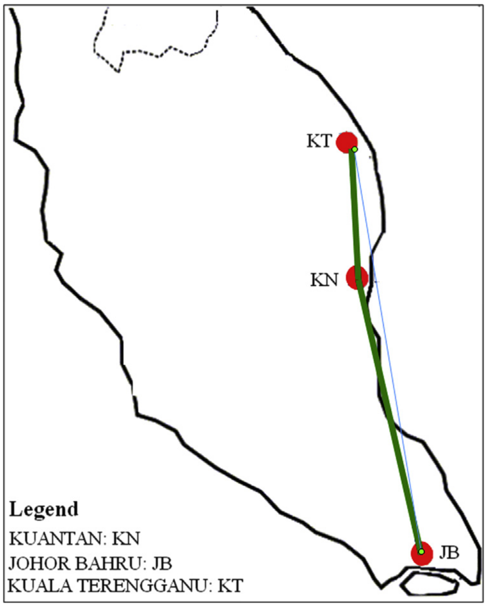

Fig. 12. The optimal path from KT to JB using MARL status per minute (min) in Case 4.

of 559 km with the actual trip time of 591 min. In real-time status, RPSA uses the received agent information, such as traffic agent data, weather agent data and other agent information from the installed MARL by roads. Therefore, by using it to determine the new route information of each route (actual trip time) and O_{TTEA} information, the optimal path from KT to JB is KT→KN→JB. By using the proposed approach for calculating travel time for the suggested optimal path by an existing approach algorithm, the travel time for Fig. 11 will be 615 min, while in Fig. 12 it is 591 min. Therefore, there is a decrease of 24 min. Based on these results, the proposed approach is better than the existing approach.

5.5. Case 5

Fig. 13 shows the Malaysian roadway network graph comprising 31 routes and 10 cities (nodes): KB, GM, KT, IP, KL, KG, KJ, KN, ML and JB. The optimal path from KB to JB in ideal route conditions is KB→GM→KL→JB, a distance of 730 km that should take 605 min of trip time. However, Fig. 14 shows that KB to JB in real-time status is a distance of 806 km with the actual trip time of 715 min. In real-time status, RPSA uses the received agent information, such as traffic agent data, weather agent data and other agent information from the installed MARL by roads. Therefore, by using it to determine the new route information of each route (actual trip time) and O_{TTEA} information, the optimal path from KB to JB is KB→GM→ML→JB. By using the proposed approach for calculating travel time for the suggested optimal path by an existing approach algorithm, the travel time for Fig. 13 will be 776 min, while in Fig. 14, it is 715 min. Therefore, there is a decrease of 61 min. Based on these results, the proposed approach is better than the existing approach.

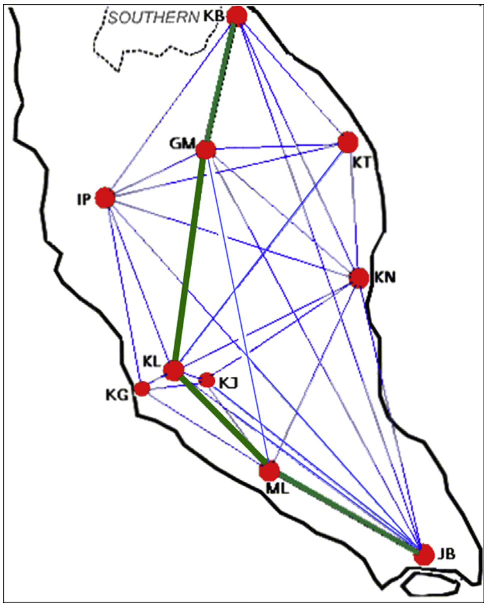

Fig. 13. The optimal path from KB to JB in ideal status per minute (min) in Case 5.

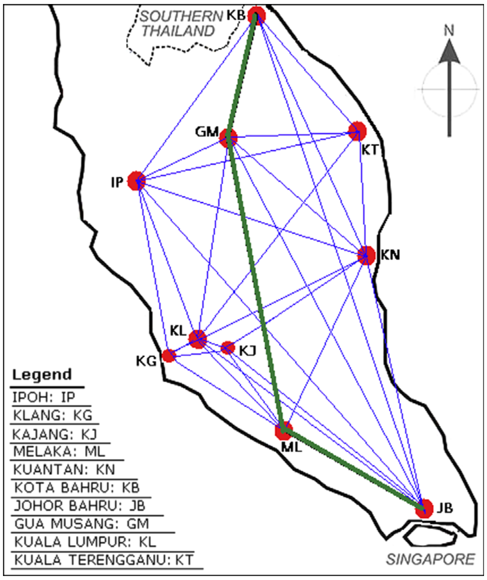

Fig. 14. The optimal path from KB to JB using MARL status per minute (min) in Case 5.

