BIBLIOGRAPHY AND REFERENCES

Abbasbandy, S., & Asady, B. (2006). Ranking fuzzy numbers by sign distance. *Inform.Sci, 176*, 2405- 2412.

Agarwal, P. K., & Shing, M. T. (1990). Algorithms for special cases of rectilinear steiner trees: Points on the boundary of a rectilinear rectangle. *Networks, 20*, 453–85.

Akkaya, K., & Younis, M. (2005). A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks, 3*, 325-349.

Akram, M. (2012). Characterizing $\Gamma-$ Semigroups by Intuitionistic Fuzzy Points. *ARPN Journal of Systems and Software, 2(12)*, 359-365.

Arora, S. (1998). Polynomial time approximation schemes for euclidean tsp and other geometric problems. *J. Assoc. Comput. Mach., 45*, 753–782.

Asady, B., & Zendehnam, A. (2007). Ranking fuzzy numbers by distance minimization. *Applied Mathematical Modeling, 31*, 2589-2598.

Atanassov, K. (June, 1983). Intuitionistic fuzzy sets. *VII ITKR's Scientific Session, Sofia*. Deposed in Central Sci. - Techn. Library of Bulg. Acad. of Sci., 1697/84 (in Bulg.).

Attanasov, K. (Sept., 2003). Intuitionistic fuzzy sets: past, present and future. *Presented at 3rd Conference of the European Society for Fuzzy Logic and Technology*, Zittau, Germany.

Atanassov, K., & Shannon, A. (1995). Intuitionstic fuzzy graphs From $\alpha-$, $\beta-$ and $(\alpha,\beta)-$levels. *Notes on Intuitionistic Fuzzy Sets, 1(1),* 32–35.

Awerbuch, B., Azar, Y., Blum, A., & Vempala, S. (1999). Improved approximation guarantees for minimum-weight k-trees and prize-collecting salesmen. *Siam J. Computing*, *28*, 254–262.

Back, T. (1994). Selective Pressure in Evolutionary Algorithms: A Characterization of Selection Mechanisms. *Proceedings of the 1st IEEE Conference on Evolutionary Computation,* 57-62.

Bang, W.Y., & Kun, C.M. (2006). *Minimum Spanning Trees.* Retrieved from: http://www.csie.ntu.edu.tw/~kmchao/tree10fall/mst.pdf

Bansal, A. (2011). Trapezoidal Fuzzy Numbers (a, b, c, d): Arithmetic Behavior. *International Journal of Physical and Mathematical Sciences, 2011*, 39-44.

Berman, P., Fobmeier, U., Karpinski, M., Kaufmann, M., & Zelikovsky, A. Z. (1994). Approaching the 5/4 - approximation for rectilinear steiner trees. *Proceedings of the European Symposium on Algorithms*, 533–542.

Berman, P., & Ramaiyer, V. (1992). Improved approximations for the steiner tree problem. *Proceedings of the ACM/SIAM Symp. on Discrete Algorithms*, San Francisco, CA, 325–334.

Berman, P., & Ramaiyer, V. (1994). Improved approximations for the steiner tree problem. *J. Algorithms, 17*, 381–408.

Bezdek, J.C. (1993). Fuzzy Models - What Are They, and Why?. IEEE Transactions on Fuzzy Systems, 1(1), 1-9.

Blelloch, G. E. (1996). Programming Parallel Algorithms. *Communications of the ACM, 39(3)*, 85-97.

Blue, M., Bush, B., & Puckett, J. (2002). Unified approach to fuzzy graph problems. *Fuzzy Sets and Systems, 125*, 355–368.

Blum, A., Chawla, S., Karger, D. R., Lane, T., Meyerson, A., & Minkoff, M. (2003). Approximation Algorithms for Orienteering and Discounted-Reward TSP. *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS'03)*, 1-10.

Boruvka, O. (1926). O jistém problému minimálním (About a certain minimal problem). *Práce mor. přírodověd. spol. v Brně III (in Czech and German summary), 3*, 37–58.

Boruvka, O. (1926). Příspěvek k řešení otázky ekonomické stavby elektrovodních sítí (Contribution to the solution of a problem of economical construction of electrical networks). *Elektronický Obzor (in Czech), 15*, 153–154.

Bustince, H., Barrenechea, E., & Pagola, M. (2006). Restricted equivalence functions. *Fuzzy Sets and Systems, 157*, 2333 – 2346.

Bustince, H., Barrenechea, E., & Pagola, M. (2008). Relationship between restricted dissimilarity functions, restricted equivalence functions and normal E_N-functions:Image thresholding invariant. *Pattern Recognition Letters, 29*, 525–536.

Bustince, H., & Burillo, P. (1996). Vague sets are intuitionistic fuzzy sets. *Fuzzy Sets and Systems, 79(3)*, 403–405.

Cagman, N., Enginoglu, S., & Citak, F. (2011). Fuzzy Soft Set Theory and its Applications. Iranian Journal of Fuzzy Systems, 8(3), 137-147.

Campos, V., Marti, R., Sanchez-Oro, J., & Duarte, A. (2013). GRASP with Path Relinking for the Orienteering Problem. *Journal of the Operational Research Society, 2013,* 1–14.

Capella, J. V., Bonastre, A., Ors, R., & Peris, M. (2013). In line river monitoring of nitrate concentration by means of a Wireless Sensor Network with energy harvesting. *Sensors and Actuators B*, *177*, 419– 427.

Carlyle, W. M., Royset, J. O., & Wood, R. K. (2008). Lagrangian Relaxation and Enumeration for Solving Constrained Shortest-Path Problems. *Networks, 52(4),* 256-270.

Chang, P.T., & Lee, E.S. (1994). Ranking fuzzy sets based on the concept of existence. *Computer Mathematics with Applications, 27*, 1–21.

Chang, P.T., & Lee, E.S. (1999). Fuzzy decision networks and deconvolution. *Computer Mathematics with Applications, 37*, 53–63.

Chao, I., Golden, B., & Wasil, E. (1996). Theory and methodology – a fast and effective heuristic for the orienteering problem. *European Journal of Operational Research*, *88*, 475–489.

Chauhan, S., Shatanawi, W., Kumar, S., & Radenovic, S. (2014). Existence and uniqueness of fixed points in modified intuitionistic fuzzy metric spaces. *Journal of Nonlinear Science and Applications, 7*, 28-41.

Chen, C. C., & Tang, H. C. (2008). Ranking of nonnormal p-norm trapezoidal fuzzy numbers with integral value. *Computer and Mathematics with Applications, 56*, 2340-2346.

Cheng, C. H. (1998). A new approach for ranking fuzzy numbers by distance method. *Fuzzy Sets and Systems, 95*, 307-317.

Chen, G., & Pham, T.T. (2001). *Introduction to Fuzzy Sets, Fuzzy Logic, and Fuzzy Control Systems*. CRC Press LLC, New York, Washington D. C.

Chen, S.H. (1985). Ranking fuzzy numbers with maximizing set and minimizing set. *Fuzzy Sets and Systems, 17*, 113–129.

Chen, S., Song, M., & Sahni, S. (2008). Two Techniques for Fast Computation of Constrained Shortest Paths. *IEEE/ACM Transactions on Networking, 16(1)*, 105-115.

Chou, S.Y., Dat, L.Q., & Yu, V.F. (2011). A revised method for ranking fuzzy numbers using maximizing set and minimizing set. *Computers & Industrial Engineering, 61*, 1342-1348.

Chuang, T.N., & Kung, J.Y. (2005). The fuzzy shortest path length and the corresponding shortest path in a network. *Computers and Operations Research, 32*, 1409-1428.

Chuang, T.N., & Kung, J.Y. (2006). A new algorithm for the discrete fuzzy shortest path problem in a network. *Appl. Math. Comput., 174*, 1660-1668.

Chu, T. C., & Tsao, C. T. (2002). Ranking fuzzy numbers with an area between the centroid point and the original point. *Computer and Mathematics with Applications, 43*, 111-117.

Coker, D., & Demirci, M. (1995). On intuitionistic fuzzy points. *Notes on Intuitionistic Fuzzy Sets, 1(2)*, 79–84.

Cormen, T.H., Leiserson, C.E., Rivest, R.L., & Stein C. (1990). *Introduction to Algorithms*. McGraw Hill Book Company.

Deng, Y., Chen, Y., Zhang, Y., & Mahadevan, S. (2012). Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment. *Applied Soft Computing, 12(3)*, 1231–1237.

Deng, Z. K. (1982). Fuzzy pseudo-metric spaces. *Journal of Mathematical Analysis and Applications*, *86*, 74-95.

Deo, N. (1974). *Graph Theory with Applications to Engineering and Computer Science*. Prentice Hall of India Pvt. Ltd., New Delhi.

De, P.K., & Das, D. (2014). A Study on Ranking of Trapezoidal Intuitionistic Fuzzy Numbers. *International Journal of Computer Information Systems and Industrial Management Applications, 6*, 437-444.

Dongrui, Wu. (June, 2012). Twelve considerations in choosing between Gaussian and trapezoidal membership functions in interval type-2 fuzzy logic controllers. *2012 IEEE International Conference on Fuzzy Systems*, Brisbane, Australia.

Dou, Y., Zhu, L., & Wang, H. S. (2012). Solving the fuzzy shortest path problem using multi-criteria decision method based on vague similarity measure. *Applied Soft Computing, 12*, 1621-1631.

Dubois, D., & Prade, H. (1978). Operations on Fuzzy Numbers. *International Journal of Systems Science, 9*, 613-626.

Dubois, D., & Prade, H. (1979). Fuzzy Real Algebra: Some Results. *Fuzzy Sets and Systems, 2*, 327-348.

Dubois, D., & Prade, H. (1980). *Fuzzy Sets and Systems: Theory and Applications*. New York: Academic Press.

Du, D.Z., Smith, J. M., & Rubinstein, J. H. (2000). *Advances in Steiner Trees*. Kluwer Academic Publishers, The Netherlands.

Dumitrescu, I., & Boland, N. (2003). Improved Preprocessing, Labeling and Scaling Algorithms for the Weight-Constrained Shortest Path Problem. *Networks, 42(3),* 135–153.

Edmonds, J. & Johnson, E.L. (1973). Matching Euler tours and the Chinese postman problem. *Mathematical Programming, 5*, 88–124.

Elizabeth, S. & Sujatha, L. (2011). Fuzzy Shortest Path Problem Based on Index Ranking. *Journal of Mathematics Research, 3(4)*, 80-88.

Faloutsos, M., Faloutsos, P., & Faloutsos, C. (1999). On Power-Law Relationships of the Internet Topology. *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, New York, USA, 251-262.

Fischetti, M., Salazar, J., & Toth, P. (1998). Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing*, *10*, 133–148.

Fister, Jr. I., Yang, X. S., Fister, I., Brest, J., & Fister, D. (2013). A Brief Review of Nature-Inspired Algorithms for Optimization. *Elektrotehniski Vestnik, 80(3),* 1–7.

Fomin, F. V., & Lingas, A. (2002). Approximation algorithms for time-dependent orienteering. *Information Processing Letters, 83*, 57–62.

Gao, J., & Lu, M. (2005). Fuzzy quadratic minimum spanning tree problem. *Applied Mathematics and Computation, 164*, 773–788.

Gau, W. L., & Buehrer, D. J. (1993). Vague sets. *IEEE Transactions on Systems, Man, and Cybernetics, 23,* 610-614.

Gendreau, M., Laporte, G., Semet, F. (1998). A tabu search heuristic for the undirected selective travelling salesman problem. *European Journal of Operational Research, 106*, 539–545.

George, A., & Veeramani, P. (1994). On some results in fuzzy metric spaces. *Fuzzy Sets and Systems, 64*, 395-399.

George, A., & Veeramani, P. (1997). On some results of analysis for fuzzy metric spaces. *Fuzzy Sets and Systems, 90*, 365-368.

Golden, B., Levy, L., & Vohra, R. (1987). The orienteering problem. *Naval Research Logistics*, *34*, 307–318.

