Available online www.jsaer.com

***Journal of Scientific and Engineering Research*, 2019, 6(10):136-148**

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

INTELLIGENT OPTIMAL ROUTE SELECTION USING GAIA AND JADE

BALE Dennis L.T.[1], Ugwu C.[2], Nwachukwu E.O.[1]

[1]Department of Computer Science, Ken Saro Wiwa Polytechnic, Bori, Rivers - Nigeria
[2]Department of Computer Science, University of Port Harcourt, Nigeria

Abstract There has been increasing cases of traffic congestion in Nigeria due to bad roads and lack of proper information management system for road users on the right route to take to avert the congestion problems. These have result to loss of man hour to work. This paper focuses on the development of an Intelligent Optimal Route Selection System using Gaia (Agent oriented software development methodology) and JADE (Java Agent Development Environment) to provide information to road users on which route to follow to avoid congestion and save time in cause of travelling by road.

Keywords Gaia, Jade, Traffic Congestion, Optimization, Route Selection

1. Introduction

Presently, there are serious demands for service delivery in economy activities especially in Nigeria where road transport system is still in its developing stage. The poor road network, poor implementation of existing transport policies and infrastructure development affects productivity negatively as more time and resources are spent on traffic deadlocks.

Inspection of the current road network system in the Port Harcourt city shows that some of the roads are usable but there is heavy traffic even at the 4-way intersections (junctions), while others are in poor state making movement on such roads difficult.

Heavy traffic build up gradually especially at pick periods thereby making movement on the roads a tedious task. In an effort to avert these problem road users usually divert to a less traffic routes without prior knowledge of the traffic situation of such routes. Sometimes they get trapped in a more serious deadlock than the previous one.

Following shortest known path to a destination do not usually yield success in Port Harcourt metropolis because of traffic and road status. Osigwe, et al [1] and Ugwu and Bale [2] proffer solution to traffic control problem at four way intersection. Creation of by-pass roads are also used to minimize congestions on roads but this introduces another problem of choice and familiarity.

The traffic control measures do not ameliorate the pains and suffering of road users because the traffic lights fail, traffic wardens leave their duty posts and phone-in programmes on radio have timing. There is presently no known intelligent route optimizer and traffic control centre in Port Harcourt city to aid the road users in their route choices. People are forced to use any known route irrespective of what will befall them on the way.

Hence, an intelligent solution is required to aid road user’s to take decision on which route to ply in other to improve arrival time, enhance productivity and service delivery. This intelligent decision is made possible by the provision of a system with intelligent Agent capability that can process all available dynamic information accurately and timely.

2. Related Literature

The dynamic nature of road transport system and its resultant effect on economic growth have attracted several researchers to explore route optimization in-view to solve the problem. Route optimizations (which is the selection of the best route among alternative routes based on certain criteria) have been applied in several areas involving routing problems; ranging from vehicle routing, travelling salesman to mail delivery problems [3-6].

Several researchers have made effort to solve problem of optimal route selection with the development of several optimization techniques like Dijkstra's Algorithm, A* search Algorithm, ALT (A* search Landmarks and Triangle inequality), Arc Flags, Contraction Hierarchies, etc Qureshi et al, [7-10]. These techniques do not incorporate any form of intelligence and thus may not be suitable for present route optimization problem solving.

In order to introduce intelligence into route optimization, soft computing techniques and Agent computing were applied in solving route optimization problems. Bale et al, [2]. Table 1 show related literatures which serve as background to the research.

Table 1: Summarized related works

Researcher	Techniques Used	Comments
Qureshi et al [7]	Weight based method and Fuzzy Logic based mechanism	Optimized route selection based on distance. Traffic events at peak period not considered.
Murat and Uludag [11]	Fuzzy Logic model (FLM) and Logistic Regression Model (LRM)	Compared both LRM and FLM using survey-based data in optimal route selection. Don not considers heavy traffic scenarios.
Salehinejad and Talebi [12]	Fuzzy Logic (FL), Any Colony and Artificial Neural Network (ANN)	Multi-parameter route selection system based on static data from TCC. Not dynamic as a result of static information.
Kammoun et al [13]	Multi Agents (MAS)	Simulate both route choice and lane change but do not consider multi-scenarios and occurrences.

3. Analysis of Intelligent Optimal Route Selection System Based on Gaia

This paper consider the use of Agent computing [14] where computing agents are used to monitor, evaluate and report traffic situations on road network for intelligent selection of route to ply based on multi-criteria. The road network are viewed in segments with agents having different roles and visibility area as shown in fig 1. The number of agents, their roles and instances are model using Gaia methodology. Gaia create an efficient series of steps, recognizable set of models, and relationship among the models, indicating when and how to utilize each model and abstraction in the building of a MAS – Multi Agents System [15].

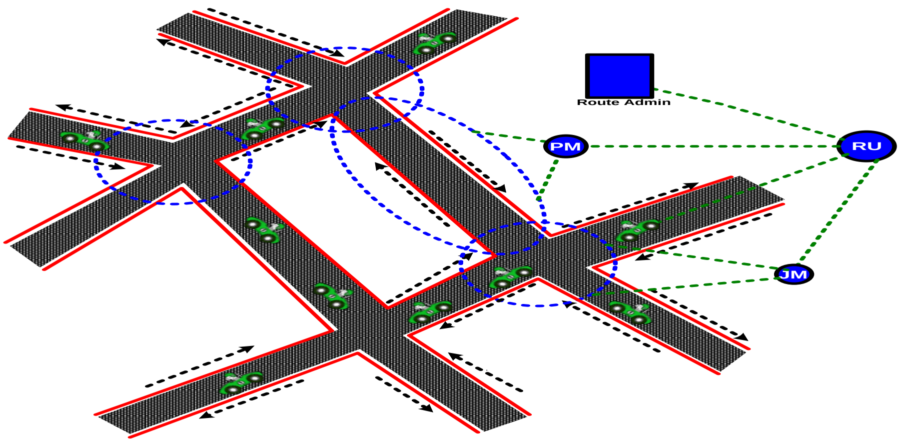

Figure 1: Road network segment and Agents Visibilities

A. Requirement Statements

The Intelligent route selection system requirement statements are:

1. User can request to move from one location to another through a valid route.

2. The system should present to the user the possible routes and suggest an optimal route based on traffic situations of the different possible routes.
3. The system monitors the routes including junctions and inform the user if there is any change in traffic situation and suggests diversion where necessary, if there are alternatives.

