Review Article **IJCN (2020) 3:8**

International Journal of Communications and Networks
(ISSN:2637-6571)

A Comparative Analysis of Four Path Optimization Algorithms

Charles OKONJI *; Olawale J. OMOTOSHO; OGBONNA, A. C.

Department of Computer Science, Babcock University

ABSTRACT

Path / route optimization for promptly moving equipment and personnel from base to disaster location has remained a nagging challenges for effective emergency response particularly within the context of developing countries. Bad road networks, poor and outdated navigation systems, faulty transportation vehicles, and traffic congestion remain among the top challenges militating against effective emergency response, and this has resulted in mounting statistics of losses for lives and properties within such jurisdictions. The pressing question has been: how can emergency response itinerary be planned and scheduled most optimally and reliably in the face of these challenges? This research paper compares four of the more popular path / route optimization algorithms (the Ant Colony Optimization Algorithm, Dijkstra's Algorithm, Bellman Ford's Algorithm, and Suurballe's Algorithm), in order to determine the trade-offs and advantages that they present with respect to each other, and propose actionable recommendations for implementation. The findings of this research would prove useful for emergency response planning, particularly within the context of developing countries where these challenges are commonplace.

Keywords: Emergency response; Emergency planning; Path optimization; Route optimization; Emergency Management.

***Correspondence to Author:**
Charles OKONJI
Department of Computer Science, Babcock University

How to cite this article:
Charles OKONJI; Olawale J. OMOTOSHO; OGBONNA, A. C..A Comparative Analysis of Four Path Optimization Algorithms.International Journal of Communications and Networks, 2020, 3:8

eSciPub LLC, Houston, TX USA.
Website: https://escipub.com/

Introduction

Emergency Management has become an important global discourse. The consequences of climate change and a rapidly degrading environment, as well as the natural and man-made disasters (some conscious, others unwitting) that continue to rock various parts of the world are testament to the fact that emergency incidents, if not managed with dispatch, effectiveness, and efficiency, can severely and negatively impact the lives of large numbers of people for extended periods of time (Government Accountability Office (GAO), 2006). For example, the Hurricanes Katrina and Rita disasters which hit in succession in 2005 devastated over 90,000 square miles in five states, resulting in estimated financial losses of over $88 billion and more than 1,500 deaths; in addition to the displacement of some at least 600,000 families (Stanton, 2007).

Mushkatel and Weschler (1985), describe four (4) stages of emergency management processes; *viz*, preparedness, reaction / response, recovery, and mitigation. While all four stages of the disaster management process remain interrelated and interconnected, Zlatanova and Fabbri (2009) thus attempt to strike some distinction between them: prevention and mitigation focus on long-term interventions that help to minimize / remove the vulnerabilities that predispose to hazards; while planning involves framing the structural and organizational structures that help to catalyse emergency response operations. This could include contingency practices like evacuation plans, early warning systems, temporary physical steps, training sessions, and field preparatory exercises. Response operations take place after a disaster occurs and it is the most difficult stage of the process due to the complicated, volatile and often dynamic nature of emergencies; recovery, the post-response phase and, in general, the standardization of an emergency situation, involves all the necessary measures to eradicate damage and long-term irrevocable losses.

In order to examine emergency events and subsequent response operations, numerous researchers on emergency management issues have embraced broad and sometimes complex sources of study. Such research studies covered emerging trends and technologies such as information systems for emergency response, the importance of human-computer communication and socio-technical processes, organizational structures and configurations, and techniques for leadership and planning. This research, however, focuses on the response and short-term recovery phases, by dealing with the key concern of how to choose the most optimal path / route for most effectively moving emergency response equipment and personnel within the shortest possible time from emergency base station to disaster location.

Related Works

The response stage is a critical, and arguably the most time-sensitive and important phase in emergency management based on current literature. The popularization of information and communication technologies (ICT) and mobile telephone has seen the emergence of new emergency response programs and procedures that feature active information systems that are capable of providing accurate and reliable information in real-time for response-critical operations (Walle & Turoff, 2007). However, a major hurdle facing emergency response agencies, particularly within the context of developing countries, relates to how valuable information can be obtained and analysed in time to inform response decisions and actions of emergency teams. Bad data governance regimes make it difficult to collect and aggregate such data, even as poor Internet facilities often hamper the timely delivery of critical response information to emergency stations and teams.

Fiedrich and Burghardt (2000) identify two major areas of research in the use of agent software, namely the agent-based simulation systems that can construct practical post-disaster environments, and the agent-based Decision Support System (DSS) that can support disaster

managers operating at different levels. In either case, however, the focus remains on the timeliness and efficacy of communication needs during any crisis. They argue that the use of agent software enables more timely and improved data acquisition and communication and thus ensures effective decision-making.