Grabiec, M. (1988). Fixed points in fuzzy metric spaces. *Fuzzy Sets and Systems*, *27*, 385-389.

Grauel, A., & Ludwig, L.A. (1999). Construction of differentiable membership functions. *Fuzzy Sets and Systems, 101(2)*, 219–225.

Grzegorzewski, P. (Sept., 2003). Distances and orderings in a family of intuitionistic fuzzy numbers. *Proceedings of the 3rd Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT, 2003)*, Germany, 223-227.

Handler, G.Y., & Zang, I. (1980). A dual algorithm for the constrained shortest path problem. Networks, 10, 293–310.

Hanss, M. (1999). On the implementation of fuzzy arithmetical operations for engineering problems. *Proceedings of the 18th International Conference of the North American fuzzy Information Processing Society – NAFIPS'99*, New York, USA, 462-466.

Hanss, M. (2005). *Applied Fuzzy Arithmetic, An Introduction with Engineering Applications*. Netherlands: Springer Berlin Heidelberg New York.

Hanss, M., & Willner, K. (2000). A fuzzy arithmetical approach to the solution of finite element problems with uncertain parameters. *Mechanics Research Communications, 27*, 257-272.

Hassin, R. (1992). Approximation Schemes for the Restricted Shortest Path Problem. Mathematics of Operation Research, 17(1), 36-42.

Hernandes, F., Lamata, M.T, Verdegay, J.S. & Yamakami, A. (2007). The shortest path problem on networks with fuzzy parameters, *Fuzzy Sets and Systems, 158*, 1561 – 1570.

Horowitz, E., Sahni, S., & Rajasekaran, S. (1999). *Computer Algorithms/C++*. Galgotia Publications Pvt. Ltd., New Delhi.

Ibarra, O., & Kim, C. (1975). Fast Approximation Algorithms for the Knapsack and Sum of Subsets Problems. *Journal of the Association for Computing Machinery, 22*, 463-468.

Itoh, T., & Ishii, H. (1996). An approach based on necessity measure to the spanning tree problem. *Journal of the Operations Research Society of Japan, 39*, 247–257.

Jain, R. (1976). Decision making in the presence of fuzzy variables. *IEEE Transactions on Systems, Man and Cybernetics, 6(10)*, 698–703.

Janiak, A., & Kasperski, A. (2008). The minimum spanning tree problem with fuzzy costs. *Fuzzy Optimization and Decision Making, 7(2)*, 105-118.

Jarník, V. (1930). O jistém problému minimálním (About a certain minimal problem). *Práce Moravské Přírodovědecké Společnosti (in Czech), 6*, 57–63.

Jarray, F. (2011). Discrete Tomography and Fuzzy Integer Programming. *Iranian Journal of Fuzzy Systems, 8*, 41-48.

Jimenez, M., Arenas, M., Bilbao, A., & Rodriguez, M. V. (2007). Linear programming with fuzzy parameters: An interactive method resolution. *European Journal of Operational Research, 177*, 1599–1609.

Ji, X., Iwamura, K., & Shao, Z. (2007). New models for shortest path problem with fuzzy arc lengths. *Applied Mathematical Modelling, 31,* 259–269.

Johnson, D., Minkoff, M., & Phillips, S. (2000). The prize collecting steiner tree problem: Theory and practice. *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, 760–769.

Joksch, H. (1966). The shortest route problem with constraints. *Journal of Mathematical Analysis and Applications, 14(1)*, 191-197.

Juttner, A., Szviatovszki, B., Mecs, I., & Rajko, Z. (2001). Lagrange relaxation based method for the QoS routing problem. *Proceedings of the IEEE INFOCOM*, 859–868.

Kahng, A. B., & Robins, G. (1990). A new family of steiner tree heuristics with good performance: The iterated 1-steiner approach. *Proceedings of the IEEE International Conference on Computer-Aided Design*, Santa Clara, CA, 428 431.

Kaleva, O., & Seikkala, S. (1984). On fuzzy metric spaces. *Fuzzy Sets and Systems*, *12*, 225-229.

Kashtiban, M.M., Khoei, A., & Hadidi, Kh. (2008). Optimization of rational-powered membership functions using extended Kalman filter. *Fuzzy Sets and Systems, 159(23)*, 3232–3244.

Katsaras, A. K. (1984). Fuzzy topological vector spaces. II. *Fuzzy Sets and Systems, 12(2)*, 143-154.

Kaymak, U., & Sousa, J.M. (2001). *Weighted Constraints in Fuzzy Optimization*. Publications in the Report Series Research in Management, ERIM Research Program: "Business Processes, Logistics and Information Systems", 1-21. Retrieved from: http://repub.eur.nl/res/pub/85/erimrs20010403174009.pdf

Klein, C.M. (1991). Fuzzy shortest paths. *Fuzzy Sets and Systems, 39*, 27-41.

Klir, G.J., & Yuan, B. (1997). *Fuzzy Sets and Fuzzy Logic Theory and Applications*. New Delhi: Prentice-Hall of India Private Limited.

Kokash, N. (2005). *An introduction to heuristic algorithms*. Technical report, Department of Informatics and Telecommunications, University of Trento, Italy.

Korkmaz, T., & Krunz, M. (2001). Multi-constrained optimal path selection. *Proceedings of the IEEE INFOCOM*, 834–843.

Kou, L., Markowsky, G., & Berman, L. (1981). A Fast Algorithm for Steiner Trees. *Acta Informatica, 15*, 141-145.

Kruskal, J.B. (1956). On the shortest spanning subtree of a graph and the travelling salesman problem. *Proceedings of the American Mathematical Society*, 48 – 50.

Kumar, A., Singh, P., Kaur, A., & Kaur, P. (2011). A new approach for ranking nonnormal p-norm trapezoidal fuzzy numbers. *Computer and Mathematics with Applications, 61*, 881-887.

Kung, J.Y., Chuang, T.N., & Lin, C.T. (October, 2006). Decision making on network problem with fuzzy arc lengths. *Proceedings of the IMACS Multiconference on Computational Engineering in Systems Applications (CESA)*, Beijing, China, 578-580.

Kwan, M.K. (1962). *Graphic Programming Using Odd or Even Points. Chinese Math., 1*, 273-277.

Lee, Y. D., & Chung, W. Y. (2009). Wireless sensor network based wearable smart shirt for ubiquitous health and activity monitoring. *Sensors and Actuators B*, *140*, 390–395.

Lefebvre, M., Puget, J., & Vilim, P. (2011). Route finder: efficiently finding k shortest paths using constraint programming. *Principles and Practice of Constraint Programming–CP, 2011,* 42–53.

Liang, Y., Konak, S. K., & Smith, A. (2002). Meta heuristics for the orienteering problem. *Proceedings of the 2002 Congress on Evolutionary Computation*, Hawaii, Honolulu, 384–389.

Li. D. F. (2010). A ratio ranking method of triangular intuitionistic fuzzy numbers and its application to madm problems. *Computer and Mathematics with Applications*, *60*, 1557–1570.

Liu, B. (2002). *Theory and Practice of Uncertain Programming*. Physica-Verlag, Heidelberg.

Liu, B. (2004). *Uncertainty Theory: An Introduction to its Axiomatic Foundations*. Springer Verlag, Berlin.

Lorenz, D., & Raz, D. (2001). A Simple Efficient Approximation Scheme for the Restricted Shortest Paths Problem. *Operations Research Letters, 28,* 213-219.

Loucks, D. P., & Beek, E. V. (2005). *Water Resources Systems Planning and Management: An Introduction to Methods, Models and Applications*. UNESCO Publishing, 135-144.

Luby, M., & Ragde, P. (1989). A Bidirectional Shortest-Path Algorithm with Good Average-Case Behavior. *Algorithmica, 4*, 551-567.

Luo, G., Huang, K., Wang, J., Hobbs, C., & Munter, E. (June, 1999). Multi-QoS Constraints Based Routing for IP and ATM Networks. *Proceedings of the IEEE Workshop on QoS Support for Real-Time Internet Applications*, Vancouver Canada.

Mahapatra, G.S., & Roy, T.K. (2013). Intuitionistic Fuzzy Number and Its Arithmetic Operation with Application on System Failure. *Journal of Uncertain Systems, 7(2),* 92-107.

Maji, P. K., Biswas, R., & Roy, A. R. (2003). Soft set theory. *Comput. Math. Appl., 45*, 555-562.

Maji, P. K., Roy, A. R., & Biswas, R. (2002). An application of soft sets in a decision making problem. *Comput. Math. Appl., 44*, 1077-1083.

Marghitu, D.B., & Dupac, M. (2012). *Advanced Dynamics: Analytical and Numerical Calculations with MATLAB*. Springer Science+Business Media, LLC, (ch. 2, 73-141).

Meguerdichian, S., Koushanfar, F., Qu, G., & Potkonjak, M. (2001). Exposure In Wireless Ad-Hoc Sensor Networks. *Proceedings of the 7th annual international conference on Mobile computing and networking*, New York, USA, 139-150.

Menger, K. (1942). Statistical metrics. *Proc. Nat. Acad. Sci.*, *28*, 535-537.

Mitchell, H. B. (2004). Ranking intuitionistic fuzzy numbers. *International Journal of Uncertainity, Fuzziness and Knowledge-Based Systems*, *12*, 377–386.

Mohanty, S. P., Biswal, S., & Pradhan, G. (2012). Minimum Spanning Tree in Fuzzy Weighted Rough Graph. *International Journal of Engineering Research and Development, 1(10)*, 23-28.

Molodtsov, D. A. (1999). Soft set theory-first results. *Comput. Math. Appl., 37*, 19-31.

Moore, R.E. (1966). *Interval Analysis*. NJ: Prentice-Hall, Englewood Cliffs.

Nan, J. X., & Li, D.F. (2010). A lexicographic method for matrix games with payoffs of triangular intuitionistic fuzzy numbers. *International Journal of Computational Intelligence Systems*, *3*, 280–289.

Nayagam, V. L., Vankateshwari, G., & Sivaraman, G. (2008). Ranking of intuitionistic fuzzy numbers. *Proceedings of the IEEE International Conference on Fuzzy Systems*, 1971–1974.

Nehi, H. M. (2010). A new ranking method for intuitionistic fuzzy numbers. *International Journal of FuzzySystems, 12*, 80–86.

Nejad, A.M., & Mashinchi, M. (2011). Ranking of fuzzy numbers based the areas on the left and the right sides of fuzzy number. *Computer and Mathematics with Applications, 61*, 431-442.

Ostrowski, K., & Koszelew, J. (2011). The Comparison of Genetic Algorithms which Solve Orienteering Problem using Complete and Incomplete Graph. *Informatyka, 8*, 61-77.

Parandin, N., Kaveh, Z., & Kousari, P. (2014). A New Method for Ranking Normal and Non-normal Fuzzy Numbers. *Journal of Interpolation and Approximation in Scientific Computing, 2014*, 1-12.

Park, J. H. (2004). Intuitionistic fuzzy metric spaces. *Chaos, Solitons and Fractals, 22*, 1039–1046.

Parvathi, R., Thilagavathi, S., Thamizhendhi, G., & Karunambigai, M. G. (2014). Index matrix representation of intuitionistic fuzzy graphs. *Notes on Intuitionistic Fuzzy Sets, 20(2)*, 100–108.

Pawlak, Z. (1982). Rough Sets. *International Journal of Computer and Information Sciences, 11*, 341-356.

Prim, R.C. (1957). Shortest connection networks and some generalizations. *Bell Systems Technology Journal, 36*, 1389 – 1401.

Ramesh, R., & Brown, K. (1991). An efficient four-phase heuristic for the generalized orienteering problem. *Computers and Operations Research*, *18*, 151–165.

Ramik, J. (2001). *Soft Computing: Overview and Recent Developments in Fuzzy Optimization*. Retrieved from: http://irafm.osu.cz/research_report/118_softco01.pdf

Ramli, N., & Mohamad, D. (2009). A Comparitive Analysis of Centroid Methods in Ranking Fuzzy Numbers. *European Journal of Scientific Research, 28(3)*, 492-501.