Steps 2 and 3 are repeated at every junction until the destination is reached.

B. Analysis Phase

From the requirement statements drawn out, four roles are noticeable for the analysis phase of the system namely; JunctionMonitor (JM), RouteUser (RU), PathMonitor (PM) and RouteAdmin (RA).

1. JunctionMonitor (JM) – Is situated at major junctions to monitor traffic situations at the junction and report to the requesting agents.
2. RouteUser (RU) – Seek to know traffic situations on its routes as provided and select optimal route from the feasible routes provided based on the traffic information gather.
3. PathMonitor (PM) – Monitor traffic on the various routes or paths that leads to the major junction and report same to the requesting agent.

RouteAdmin (RA) – Oversee the entire road network and provide possible routes based on the origin and destination (O/D) of the requesting agents.

1. Role Model

Each agent has one or two roles. The roles of the agents defined above are model in the role schema as shown in fig 2 to fig 5.

Role Schema: JUNCTIONMONITOR
Description: This role monitors traffic situation at the major junction and informs the RouterUser
Protocols and Activities: <u>CheckJunctionTraffic</u>, InformRouteUser, AwaitRequest
Permissions: reads RouteUser data structure trafficSituation
Responsibilities: Liveness: JUNCTIONMONITOR = (JTraffic) $^{\omega}$ JTRAFFIC = <u>CheckJunctionTraffic</u>. InformRouteUser. AwaitRequest Safety: • Sensor must be active

Figure 2: Schema for role JUNCTIONMONITOR (JM)

Figure 2 defined the role of JunctionMonitor agent, which is to report traffic situation at the junction. Detail explanations are;

1. **Protocol and Activities** for this role are;
   - i CheckJunctionTraffic – this is an activity which is to monitor junction traffic events.
   - ii InformRouteUser – this is a protocol which report the information gathered to the requesting agent.
   - iii AwaitRequest – this is also a protocol, which check to receive request from other agents.
2. **Permissions** for this role;
   - i reads RouteUser data structure – this is to check the routes of the user to determine whether it will pass through the junction.
   - ii reads trafficSituation – this is to access the traffic value or data of the junction in order to determine traffic events at the junction.
3. **Responsibilities** of the role are functions or actions to be taken in the correct order of execution. They are;

i **Liveness** – JunctionMonitor is assigned $(JTraffic)^{\omega}$ which occurs most often endlessly as indicated by $^{\omega}$. The JTraffic consist of Checkjunction. InformRouteUser. AwaitRequest, which execute in this order because of the period (.). The period (.) indicates that the first executes before the second and second before the third.

ii **Safety** – this has to do with the precaution taken to ensure accurate result. In this case, the sensor must be active.

Role Schema: ROUTEUSER
Description: Act on behalf of user, it announces users origin and destination (OD). Get available feasible routes in respect of OD and select optimal route based on traffic information on those routes.
Protocols and Activities: <u>SendUserOD</u>, RequestFeasibleRoutes, ReceiveRoutes, QueriesPM, QueriesJM, <u>SelectOptimalRoute</u>.
Permissions: reads, update user data structure reads PathStatus JunctionStatus
Responsibilities: Liveness: ROUTEUSER = <u>SendUserOD</u>$^{+}$. $(SeekRoute)^{\omega}$. $(QTraffic)^{\omega}$. <u>SelectOptimalRoute</u>$^{\omega}$ SEEKROUTE = RequestFeasibleRoutes.ReceiveRoutes QTRAFFIC = QueriesPM||QueriesJM Safety: • OD must not be null

Figure 3: Schema for role ROUTEUSER (RU)

Figure 3 defined the role of RouteUser agent, which acts on behalf of user; it announces user's origin and destination (O/D). Get available feasible routes in respect of O/D and select optimal route based on traffic information on those routes. Detail explanations are;

1. **Protocol and Activities** for this role are;

i SendUserOD – this is an activity which send user's origin and destination (O/D) to directory facilitator (df).

ii RequestFeasibleRoutes – this is a protocol which seek to know the feasible routes between O/D on the road map.

iii ReceiveRoutes – this is also a protocol, which receive feasible routes between O/D from RouteAdmin agent.

iv QueriesPM – this protocol asked for traffic information and status on all the paths connecting between O/D.

v QueriesJM - this protocol asked for traffic information at all the junctions on the feasible routes between O/D.

vi SelectOptimalRoute - this is an activity which determine the optimal route based on traffic information gathered about all the individual routes that lead to destination, from other agents.

2. **Permissions** for this role;

i reads, update user data structure – this is to access the origin and destination of the user, make changes to it where necessary.

ii reads PathStatus – this is to access the traffic value or data and state of the various paths.

iii reads JunctionStatus – this is to access the traffic value or data of the various junctions.