Further, the Ensayo Virtual EOC (vEOC) project (Becerra-Fernandez & Prietula, 2006) (Becerra-Fernandez, et al., 2007) aimed at addressing the basic needs of emergency response organizations. This project required a broad, multidisciplinary effort to build a digital computing system to simulate the different functions and processes of an emergency management incident. One important contribution of this work was to help better train emergency management personnel on how to tackle emergencies effectively and efficiently if they happen.

There are several types of other systems that play crucial roles in disaster management in the modern era (Kapoor, 2009). These systems include, but are not limited to: the use of maps for planning disaster response regimens. These maps help to layout the topographies, population distribution, as well as seismic movements and patterns for tracking hurricanes. Similar systems that feature in modern disaster management operations include: aerial photography systems for monitoring, risk assessment, and evaluating the scale of disasters; electronic communication systems for delivering commands, reporting, monitoring and evaluation; logistics and inventory systems used for scheduling, inventory management, and delivery of supplies and accounting procedures; among others. All these digital and electronic systems have greatly helped to minimize the challenges that have often been associated with effective disaster management in modern times.

One other challenge, however, that relevantly applies to the focus of this research and continue to impact on disaster management, particularly in the context of developing countries, is that of navigation and transportation. Poor town planning, high population densities, and unstructured urban planning / development practices can sometimes result in great difficulty for the transportation and mobility of emergency equipment and personnel in the face of disaster that is localized to particular area(s). In busy cities like Beijing (China), Hong Kong (Japan), and Lagos (Nigeria), to mention a few, transportation and mobility during emergency response operations can sometimes prove a big challenge due to large volumes of regular human and vehicular traffic.

Methodology Overview

In light of this challenge, this research advances knowledge by presenting a qualitative comparative analysis / evaluation of selected common path / route optimization algorithms / systems that can be used to identify the most optimal path / route for effective operations during disaster situations. The exploratory approach is adopted to traverse the Ant Colony Optimization Algorithm, Dijkstra's Algorithm, Bellman Ford's Algorithm, and Suurballe's Algorithm; presenting the operating principles, trade-offs, and advantages that each present over the other. Recommendations, are then proffered as a way forward in the implementation of path / route optimization algorithms for effective disaster management in light of the research findings.

Path / Route Optimization

Path / route optimization remains the "holy grail" of navigation systems. It is often referred to as the shortest path problem (SPP), and is known to be among the most fundamental and well-known combinatorial problems of modern science and engineering (Broumi, Bakal, Talea, Smarandache, & Vladareanu, 2016). The most effective navigation systems in use today are able to estimate the most optimal path / route from a source point to a destination point on a mapped grid. This is often achieved, in its basic sense, by deploying a search mechanism that calculates the approximate distance of all pathways or linkages connecting the source and the destination points on the mapped grid, and

then returning the most optimal path / route (which is usually either the shortest, or that which offers the least resistance) that links to the destination (often the goal of the search mechanism). This same thinking has been successfully applies in various other domains, such as vehicle parking systems, assembly lines, manufacturing plants, and traffic redirection, to mention a few.

Many different algorithms have been proposed to this end, and have been widely applied in various navigation and transportation systems for the purpose of efficiency and optimization. Popular amongst these are: the Ant Colony Optimization Algorithm, Dijkstra's Algorithm, Bellman Ford's Algorithm, and Suurballe's Algorithm. These selected algorithms are individually discussed in the following sub-sections of the research.

Ant Colony Optimization (ACO) Algorithm

In real life, ants are known to live in colonies, and their behaviour is driven by the objective of survival for the entire colony rather than that of an individual ant. Ants initially explore the environment around their nest in a random manner while searching for food, leaving on the floor a chemical pheromone trail as they crawl along. The consideration here is that high concentrations of this pheromone determine the smell of the ants, and high concentrations too. The quantity of pheromone left on the ground by an ant may depend on the quantity and quality of the food during the return trip; and this points the way to the food source for other ants. Figure 1 illustrates this behavioural principle.

Apparently inspired by this principle, Marco Dorigo and colleagues proposed the Ant Colony Optimization Algorithm in the 1990s, as an optimizing metaheuristic technique modelled after the foraging behaviour of real ant colonies and, in general, how ants can seek the shortest paths between food sources and their nest (Dorigo, Maniezzo, & Colorni, 1996). Metaheuristic methods usually attempt to find good solutions within a reasonable timeframe for complex problems (Talbi, 2009). By integrating simple behavioural processes that are observed at work in real life (Corne, Dorigo, & Glover, 1999), it is possible to solve complex optimization problems.