Rao, P. P. B, & Shankar, N. R. (2011). Ranking Fuzzy Numbers with a Distance Method using Circumcenter of Centroids and an Index of Modality. *Advances in Fuzzy Systems, 2011*, 1-7.

Ravindran, R., Thulasiraman, K., Das, A., Huang, K., Luo, G., & Xue, G. (2002). Quality of Services Routing: Heuristics and Approximation Schemes with a Comparative Evaluation. *Proceedings of the ISCAS, 2002*.

Razali, N. M., & Geraghty, J. (July 6 – 8, 2011). Genetic Algorithm Performance with Different Selection Strategies in Solving TSP. *Proceedings of the World Congress on Engineering 2011*, London, U.K., 1134-1139.

Royset, J., Carlyle, W., & Wood, R. (2009). Routing military aircraft with a constrained shortest-path algorithm. *Military Operations Research, 14(3),* 31–52.

Saeidifar, A. (2011). Application of weighting functions to the ranking of fuzzy numbers. *Computers and Mathematics with Applications, 62*, 2246-2258.

Sahni, S. (1977). General Techniques for Combinatorial Approximation. *Operations Research, 25*, 920-936.

Santhosh, C. P., & Ramakrishnan, T. V. (2011). Norm and Inner Product on Fuzzy Linear Spaces over Fuzzy Fields. *Iranian Journal of Fuzzy Systems, 8(1)*, 135-144.

Sardar, S. K., Mandal, M., & Majumder, S. K. (2011). Intuitionistic Fuzzy Points in Semigroups. *International Scholarly and Scientific Research & Innovation, 5 (3)*, 1343-1348.

Schilde, M., Doerner, K. F., Hartl, R. F., & Kiechle, G. (2009). Metaheuristics for the bi-objective orienteering problem. *Swarm Intell., 3*, 179-201.

Seda, M. (2008). Fuzzy Shortest Paths Approximation for Solving the Fuzzy Steiner Tree Problem in Graphs. *International Journal of Computer, Electrical, Automation, Control and Information Engineering, 2(2)*, 437-441.

Shannon, A., & Atanassov, K. (Sept., 1994). A first step to a theory of the intuitionistic fuzzy graphs. *Proceedings of the First Workshop on Fuzzy Based Expert Systems*, Sofia, 59–61.

Shepherd, R., Beirne, S., Lau, K. T., Corcoran, B., & Diamond, D. (2007). Monitoring chemical plumes in an environmental sensing chamber with a wireless chemical sensor network. *Sensors and Actuators B*, *121*, 142–149.

Shukla, K. K., & Sah, S. (2013). Construction and maintenance of virtual backbone in wireless networks. *Wireless Networks, 19*, 969-984.

Sivaraj, R., & Ravichandran, T. (2011). A Review of Selection Methods in Genetic Algorithm. *International Journal of Engineering Science and Technology (IJEST), 3(5),* 3792-3797.

Sujatha, L., & Elizabeth, S. (2011). Fuzzy Shortest Path Problem Based on Similarity Degree. *Applied Mathematical Sciences, 5(66)*, 3263-3276.

Su, J., & Li, A. (2009a). Approach to the Shortest Path with Fuzzy Constraints by Simulated Annealing Algorithm. *Global Congress on Intelligent System, 1*, 516 – 520.

Su, J., & Li, A. (2009b). Simulated Annealing Approach for the Constrained Shortest Path with Fuzzy Arc and Node Weights. *World Congress on Computer Science and Information Engineering, 4,* 754-758.

Tasgetiren, M. (2001). A genetic algorithm with an adaptive penalty function for the orienteering problem. *Journal of Economic and Social Research*, *4*, 1–26.

Tsiligirides, T. (1984). Heuristic methods applied to orienteering. *Journal of the Operational Research Society, 35*, 797–809.

Vansteenwegen, P., Souffriau, W., & Oudheusden, D. V. (2011). The orienteering problem: A survey. *European Journal of Operational Research, 209*, 1–10.

Varghese, A., & Kuriakose, S. (2012). Centroid of an intuitionistic fuzzy number. *Notes on Intuitionistic Fuzzy Sets, 18(1)*, 19-24.

Walczak, B. & Massart, D.L. (1999). Rough sets theory. *Chemometrics and Intelligent Laboratory Systems, 47*, 1–16.

Wang, Q., Sun, X., Golden, B., & Jia, J. (1995). Using artificial neural networks to solve the orienteering problem. *Annals of Operations Research*, *61*, 111–120.

Wang, X., & Kerre, E. E. (2001a). Reasonable properties for the ordering of fuzzy quantities (I). *Fuzzy Sets and Systems, 118(3)*, 375–385.

Wang, X., & Kerre, E. E. (2001b). Reasonable properties for the ordering of fuzzy quantities (II). *Fuzzy Sets and Systems, 118(3)*, 387–405.

Wang, Y. J., & Lee, S. H. (2008). The revised method of ranking fuzzy numbers with an area between the centroid and original points. *Computer and Mathematics with Applications, 55*, 2033-2042.

Wang, Z. X., Liu, Y. T., Fan, Z. P., & Feng, B. (2009). Ranking L-R fuzzy number based on deviation degree. *Inform.Sci, 179,* 2070-2077.

Williamson, D.P., & Shmoys, D.B. (2010). *The Design of Approximation Algorithms.* Cambridge University Press.

Xiangning, F., & Yulin, S. (2007). Improvement on LEACH Protocol of Wireless Sensor Network. *Proceedings of International Conference on Sensor Technologies and Applications*, 260-264.

Xiao, Y., Thulasiraman, K., Xue, G., & Juttner, A. *The Constrained Shortest Path Problem: Algorithmic Approaches and an Algebraic Study with Generalization*. Retrieved from: http://www.cs.ou.edu/~thulasi/Misc/AKCE%20October%2025.pdf

Xia, Z. Q., & Guo, F. F. (2004). Fuzzy Metric Spaces. *J. Appl. Math. and Computing*, 16(1-2), 371 – 381.

Xue, G. (2000). Primal-dual algorithms for computing weight-constrained shortest paths and weight-constrained minimum spanning trees, *Proceedings of the IEEE IPCCC'00*, 71–277.

Yager, R. R. (1980). On a general class of fuzzy connectives. *Fuzzy Sets and Systems, 4*, 235-242.

Yang, X.S. (2012). Flower Pollination Algorithm for Global Optimization. *LNCS 7445,* 240–249.

Yang, X.S., Karamanoglu, M. & He, X. (2013). Multi-objective Flower Algorithm for Optimization. *Proceedings of the International Conference on Computational Science, ICCS 2013*, 861 – 868.

Yao, J. S., & Lin, F. T. (2000). Fuzzy critical path method based on signed distance ranking fuzzy numbers. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 30*, 76-82.

Yao, J.S., & Lin, F.T. (2003). Fuzzy Shortest-Path Network Problems With Uncertain Edge Weights. *Journal of Information Science and Engineering, 19*, 329-351.

Yao, J. S., & Wu, K. (2000). Ranking fuzzy numbers based on decomposition principle and signed distance. *Fuzzy sets and systems, 116*, 275-288.

Ye, J. (2011). Expected value method for intuitionistic trapezoidal fuzzy multicriteria decision-making problems. *Expert Systems with Applications, 38*, 11730-11734.

Yu, J.R., & Wei, T.H. (2007). Solving the Fuzzy Shortest Path Problem by Using a Linear Multiple Objective Programming. *Journal of the Chinese Institute of Industrial Engineers, 24(5),* 360-365.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control, 8(3)*, 338–353.

Zhao, F., & Guibas, L. J. (2004). *Wireless Sensor Networks: An Information Processing Approach*. San Francisco.

Zhuiykov, S. (2012). Solid-state sensors monitoring parameters of water quality for the next generation of wireless sensor networks. *Sensors and Actuators B*, *161*, 1– 20.

Zimmermann, H. J. (2010). Fuzzy set theory. *WIREs Computational Statistics, 2*, 317-332.

List of Publications

Journals

1. **M. Verma, K. K. Shukla, "Fuzzy Metric Space Induced by Intuitionistic Fuzzy Points and Its Application to the Orienteering Problem", IEEE Transactions on Fuzzy Systems (SCI Indexed, IF-8.746). Available online as early access.**

2. M. Verma, K. K. Shukla, " A Greedy Algorithm for Fuzzy Shortest Path Problem using Quasi- Gaussian Fuzzy Weights", International Journal of Fuzzy System Applications, Vol. 3(2), pp. 55-70, 2013.

3. M. Verma, M. Gupta, B. Pal, K. K. Shukla, "Roulette Wheel Selection based Heuristic Algorithm for the Orienteering Problem", International Journal of Computers and Technology, Vol. 13(1), pp. 4127-4145, 2014.

4. M. Verma, K. K. Shukla, " Fuzzy Minimum Spanning Tree Problem - A Greedy Algorithm using Quasi- Gaussian Fuzzy Weights", Journal of Software Engineering Tools and Technology Trends, Vol. 1(3), pp. 15-22, 2014.

5. M. Verma, K.K. Shukla, "Application of Fuzzy Optimization to the Orienteering Problem", Advances in Fuzzy Systems, Vol. 2015, pp.1-12, 2015.

Conferences

1. M. Verma, K. K. Shukla, "A New Algorithm for solving Fuzzy Constrained Shortest Path Problem using Intuitionistic Fuzzy Numbers", In Proc. of 2nd International Conference On Advances in Computing, Electronics and Electrical Technology (CEET-2014), pp. 1-5, Kuala Lumpur, Malaysia, 2014. [Also published in International Journal of Artificial Intelligence and Neural Networks, Vol. 5(1), pp. 38-42, 2015].

2. M. Verma, K. K. Shukla, " Fuzzy Constrained Shortest Path Algorithm using Circumcenter of Centroids", In Proc. of 3rd IEEE International Advance Computing Conference (IACC-2013), pp. 657-660, Ghaziabad, U.P., 2013.

3. M. Verma, K. K. Shukla, "Evaluation of Ranking Methods for the Constrained Fuzzy Shortest Path Problem", In Proc. of 2nd International Conference on Emerging Research in Computing, Information, Communication and Applications (ERCICA-2014), pp. 812-822, Bangalore, Karnataka, 2014.

4. M. Verma, B. Pal, M. Gupta, K. K. Shukla, "A Stochastic Greedy Heuristic Algorithm for the Orienteering Problem", In Proc. of 5th IEEE International Conference on Computer and Communication Technology (ICCCT-2014), pp. 59-65, Allahabad, U.P., 2014.

5. M. Verma, K. K. Shukla, " A Solution of the Fuzzy Steiner Tree using Quasi- Gaussian Fuzzy Weights", In Proc. of International Conference on Artificial Intelligence and Soft Computing (AISC-2012), pp. 75-80, Varanasi, U.P., 2012.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2015.2452948, IEEE Transactions on Fuzzy Systems

Fuzzy Metric Space Induced by Intuitionistic Fuzzy Points and Its Application to the Orienteering Problem

Madhushi Verma and K. K. Shukla

Abstract— **In this paper, a new definition for Atanassov intuitionistic fuzzy metric space is presented using the concept of Atanassov intuitionistic fuzzy point and Atanassov intuitionistic fuzzy scalars. The distance metric introduced here is then applied to an interesting problem called the orienteering problem that finds application in several industries like the home delivery system, robot path planning, tourism industry etc. and in each of these practical applications, the two parameters involved i.e. score and distance travelled, as well as the position of locations cannot be predicted precisely. To tackle these uncertainties we use trapezoidal Atanassov intuitionistic fuzzy numbers for representing the parameter score. The uniqueness of this paper is the consideration of uncertainty in the position of a city or a location and handling this type of uncertainty using the idea of Atanassov intuitionistic fuzzy points and the distance metric between Atanassov intuitionistic fuzzy points. Further, a method for ranking trapezoidal Atanassov intuitionistic fuzzy numbers has been presented and used for modelling the scores.**

Index Terms— **Intuitionistic fuzzy metric space, intuitionistic fuzzy point, intuitionistic fuzzy scalars, orienteering problem, ranking, trapezoidal intuitionistic fuzzy numbers.**

I. Introduction

THE concept of fuzzy set, which deals with the non-probabilistic uncertainty, was introduced by Zadeh in 1965 [1]. Since then many researchers have defined the concept of fuzzy metric that has been widely used in the fields of pattern recognition, routing, scheduling, transportation, fuzzy optimization etc. [2]-[4]. A metric space can be defined as (X, d) where X is a set of points and d is a metric on X i.e. a function that defines the distance between every pair of elements belonging to X. Two methods of defining fuzzy metric space include: (1) use of fuzzy numbers for defining a metric in an ordinary space and (2) use of real numbers for measuring the distance between fuzzy sets. Using these two approaches, fixed point theorems were given [5], Hausdorff topology [6]-[7], and fuzzy normed spaces [8]-[9] were defined by several authors.