3. **Responsibilities** of the role are functions or actions to be taken in the correct order of execution. They are;

i **Liveness** – ROUTEUSER is assigned SendUserOD^{+}. (SeekRoute)$^{\omega}$. (QTraffic) $^{\omega}$. SelectOptimalRoute $^{\omega}$. The SendUserOD^{+} means that SendUserOD must execute at least ones and it must be executed before SeekRoute, QTraffic and SelectOptimalRoute in that order. The SeekRoute consist of RequestFeasibleRoutes.ReceiveRoutes meaning that RequestFeasibleRoutes must occur before ReceiveRoutes. QTraffic is expanded as QueriesPM||QueriesJM which means that both can interleave.

ii **Safety** – this has to do with the precaution taken to ensure accurate result. In this case, O/D must not be null.

Role Schema: ROUTEADMIN
Description: Is aware of the entire road network (map) and provide feasible routes to requesting agent based on agent OD.
Protocols and Activities: ReceiveRouteUserOD, <u>CheckRoutes</u>, SendFeasibleRoutes
Permissions: reads RoadNetwork RouteUser data structure create FeasibleRoutes
Responsibilities: Liveness: ROUTEADMIN = ReceiveRouteUserOD^{+}. <u>CheckRoutes</u>$^{\omega}$. SendFeasibleRoutes$^{\omega}$ Safety: • RouteUserOD must not be nil

Figure 4: Schema for role ROUTEADMIN (RA)

Figure 4 defined the role of RouteAdmin agent, which is aware of the entire road network (map) and provide feasible routes to requesting agent based on agent's O/D. Detail explanations are;

1. **Protocol and Activities** for this role are;

i ReceiveRouteUserOD – this is a protocol which receives origin and destination (O/D) from RouteUser agent.

ii CheckRoutes – this is an activity that checks the road map to determine feasible routes between O/D on the road map.

iii SendFeasibleRoutes – this is also a protocol, which sends feasible routes between O/D to RouteUser agent.

2. **Permissions** for this role;

i reads RoadMap – this is to access the digital road map of the metropolis.

ii reads RouteUser data structure – this is to determine O/D of a particular RouteUser.

iii create FeasibleRoutes – this is to generate feasible routes for a particular O/D.

3. **Responsibilities** of the role are functions or actions to be taken in the correct order of execution. They are;

i **Liveness** – ROUTEADMIN is assigned ReceiveRouteUserOD^{+}. CheckRoutes$^{\omega}$.SendFeasibleRoutes$^{\omega}$ this means that ReceiveRouteUserOD occurs at least ones before CheckRoutes$^{\omega}$ and SendFeasibleRoutes$^{\omega}$ can occur in sequences. But both CheckRoutes and SendFeasibleRoutes can occur often endlessly.

ii **Safety** – this has to do with the precaution taken to ensure accurate result. In this case, RouteUserOD must not be nil.

Role Schema: PATHMONITOR
Description: Monitors the traffic status of the path between two major junctions and road situation. It receives request for path status from RouteUser and report situation.

Protocols and Activities: AwaitRequest, CheckPathTrafficStatus, CheckPathSituationStatus, SendPathStatus
Permissions: reads, update its data structure reads RouteUser data structure
Responsibilities: Liveness: PATHMONITOR = AwaitRequest $^{\omega}$. (TStatus)$^{\omega}$. SendPathStatus$^{\omega}$ TSTATUS = CheckPathTrafficStatus || CheckPathSituationStatus Safety: • RouteUserRequest must not be nil

Figure 5: Schema for role PATHMONITOR (PM)

Figure 5 defined the role of PathMonitor agent, which is to observe path traffic and road status, report such to the requesting agent. Detail explanations are;

1. **Protocol and Activities** for this role are;
   - i AwaitRequest – this is a protocol that waits for request from other agents.
   - ii CheckPathTrafficStatus – this is an activity responsible for providing traffic information on the path.
   - iii CheckPathSituationStatus – this is also an activity, whose duty is to observe the state of the paths.
   - iv SendPathStatus – this protocol reply the requesting agent with information concerning traffic events and state of the road.
2. **Permissions** for this role;
   - i reads, update its data structure – this is to check and change its traffic information on the paths.
   - ii reads RouteUser data structure – this is to access the paths relevant to a particular set of routes.
3. **Responsibilities** of the role are functions or actions to be taken in the correct order of execution. They are;
   - i **Liveness** – PATHMONITOR is assigned AwaitRequest $^{\omega}$. (TStatus)$^{\omega}$. SendPathStatus$^{\omega}$ which occurs in this order.
     TSTATUS = CheckPathTrafficStatus ||CheckPathSituationStatus
   - ii **Safety** – this has to do with the precaution taken to ensure accurate result. In this case, RouteUserRequest must not be nil.

2. Interaction Model

This model shows the communication or links among the different roles. The protocol definitions are used to model the interaction between roles. Here we present some protocols of the system in fig. 6 to fig. 8.