For example, in the case of ants, they can collectively decide the shortest route from their nest to a food source without using visual signals; but instead using simple individual interactions mediated by a pheromone. The ACO algorithm has been used successfully to solve many complex problems of combinatorial optimization, such as the problem of moving a sales-man (the Travelling Salesman Problem [TSP]), the problem of vehicle routing, the problem of graph colouring, the problem of quadratic allocation, the problem of network-traffic optimization, and the problem of job-shop planning, to mention a few (Shtovba, 2005).

In practice, therefore, artificial ants are looking probabilistically for the answer in the ACO algorithm to construct candidate solutions, which are then analysed and modified based on the associated quantity / intensity of each pheromone trail. It should be noted that while the concentration of the pheromone trail may reduce overtime due to evaporation, the trail with the highest pheromone value is still likely to remain the optimal solution to the problem. This operating principle is illustrated in Figure 2.

Hence, given a problem of colony optimization (CO) to be solved, a finite set of *C* components of the solution must first be derived in order to assemble possible solutions to the *CO* problem. First, it would be important to describe a set of *T* pheromone values, also known as the pheromone model, which is essentially a parameterized probabilistic model and one of the metaheuristic core components of the ACO that would be correlated with components of the solution. The pheromone model is then used to assemble values from the set of solution components, in order to probabilistically produce solutions to the problem under consideration.

However, a number of domains and applications still pose formidable challenges for implementations of ACO. These have been

observed to include: its performance and competence in process handling for complex production systems (Kucukkoc & Zhang, 2015); combinatorial optimization while solving Job Shop scheduling and probabilistic existence in metaheuristic algorithms (Prakasam & Savarimuthu, 2015); network optimization for routing and scheduling in Convoy Movement Problems (CMPs), which is essentially an NP-hard problem (Krishna & Kumar, 2015); the Multidimensional Knapsack Problem (MKP) that extends the basic Knapsack problem with two or more limitations (Fingler, Cáceres, Mongelli, & Song, 2014); and specialized applications for navigational decision support (Lazarowska, 2014); to mention a few.

However, ACO is known to be time-inefficient; having a time complexity of $O(n^2 \times number\ of\ iterations)$ (Ismkhan, 2017).

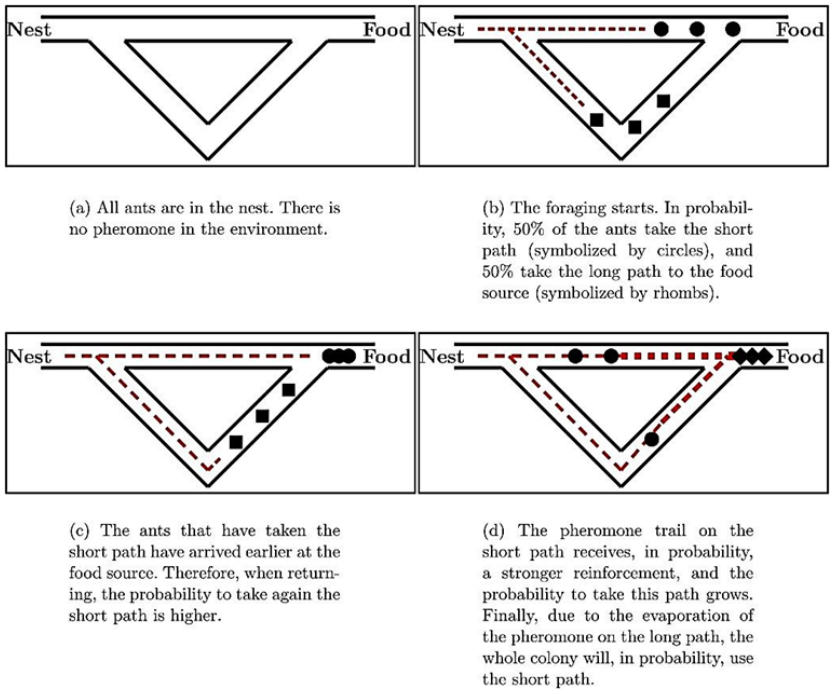

Figure 1: An experimental setting that demonstrates the shortest path finding capability of ant colonies. Between the ants' nest and the only food source exist two paths of different lengths. In the four graphics, the pheromone trails are shown as dashed lines whose thickness indicates the trails' strength. (Blum, 2005)