Table 5
The gaps computed based on using existing approach and proposed approach (α, $\gamma=0$).

Temp. rate	Case	Existing method			Proposed method			TimGap(%)
		Optimal path	Ref.	ATTim (min)	Optimal path	Ref.	ATTim (min)	
$\tau=0$	1	KG→KJ→JB	Fig. 5	294	KG→ML→JB	Fig. 6	282	4.08
	2	IP→KL→ML	Fig. 7	330	IP→KG→ML	Fig. 8	321	2.73
	3	KB→GM→KL→ML	Fig. 9	638	KB→GM→ML	Fig. 10	569	10.82
	4	KT→JB direct	Fig. 11	615	KT→KN→JB	Fig. 12	591	3.90
	5	KB→GM→KL→JB	Fig. 13	776	KB→GM→ML→JB	Fig. 14	715	7.86
$\tau=10$	1	KG→KJ→JB	Fig. 5	296	KG→ML→JB	Fig. 6	282	4.73
	2	IP→KL→ML	Fig. 7	332	IP→KG→ML	Fig. 8	322	3.01
	3	KB→GM→KL→ML	Fig. 9	649	KB→GM→ML	Fig. 10	569	12.33
	4	KT→JB direct	Fig. 11	618	KT→KN→JB	Fig. 12	605	2.10
	5	KB→GM→KL→JB	Fig. 13	778	KB→GM→ML→JB	Fig. 14	715	8.10
$\tau=30$	1	KG→KJ→JB	Fig. 5	301	*KG*→ML→JB	Fig. 6	282	6.31
	2	IP→KL→ML	Fig. 7	333	*IP*→KL→ML	Fig. 7	333	0.00
	3	KB→GM→KL→ML	Fig. 9	649	KB→GM→ML	Fig. 10	574	11.56
	4	KT→JB direct	Fig. 11	622	KT→KN→JB	Fig. 12	617	0.80
	5	KB→GM→KL→JB	Fig. 13	780	KB→GM→ML→JB	Fig. 14	728	6.67
$\tau=50$	1	KG→KJ→JB	Fig. 5	304	KG→ML→JB	Fig. 6	282	7.24
	2	IP→KL→ML	Fig. 7	334	IP→KL→ML	Fig. 7	334	0.00
	3	KB→GM→KL→ML	Fig. 9	653	KB→GM→ML	Fig. 10	583	10.72
	4	KT→JB direct	Fig. 11	627	KT→JB direct	Fig. 11	627	0.00
	5	KB→GM→KL→JB	Fig. 13	781	KB→GM→ML→JB	Fig. 14	738	5.51

- *Temp.Rate* is the temperature date parameter.
- *PropMthod* is the proposed policy method using MARL data.
- *ATTim* is the average travel time computed by proposed approach using MARL.
- *TimGap* is the calculated gap between using existing approach and proposed approach strategies method (13).

Table 6
The gaps computed based on using existing and proposed approaches ($\alpha=0.5$, $\gamma=0$).

Case	ExistMthod ATTim (min)	PropMthod ATTim (min)	(%)
1	136	59	56.62
2	191	185	3.14
3	213	191	10.33
4	205	189	20.40
5	213	191	10.33

- *ExistMthod* is the existing policies method using data.
- *PropMthod* is the proposed policy method using MARL data.
- *ATTim* is the average travel time computed by proposed approach using MARL.
- *TimGap* is the calculated gap between using existing and proposed approaches (13).

6. Simulation results and experimental comparison

In this section, the simulation experiments are presented. These experiments were carried out on five different Malaysia RTN topologies, which consisted of 3–10 nodes with 3–30 different links. For example, the results of the new approach described in Section 4 were assessed by using RPSA results and illustrated using several cases. In all the cases, the temperature parameter of the Boltzmann distribution controlled the impact of the Q-values on route generation. This was an important parameter for determining the optimal route. In this section, in ideal status, which uses existing shortest path approaches, such as the Dijkstra algorithm to calculate the shortest path, and the proposed approach, which is using a MARL method to calculate the shortest path and accounts for traffic congestion, will be discussed. In addition, four different temperature parameter strategies ($\tau=0$, 10, 30, 50) were assessed for evaluating the routes under different traffic conditions. This comparison in a real-world network evaluates the proposed method using MARL. In regard to Eq. (17), the TimGap column reported the gap between the real times determined by ideal status (ExistMthod), and the proposed approach (PropMthod). The TimGap equation can be defined as follows:

$$TimGap = \frac{ExistMthod - PropMthod}{ExistMthod} * 100 \tag{17}$$

Finally, in all the experimental cases (Table 5 and Fig. 17), the time gap obtained by using the proposed method was less than the time gap resulting from existing approaches (in ideal status).