Figure 6: Protocol related to JunctionMonitor Role

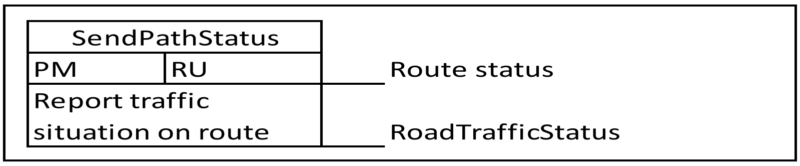

Figure 7: Protocol related to PathMonitor Role

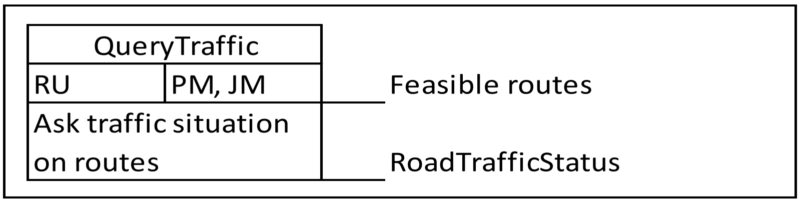

Figure 8: Protocol related to RouteUser Role

The explanations of protocol definition as presented in figure 6 are

1. *Purpose*: Inform
2. *Initiator*: JunctionMonitor agent (JM)
3. *Responder*: RouteUser agent (RU)
4. *Input*: Traffic at the junctions
5. *Output*: Junction traffic status update
6. *Processing*: Inform RU of traffic events at the junctions.

The explanations of protocol definition as presented in figure 7 are

1. *Purpose*: Send path status
2. *Initiator*: PathMonitor agent (PM)
3. *Responder*: RouteUser agent (RU)
4. *Input*: Route status
5. *Output*: Path traffic status report
6. *Processing*: Report traffic situation on route.

The explanations of protocol definition as presented in figure 8 are

1. *Purpose*: Query Traffic
2. *Initiator*: RouteUser agent
3. *Responder*: PathMonitor and JunctionMonitor agents
4. *Input*: Feasible routes
5. *Output*: Routes traffic status report
6. *Processing*: Seek traffic situation report on routes.

C. Design Phase

In this phase, the abstract entities in the role and interaction models of the analysis phase are converted into concrete entities the system. The Agent and Service model are presented here;

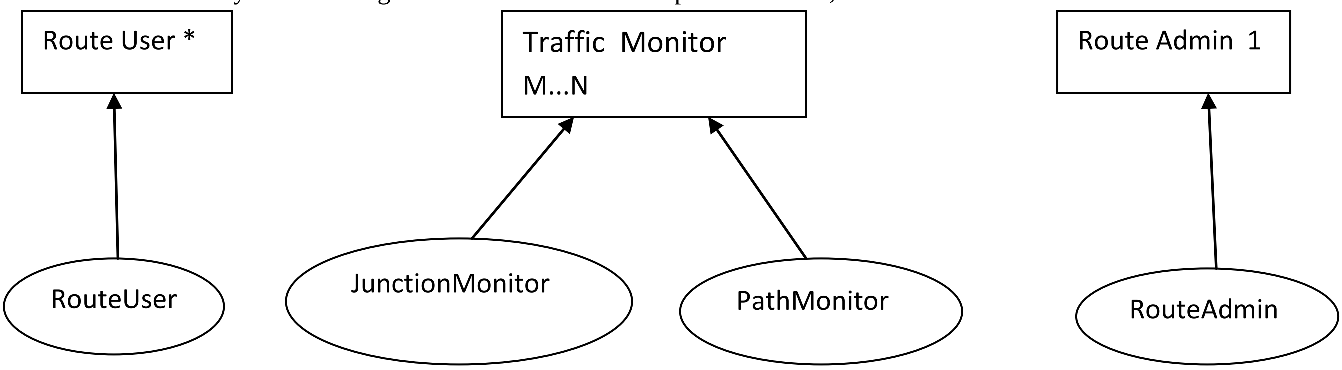

Figure 9: Agent Type Model

1. Agent Model

In our system we identify three agent types from the agent roles presented in section B, they are Traffic Monitor, Route User and Route Admin agent type. Fig 9 shows the Agent Types, their roles mapping and instances. Route User Agent takes on the role RouteUser (RU) and there can be zero or more instances of it. Traffic

Monitor Agent takes on the roles JunctionMonitor (JM) and PathMonitor (PM) and there can be between *m* and *n* instances. Route Admin Agent takes on the role RouteAdmin (RA) and there can be exactly 1 instance.

2. Service Model

The services related to each agent role, and the indication of the core services properties are recognized in the Gaia services model. A function can be considered as a service. Service is basically a particular, logical block of activity which is engage by an agent. The inputs, pre-conditions, post-conditions and outputs of each service are usually documented in this model. Table 2 contain the service model of the system.

Table 2: The Service model

Service	Input	Output	Pre-condition	Post-condition
Get route	Origin and Destination (OD)	Feasible routes	OD not nil	Road map exist
Traffic Information	Traffic events on feasible routes	Traffic situation on routes	OD not nil	
Inform User	Traffic situation on routes	Selected Route	Traffic information not null	Valid route

3. Acquaintance Model

The model is viewed as a directed graph (showing interaction links that is present among agent types) wherein nodes correspond to agent types and arcs represent interaction pathways. It does not define what and when messages are sent but just indicate existence of pathways.

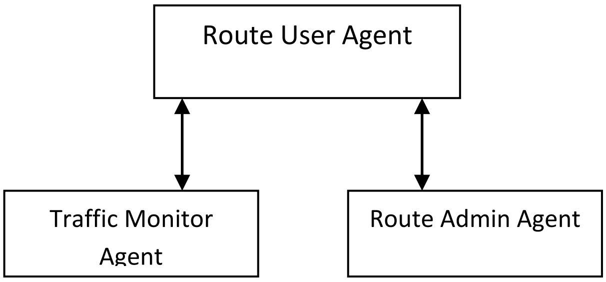

Figure 10: Acquaintances Model

From fig. 10 it is clear that two way interactions exist between Traffic Monitor Agent and Route User Agent, and also Route User Agent and Route Admin Agent. But there is no direct communication between Traffic Monitor Agent and Route Admin Agent. There are no potential communication bottlenecks as showed in fig 10.