The idea of Atanassov intuitionistic fuzzy sets (AIFS) which is a generalization of fuzzy sets was proposed by Atanassov [10]. An AIFS K in U can be denoted as $K = \{\langle x, \mu_K(x), \nu_K(x)\rangle | x \in U\}$ where $\mu_K(x), \nu_K(x) \in [0,1]$ with the constraint that $0 \leq \mu_K(x) + \nu_K(x) \leq 1 \quad \forall x \in U$ and represents the degree of membership and non-membership respectively. Though AIFS is an extension of fuzzy sets proposed by Zadeh, there are several situations that can be modelled using AIFS but cannot be represented using ordinary fuzzy sets. For example, let us consider a travelling salesman who has a limit on distance travelled that prevents him from visiting all the cities but he has some information about the cities where maximum sales can take place. So the aim here is to maximize the total sales within the limit on distance travelled. If E is a set consisting of all those cities which can be visited by the salesman and $m \in E$ represents a city visited by the salesman, then the membership degree of the cities visited by the salesman can be stated as $\mu(m)$. In this case, an ordinary fuzzy set can be used but this ordinary fuzzy set cannot represent the situation where the need is to evaluate the number of cities that could not be visited by the salesman. For this, we need AIFS where the degree of non-membership can be computed as $\nu(m) = 1 - \mu(m)$. Moreover, there can be a situation where the salesman visited a city but could not sell his product due to unavailability of the customer or shop being closed, then such situations can be represented through the hesitancy degree as $\pi(m) = 1 - \big(\mu(m) + \nu(m)\big)$. Also, there are a few operators like the modal operators that can be defined for AIFS but not for ordinary fuzzy sets. These operators provide for a detailed estimation of the available information. Also, AIFS is more powerful as it allows geometrical interpretation and helps in managing the existing uncertainty in a much better way [11]. In 1994, Shannon and Atanassov introduced the idea of Atanassov intuitionistic fuzzy graphs (AIFGs) [16]. Some properties of AIFGs were presented in [17]. In 2014, a modified definition of AIFG and some operations on AIFG were proposed by Parvathi et al [18]. Later the concept of Atanassov intuitionistic fuzzy metric space was introduced using continuous t-norms and continuous t-conorms [12]. Some theorems and properties were also stated for Atanassov intuitionistic fuzzy metric space [19].

The first author would like to acknowledge the financial support by IIT (BHU) in terms of teaching assistantship and research support grant.

Madhushi Verma, Research Scholar, is with the Department of Computer Science and Engineering, IIT (BHU), Varanasi (email: madhushi.rs.cse@itbhu.ac.in).

K. K. Shukla, Professor, is with the Department of Computer Science and Engineering, IIT (BHU), Varanasi (email: kkshukla.cse@itbhu.ac.in).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2015.2452948, IEEE Transactions on Fuzzy Systems

In 2004, Xia and Guo suggested a method of defining fuzzy metric spaces. They used the concept of fuzzy points and fuzzy scalars to measure the distance between the fuzzy points [13]. The concept of Atanassov intuitionistic fuzzy point (AIFP) was introduced by Coker and Demirci in [14]. The idea of AIFP was then used by researchers to prove some relations and theorems [20]-[21]. The concept of AIFP can be used to study some general structures like those introduced in [22]-[23]. In this paper, we use the concept of AIFP and Atanassov intuitionistic fuzzy scalars for proposing a new definition of the Atanassov intuitionistic fuzzy metric space. The proposed definition of distance metric is then applied to the orienteering problem resulting in the formulation of the Atanassov intuitionistic fuzzy orienteering problem (AIFOP) where for a given graph $G(V,E)$, the task is to compute a Hamiltonian path P that connects the stated source(v_1) and target(v_N) along with a subset of vertices (V') of the vertex set V and also satisfies the upper limit on the distance covered (D_{max}). The Atanassov intuitionistic fuzzy version of the problem has been considered for the first time and we deal with both, the uncertainty in the parameter score using trapezoidal Atanassov intuitionistic fuzzy number and the uncertainty in the position of a city/location using Atanassov intuitionistic fuzzy points. Further, an algorithm for solving AIFOP has been suggested in section IV (B) of this paper. An Atanassov intuitionistic fuzzy point $c(\lambda,\alpha)$ can be defined in the following form [14]:

$c(\lambda,\alpha) = \{\langle x, c_\lambda, 1-c_{1-\alpha}\rangle | x \in U\}$ (Here, $c \in U$ is the support of $c(\lambda,\alpha)$ and λ and α represents the degree of membership and degree of non-membership respectively of $c(\lambda,\alpha)$).

Where U is a nonempty set and $c \in U$. $\lambda \in (0,1]$ and $\alpha \in [0,1)$ are two real numbers such that $\lambda + \alpha \leq 1$.

In this paper the notation used to signify an Atanassov intuitionistic fuzzy point is (x,λ,i). $P_{IF}(U)$ denotes the set of all Atanassov intuitionistic fuzzy points defined on U. When $U = R$, Atanassov intuitionistic fuzzy points are called Atanassov intuitionistic fuzzy scalars and $S_{IF}(R)$ denotes the set of all Atanassov intuitionistic fuzzy scalars. An Atanassov intuitionistic fuzzy point is said to belong to an Atanassov intuitionistic fuzzy set K if

$$K = \{(x,\lambda,i) | \mu_K(x) \geq \lambda, \nu_K(x) \leq i\}$$

II. Atanassov Intuitionistic Fuzzy Metric Space

In this section a few necessary definitions are presented:

Definition 1: Let (a,λ,i) and (b,γ,j) be two Atanassov intuitionistic fuzzy scalars then we say that:

(1) $(a,\lambda,i) \succcurlyeq (b,\gamma,j)$ if $a > b$ or $(a,\lambda,i) = (b,\gamma,j)$.
(2) (a,λ,i) is said to be no less than (b,γ,j) if $a \geq b$ denoted by $(a,\lambda,i) \succ (b,\gamma,j)$ or $(b,\gamma,j) \prec (a,\lambda,i)$.
(3) (a,λ,i) is said to be non-negative if $a \geq 0$. The set of all the non-negative Atanassov intuitionistic fuzzy scalars is denoted by $S_{IF}^{+}(R)$.

Here, both the operators $\succ$ and $\succcurlyeq$ denote partial ordering.

Definition 2: Let U be a nonempty set and $d_{IF}: P_{IF}(U)\ X\ P_{IF}(U) \to S_{IF}^{+}(R)$ be a mapping. For any $\{(x,\lambda,i),(y,\gamma,j),(z,\rho,l)\} \subset P_{IF}(U)$, if d_{IF} satisfies the following three conditions:

(1) *Non Negative:* $d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) \geq 0$ and $d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) = 0$ iff $x = y,\ \lambda = \gamma = 1, i = j = 0$ (in $d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) = 0$, 0 denotes the Atanassov intuitionistic fuzzy scalar with membership degree 1 and non-membership degree 0).
(2) *Symmetric:* $d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) = d_{IF}\big((y,\gamma,j),(x,\lambda,i)\big)$.
(3) *Triangle Inequality:* $d_{IF}\big((x,\lambda,i),(z,\rho,l)\big) \prec d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) + d_{IF}\big((y,\gamma,j),(z,\rho,l)\big)$.

Here, the summation is defined as:

$(x,\lambda,i) + (y,\gamma,j) = (x+y, min\{\lambda,\gamma\}, max\{i,j\})$.

Then, $(P_{IF}(U), d_{IF})$ is called an Atanassov intuitionistic fuzzy metric space. Here, $(x,\lambda,i),(y,\gamma,j),(z,\rho,l)$ are Atanassov intuitionistic fuzzy points, d_{IF} is the Atanassov intuitionistic fuzzy metric defined in $P_{IF}(U)$ and $d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big)$ is the Atanassov intuitionistic fuzzy distance between two Atanassov intuitionistic fuzzy points.

Proposition 1: Let (U,d) be an ordinary metric space. If (x,λ,i) and (y,γ,j) are two Atanassov intuitionistic fuzzy points in $P_{IF}(U)$, then we define the distance between them as

$$d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) = (d(x,y), min\{\lambda,\gamma\}, max\{i,j\})$$

Here, $d(x,y)$ denotes the distance between x and y defined in (X,d). Therefore, $(P_{IF}(U), d_{IF})$ is an Atanassov intuitionistic fuzzy metric space if it satisfies the three necessary conditions stated in Definition 2.

Proof: Here we show that d_{IF} obeys the three conditions given in Definition 2:

(a) *Non-Negative*: Let (x,λ,i) and (y,γ,j) be two AIFPs in $P_{IF}(U)$. If $d(x,y)$ is the distance between x and y then we can say $d(x,y) \geq 0$. From Definition 1, it follows that $d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) = (d(x,y), min\{\lambda,\gamma\}, max\{i,j\})$ is a non-negative Atanassov intuitionistic fuzzy scalar and $d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) = 0$ iff $d(x,y) = 0, min\{\lambda,\gamma\} = 1, max\{i,j\} = 0$ i.e. $x = y, \lambda = \gamma = 1, i = j = 0$.

(b) *Symmetric*: Let (x,λ,i) and (y,γ,j) be two AIFPs in $P_{IF}(U)$, then we have

$$\begin{aligned} d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) &= (d(x,y), min\{\lambda,\gamma\}, max\{i,j\}) \\ &= (d(y,x), min\{\gamma,\lambda\}, max\{j,i\}) \\ &= d_{IF}\big((y,\gamma,j),(x,\lambda,i)\big). \end{aligned}$$

(c) *Triangle Inequality*: Let $(x,\lambda,i),(y,\gamma,j)\ and\ (z,\rho,l)$ be three AIFPs, then

$$\begin{aligned} d_{IF}\big((x,\lambda,i),(z,\rho,l)\big) &= (d(x,z), min\{\lambda,\rho\}, max\{i,l\}) \\ &\prec (d(x,y)+d(y,z), min\{\lambda,\rho,\gamma\}, max\{i,l,j\}) \\ &= (d(x,y), min\{\lambda,\gamma\}, max\{i,j\}) + \\ &\qquad (d(y,z), min\{\gamma,\rho\}, max\{j,l\}) \\ &= d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) + d_{IF}\big((y,\gamma,j),(z,\rho,l)\big) \end{aligned}$$

Proposition 2: Let R^n be the $n-$dimensional Euclidean space and T an Atanassov intuitionistic fuzzy linear space defined in R^n. Suppose (x,λ,i) and (y,γ,j) be two arbitrary Atanassov intuitionistic fuzzy points $belonging\ to\ T$, then the distance between them can be defined as:

$$d_{IFE}\big((x,\lambda,i),(y,\gamma,j)\big) = (d_E(x,y), min\{\lambda,\gamma\}, max\{i,j\})$$

Here, d_E denotes the Euclidean distance. So, (T, d_{IFE}) is also an Atanassov intuitionistic fuzzy metric space, where T can be

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2015.2452948, IEEE Transactions on Fuzzy Systems

considered as the set of Atanassov intuitionistic fuzzy points that $belong\ to$ the Atanassov intuitionistic fuzzy set T.