6.1. Using MARL for RPS evaluation

This section presents the results obtained by using RPSA and MARL for RPS in the experimental cases as listed in Table 5.

Case 1: As depicted in Section 5.1, the RTN graph of Case 1 shows five routes with four nodes (cities) which were used to generate a graph displaying the optimal path based on Google maps and the proposed method. Table 5 and Fig. 16 compare the results of the proposed method with the existing approach in different temperature rates ($\tau=0$, 10, 30, and 50). Considering that α and γ to be 0, the time gap was calculated to be 4.08%, 4.73%, 6.31%, and 7.24%. We tested the cases for different learning rates by considering $\alpha=0.5$ and $\gamma=0$. In this case, the time gap was 56.62% (Table 6).

Case 2: As depicted in Section 5.2, the RTN graph of Case 2 shows 12 routes with 6 nodes (cities) which were used to generate a graph displaying the optimal path based on Google maps and the proposed method. Table 5 and Fig. 16 compare the results of the proposed method with the existing approach in different temperature rates ($\tau=0$, 10, 30, and 50). Considering that α and γ to be 0, the time gap was calculated to be 2.73%, 3.01%, 0.00%, and 0.00%. We tested the cases for different learning rates by considering $\alpha=0.5$ and $\gamma=0$. In this case, the time gap was 3.14% (Table 6).

Case 3: As depicted in Section 5.3, the RTN graph of Case 3 had 23 routes with 9 cities and it was used to generate a graph displaying the optimal path. Table 5 and Fig. 16 show the

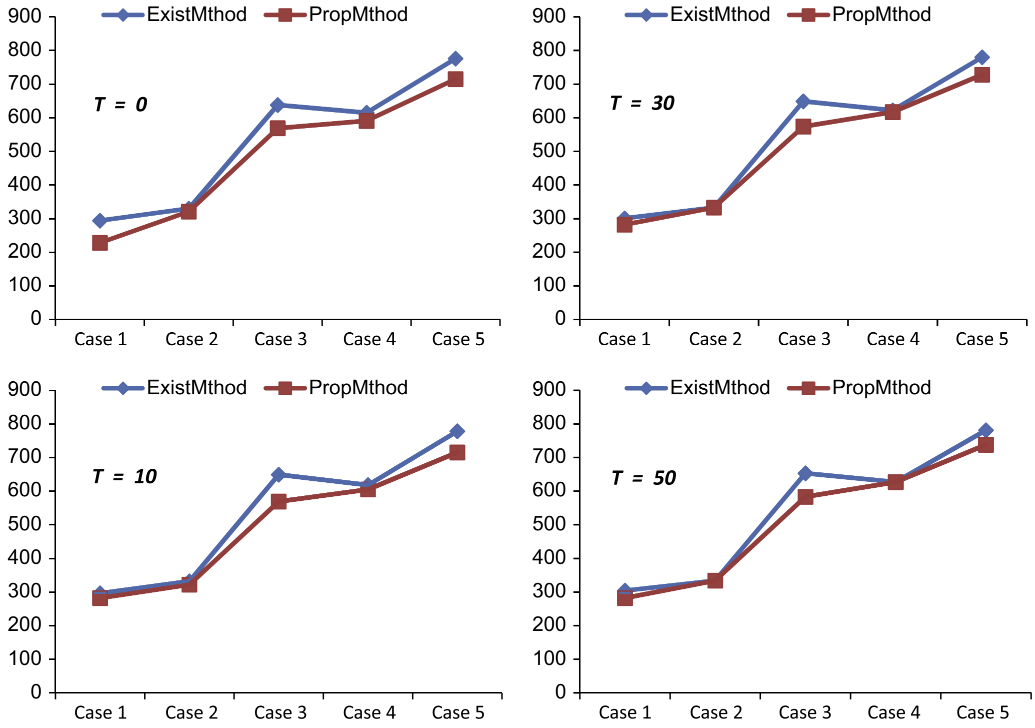

Fig. 15. The average travel time comparison among Cases 1, 2, 3, 4, and 5 (α and γ=0).