4. Implementation

The designed intelligent optimal route selection system (IORS) is implemented with Jade. JADE is formed from the term Java Agent DEvelopment Framework which is a framework to build up multi-agent systems (MAS) in accordance with FIPA (Foundation for Intelligent Physical Agent) stipulations. It is a software platform fully implemented in Java which offers vital middleware-layer functionalities that are free of particular application. JADE made easy the actualization of the software agent abstraction in distributed applications and can be used to easily translate the Gaia design models into a MAS [16-17]. The interesting issues about JADE is that it shield the developer from detail expertise in agent theory by providing facilitates that require minimal effort in development of MAS.

The implemented intelligent optimal route selection system was tested over selected major road network in Port Harcourt (PH) metropolis. Fig 11 shows the road network map used to test the system. The sketch map was obtained from a road survey carried out during the study.

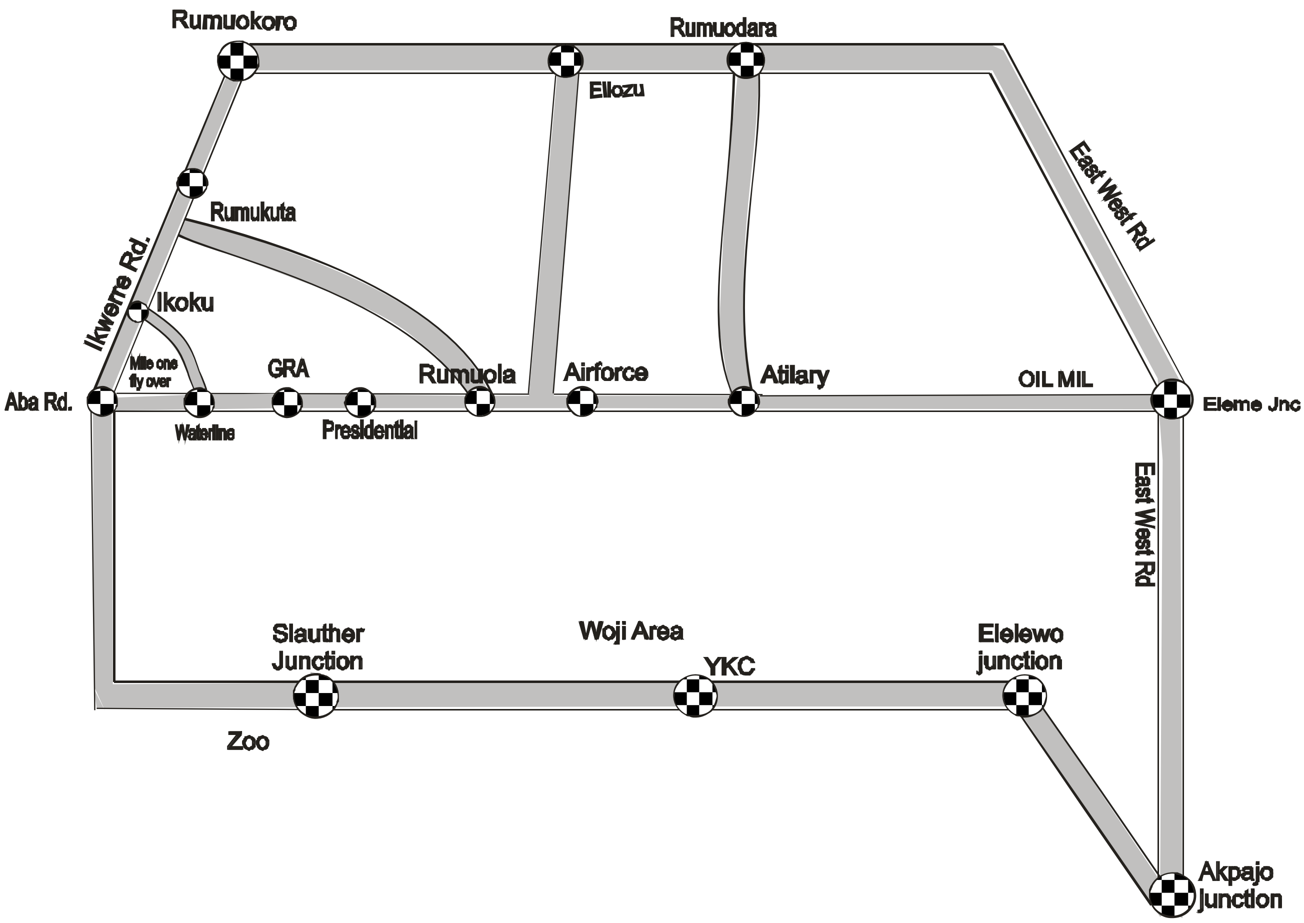

Figure 11: Selected Road Network in PH

The User sends origin and destination (O/D) through the RU to the RA and based on RA's reply of possible routes between O and D, RU sends traffic situation request to the relevant TrafficMonitor agents (JMs and PMs). The PMs and JMs in turn reply with traffic situation reports of their various paths and junctions respectively. Optimal route is obtained after evaluation of the information received. Table 3 shows various optimal routes as generated from the system at run time.