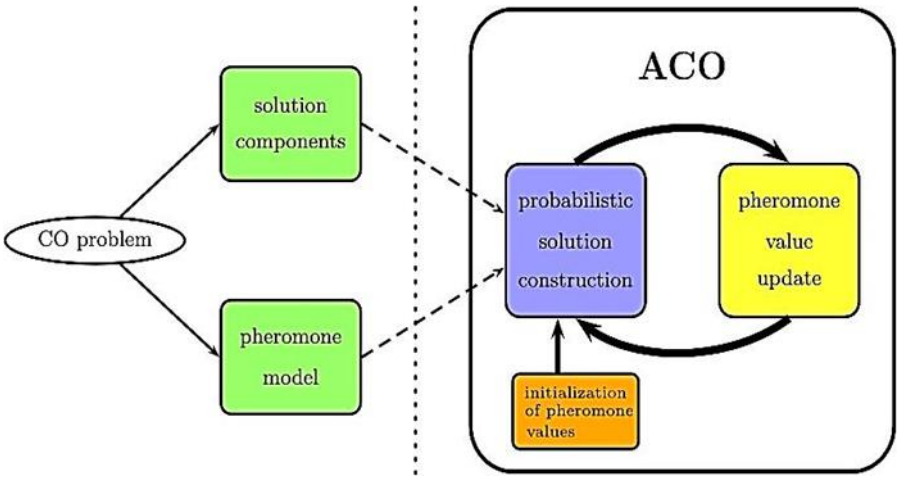

Figure 2: The working of the ACO metaheuristic (Christian Blum, 2005)

Dijkstra's Algorithm

Dijkstra's Algorithm is one of the most well-known and widely-used algorithms that have been deployed towards solving the shortest path problems. It was proposed by Edsger Wybe Dijkstra (1959) in a research note describing the solutions to two graph-search problems in edge-weighted graphs with connected vertices. The second search problem described in this note was the shortest path problem (SPP). In a network setting, SPPs are modelled as interconnected graphs with paths running from source to destination, and weights attached to each graph edge. The length of edges may represent real-life quantities such as time, cost, process, or others. For conventional implementations, a decision-maker or path-seeker faced with certain constraint conditions and some certainty about the parametric values of these quantities attempts to find the path connecting a source edge to a destination edge with minimum weight values (Broumi, Bakal, Talea, Smarandache, & Vladareanu, 2016).

Dijkstra describes the shortest-length path from a single source vertex, *s*, to another vertex, *v*, that can be trivially extended to other multiple sources, as a non-empty subset *S* of vertices, that contain a path from a $u \in S$, to *v*, whose length does not exceed the length of any other path from a $w \in S$, to *v*. This extension can be achieved by adding to the original graph the dummy vertex, *s*, connecting it to each vertex in *S* by an edge of weight zero, and finding for this extended graph the shortest-length path from *s* to *v* (Ciesielski, Falcão, & Miranda, 2018); with a time complexity of $n^2 + m = O(n^2)$ (Magzhan & Jani, 2013).

Many researches have now proposed modifications, advancements, enhancements to the original Dijkstra's Algorithm for better performance in various application or problem domains. For example, (Broumi, Bakal, Talea, Smarandache, & Vladareanu, 2016) proposes a version of the algorithm for SPPS in networks having edge weights that are characterized by single-valued neutrosophic numbers. Also, (Ciesielski, Falcão, & Miranda, 2018) describe the parameters that are essential for Dijkstra's Algorithm to return an optimal mapping of a path-value function defined on an arbitrary finite graph, with the properties of such a function. These are to mention a few.

Bellman Ford's Algorithm

The Bellman Ford Algorithm (BFA) was first proposed by Alfonso Shimbel in a 1955 Symposium Paper titled "Structure in communication nets" (Shimbel, 1955). However, it was later published by Richard Bellman (1958) and Lester Ford Jr. (Ford Jr, 1956), respectively. Thus, it became popularly known as the Bellman-Ford Algorithm. The reason why it is also sometimes referred to as the Bellman–Ford–Moore algorithm by some sources is because Edward Moore is known to have also published a version of the same Algorithm in 1957.

The Bellman Ford algorithm is used in a weighted graph to find the shortest paths from a source or origin vertex to all other (destination) vertices. This algorithm relies on the theory of relaxation where the shortest distance is gradually replaced by more accurate values for all vertices until the optimal solution is finally reached. At the start, all vertices are assumed to have a distance of "Infinity"; except the distance of the source vertex, which is assumed to be "zero" (0). Then, all the related vertices are updated with the new distances according to the formula $(source\ vertex\ distance + edge\ weights)$. The same procedure is then repeated with new distances to the new vertices and so on.

BFA is one of the preferred algorithms that have been widely applied for solving optimization problems in directed hypergraphs. One reason why it is particularly preferred is owing to the fact that beyond its robustness to negative edge weights, it is also easily adaptable to parallelized search problems (Weber, Kreuzer, & Knoll, 2020); even though it also performs considerably slower than popular path optimization algorithms like Dijkstra's.