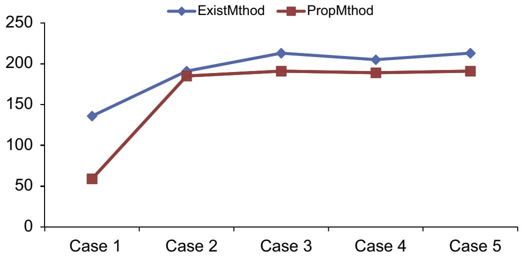

Fig. 16. The average travel time comparison among Cases 1, 2, 3, 4, and 5 (α=0.5 and γ=0).

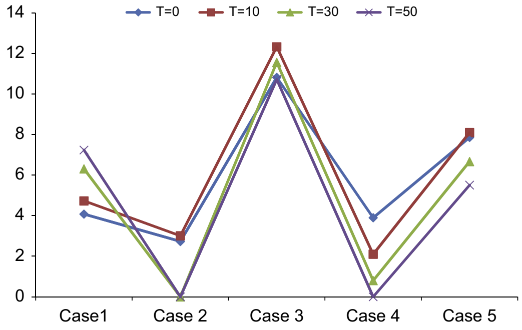

Fig. 17. The Gap comparison among Cases 1, 2, 3, 4, and 5 (α and γ=0).

comparison of the results of the proposed approach and the existing approach considering different temperature rates (τ=0, 10, 30, and 50), α and γ=0. The time gaps were 10.82%, 12.33%, 11.56%, and 10.72%. We tested the two policies in different learning rates, by considering α=0.5 andγ=0, resulting in a time gap of 10.33% (Table 6).

Case 4: As depicted in Section 5.4, the RTN graph of Case 4 had three routes with three cities and it was used to generate a graph displaying the optimal path. Table 5 and Fig. 16 show the comparison of the results of the proposed approach and the existing approach considering different temperature rates (τ=0, 10, 30, and 50), α and γ=0. The time gaps were 3.90%, 2.10%, 0.80%, and 0.00%. We tested the two policies in different learning rates, by considering α=0.5 and γ=0, resulting in a time gap of 20.40% (Table 6).

Case 5: As depicted in Section 5.5, the RTN graph of Case 5 had 31 routes and 10 cities (Table 5). Table 5 and Fig. 16 show that when different temperature rates (τ=0, 10, 30, and 50), α and γ=0 were considered, the time gaps between the results of proposed approach and existing approach were 7.86%, 8.10%, 6.67%, and 5.51%. We tested the two policies in different learning rates, by considering α=0.5 and γ=0. Here the time gap was 10.33% (Table 6).

This study attempted to calculate or estimate real-world travel time in RPS using MARL for RTN to offer new solutions, including a new algorithm based on MARL and a new model for finding the optimal path in the RTN to conduct vehicles to their destination. A comparison of the results shows that the proposed method using MARL and Q-value based on real-world RTN reduced traffic congestion and improved road traffic systems compared to the greedy strategy method. However, to assess the MARL for the RPS method, simulated case studies were used (Figs. 15 and 16) and the performance of the two methods was compared. In all experimental cases, the proposed method result times were less than the result times achieved by the existing approach. It was also revealed that the existing approach based on Google maps data was not always realistic or accurate. The trip durations for the case studies predicted by Google maps were between 0.00 and 12.33% off from the actual travel times. As a result, a new RPS approach was developed, which used the proposed method.

7. Conclusion

In this study, the circumstances for applying agent-oriented techniques that focus on the use of RL methods for vehicle routing problem was presented for traffic networks. For this purpose, we presented a conceptual framework for route planning systems that would route vehicles based on MARL. This framework identified the various components of the issue, by calculating traffic routes using a number of agents in a static network situation and extended all this to a real dynamic network in Malaysia. The important achievement of the study was to resolve the RPS problems using simulation methods and MASs with learning abilities, in order to make decisions about routing vehicles between Malaysia′s cities. This study presented a new paradigm that included new RPSA and Q-values based on MARL for finding the optimal path to reduce traffic congestion and conduct the vehicles to their destinations in the RTN. It also introduced a conceptual model of RPS using MARL in the RTN as well as showing that agent learning technology can optimize RPS for RTN by reviewing agent applications for RTN optimization. Illustrating how a MARL can optimize the performance and demonstrating how a MARL is a coupled network of software learning agents that interact to solve RTN problems beyond the knowledge of each individual problem solving component were two further achievements obtained by this study. This research has also demonstrated that agent technology is suitable for solving communication concerns in a distributed RTN environment. The novelty of this study is the use of MARL for RPS, which can be employed by RTN in Malaysia to offer access to RTN data resources. MARL attempted to solve RTN problems by collaborating between agents, resulting in answers to complex RTN problems. In this study, each agent performed a special function of the RTN and shared its knowledge with other agents. Given the above described results, our contributions are as follows:

1. The research work modeled a new route planning system based on QVDP with the MARL algorithm in order to reduce vehicle trip times and costs by giving a priority trip plan to vehicles.
2. The research uses the ability of learning models to propose several route alternatives to reduce time and minimize travel costs during driving.
3. The paper is important for vehicle trip planning by deploying the MAS to predict the road and environmental conditions.
4. The study provides high quality travel planning service to the vehicle drivers.
5. The paper had results with enough sizes and dimensions that included three important issues (RPS, MARL, and RTN) in Computer Science.