Proof: In the ordinary sense, R^n is a metric space and T can be viewed as a subset of $P_{IF}(R^n)$. Therefore, d_{IFE} is an Atanassov intuitionistic fuzzy metric. (from the proof of Proposition 1)

$S_{IF}^{+}(R)$ is not a complete ordered set. Therefore, in Definition 2 for triangle inequality $\leq$ is replaced by $\prec$. $\prec$ is much weaker than $\leq$, so the natural query that arises is as follows:

Is there some kind of Atanassov intuitionistic fuzzy metric space that can satisfy the triangle inequality with some partial order which is stronger than $\prec$ like $\preccurlyeq$.

The answer to the query is YES and such kind of metric spaces are called strong Atanassov intuitionistic fuzzy metric spaces.

Definition 3: Let U be a nonempty set and $d_{IF}: P_{IF}(U)\ X\ P_{IF}(U) \to S_{IF}^{+}(R)$ be a mapping. $(P_{IF}(U), d_{IF})$ can be called a strong Atanassov intuitionistic fuzzy metric space if it fulfils the first two conditions of Definition 2 and for any $(x,\lambda,i),(y,\gamma,j)\ and\ (z,\rho,l)$ in $P_{IF}(U)$ we have:

$(3')\ d_{IF}\big((x,\lambda,i),(z,\rho,l)\big) \preccurlyeq d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) + d_{IF}\big((y,\gamma,j),(z,\rho,l)\big)$

Proposition 3: Suppose T is an Atanassov intuitionistic fuzzy linear space defined in R^n, then the distance between any two Atanassov intuitionistic fuzzy points on T can be stated as:

$$d_{IFE}\big((x,\lambda,i),(y,\gamma,j)\big) = (d_E(x,y), min\{\lambda,\gamma\}, max\{i,j\}) \quad (1)$$

d_E denotes the Euclidean distance, (T, d_{IFE}) is a strong Atanassov intuitionistic fuzzy metric space and T is the set of Atanassov intuitionistic fuzzy points on the Atanassov intuitionistic fuzzy set T.

***Proof*:** Here we only consider the third property of triangle inequality as the first two properties can be proved in the similar way as shown in Proposition 1.

Suppose $(x,\lambda,i),(y,\gamma,j)\ and\ (z,\rho,l)$ are three arbitrary Atanassov intuitionistic fuzzy points on T. As stated earlier, (R^n, d_E) is a metric space, so

$$d_E(x,z) \leq d_E(y,z) + d_E(x,y) \quad (2)$$

If the above stated inequality (2) holds strictly, then condition $(3')$ is obviously satisfied from Definition 1(1). For the other case, where the " = " relation is considered, there must exist some $\lambda \in (0,1]$ such that $y = (1-\lambda)x + \lambda z$. Let $\alpha = min\{\lambda,\rho\}$ and $\beta = max\{i,l\}$. Then we can say that $\{x,z\} \subset T_{\alpha,\beta}$. As, T is an Atanassov intuitionistic fuzzy linear space, $T_{\alpha,\beta}$ is a linear subspace of R^n. It follows that $y \in T_{\alpha,\beta}$ i.e. $\gamma = \mu_T(y) \geq \alpha = min\{\lambda,\rho\}$. This implies that $min\{\lambda,\rho,\gamma\} = min\{\lambda,\rho\}$. Similarly, $j = \nu_T(y) \leq \beta = max\{i,l\}$. This implies that $max\{i,l,j\} = max\{i,l\}$.

Thus it can be stated that:

$$d_{IFE}\big((x,\lambda,i),(z,\rho,l)\big) = (d_E(x,z), min\{\lambda,\rho\}, max\{i,l\})$$
$$= (d_E(x,y) + d_E(y,z), min\{\lambda,\rho,\gamma\}, max\{i,l,j\})$$
$$= d_{IFE}\big((x,\lambda,i),(y,\gamma,j)\big) + d_{IFE}\big((y,\gamma,j),(z,\rho,l)\big)$$

III. Ranking of Trapezoidal Atanassov Intuitionistic Fuzzy Numbers

The centroid of a fuzzy number signifies its geometric centre and is denoted using the formula: $\int_{-\infty}^{\infty} xf(x)dx / \int_{-\infty}^{\infty} f(x)dx$. A trapezoid can be divided into three parts i.e. one rectangle and two triangles and centroid of each when joined forms a triangle. The centroid of this resultant triangle can be considered to be a better point of reference for the purpose of ranking. A trapezoidal Atanassov intuitionistic fuzzy number $\langle (a,b,c,d),(e,f,g,h)\rangle$ can be seen as two trapezoids, one representing the membership function and the other representing the non-membership function. Each of the trapezoids can be further divided into three parts. For the trapezoid denoting the membership function, g_1, g_3 and g_2 are the centroids of the two triangles and one rectangle respectively. Similarly, g_4, g_6 and g_5 signifies the centroids of the two triangles and one rectangle respectively of the trapezoid representing the non-membership function as shown in Fig. 1. The centroid values of these two resultant triangles can be computed using the following formulas:

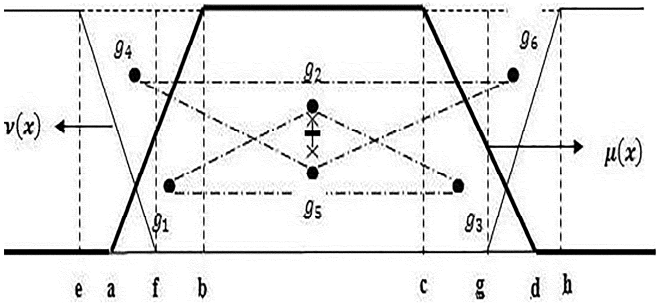


Fig. 1. The point of reference (CoC) used for ranking a trapezoidal Atanassov intuitionistic fuzzy number.

The centroid of the triangle $g_1g_2g_3 = C_1$

$$C_1 = (x_1, y_1) = \left[\frac{(2a+b+7c+2d)}{18}, \frac{7}{18}\right] \quad (3)$$

The centroid of the triangle $g_4g_5g_6 = C_2$

$$C_2 = (x_2, y_2) = \left[\frac{(2e+f+2h+7g)}{18}, \frac{11}{18}\right] \quad (4)$$

Then, the value that can be used for ranking can be evaluated using the following formula:

$$Centroid\ of\ centroids\ (CoC) = \sqrt{\left(\frac{x_1+x_2}{2}\right)^2 + \left(\frac{y_1+y_2}{2}\right)^2} \quad (5)$$

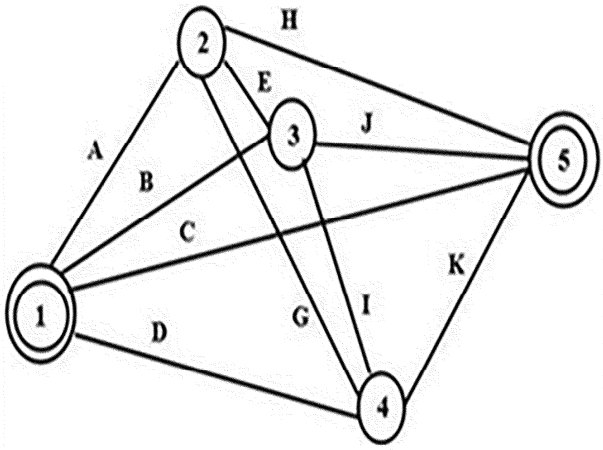
After the CoC values are computed for the total collected score of each feasible path, then $Rank\ (R)$ is assigned to each of the paths i.e. the path with higher CoC value gets a higher rank.

IV. Atanassov Intuitionistic Fuzzy Orienteering Problem (AIFOP)

A. Formulation of AIFOP

We apply the above derived formula of distance between two AIFPs (Proposition 1) on Orienteering Problem (OP) which can be represented as an undirected graph $G(V,E)$ where E signifies the set of edges and $V = \{v_1, \ldots\ldots, v_N\}$ is the set of vertices. Let the two functions i.e. distance function on edges be denoted as $d: E \to \Re^+$ and the function for score on vertices can be stated as $S: V \to \Re^+$. The goal here is to determine a Hamiltonian Path (P) that satisfies the distance bound (D_{max}), connects the source (v_1), target (v_N) and a subset of vertices (V') of V such that the total collected score

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2015.2452948, IEEE Transactions on Fuzzy Systems



Node	Label	Intuitionistic Fuzzy Point Values $((x,y),\lambda,i)$	Intuitionistic Fuzzy Score Values $\langle(\mu(x)),(\nu(x))\rangle$
v_1	1	$((10.5,14.4),0.6,0.3)$	$\langle(1,2,8,9),(0,2,8,10)\rangle$
v_2	2	$((11.2,14.1),0.8,0.2)$	$\langle(8,9,11,12),(6,8,12,14)\rangle$
v_3	3	$((18,15.9),0.5,0.5)$	$\langle(3,5,9,11),(1,4,10,13)\rangle$
v_4	4	$((18.3,13.3),0.7,0.1)$	$\langle(17,20,24,27),(14,18,26,30)\rangle$
v_5	5	$((16.5,9.3),0.9,0.1)$	$\langle(1,2,4,5),(0,1,5,6)\rangle$
$D_{max} = (20,0.8,0.2)$			

Fig. 2. Input graph G with number of nodes $(N) = 5$, source $(v_1) = 1$, target $(v_N) = 5$ and the co-ordinate values (Atanassov intuitionistic fuzzy points) and the score values (trapezoidal Atanassov intuitionistic fuzzy numbers) of each node.

is maximized [15]. OP finds several real life applications which include the fields of logistics, home delivery system, disaster management system, tourism industry etc. In each of the stated applications, the parameters involved i.e. score and distance cannot be determined precisely. Also, there exists some uncertainty in the position of the city or location. This leads to the need of using some technique to tackle this uncertainty. One such method is the use of Atanassov intuitionistic fuzzy points to model the uncertainty in the position and Atanassov intuitionistic fuzzy numbers to tackle the imprecise nature of the parameter score.

B. An Algorithm for solving AIFOP

Input: A given graph G with nodes represented as Atanassov intuitionistic fuzzy points and score of each node represented as a trapezoidal Atanassov intuitionistic fuzzy number.

Output: A path (P) that satisfies the distance bound D_{max} and maximizes the total collected score.

***Step 1*:** Calculate the Atanassov intuitionistic fuzzy distance values (i.e. the weight of each edge denoted as (d_{ij}) using the following formula:

$$d_{IF}\big((x,\lambda,i),(y,\gamma,j)\big) = (d(x,y), min\{\lambda,\gamma\}, max\{i,j\}) \quad (6)$$

Where, $d(x,y)$ denotes the Euclidean distance between two nodes.

***Step 2*:** Determine all possible paths connecting the source and target. For each path calculate:

(a) the total distance covered by the path using the following formula: (addition of two Atanassov intuitionistic fuzzy points)

$$(x,\lambda,i) + (y,\gamma,j) = \big((x+y), min\{\lambda,\gamma\}, max\{i,j\}\big) \quad (7)$$

(b) the total score collected on traversing a path using the following formula: (addition of two trapezoidal Atanassov intuitionistic fuzzy numbers).

$$\langle(a_1,b_1,c_1,d_1),(e_1,f_1,g_1,h_1)\rangle + \langle(a_2,b_2,c_2,d_2),(e_2,f_2,g_2,h_2)\rangle = \left\langle\begin{matrix}(a_1+a_2,b_1+b_2,c_1+c_2,d_1+d_2),\\(e_1+e_2,f_1+f_2,g_1+g_2,h_1+h_2)\end{matrix}\right\rangle \quad (8)$$

***Step 3*:** Discard those paths from the set of all possible paths that do not satisfy the D_{max} values (using Definition 1(1)).