As a result, the Time Complexity of the Bellman Ford algorithm is relatively high: n^3 (Magzhan & Jani, 2013).

BFA has been widely applied to solving a broad range of problems across several domains. These include: single link failure recovery in Software Defined Networks (Waleed, Faizan, Iqbal, & Anis, 2017); parallel execution in the Symbolic controller synthesis Scheme (Weber, Kreuzer, & Knoll, 2020); single valued membership degree values under neutrosophic environments, which was solved using a trapezoidal interval valued neutrosophic version of Bellman's algorithm (Broumi, et al., 2019); as well as route planning for better efficiency of electric vehicles fitted with an embedded GPU (Schambers, Eavis-O'Quinn, Roberge, & Tarbouchi, 2018). (Kalpana & Tyagi, 2017), also integrated the global positioning system (GPS) into BFA for solving shortest path problems with more efficiency. These are to mention a few.

Suurballe's Algorithm

The Suurballe Algorithm is one that is used in finding two separate and unconnected paths on a directed graph that is not negatively-weighted, such that both paths connect the same pair of vertices and have a minimum total length (Bhandari, 1999). This Algorithm was first conceived by John W. Suurballe, and published first in 1974 (Suurballe, 1974), when the runtime was estimated to be $O(n^2\ logn)$. Later, in 1984, Suurballe and Tarjan modified the Algorithmic runtime to become $O(m\ log\ (_{1+\ m/n})^n)$ (Suurballe & Tarjan, 1984).

Suurballe's Algorithm is defined by the relationship: Let *G* be a weighted directed graph with vertex set *V* and edge set *E*; let *s* be a designated source vertex in *G*, and let *t* be a designated destination vertex. Let each edge (u,v) in *E*, from vertex *u* to vertex *v*, have a non-negative cost $w(u,v)$. Define $d(s,u)$ to be the cost of the shortest path to vertex *u* from vertex *s* in the shortest path tree rooted at *s*.

The operating principle of Suurballe's Algorithm is to use Dijkstra's Algorithm to locate one path on the graph, and then modify the weighted edges of the graph, before running Dijkstra's Algorithm the second time to locate the other path. The algorithm's output is then generated by combining these two paths, discarding the edges crossed by the paths in opposite directions, and using the remaining edges to form the two paths to return as the output. Changing the weights helps to maintain the weights' non-negativity while allowing Dijkstra's second-run instance to find the right second path.

Theoretically, this implies finding the shortest path tree *T* rooted at node *s* by running Dijkstra's algorithm. This tree contains for every vertex *u*, a shortest path from *s* to *u*. Let P_1 be the shortest cost path from *s* to *t*. The edges in *T* are called tree edges and the remaining edges are called non-tree edges; where all tree edges have a cost of *0*, and non-tree edges have a non-negative cost. Suurballe has given a time complexity of $O(n^2\ logn)$ for this problem.

The Suurballe's Algorithm can be viewed as a special case of a minimum cost flow algorithm which repeatedly moves the maximum amount of flow along the shortest increase path. The first path found by Suurballe's Algorithm is the shortest increase path for the initial (zero) flow, and the second path found by Suurballe's Algorithm is the shortest increase path left after moving one flow unit along the first route.

Recent research has seen the Suurballe Algorithm being applied to solve a variety of problems across different domains; such as: the problem of Generalized Diversity Coding for survivable routing algorithms (Pašić, et al., 2020); multi-constraint quality of service problems of disjoint multipath routing in Software Defined Networks (Doshi & Kamdar, 2018); energy-optimal routing problems in wireless ad-hoc networks (Baugh, Calinescu, Rincon-Cruz, & Qiao, 2016); as well as multipath route selection over the Internet (Helfert, Niedermayer, & Carle, 2018). These are to mention a few.

Comparative Analysis

Table 1 presents a summary of the Comparative Analysis of all four path / route optimization algorithms that have been widely applied to solving the shortest path problem: Ant Colony Optimization Algorithm, Dijkstra's Algorithm, Bellman Ford's Algorithm, and Suurballe's Algorithm.