References

Adler, J.L., Blue, V.J., 2002. A cooperative multi-agent transportation management and route guidance system. J. Transp. Res. Part C Emerg. Technol. 10 (5–6), 433–454.

Adler, J., Satapathy, G., Manikonda, V., Bowles, B., Blue, V., 2005. A multi-agent approach to cooperative traffic management and route guidance. Transp. Res. Part B Methodol. 39 (4), 297–318.

Akchurina, N., 2010. Multi-Agent Reinforcement Learning Algorithms (Ph.D. Thesis). University of Paderborn, pp. 1–182.

Almejalli, K., Dahal, K., Hossain, A., 2009. An intelligent multi-agent approach for road traffic management systems. In: 2009 IEEE International Conference on Control Applications/International Symposium on Intelligent Control, vol. 1–3, July 08–10, Petersburg, Russia, pp. 825–830.

Arel, I., Liu, C., Urbanik, T., Kohls, A.G., 2010. Reinforcement learning based multi-agent system for network traffic signal control. Intell. Transp. Syst. 4, 128–135.

Awad, E., 2011. Learning to Share: A Study of Multi-agent Learning in Transportation Systems (Master′s Thesis). Computing and Information Science Program, Masdar Institute of Science and Technology.

Bakker, B., Kester, L., 2006. DOAS 2006 Project: Reinforcement Learning of Traffic Light Controllers Adapting to Accidents.

Bakker, B., Steingrover, M., Schouten, R., Nijhuis, E., Kester, L., 2005. Cooperative multiagent reinforcement learning of traffic lights. In: 16th European Conference on Machine Learning (ECML-05) on Workshop on Cooperative Multi-Agent Learning, Porto, Portugal.

Balaji, P.G., Sachdeva, G., Srinivasan, D., Tham, C.K., 2007. Multi-agent system based urban traffic management. In: 2007 IEEE Congress on Evolutionary Computation, Sep 25–28, Singapore, Singapore, vol. 1–10, pp. 1740–1747.

Barbucha, D., 2011. An agent-based guided local search for the capacited vehicle routing problem. In: Proceedings of the 5th KES International Conference on Agent and Multi-Agent Systems: Technologies and Applications, KES-AMSTA 11, Springer-Verlag, Berlin, Heidelberg, ISBN 978-3-642-21999-3, pp. 476–485.

Busoniu, L., De Schutter, B., Babuska, R., 2005. Learning and coordination in dynamic multiagent systems. Delft Center for Systems and Control, Delft University of Technology, no. 05-019, Delft, The Netherlands.

Bussoniu, L., Babuska, R., De Schutter, B., 2010. Multi-agent reinforcement learning: an overview. In: Innovations in Multi-Agent Systems and Applications—1. Series of Studies in Computational Intelligence, vol. 310, Springer, Berlin, Germany, pp. 183-22 (Chapter 7).

Camiel, E., 2011. Road Safety Strategic Plan 2008–2020. Ministry of Transport, Public Works and Water Management.

Chabrol, M., Sarramia, D., Tchernev, N., 2006. Urban traffic systems modelling methodology. Int. J. Prod. Econ. 99 (1–2), 156–176.

Zhou, Changfeng, Yan, L., Yuejin, T., Liangcai, L., 2006. Dynamic Vehicle Routing and Scheduling with Variable Travel Times in Intelligent Transportation System, pp. 8707–8711.

Chang-Qing, C., Zhao-Sheng, Y., 2007. Study urban traffic management based multi-agent system. In: Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, IEEE, Hong Kong, pp. 25–29.

Chen, B., Cheng, H., 2010. A review of the applications of agent technology in traffic and transportation systems. Trans. Intell. Transp. Syst. 11 (2), 485–497.

Chen, B., Cheng, H.H., Palen, J., 2009. Integrating mobile agent technology with multi-agent systems for distributed traffic detection and management systems. Transp. Res. Part C 17, 1–10.

Chowdhury, M.A., Sadek, A.W., 2003. Fundamentals of Intelligent Transportation Systems Planning. Boston, Artech House

Chuan-hua, Z., Qing-song, L., 2004. Route selecting in the light of the theory of multimode transportation based on multi-agent. In: 2004 IEEE International Confixence on Systems, Man and Cybernetics, vol. 4, pp. 4023–4027.