***Step 4*:** Rank the scores of the remaining paths to determine the most desirable path (i.e. the path that satisfies the upper limit on the distance covered and has the best total collected score value) using (3)-(5).

So, the path with $R = 1$ is considered to be the most desirable path and returned as output. If two paths have the same score values then their rank values are computed on the basis of the total distance covered by those paths (i.e. the path that covers less distance, gets higher ranking).

V. ILLUSTRATIVE EXAMPLE

Consider the network in Fig. 2.

***Step 1*:** The d_{ij} value of each edge is computed using (6) and the values are presented in Table I.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2015.2452948, IEEE Transactions on Fuzzy Systems

TABLE I
THE d_{ij} VALUE OF EACH EDGE

Edge	Label	Distance Values of each Edge (d_{ij})
$1-2$	A	$(0.76, 0.6, 0.3)$
$1-3$	B	$(7.64, 0.5, 0.5)$
$1-4$	D	$(7.88, 0.6, 0.3)$
$1-5$	C	$(9.38, 0.6, 0.3)$
$2-3$	E	$(7.03, 0.5, 0.5)$
$2-4$	G	$(7.14, 0.7, 0.2)$
$2-5$	H	$(7.15, 0.8, 0.2)$
$3-4$	I	$(2.62, 0.5, 0.5)$
$3-5$	J	$(6.77, 0.5, 0.5)$
$4-5$	K	$(4.39, 0.7, 0.1)$

TABLE II
THE VALUE OF TOTAL DISTANCE COVERED AND TOTAL SCORE COLLECTED ON TRAVERSING EACH PATH

Path	Total Distance Covered by the Path	Total Score Collected by the Path
$P_1: 1-5$	(9.38, 0.6, 0.3)	⟨(1,2,8,9), (0,2,8,10)⟩
$P_2: 1-2-5$	(7.91, 0.6, 0.3)	⟨(9,11,19,21), (6,10,20,24)⟩
$P_3: 1-3-5$	(14.41,0.5,0.5)	⟨(4,7,17,20), (1,6,18,23)⟩
$P_4: 1-4-5$	(12.27,0.6,0.3)	⟨(18,22,32,36), (14,20,34,40)⟩
$P_5: 1-2-3-5$	(14.56,0.5,0.5)	⟨(12,16,28,32), (7,14,30,37)⟩
$P_6: 1-3-2-5$	(21.82,0.5,0.5)	⟨(12,16,28,32), (7,14,30,37)⟩
$P_7: 1-2-4-5$	(12.29,0.6,0.3)	⟨(26,31,43,48), (20,28,46,54)⟩
$P_8: 1-4-2-5$	(22.17,0.6,0.3)	⟨(26,31,43,48), (20,28,46,54)⟩
$P_9: 1-3-4-5$	(14.65,0.5,0.5)	⟨(21,27,41,47), (15,24,44,53)⟩
$P_{10}: 1-4-3-5$	(17.27,0.5,0.5)	⟨(21,27,41,47), (15,24,44,53)⟩
$P_{11}: 1-2-3-4-5$	(14.8,0.5,0.5)	⟨(29,36,52,59), (21,32,56,67)⟩
$P_{12}: 1-2-4-3-5$	(17.29,0.5,0.5)	⟨(29,36,52,59), (21,32,56,67)⟩
$P_{13}: 1-3-4-2-5$	(24.55,0.5,0.5)	⟨(29,36,52,59), (21,32,56,67)⟩
$P_{14}: 1-4-3-2-5$	(24.68,0.5,0.5)	⟨(29,36,52,59), (21,32,56,67)⟩
$P_{15}: 1-4-2-3-5$	(28.82,0.5,0.5)	⟨(29,36,52,59), (21,32,56,67)⟩
$P_{16}: 1-3-2-4-5$	(26.2,0.5,0.5)	⟨(29,36,52,59), (21,32,56,67)⟩

TABLE III
THE SOLUTION SET AFTER DISCARDING THOSE PATHS THAT DO NOT SATISFY THE DISTANCE BOUND (D_{max})

Path	Total Distance Covered by the Path	Total Score Collected by the Path
$P_1: 1-5$	(9.38, 0.6, 0.3)	⟨(1,2,8,9), (0,2,8,10)⟩
$P_2: 1-2-5$	(7.91, 0.6, 0.3)	⟨(9,11,19,21), (6,10,20,24)⟩
$P_3: 1-3-5$	(14.41,0.5,0.5)	⟨(4,7,17,20), (1,6,18,23)⟩
$P_4: 1-4-5$	(12.27,0.6,0.3)	⟨(18,22,32,36), (14,20,34,40)⟩
$P_5: 1-2-3-5$	(14.56,0.5,0.5)	⟨(12,16,28,32), (7,14,30,37)⟩
$P_7: 1-2-4-5$	(12.29,0.6,0.3)	⟨(26,31,43,48), (20,28,46,54)⟩
$P_9: 1-3-4-5$	(14.65,0.5,0.5)	⟨(21,27,41,47), (15,24,44,53)⟩
$P_{10}: 1-4-3-5$	(17.27,0.5,0.5)	⟨(21,27,41,47), (15,24,44,53)⟩
$P_{11}: 1-2-3-4-5$	(14.8,0.5,0.5)	⟨(29,36,52,59), (21,32,56,67)⟩
$P_{12}: 1-2-4-3-5$	(17.29,0.5,0.5)	⟨(29,36,52,59), (21,32,56,67)⟩

TABLE IV
RANKS ASSIGNED TO THE PATHS TO DETERMINE THE MOST DESIRABLE PATH

Path	Total Distance Covered by the Path	CoC Values for Score	Rank
$P_1: 1-5$	(9.38, 0.6, 0.3)	4.417	10
$P_2: 1-2-5$	(7.91, 0.6, 0.3)	11.51	8
$P_3: 1-3-5$	(14.41,0.5,0.5)	9.846	9
$P_4: 1-4-5$	(12.27,0.6,0.3)	20.006	6
$P_5: 1-2-3-5$	(14.56,0.5,0.5)	17.396	7
$P_7: 1-2-4-5$	(12.29,0.6,0.3)	27.171	3
$P_9: 1-3-4-5$	(14.65,0.5,0.5)	25.504	4
$P_{10}: 1-4-3-5$	(17.27,0.5,0.5)	25.504	5
$\boldsymbol{P_{11}: 1-2-3-4-5}$	(14.8,0.5,0.5)	32.670	**1**
$P_{12}: 1-2-4-3-5$	(17.29,0.5,0.5)	32.670	2

***Step 2*:** All the possible paths are determined and the following values are calculated and presented in Table II.

(a) Total distance covered by the path using (7).

(b) Total score collected on traversing a path using (8).

***Step 3*:** Those paths that do not satisfy the distance bound (D_{max}) are discarded using Definition 1(1) and the remaining paths that form the solution set are stated in Table III.

***Step 4*:** The paths obtained after Step 3 are ranked using (3)-(5) to determine the most desirable path and the values are stated in Table IV.

So, the most desirable path is P_{11} that collects the maximum possible score within the specified distance constraint.

VI. CONCLUSION

In this paper, we proposed a new definition for Atanassov intuitionistic fuzzy metric space using the idea of Atanassov intuitionistic fuzzy point and Atanassov intuitionistic fuzzy scalars. We use the concept of AIFP and apply the distance measure thus proposed for the first time on the Atanassov intuitionistic fuzzy orienteering problem to model the uncertainty present in the position or location and the two parameters involved i.e. distance and score. Also, with the help of an illustrative example, we suggested a method to solve AIFOP. In future, we plan to use some heuristic or meta-heuristic like the ant colony optimization algorithm, firefly algorithm, flower pollination algorithm etc. to replace the exhaustive search method and apply the proposed technique for larger graphs.

ACKNOWLEDGMENT

The authors gratefully acknowledge Prof. Rekha Srivastava, Department of Mathematical Sciences, IIT (BHU) for fruitful discussions. The valuable suggestions by anonymous referees have resulted in improving the quality of this paper. We gratefully acknowledge the same.

REFERENCES

[1] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353, 1965.

[2] Z. K. Deng, "Fuzzy pseudo-metric spaces," *Journal of Mathematical Analysis and Applications*, vol. 86, pp. 74-95, 1982.

[3] O. Kaleva and S. Seikkala, "On fuzzy metric spaces," *Fuzzy Sets and Systems*, vol. 12, pp. 225-229, 1984.

[4] K. Menger, "Statistical metrics," *Proc. Nat. Acad. Sci.*, vol. 28, pp. 535-537, 1942.

[5] M. Grabiec, "Fixed points in fuzzy metric spaces," *Fuzzy Sets and Systems*, vol. 27, pp. 385-389, 1988.

[6] A. George and P. Veeramani, "On some results in fuzzy metric spaces," Fuzzy *Sets and Systems*, vol. 64, pp. 395-399, 1994.

[7] A. George and P. Veeramani, "On some results of analysis for fuzzy metric spaces," *Fuzzy Sets and Systems*, vol. 90, pp. 365-368, 1997.

[8] A. K. Katsaras, "Fuzzy topological vector spaces. II," *Fuzzy Sets and Systems*, vol. 12, no. 2, pp. 143-154, 1984.

[9] C. P. Santhosh and T. V. Ramakrishnan, "Norm and Inner Product on Fuzzy Linear Spaces over Fuzzy Fields," *Iranian Journal of Fuzzy Systems*, vol. 8, no. 1, pp. 135-144, 2011.

[10] K. Atanassov, "Intuitionistic fuzzy sets," VII ITKR's Scientific Session, Sofia, June 1983. Deposed in Central Sci. - Techn. Library of Bulg. Acad. of Sci., 1697/84 (in Bulg.), 1983.
[11] K. Atanassov, "Intuitionistic fuzzy sets: past, present and future," Presented at 3rd Conference of the European Society for Fuzzy Logic and Technology, 2003, Sept., Zittau, Germany.
[12] J. H. Park, "Intuitionistic fuzzy metric spaces," *Chaos, Solitons and Fractals*, vol. 22, pp. 1039–1046, 2004.
[13] Z. Q. Xia and F. F. Guo, "Fuzzy Metric Spaces," *J. Appl. Math. and Computing*, vol. 16, no. 1 – 2, pp. 371 – 38, 2004.
[14] D. Coker and M. Demirci, "On intuitionistic fuzzy points," *Notes on Intuitionistic Fuzzy Sets*, vol. 1, no. 2, pp. 79–84, 1995.
[15] P. Vansteenwegen, W. Souffriau and D. V. Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, pp. 1-10, 2011.
[16] A. Shannon and K. Atanassov, "A first step to a theory of the intuitionistic fuzzy graphs," in *Proc. of the First Workshop on Fuzzy Based Expert Systems*, Sofia, 1994, Sept., pp. 59–61.
[17] K. Atanassov and A. Shannon, "Intuitionstic fuzzy graphs From $\alpha-$, $\beta-$ and (α,β) −levels," *Notes on Intuitionistic Fuzzy Sets*, vol. 1, no. 1, pp. 32–35, 1995.
[18] R. Parvathi, S. Thilagavathi, G. Thamizhendhi and M. G. Karunambigai, "Index matrix representation of intuitionistic fuzzy graphs," *Notes on Intuitionistic Fuzzy Sets*, vol. 20, no. 2, pp. 100–108, 2014.
[19] S. Chauhan, W. Shatanawi, S. Kumar and S. Radenovic, "Existence and uniqueness of fixed points in modified intuitionistic fuzzy metric spaces," *Journal of Nonlinear Science and Applications*, vol. 7, pp. 28-41, 2014.
[20] M. Akram, "Characterizing $\Gamma-$ Semigroups by Intuitionistic Fuzzy Points," *ARPN Journal of Systems and Software*, vol. 2, no. 12, pp. 359-365, 2012.
[21] S. K. Sardar, M. Mandal and S. K. Majumder, "Intuitionistic Fuzzy Points in Semigroups," *International Scholarly and Scientific Research & Innovation*, vol. 5, no. 3, pp. 1343-1348, 2011.
[22] H. Bustince, E. Barrenechea and M. Pagola, "Restricted equivalence functions," *Fuzzy Sets and Systems*, vol. 157, pp. 2333 – 2346, 2006.
[23] H. Bustince, E. Barrenechea and M. Pagola, "Relationship between restricted dissimilarity functions, restricted equivalence functions and normal E_N-functions:Image thresholding invariant," *Pattern Recognition Letters*, vol. 29, pp. 525–536, 2008.