Table 1: Comparative Analysis of Selected Path Optimization Algorithms

Algorithms	Operating Principle	Time Complexity
Ant Colony Optimization (ACO) Algorithm	Given an optimization problem to be solved, a finite set of *C* components of the solution are derived in order to assemble possible solutions to the problem. A set of *T* pheromone values are described to create a probabilistic (metaheuristic) pheromone model. The pheromone model is then used to assemble values from the set of solution components, in order to probabilistically produce solutions to the problem under consideration.	$O(n^2 \times no.of\ iters.)$
Dijkstra's Algorithm	Dijkstra describes the shortest-length path from a single source vertex, *s*, to another vertex, *v*, that can be trivially extended to other multiple sources, as a non-empty subset *S* of vertices, that contain a path from a $u \in S$, to *v*, whose length does not exceed the length of any other path from a $w \in S$, to *v*. This extension can be achieved by adding to the original graph the dummy vertex, *s*, connecting it to each vertex in *S* by an edge of weight zero, and finding for this extended graph the shortest-length path from *s* to *v*.	$n^2 + m = O(n^2)$
Bellman Ford's Algorithm	This algorithm relies on the theory of relaxation where the shortest distance is gradually replaced by more accurate values for all vertices until the optimal solution is finally reached. At the start, all vertices are assumed to have a distance of "Infinity"; except the distance of the source vertex, which is assumed to be "zero" (0). Then, all the related vertices are updated with the new distances according to the formula $(source\ vertex\ distance + edge\ weights)$. The same procedure is then repeated with new distances to the new vertices and so on.	n^3
Suurballe's Algorithm	Let *G* be a weighted directed graph with vertex set *V* and edge set *E*; let *s* be a designated source vertex in *G*, and let *t* be a designated destination vertex. Let each edge (u, v) in *E*, from vertex *u* to vertex *v*, have a non-negative cost $w(u, v)$. Define $d(s, u)$ to be the cost of the shortest path to vertex *u* from vertex *s* in the shortest path tree rooted at *s*. Suurballe's Algorithm uses Dijkstra's Algorithm to locate one path on the graph, and then modifies the weighted edges of the graph, before running Dijkstra's Algorithm the second time to locate the other path. The algorithm's output is then generated by combining these two paths, discarding the edges crossed by the paths in opposite directions, and using the remaining edges to form the two paths to return as the output.	$O(n^2\ logn)$ $O(m\ log\ (_{1+\ m/n})^n)$

From the analysis shown in Table 1, it is intuitively clear that Suurballe's Algorithm presents the most efficient time constraint for a solution to the shortest path problem. Next to this in performance is Dijkstra's Algorithm. The Ant Colony Optimization Algorithm, and the Bellman Ford's Algorithm present the least efficiency in terms of time constraint for solutions to the shortest path problem.

Summary & Recommendations

In summary, the Dijkstra's Algorithms seems to be the foundations of a number of emerging algorithms that are now able to successfully solve the shortest path problem, even with better efficiency (such as the Suurballe's Algorithm that

optimizes and incorporation Dijkstra's Algorithm).

However, in recent times, we have also begun to see hybridizations of one or more path / route algorithms being applied to various problems; and yielding results that far outperforms the each of the component algorithms in individual applications. One example is (Dinitz & Itzhak, 2017), which hybridized the Bellman Ford and Dijkstra Algorithms towards better performance for single-source problems of path optimization a digraphs that allow negative edge costs. Similarly, (Swathika, Hemamalini, Mishra, Pophali, & Barve, 2016), which hybridized Bellman Ford and Dijkstra Algorithms for better performance in shortest path identification in reconfigurable micro-grids for power distribution networks.

This practice is highly recommended; especially as path optimization problems continue to present contemporary challenges that have been seen to be more complicated and complex.

Conclusion

In conclusion, path / route optimization continue to present a contemporaneous challenge for theoretical computer science. This research has been able to undertake a comperation mon algorithms that have been widely deployed for solving problems of path / route selection and optimization across various domains; *viz*.: Ant Colony Optimization Algorithm, Dijkstra's Algorithm, Bellman Ford's Algorithm, and Suurballe's Algorithm.

While each of these algorithms have been seen to present various advantages and disadvantages in comparation with each other, particularly in terms of performance, time, and space complexities, it is recommended that they be hybridized for modern applications; because the practice of combining multiple individual efficiencies is able to upscale performance exponentially.