Claes, R., Holvoet, T., Weyns, D., 2011. A decentralized approach for anticipatory vehicle routing using delegate multiagent systems. IEEE Trans. Intell. Transp. Syst. 12 (2), 364–373.

Dangelmaier, W., Klopper, B., Rungerer, N., Aufenanger, M., 2008. Aspects of Agent Based Planning in the Demand Driven Railcab Scenario. Dynamics in Logistics, pp. 171–178.

Delling, D., 2009. Engineering and Augmenting Route Planning Algorithms (Ph.D. Thesis). University of Karlsruhe.

Demircan, S., Aydin, M., Durduran, S.S., 2008. Route optimization with Q-learning. In: Proceedings of the 8th Conference on Applied Computer Science (ACS08), World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA, pp. 416–419.

Dominik, B., Katharina, S., Axel, T., 2011. Multi-agent-based transport planning in the newspaper industry. Int. J. Prod. Econ. 131 (1), 146–157.

Dong, B., Li, K., Liao, M., Wang, H., 2007. The route choice model under the traffic information guide environment based on game theory. J. Beihua Univ. (Nat. Sci.) 8 (1), 88–91.

Ferdowsizadeh-Naeeni, A., 2004. Advanced Multi-Agent Fuzzy Reinforcement Learning (Master Thesis). Dalarna University, Sweden.

Flinsenberg, I., 2004. Route Planning Algorithms for Car Navigation (Ph.D. Thesis), Technische Universiteit Eindhoven.

Gehrke, J., Wojtusiak, J., 2008. Traffic prediction for agent route planning. In: Proceedings of the 8th International Conference on Computational Science, Part III, ICCS 08, vol. 5103, Springer-Verlag, Berlin, Heidelberg, ISBN 978-3-540-69388-8, pp. 692–701.

Geisberger, R., 2011. Advanced Route Planning in Transportation Networks (Ph.D. thesis). Karlsruhe Institute of Technology, Karlsruhe, pp. 1–227.

Isabel, M., Vicente, R.T., Garcia, L.A., Martinez, J.J., 2009. A Rule-Based Multi-agent System for Local Traffic Management. Intelligent Data Engineering and Automated Learning, vol. 5788. Burgos, Spain, pp. 502–509.

Jie, L., Meng-yin, F., 2003. Research on route planning and map-matching in vehicle GPS/dead-reckoning/electronic map integrated navigation system. IEEE Intell. Transp. Syst. 2, 1639–1643.

Ji, M., Yu, X., Yong, Y., Nan, X., Yu, W., 2012. Collision-avoiding aware routing based on real-time hybrid traffic informations. J. Adv. Mater. Res. 396–398, 2511–2514.

Jones, K., 2010. A Trust Based Approach to Mobile Multi-Agent System Security (Ph. D. Thesis). The Faculty of Technology, De Montfort University, Leicester.

Khanjary, M., Hashemi, S., 2012. Route Guidance Systems: Review and Classification. In: 6th IEEE/ACM Euro American Conference on Telematics and Information Systems (EATIS), May, ACM, New York, NY, USA, pp. 269–275.

Kiam, T.S., Lee, D., 2010. Performance of multiagent taxi dispatch on extended-runtime taxi availability: a simulation study. IEEE Trans. Intell. Transp. Syst. 11 (March), 231–236.

Kosicek, M., Tesar, R., Darena, F., Malo, R., Motycka, A., 2012. Route planning module as a part of supply chain management system. Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis, vol. LX (2), pp. 135–142.

Kovalchuk, Y., 2009. A Multi-Agent Decision Support System for Supply Chain Management (Ph.D. Thesis). School of Computer Science and Electronic Engineering, University of Essex, July.

Kuyer, L., Whiteson, S., Bakker, B., Vlassis, N., 2008. Multiagent reinforcement learning for urban traffic control using coordination graphs. In: Proceedings of the 19th European Conference on Machine Learning, pp. 656–671.

Lauer, M., Riedmiller, M., 2000. An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In: Proceedings 17th International Conference on Machine Learning (ICML-00), Stanford University, US, pp. 535–542.

Lecce, D., Amato, V., 2008. Multi agent negotiation for a decision support system in route planning. In: Proceedings of the International Conference on Computational Intelligence for Modelling Control & Automation, pp. 458–463.

Ligtenberg, A., 2006. Exploring the Use of Multi-Agent Systems for Interactive Multi-Actor Spatial Planning (Ph.D. Thesis). Wageningen University.