Table 3: Summarized Optimal Routes generated from the system

Instance	User	Origin	Destination	No of Possible Routes	Optimal Route	Optimal Route Description
1	1	Mile One	Eleme Junction	33	Route 7	Mileone/Ikoku/Rumuokuta/Rumuola/Airforce/Artilary/ElemeJunction
2	1	Mile One	Eleme Junction	33	Route 33	Mileone/Slaughter/YKC/Elelewo/Akpajo/ElemeJunction
3	2	Rumuokuta	Akpajo	26	Route 26	Rumuokuta/Rumuola/Airforce/Artilary/Elemejunction/Akpajo
4	2	Rumuokuta	Akpajo	26	Route 11	Rumuokuta/Rumuokoro/Eliozu/Rumuodara/Elemejunction/Akpajo
5	3	Waterline	Eliozu	30	Route 17	Waterline/Ikoku/Rumuokuta/Rumuokoro/Eliozu
6	3	Waterline	Eliozu	30	Route 26	Waterline/GRA/Presidential/Rumuola/Airforce/Eliozu
7	4	Mile One	Mile One	0	0	Nil
8	4	Presidential	Elelewo	34	Route 19	Presidential/Rumuola/Rumuokuta/Ikoku/MileOne/Slaughter/YKC/Elelewo
9	4	Presidential	Elelewo	34	Route 9	Presidential/GRA/Waterline/Mileone/

						Slaughter/YKC/Elelewo
10	1	Rumuokoro	YKC	32	Route 5	Rumuokoro/Rumuokuta/Ikoku/Mileone/Slaughter/YKC
11	3	Rumuokoro	YKC	32	Route 5	Rumuokoro/Rumuokuta/Ikoku/Mileone/Slaughter/YKC
12	2	Eleme Junction	GRA	32	Route 6	Elemejunction/Rumuodara/Eliozu/Airforce/Rumuola/Presidential/GRA

5. Results and Discussion

Table 3 show summarized Optimal Routes generated from the system when inputted with different Origin and Destination at different instances. Fig 12 and 13 display the interfaces of the system at runtime.