References

1. Baugh, S., Calinescu, G., Rincon-Cruz, D., & Qiao, K. (2016). Improved Algorithms for Two Energy-Optimal Routing Problems in Ad-Hoc Wireless Networks. *2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom)(BDCloud-SocialCom-SustainCom)* (pp. 509-516). Atlanta, GA, USA: IEEE. doi:https://doi.org/10.1109/BDCloud-SocialCom-SustainCom.2016.80
2. Becerra-Fernandez, I., & Prietula, M. (2006). Project Ensayo: Integrating simulation, training, discovery, and support. *North American Association for Computational Social and Organizational Science (NAACSOS).* Notre Dame, Indiana.
3. Becerra-Fernandez, I., Prietula, M., Madey, G., Rodriguez, D., Gudi, A., & Rocha, J. (2007). Project Ensayo: A Virtual Emergency Operations Center. *16th International Conference on Management of Technology.* Miami Beach, Florida.
4. Bellman, R. (1958). On a routing problem. *Quarterly of Applied Mathematics, 16*, 87–90. doi:https://doi.org/10.1090/qam/102435
5. Bhandari, R. (1999). Suurballe's disjoint pair algorithms. In *Survivable Networks: Algorithms for Diverse Routing* (pp. 86–91). Springer-Verlag.
6. Blum, C. (2005). Ant colony optimization: Introduction and recent trends. *Physics of Life Reviews, 2*(4), 353-373. doi:https://doi.org/10.1016/j.plrev.2005.10.001
7. Broumi, S., Bakal, A., Talea, M., Smarandache, F., & Vladareanu, L. (2016). Applying Dijkstra algorithm for solving neutrosophic shortest path problem. *2016 International Conference on Advanced Mechatronic Systems (ICAMechS)* (pp. 412-416). Melbourne, VIC, Australia: IEEE. doi:https://doi.org/10.1109/ICAMechS.2016.7813483
8. Broumi, S., Nagarajan, D., Lathamaheswari, M., Talea, M., Bakali, A., & Smarandache, F. (2019). Bellman-Ford Algorithm Under Trapezoidal Interval Valued Neutrosophic Environment. *International Conference on Computing* (pp. 174-184). https://doi.org/10.1007/978-3-030-36368-0_15.
9. Ciesielski, K. C., Falcão, A. X., & Miranda, P. A. (2018). Path-value functions for which Dijkstra's algorithm returns optimal mapping. *Journal of Mathematical Imaging and Vision, 60*(7), 1025-1036. doi:https://doi.org/10.1007/s10851-018-0793-1

10. Corne, D., Dorigo, M., & Glover, F. (1999). *New Ideas in Optimization.* New York, NY: McGraw-Hill.
11. Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*(1), 269–271. doi:https://doi.org/10.1007/BF01386390
12. Dinitz, Y., & Itzhak, R. (2017). Hybrid Bellman–Ford–Dijkstra algorithm. *Journal of Discrete Algorithms, 42*, 35-44. doi:https://doi.org/10.1016/j.jda.2017.01.001
13. Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 26*(1), 29-41. doi:https://doi.org/10.1109/3477.484436
14. Doshi, M., & Kamdar, A. (2018). Multi-constraint QoS disjoint multipath routing in SDN. *2018 Moscow Workshop on Electronic and Networking Technologies (MWENT)* (pp. 1-5). Moscow, Russia: IEEE. doi:https://doi.org/10.1109/MWENT.2018.8337305
15. Fiedrich, F., Gehbauer, F., & Rickers, U. (2000). Optimized resource allocation for emergency response after earthquake disasters. *Safety Science, 35*(1-3), 41-57. doi:https://doi.org/10.1016/S0925-7535(00)00021-7
16. Fingler, H., Cáceres, E. N., Mongelli, H., & Song, S. W. (2014). A CUDA based Solution to the Multidimensional Knapsack Problem Using the Ant Colony Optimization. *Procedia Computer Science, 29*, 84–94. doi:https://doi.org/10.1016/j.procs.2014.05.008
17. Ford Jr, L. R. (1956). *Network Flow Theory.* Santa Monica, California: RAND Corporation.
18. Government Accountability Office (GAO). (2006, March 08). *Hurricane Katrina: GAO's preliminary observations regarding preparedness, response, and recovery.* Retrieved January 31, 2020, from US Government Accountability Office: https://www.gao.gov/new.items/d06442t.pdf
19. Helfert, F., Niedermayer, H., & Carle, G. (2018). Evaluation of algorithms for multipath route selection over the Internet. *21st Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)* (pp. 1-8). Paris, France: IEEE. doi:https://doi.org/10.1109/ICIN.2018.8401640
20. Ismkhan, H. (2017). Effective heuristics for ant colony optimization to handle large-scale problems. *Swarm and Evolutionary Computation, 32*, 140-149. doi:https://doi.org/10.1016/j.swevo.2016.06.006
21. Kalpana, M. A., & Tyagi, M. A. (2017). Bellman ford shortest path algorithm using global positioning system. *International Research Journal of Engineering and Technology (IRJET), 4*(04), 2503-2507. Retrieved February 01, 2020, from https://67.209.122.217/archives/V4/i4/IRJET-V4I4624.pdf
22. Kapoor, M. (2009). *Disaster Management.* New Delhi, India: Motilal Banaisidasi Publishers Pvt. Ltd.
23. Krishna, K. S., & Kumar, P. R. (2015). On the amenability and suitability of ant colony algorithms for convoy movement problem. *Procedia-Social and Behavioral Sciences, 189*, 3-16. doi:https://doi.org/10.1016/j.sbspro.2015.03.187
24. Kucukkoc, I., & Zhang, D. Z. (2015). Type-E parallel two-sided assembly line balancing problem: Mathematical model and ant colony optimisation based approach with optimised parameters. *Computers & Industrial Engineering, 84*, 56-69. doi:https://doi.org/10.1016/j.cie.2014.12.037
25. Lazarowska, A. (2014). Ant colony optimization based navigational decision support system. *Procedia Computer Science, 35*, 1013-1022. doi:https://doi.org/10.1016/j.procs.2014.08.187
26. Magzhan, K., & Jani, H. M. (2013). A review and evaluations of shortest path algorithms. *International Journal of Scientific and Technology Research, 2*(6), 99 -104.
27. Mushkatel, A. H., & Weschler, L. F. (1985). Emergency management and the intergovernmental system. *Public Administration Review, 45*, 49-56. doi:https://doi.org/10.2307/3134997
28. Pašić, A., Babarczi, P., Tapolcai, J., Bérczi-Kovács, E. R., Király, Z., & Rónyai, L. (2020). Minimum Cost Survivable Routing Algorithms for Generalized Diversity Coding. *IEEE/ACM Transactions on Networking, Online First*, 1-12. doi:https://doi.org/10.1109/TNET.2019.2963574
29. Prakasam, A., & Savarimuthu, N. (2015). Metaheuristic algorithms and polynomial Turing reductions: A case study based on ant colony optimization. *Procedia Computer Science, 46*, 388-395. doi:https://doi.org/10.1016/j.procs.2015.02.035
30. Schambers, A., Eavis-O'Quinn, M., Roberge, V., & Tarbouchi, M. (2018). Route planning for electric vehicle efficiency using the Bellman-Ford