A Greedy Algorithm for Fuzzy Shortest Path Problem using Quasi-Gaussian Fuzzy Weights

Madhushi Verma, Department of Computer Engineering, IIT(BHU), Varanasi, India

K. K. Shukla, Department of Computer Engineering, IIT(BHU), Varanasi, India

ABSTRACT

Several algorithms exist to determine the shortest path in a network for the crisp case where the weights are real numbers. In the real world, these weights represent parameters like cost, packet arrival time, link capacity etc which are not naturally precise. To model the uncertainty involved, for the first time we use the Gaussian fuzzy numbers as weights and a method has been presented in this paper to determine the fuzzy shortest path. Gaussian membership functions are preferred over other membership functions (triangular and trapezoidal) that are easy to analyze because it is continuous and differentiable enabling efficient gradient based optimization and it is simpler to represent because it requires fewer parameters. The issue of performing fuzzy arithmetic operations to calculate the fuzzy shortest path length and the corresponding fuzzy shortest path in the network has been addressed and to tackle it the concept of decomposed fuzzy numbers has been used. Also, a greedy algorithm which is an extension of Dijkstra's algorithm for fuzzy shortest path problem has been proposed.

Keywords: *Decomposed Fuzzy Numbers, Fuzzy Shortest Path, Fuzzy Shortest Path Length, Link Preference Index, Quasi-Gaussian Fuzzy Numbers, Ranking*

INTRODUCTION

In the field of mathematics and computer science, graphs are structures used to depict the pair wise relations between the objects from a collection. A graph can be represented as $G = \left(V, E\right)$ where V is the set of vertices and E is the set of edges connecting pair of vertices. In a network, a path is an alternating sequence of vertices and edges connecting a source node and a destination node. In general, there can be more than one path connecting the source node and the destination node, the shortest path is one where the sum of the weights assigned to the edges is minimum. The problem of finding the shortest path is of great importance in graph theory as it finds various real life applications like communication, computer networks, transport, routing, supply chain management etc. (Elizabeth & Sujatha, 2011; Hernandes et al., 2007; Ravishankar et al., 2012). An example of finding the shortest path could be to determine the route between city A and city B on a road map with minimum travel time. Here the various connecting cities between city A

DOI: 10.4018/ijfsa.2013040104

and city B can be represented as vertices and their connecting road segments can be represented as edges connecting the vertices. In the real world the weights assigned to these edges correspond to cost, time, capacities or demand which are represented as real numbers. However these parameters are not naturally precise and the uncertainty involved cannot be modelled using real numbers. This gives rise to the fuzzy shortest path problem (FSPP) where the weights are taken as fuzzy numbers in order to deal with the uncertainty. In a graph, various types of fuzziness are possible which are given by Blue et al (2002) as:

- **Type I:** Fuzzy set of crisp graphs;
- **Type II:** Crisp vertex set and fuzzy edge set;
- **Type III:** Crisp vertices and edges with fuzzy connectivity;
- **Type IV:** Fuzzy vertex set and crisp edge set;
- **Type V:** Crisp graph with fuzzy weights.

To tackle the non-statistical uncertainty in problems, fuzzy set theory was proposed by Zadeh (1978) and using this theory a lot of work has been done in the area of shortest path problem. Dubois & Prade (1980) first introduced the "Fuzzy Shortest Path Problem" and found the fuzzy shortest path length (FSPL) in a network using the Floyd's algorithm and the Ford's algorithm. However, the corresponding shortest path may not be present in the network. A dynamical programming recursion-based fuzzy algorithm was proposed by Klein (1991) and in 2006 another dynamic programming approach was used to solve FSPP using triangular fuzzy numbers (Kung et al., 2006). Later, an attempt was made to determine the fuzzy shortest path present in the network corresponding to the calculated FSPL using different fuzzy numbers (triangular, trapezoidal and discrete fuzzy numbers) as arc lengths and was successfully accomplished by using the concept of "Similarity Measure" where a comparison was made between each path length and the shortest path length, the one with the highest similarity degree was concluded as the fuzzy shortest path (FSP) (Chuang & Kung, 2005; Chuang & Kung, 2006; Sujatha & Elizabeth, 2011). Other than similarity measure, "ranking index" has also been used for determining the FSP in the network by comparing individual path lengths with the shortest path length and assigning ranks in order to determine the FSP (Elizabeth & Sujatha, 2011).

All of the reported research work is based on piece-wise linear membership functions like trapezoidal or triangular (Ebrahimnejad, 2012; Elizabeth & Sujatha, 2011). Although these membership functions are easy to analyze, they are unsuitable for applying gradient based parameter optimization methods (Dongrui, 2012). Gaussian membership function is simpler to represent since it requires fewer parameters. It is continuous and differentiable enabling efficient gradient based optimization in cases where actual measurement data is used to identify the parameters of membership functions while using parameter estimation and system identification techniques, continuity and differentiability and fewer parameters are highly desirable (Grauel & Ludwig, 1999). Kashtiban et al. (2008) have used a recursive extended Kalman filter to obtain very good estimates of membership function from observed data, their method also requires the membership function to be continuous and differentiable leading to a choice of Gaussian functions.

The proposed method is applicable to all problems related with computer networks, vehicle routing, job scheduling etc. In all these applications, the parameters are not exactly known but estimated from samples of past observations. For example, in computer networks, specially wireless networks, the signal to noise ratio (SNR) is a crucial parameter. Networks suffer from several noise types- thermal, in-

duced, crosstalk, impulse or atmospheric noise. In such a case, the Shannon capacity of a link is given by:

$$capacity = bandwidth \times \log_2\left(1 + SNR\right)$$

Since in practical situations noise is a net effect of several independent phenomena, it tends to follow Gaussian distribution (the central limit theorem). By limiting the Gaussian function to 3σ on either side of the mean gives a 6σ design where 99.99966% values are taken care of by the model. Similarly, in vehicle routing applications, traffic conditions and in job scheduling processing times are naturally modelled by the proposed Quasi-Gaussian fuzzy numbers. Thus, we find that Quasi-Gaussian fuzzy numbers represent a realistic model for many real applications.

In this paper we first give a description of the problem and justify the techniques used. Then we provide the definition and explanation of the terms and techniques used in this paper. Afterwards an explanation of the steps and process of solving the FSPP using an illustrative example is presented. Next we present an extension of the classical greedy shortest path algorithm for the case of fuzzy numbers. Finally, the paper is concluded in the last section.

PROBLEM FORMULATION

In this paper, type V graph is considered where the weights assigned to the edges are fuzzy numbers. There are several types of fuzzy numbers (triangular, trapezoidal, and Gaussian etc.) and the one used here is a variation of Gaussian fuzzy numbers called Quasi-Gaussian fuzzy numbers which are taken as weights allotted to the edges of the graph.

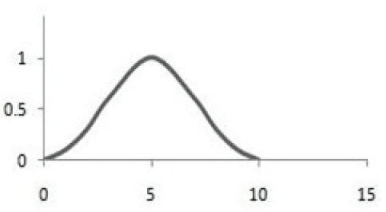
It is therefore highly desirable to have a shortest path algorithm that models uncertainty in link weights with Gaussian function expressed as:

$$\mu\left(x;\overline{x},\sigma,m\right) = exp\left[-\frac{1}{2}\left|\frac{x-\overline{x}}{\sigma}\right|^m\right]$$

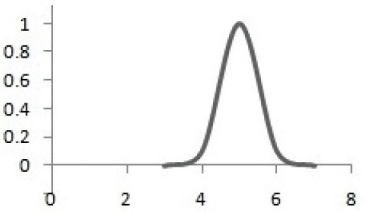
where, $\overline{x}$ is the centre, σ is the width and m is the fuzzification factor. Various shapes come into existence by varying the values of these parameters showing different degree of fuzziness which are shown in Figure 1:

To model the FSPP, here we use a variation of Gaussian membership function called Quasi-Gaussian membership function, the definition and explanation of which has been presented in the next section of preliminaries. The concept of decomposed fuzzy numbers is used to execute the fuzzy arithmetic operations on these Quasi-Gaussian fuzzy numbers and the process of conversion and execution of arithmetic operations has been explained in a later section of this paper. The technique used for determining the minimum of two decomposed fuzzy numbers using the concept of lattice of fuzzy numbers and α -cuts is preferred over other ranking methods like those discussed in the paper by Ramli and Mohamad (2009) using the centroid index that falls under the category of fuzzy scoring techniques. These centroid indexes can be used for ranking of triangular and trapezoidal fuzzy numbers as most of it involves calculating the area under the curve which requires less number of calculations in case of triangular and trapezoidal fuzzy numbers and more calculations when Gaussian fuzzy numbers are involved, thereby increasing the complexity of the operations. Another reason for using this method of determining the minimum of two decomposed fuzzy numbers is that the centroid methods fail to rank fuzzy numbers with the same centre values $\left(\overline{x}\right)$ but different spread or width values $\left(\sigma\right)$ but in our case this situation can be easily handled and the minimum of two fuzzy numbers can be generated as the concept of α -cuts is used here.

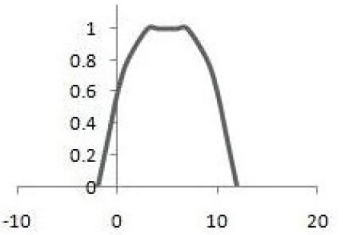
Figure 1. Different shapes obtained by varying the value of fuzzification factor m



1(a): $\bar{x} = 5, \sigma = 2, m = 2$



1(b): $\bar{x} = 5, \sigma = 0.5, m = 2$



1(c): $\bar{x} = 5, \sigma = 5, m = 5$

PRELIMINARIES

In this section some definitions are stated and some fuzzy arithmetic operations and ranking indexes are reviewed which are the requisites for solving the FSPP.

Definition 1: Quasi Gaussian Fuzzy Number

Fuzzy numbers find a lot of practical applications and Gaussian fuzzy numbers can be applied in various fields so it is beneficial to limit the Gaussian fuzzy numbers and derive a new category called "Quasi-Gaussian fuzzy number" (QGFN). QGFN is a Gaussian fuzzy number with finite support i.e. the value of x beyond $\bar{x} - 3\sigma_l$ and $\bar{x} + 3\sigma_r$ is zero where $\bar{x}$ is the modal value, σ_l and σ_r denote the left and right spreads respectively corresponding to the Gaussian distribution's standard deviation. We use Hanss (2005) notation as follows:

$$p = gfn^{*}\left(\bar{x}, \sigma_l, \sigma_r\right) \quad (1)$$

Here p is a QGFN and the membership function $\mu_p\left(x\right)$ is defined as the equation shown in Box 1.