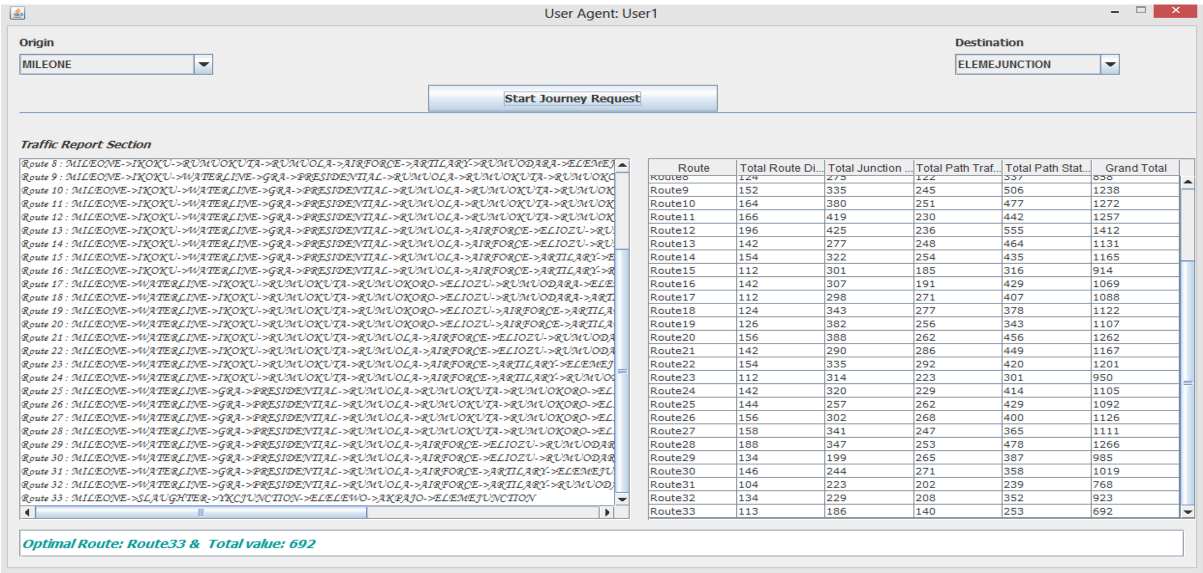

Figure 12: Screenshot UserAgent 1 showing O/D and Optimal Route 33

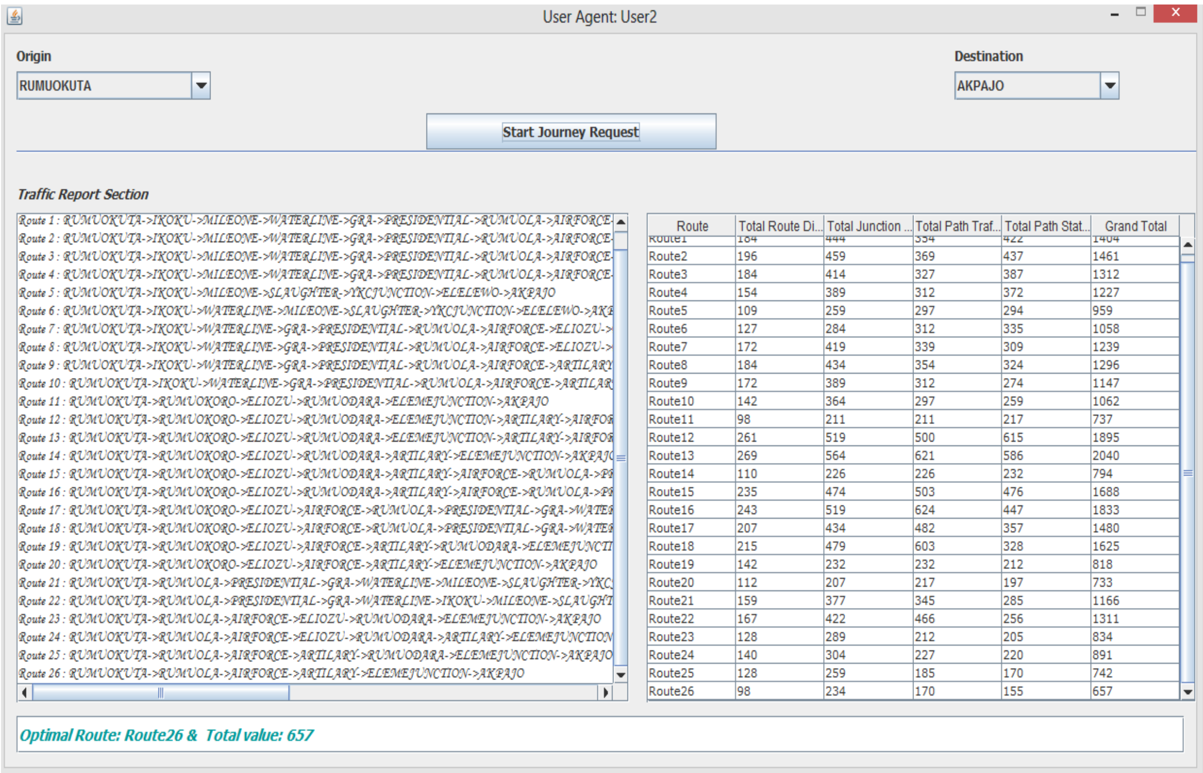

Figure 13: Screenshot UserAgent 2 showing O/D and Optimal Route 26

Different O/D was inputted into IORS at different instances the results obtained are presented in table 3. From table 3, instance 1 and 2 have the same User (User 1), origin, destination, number of possible routes but different optimal route. The change in the optimal route is caused by the traffic information received as at that instance. When both O/D are the same that is you are at your destination in real sense, the system display route as nil as indicated in instance 7 of table 3. Instances 10 and 11 generated by User 1 and 3 route request, produces the same route because O/D and traffic information of both instances are the same. This means that during the query period there is no remarkable change in traffic information.

The system developed in this paper was compared with other route optimization techniques like Shortest path and Ant Colony algorithm. Fig 14 and fig 15 display the result of the comparison.

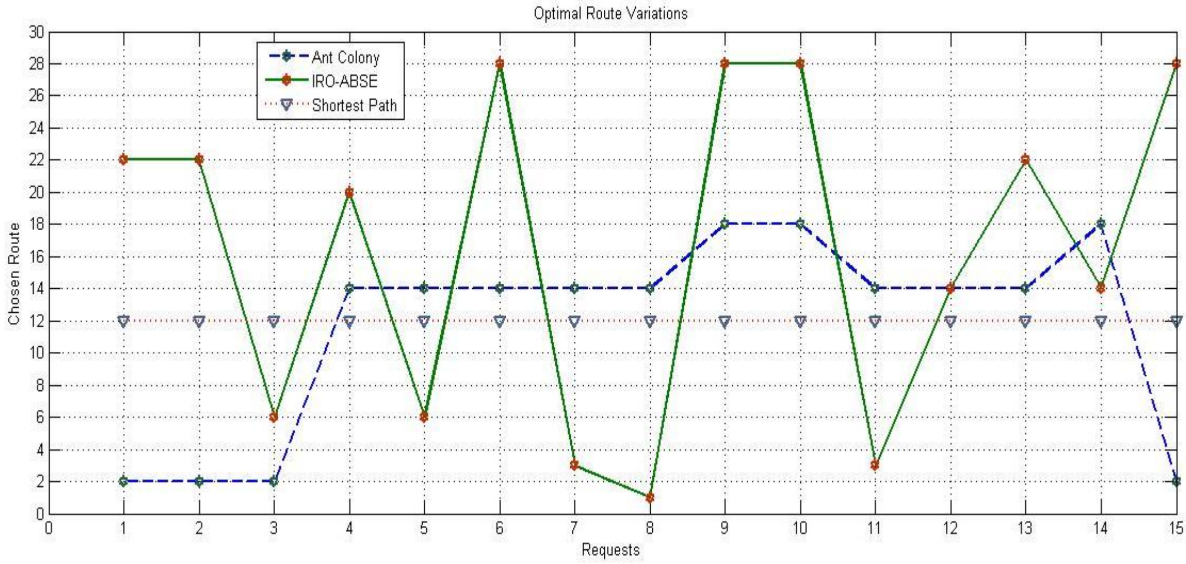

Figure 14: Optimal Route Variations (Rumuokoro and YKC)