algorithm on an embedded GPU. *4th International Conference on Optimization and Applications (ICOA)* (pp. 1-6). Mohammedia, Morocco: IEEE. doi:https://doi.org/%2010.1109/ICOA.2018.8370584

31. Shimbel, A. (1955). Structure in communication nets. *Proceedings of the Symposium on Information Networks* (pp. 199–203). New York, NY: Polytechnic Press of the Polytechnic Institute of Brooklyn.
32. Shtovba, S. D. (2005). Ant algorithms: Theory and applications. *Programming and Computer Software, 31*(4), 167-178.
33. Stanton, T. H. (2007, January 12). *Delivery of benefits in an Emergency: Lessons from hurricane.* Retrieved January 31, 2020, from IBM Center for the Business of Government: http://www.businessofgovernment.org/sites/default/files/StantonKatrinaReport.pdf
34. Suurballe, J. W. (1974). Disjoint paths in a network. *Networks, 4*(2), 125–145. doi:https://doi.org/10.1002%2Fnet.3230040204
35. Suurballe, J. W., & Tarjan, R. E. (1984). A quick method for finding shortest pairs of disjoint paths. *Networks, 14*(2), 325-336. doi:https://doi.org/10.1002/net.3230140209
36. Swathika, O. V., Hemamalini, S., Mishra, S., Pophali, S. M., & Barve, N. A. (2016). Shortest Path Identification in Reconfigurable Microgrid Using Hybrid Bellman Ford-Dijkstra's Algorithm. *Advanced Science Letters, 22*(10), 2932-2935.
37. Talbi, E. G. (2009). *Metaheuristics: From design to implementation (volume 74).* New Jersey, NJ: John Wiley & Sons.
38. Waleed, S., Faizan, M., Iqbal, M., & Anis, M. I. (2017). Demonstration of single link failure recovery using Bellman Ford and Dijikstra algorithm in SDN. *International Conference on Innovations in Electrical Engineering and Computational Technologies (ICIEECT)* (pp. 1-4). Karachi, Pakistan: IEEE. doi:https://doi.org/10.1109/ICIEECT.2017.7916533
39. Walle, B. V., & Turoff, M. (2007, March). Introduction. *Communications of the ACM (Emergency response information systems: Emerging trends and technologies), 50*(3), 29-31. Retrieved from http://doi.org/10.1145/1226736.1226760
40. Weber, A., Kreuzer, M., & Knoll, A. (2020). A generalized Bellman-Ford Algorithm for Application in Symbolic Optimal Control. *arXiv preprint, arXiv:2001.06231*, 1-8. Retrieved February 09, 2020, from https://arxiv.org/pdf/2001.06231.pdf
41. Zlatanova, S., & Fabbri, A. G. (2009). Geo-ICT for risk and disaster management. In H. J. Scholten, R. van de Velde, & N. van Manen (Eds.), *Geospatial Technology and the Role of location in Science* (Vols. 96, GeoJournal Library, pp. 239-266). Springer, Dordrecht. doi:https://doi.org/10.1007/978-90-481-2620-0_13