Lucian, B., Robert, B., Bart, D., Damien, E., 2010. Reinforcement Learning and Dynamic Programming Using Function Approximators. CRC Press, Boca Raton, Florida.

Marcelo, C., Cristina, B., Aura, C., 2010. A Multi-agent System for Dynamic Path Planning. In: BWSS Brazilian Workshop on Social Simulation-SBC, Joint Conference of XX Brazilian Symposium on Artificial Intelligence-SBIA. FEI University Campus, Sao Bernardo do Campo, Brazil.

Michael, W., 2009. An Introduction to MultiAgent Systems. May, second ed. John Wiley and Sons. ISBN-13: 978-0470519462.

Negenborn, R., De Schutter, B., Hellendoorn, J., 2008a. Multi-Agent Model Predictive Control For Transportation Networks: Serial Versus Parallel Schemes. Engineering Applications of Artificial Intelligence. Elsevier 21(3), pp. 353–366.

Paternina-Arboleda, C.D., Das, T.K., 2005. A multi-agent reinforcement learning approach to obtaining dynamic control policies for stochastic lot scheduling problem. Simul. Model. Pract. Theory 13 (5), 389–406.

Pellazar, M.S., 1998. Vehicle route planning with constraints using genetic algorithms. In: Proceedings of the IEEE 1998 National Aerospace and Electronics Conference (NAECON), 13–17 July, pp. 392–399.

Riedmiller, M.A., Moore, A.W., Schneider, J.G., 2000. Reinforcement learning for cooperating and communicating reactive agents in electrical power grids. In: Hannebauer, M., Wendler, J., Pagello, E. (Eds.), Balancing Reactivity and Social Deliberation in Multi-Agent Systems. Springer, pp. 137–149

Robu, V., Noot, H., Poutre, H.L., van Schijndel, W.J., 2011. A multi-agent platform for auction-based allocation of loads in transportation logistics. Expert Syst. Appl. 38 (4), 3483–3491.

Rothkrantz, L., 2009. Dynamic routing using the network of car drivers. In: EATIS 09: Proceedings of the 2009 Euro American Conference on Telematics and Information Systems, New York, NY, USA, ACM, ISBN 978-1-60558-398-3, pp. 1–8.

Ruimin, L., Qixin, S., 2003. Study on integration of urban traffic control and route guidance based on multi-agent technology. IEEE Intell. Transp. Syst. 2, 1740–1744.

Ryan, J., Donald, C., 2008. Divide and Conquer Evolutionary TSP Solution for Vehicle Path Planning. In: Proceedings of IEEE Congress on Evolutionary Computation. Hong Kong, June 2008, EC0172, pp. 676–681.

Schmitt, E., Jula, H., 2006. Vehicle route guidance systems: classification and comparison. In: Intelligent Transportation Systems Conference, pp. 242–247.

Schultes, D., 2008. Route Planning in Road Networks (Ph.D. Thesis). Fakultat fur Informatik, Universitat Karlsruhe (TH).

Seow, K., Dang, N., Lee, D., 2007. Towards an Automated Multiagent Taxi-Dispatch System. In: IEEE International Conference, 2007, pp. 1045–1050.

Shanqing, Y., Jing, Z., Bing, L., Mabu, S., Hirasawa, K., 2012. Q value-based dynamic programming with SARSA learning for real time route guidance in large scale road networks. In: The 2012 International Joint Conference on Neural Networks (IJCNN), vol. 10–15, pp. 1–7.

Shi, A., Na, C., Chun-b., H., 2008. Simulation and analysis of route guidance strategy based on a multi-agent-game approach. In: Management Science and Engineering, September 2008, pp. 140–146.

Shimohara, X., Hu, J., Yangsheng, X., Song, J., 2005. A simulation study on agent-network based route guidance system. In: 2005 IEEE Intelligent Transportation Systems Conference (ITSC).

Shoham, Y., Leyton-Brown, K., 2008. Multiagent Systems: Algorithmic, Game Theoretic and Logical Foundations. Cambridge University Press 2008, ISBN 978-0-521-89943-7, pp. I-XX, 1–483.

Srinivasan, S., Kumar, D., Jaglan, V., 2010. Multi-agent system supply chain management in steel pipe manufacturing. Int. J. Comput. Sci. Issues 7 (4), 30–34.

Stephan, W., Yunhui, W., 2008. Towards a conceptual model of talking to a route planner. In: The 8th International Symposium on Web and Wireless Geographical Information Systems (W2GIS 08), Springer-Verlag, Berlin, Heidelberg, pp. 107–123.