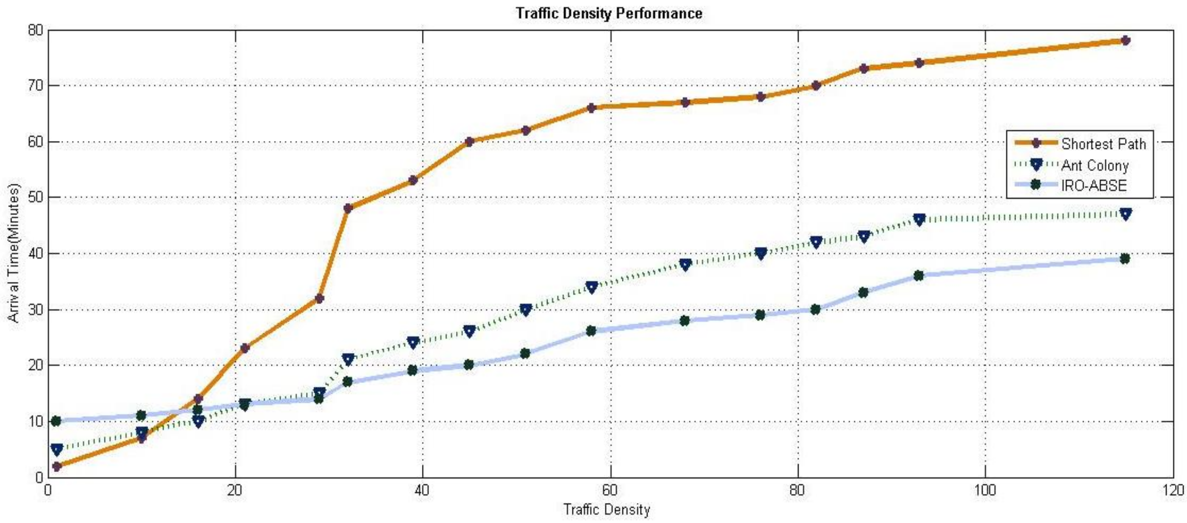

Figure 15: Traffic Density Performance

The optimal route variations in fig 14 show that at route request 1, 2, and 3 of IRO-ABSE, optimal route is 3 for both 1 and 2 but changes to 28 for request 3. For Ant colony, route request 1, 2, 3 produces the same optimal route 22, but changes at request 4 to route 8. While Shortest path maintain the same optimal route 4 for all route requests. The percentage index for the variations in optimal routes in figure 4.7 is; Shortest path is 7%, Ant colony is 26% and IRO-ABSE is 87%.

When the traffic situation (density) on the routes are high it takes longer time to arrive at destination for shortest-path but for Ant colony it takes less time, because the system finds another route especially in a total deadlock occurrence. While IRO-ABSE takes lesser time to get to the destination when compared with shortest-path and Ant colony because the traffic situation is noticed on time and efforts are made to avoid routes with heavy traffic that may affect optimality. This is presented in fig 15 and based on this it is clear that in the computation of optimal route, IRO-ABSE considers multi criteria in multi scenario, hence giving it a better performance index over others.

5. Conclusion

The designed intelligent optimal route selection system based Gaia methodology and implemented in JADE proffer solution to route optimization. The developed IORS generates optimal route based on traffic information obtained from various routes on the road network. It also furnished user with possible routes between origin and destination. The results obtained from IORS when fed with traffic data shows better performance in solving route optimization problem. In other to achieve the objectives of this research a performance evaluation of IORS and other optimization techniques was conducted which revealed that Intelligent Optimal Route Selection was a better approach to solve route optimization problem on road network in a dynamic environment.

References

[1]. Osigwe, U. C., Oladipo, F. O., Onibere, E. A. (2011). Design and Simulation of an Intelligent Traffic Control System. *International Journal of Advances in Engineering & Technology Vol. 1, Issue 5, pp. 47-57.*

[2]. Ugwu, C. and Bale, D. (2014). An Application of Fuzzy Logic Model in Solving Road Traffic Congestion. International Journal of Engineering Research & Technology (IJERT).

[3]. Caric, T. and Gold, H. (2008), Vehicle Routing Problem, In-Teh Croatian branch, I-Tech Education and Publishing KG, Vienna, Austria.

[4]. Wei-Cheng, X. Chang-Min, C., Song-Song, F. and Ling-Ling L. (2014). A Vehicle Routing Optimization Method with Constraints Condition based on Max-Min Ant Colony Algorithm. Applied Mathematics & Information Sciences International Journal.

[5]. Andersson, M. and Lindroth, P. (2005). Route optimization applied to school transports –Amethod combining column generation with greedy heuristics. SE-412 96 Göteborg, Sweden.

[6]. Weisstein, E. (2015). Traveling Salesman Problem. Retrieved July, 2015 http://mathworld.wolfram.com/TravelingSalesmanProblem.html

[7]. Qureshi, M. F., Ali Shah, S. M., and Al-Matroushi, G. I. G. (2013). A Comparative Analysis of Multi-Criteria Road Network. European Centre for Research Training and Development UK.

[8]. Geisberger, R. (2011). Advanced Route Planning in Transportation Networks. des Karlsruher Instituts für Technologie.

[9]. Goldberg, A. V. and Harrelson, C. (2005). Computing the Shortest Path: A* Search Meets Graph Theory. In Proceedings of the 16th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'05), pp 156–165

[10]. Baum, M. (2011). On Preprocessing the Arc-Flags Algorithm. KIT – University of the State of Baden-Wuerttemberg and National Laboratory of the Helmholtz Association.

[11]. Murat, Y. and Uludag, N. (2008). Route choice modelling in urban transportation networks using fuzzy logic and logistic regression methods. Journal of Scientific & Industrial Research, vol. 67, pp. 19-27.

[12]. Salehinejad, H. and Talebi, S. (2010). Dynamic Fuzzy Logic-Ant Colony System-Based Route Selection System. Applied Computational Intelligence and Soft Computing, Hindawi Publishing Corporation.

[13]. Kammoun, H. M., Kallel, I., Casillas, J., and Alimi, A. M. (2008). A Road Traffic MultiAgent Simulation using TurtleKit under MadKit. Retrieved on 18/8/2015, kammoun.regim.org/wp-content/uploads/2012/01/2008_ICAISC_HIJA.pdf

[14]. Jennings, N. R. (2000). On Agent-Based Software Engineering. Artificial Intelligence 117 (2000) 277–296.

[15]. Wooldridge, M., Jennings, N. R. and Kinny, D. (2000). The Gaia Methodology for Agent-oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent System, vol. 3(3), pp. 285–312.

[16]. Giovanni, C. (2009). JADE Tutorial: Jade Programming For Beginners. Telecom Italia S.p.A.

[17]. Bellifemine, F. Giovanni, C., Greenwood, D. (2007). Developing Multi-Agent Systems with JADE. John