Suzuki, N., Araki, D., Higashide, A., Suzuki, T., 1995. Geographical route planning based on uncertain knowledge. In: Proceedings of Seventh International Conference on Tools with Artificial Intelligence, 5–8 November, pp. 434–441.

Tampere, C., Immers, B., Stada, J., Janssens, B., 2008. Multi-agent control in road Traffic management. In: 7th International Conference Environmental Engineering, vol. 1–3, Lithuania, pp. 1052–1061.

Tesauro, G., Jong, N.K., Das, R., Bennani, N., 2006. A hybrid reinforcement learning approach to autonomic resource allocation. In: ICAC 06, Dublin, Ireland, pp. 65–73.

Torrey, L., 2010. Crowd simulation via multi-agent reinforcement learning. In: Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE), 2010, Stanford, California, USA, The AAAI Press.

Tu, H., 2008. Monitoring Travel Time Reliability on Freeways. Transportation and Planning (Ph.D. Thesis), Delft University of Technology, Delft.

Uppin, M.S., 2010. Multi agent system model of supply chain for information sharing. Contemp. Eng. Sci. 3 (1), 1–16.

Vasan, A., Slobodan, P., 2010. Optimization of water distribution network design using differential evolution. J. Water Resour. Plan. Manag. 136 (2), 279–287.

Vasirani, M., Ossowski, S., 2009. A market-inspired approach to reservation based urban road traffic management. In: AAMAS 09: Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems, Richland, pp. 617–624.

Vlassis, N., 2007. A concise introduction to multiagent systems and distributed artificial intelligence. In: Synthesis Lectures in Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers.

Volf, P., Sislak, D., Pechoucek, M., 2011. Large-scale high-fidelity agent based simulation in air traffic domain. Cybern. Syst. 42, 502–525.

Watanabe, T., Lijiao, C., 2011. Vehicle routing based on traffic cost at intersection. In: Shea, J., Nguyen, N., Crockett, K., Howlett, R., Jain, L. (Eds.) Agent and Multi-Agent Systems: Technologies and Applications. Lecture Notes in Computer Science, vol. 6682. Springer, Berlin, Heidelberg, ISBN 978-3-642-21999-3, pp. 592–601.

Wedde, H., Lehnhoff, S., van Bonn, B., Bay, Z., Becker, S., Bottcher, S., Brunner, C., Buscher, A., Furst, T., Lazarescu, A., Rotaru, E., Senge, S., Steinbach, B., Yilmaz, F., Zimmermann, T., 2007. A novel class of multi-agent algorithms for highly dynamic transport planning inspired by honey bee behavior. In: IEEE Conference on Emerging Technologies and Factory Automation, IEEE, Greece, pp. 1157–1164.

Weyns, D., Holvoet, T., Helleboogh, A., 2007. Anticipatory vehicle routing using delegate multi-agent systems. In IEEE Intelligent Transportation System Conference, USA, pp. 87–93.

Wilson, A., Fern, A., Tadepalli, P., 2010. Bayesian role discovery for multi-agent reinforcement learning. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, vol. 1, pp. 1587–1588.

Wu, J., Jin, S., Ji, H., Srikanthan, T., 2011a. Algorithm for time-dependent shortest safe path on transportation networks. Procedia CS 4, 958–966.

Yu, D., Yang, Z., Wang, Y., Sun, J., 2006. Urban road traffic control system and its coordinate optimization based on multi-agent system. J. Jilin Univ. (Eng. Technol. Ed.) 36 (1), 113–118.

Yunhui, W., Stephan, W., 2007. Agent Behaviour in Peer-to-Peer Shared Ride Trip Planning (Master′s Thesis). Department of Geomatics, The University of Melbourne, Australia.

Zafar, K., Baig, A.R., 2012. Optimization of route planning and exploration using multi agent system. Multimed. Tools Appl. 56 (2), 245–265.

Zegeye, K.G., 2010. A Dynamic Prediction of Travel Time for Transit Vehicles in Brazil Using GPS Data. Department of Civil Engineering and Management, University of Twente, Enschede, The Netherlands.

Zi, Z., Jian-Min, X., 2005. A dynamic route guidance arithmetic based on reinforcement learning. In: Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, pp. 3607–3611.

Zolfpour-Arokhlo, M., Selamat, A., Mohd-Hashim, S.Z., 2011. Multi-agent reinforcement learning for route guidance system. Int. J. Adv. Comput. Technol. 3 (6), 224–232.

Zou, L., Xu, J., Zhu, L., 2007. Application of genetic algorithm in dynamic route guidance system. J. Transp. Syst. Eng. Inf. Technol. 7 (3), 45–48.