Definition 2: Decomposed Fuzzy Numbers (DFN)

Fuzzy arithmetical operations like addition, subtraction etc. can be performed in several ways using Zadeh's extension principle, LR fuzzy numbers (Dubois & Prade, 1978; Dubois & Prade, 1979), discretized fuzzy numbers (Hanss, 1999; Hanss & Willner, 2000) and decomposed fuzzy numbers (Moore, 1966). Here we prefer decomposed fuzzy numbers which implements interval arithmetic because in case of Zadeh's extension principle, the same output is generated by an infinite number of combinations of the input values and thus it is

Box 1.

$$\mu_p(x) = \begin{cases} 0 & for\ x \le \overline{x} - 3\sigma_l \\ \exp\left[-\dfrac{(x-\overline{x})^2}{2\sigma_l^{\,2}}\right] & for\ \overline{x} - 3\sigma_l < x < \overline{x} \\ & \forall x \in R \\ \exp\left[-\dfrac{(x-\overline{x})^2}{2\sigma_r^{\,2}}\right] & for\ \overline{x} \le x < \overline{x} + 3\sigma_r \\ 0 & for\ x > \overline{x} + 3\sigma_r \end{cases} \quad (2)$$

not useful for practical implementations. When L-R fuzzy numbers are used in fuzzy arithmetic for performing the elementary operations, it is necessary to introduce certain approximation procedures due to which significant amount of information is lost and therefore it is not suitable for practical implementation. To overcome the drawbacks of the above stated two methods, the concept of discretized fuzzy numbers was introduced where the continuous membership functions of the fuzzy numbers were discretized and the μ-axis was divided into intervals. But it was observed that after performing the elementary operations when the continuous counterpart of the result was reconstructed by joining the lines between the elements of the discretized fuzzy number (output), the resultant fuzzy number did not satisfy the convexity property of fuzzy numbers so some more operations were performed to eliminate the invalid elements from the final discretized fuzzy number generated as the output of the elementary operations. To avoid the extra calculation involved in case of discretized fuzzy numbers, the idea of decomposed fuzzy numbers came into existence that implements the concept of interval arithmetic and performs the elementary operations using the α-cut values and is well suited for practical applications (Hanss, 2005). This is the approach taken in the present paper as well.

A fuzzy set A can be represented as sequence of α-cuts. An α-cut $\left(A_\alpha\right)$ of A is defined as (Hanss, 2005):

$$A_\alpha = \alpha\text{-cut (A)} = \left\{x | \mu_A(x) \ge \alpha\right\}$$

where $\alpha \in [0,1]$.

To reduce the infinite number of α-cuts and make the decomposed fuzzy numbers usable for practical applications, the infinite set is reduced to a finite one by selecting some discrete values $\alpha_j = \mu_j$ for α [18].

The finite set is created by dividing the interval [0, 1] into k sub-intervals by applying the following formula (Hanss, 2005):

$$\Delta\mu = \frac{1}{k}$$

Now the discrete values are:

$$\mu_j = \frac{j}{k},\ j = 0,1,\ldots\ldots\ldots k$$

with the properties:

$$\mu_0 = 0, \mu_1 = 1$$

$$\mu_{j+1} = \mu_j + \Delta\mu, \quad j = 0,1,\ldots\ldots\ldots\ldots k-1$$

The parameter k which controls the degree of refinement is called the *decomposition number*. The decomposed form of the fuzzy number p_i that corresponds to the finite number of α -cuts is represented by the set:

$$P_i = \left\{X_i^{(0)}, X_i^{(1)}, \ldots\ldots\ldots\ldots, X_i^{(k)}\right\} \text{ of } \left(k+1\right) \text{ intervals} \quad (3)$$

where:

$$X_i^{(j)} = \left[a_i^{(j)}, b_i^{(j)}\right] = cut_{\mu_j}\left(p_i\right)$$

$$a_i^{(j)} \leq b_i^{(j)}, \; j = 1,2,\ldots\ldots\ldots\ldots\ldots k$$

These $\left(k+1\right)$ intervals are called *intervals of confidence.*

Definition 3: Addition Operation Using DFN

If two decomposed fuzzy numbers $X_1^{(j)}$ and $X_2^{(j)}$ represented by the interval $\left[a_1^{(j)}, b_1^{(j)}\right]$ and $\left[a_2^{(j)}, b_2^{(j)}\right]$ respectively, are to be added to form the result $Z^{(j)}$ shown by the interval $\left[a^{(j)}, b^{(j)}\right]$ then:

$$\left[a_1^{(j)}, b_1^{(j)}\right] + \left[a_2^{(j)}, b_2^{(j)}\right] = \left[\underset{a^{(j)}}{a_1^{(j)} + a_2^{(j)}}, \underset{b^{(j)}}{b_1^{(j)} + b_2^{(j)}}\right] \quad (4)$$

The concept of interval arithmetic induced by the decomposed fuzzy numbers is preferred over other methods of performing fuzzy arithmetic operations due to its less complex implementation (Hanss, 2005).

Definition 4: Minimum Operation Using DFN

As stated by Klir and Yuan (1997), the set of real numbers R is linearly ordered and a pair of value p and q can be stated as either $p \leq q$ or $q \leq p$. Here, the lattice $\left(R, \leq\right)$ can be presented by an operation:

$$min\left(p,q\right) = \begin{cases} p & if\ p \leq q \\ q & if\ q \leq p \end{cases}$$

for every $p, q \in R$

However, this kind of linear ordering cannot be applied on fuzzy numbers and the minimum operation on two fuzzy numbers A, B is expressed as $MIN\left(A, B\right)$.

Let S denote the set of all fuzzy numbers then the triple $< S, MIN, MAX >$ is a distributive lattice in which MIN corresponds to *meet* and MAX corresponds to *join* operation (Klir & Yuan, 1997).

It is possible to define partial ordering on the lattice $< S, MIN, MAX >$.

The decomposed fuzzy numbers are represented in terms of α -cuts and these α -cuts can also be utilized in finding the partial ordering of the fuzzy numbers.

For any $A, B \in S$ and $\alpha \in \left[0,1\right]$ if the α -cut of A is expressed as A_α and that of B as B_α where $A_\alpha = \left[a_1, a_2\right]$ and $B_\alpha = \left[b_1, b_2\right]$. Then:

$$MIN\left(A_\alpha, B_\alpha\right) = \left[min\left(a_1, b_1\right), min\left(a_2, b_2\right)\right]$$

Similarly, in case of decomposed fuzzy numbers $X_1^{(j)} and X_2^{(j)}$ represented by the interval $\left[a_1^{(j)}, b_1^{(j)}\right]$ and $\left[a_2^{(j)}, b_2^{(j)}\right]$ respectively, the minimum of the two fuzzy numbers can be represented as $Z_{min}^{(j)}$ stated as interval $\left[a^{(j)}, b^{(j)}\right]$ then:

$$Z_{min}^{(j)} = \left[a^{(j)}, b^{(j)}\right] = \left[min\left(a_1^{(j)}, a_2^{(j)}\right), min\left(b_1^{(j)}, b_2^{(j)}\right)\right] \quad (5)$$

Definition 5: Link Preference Index (LPI)

The LPI defines the ranking order where each path length $\left(Z_I\right)$ is compared with the FSPL $\left(Z_{min}\right)$ which is shown in Figure 2. The path $\left(Z_I\right)$ with the highest LPI is preferred to all other paths and is the fuzzy shortest path i.e.:

$$Z_1 < Z_2 \ \ iff \ LPI\left(Z_{min} < Z_1\right) > LPI\left(Z_{min} < Z_2\right)$$

The formula for LPI using QGFN is derived as follows:

$$Z_{min} = gfn^*\left(\overline{x}, \sigma_l, \sigma_r\right)$$

$$Z_I = gfn^*\left(\overline{x_I}, \sigma_{I_l}, \sigma_{I_r}\right)$$

Let μ_d be the membership function.

If $\overline{x} \le x < \overline{x} + 3\sigma_r$:

$$\mu_d = exp\left[-\left(x-\overline{x}\right)^2 / 2{\sigma_r}^2\right]$$

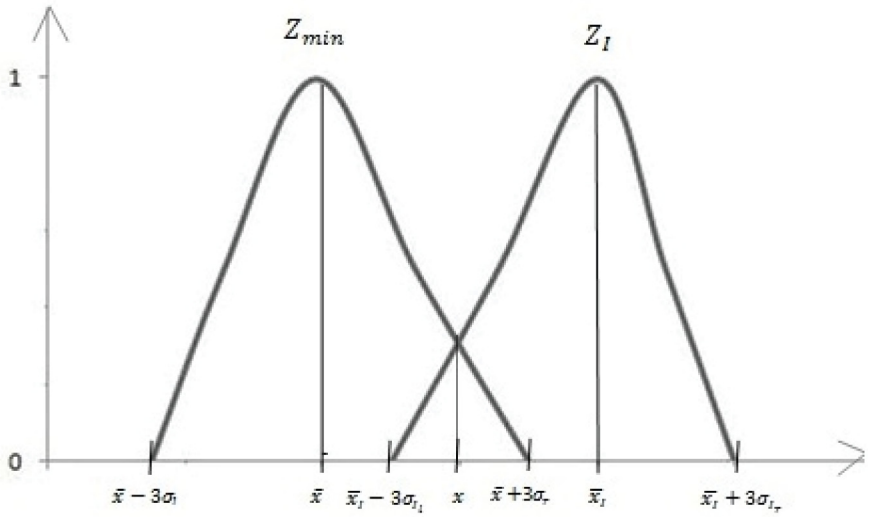
$$=> \log\frac{1}{\mu_d} = \frac{\left(x-\overline{x}\right)^2}{2{\sigma_r}^2}$$

$$=> 2{\sigma_r}^2\left(\log\frac{1}{\mu_d}\right) = \left(x-\overline{x}\right)^2$$

$$=> \sigma_r\sqrt{2\left(\log\frac{1}{\mu_d}\right)} = x-\overline{x}$$

$$=> x = \left[\sigma_r\sqrt{2\left(\log\frac{1}{\mu_d}\right)} + \overline{x}\right] \quad (6)$$

Figure 2. Link preference index diagram



If $\bar{x} - 3\sigma_{I_l} < x < \overline{x_I}$:

$$\mu_d = exp\left[-\left(x - \overline{x_I}\right)^2 / 2{\sigma_{I_l}}^2\right]$$

$$=> x = \left[\sigma_{I_l}\sqrt{2\left(\log\frac{1}{\mu_d}\right)} + \overline{x_I}\right] \quad (7)$$

Equating (6) and (7):

$$=> \sigma_r\sqrt{2\left(\log\frac{1}{\mu_d}\right)}$$

$$+\bar{x} = \sigma_{I_l}\sqrt{2\left(\log\frac{1}{\mu_d}\right)} + \overline{x_I}$$

$$=> \bar{x} - \overline{x_I} = \sigma_{I_l}\sqrt{2\left(\log\frac{1}{\mu_d}\right)}$$

$$-\sigma_r\sqrt{2\left(\log\frac{1}{\mu_d}\right)}$$

$$=> \bar{x} - \overline{x_I} = \sqrt{2\log\frac{1}{\mu_d}}\left(\sigma_{I_l} - \sigma_r\right)$$

$$=> \sqrt{2\left(\log\frac{1}{\mu_d}\right)} = \frac{\left(\bar{x} - \overline{x_I}\right)}{\left(\sigma_{I_l} - \sigma_r\right)}$$

Squaring on both sides:

$$=> 2\left(\log\frac{1}{\mu_d}\right) = \left[\left(\bar{x} - \overline{x_I}\right) / \left(\sigma_{I_l} - \sigma_r\right)\right]^2$$

$$=> \log\frac{1}{\mu_d} = \frac{1}{2}\left[\left(\bar{x} - \overline{x_I}\right) / \left(\sigma_{I_l} - \sigma_r\right)\right]^2$$

$$\mu_d = exp\left[-\left(\bar{x} - \overline{x_I}\right)^2 / 2\left(\sigma_{I_l} - \sigma_r\right)^2\right], \quad (8)$$

$where\ \sigma_{I_l} \neq \sigma_r$

We take:

$$LPI = exp\left[-\left(\bar{x} - \overline{x_I}\right)^2 / 2\left(\sigma_{I_l} - \sigma_r\right)^2\right] \quad (9)$$

$if\ \sigma_{I_l} \neq \sigma_r$

The above stated formula can accurately rank the Quasi-Gaussian fuzzy numbers but cannot tackle situations where the left spread $\left(\sigma_{I_l}\right)$ of Z_I becomes equal to the right spread $\left(\sigma_r\right)$ of Z_{min}. To deal with such a situation, we consider the following three cases:

Case I:

$$if\ \ \overline{x_I} - 3\sigma_{I_l} < \bar{x} + 3\sigma_r$$

$$=> \overline{x_I} - \bar{x} < 3\left(\sigma_{I_l} + \sigma_r\right)$$

$$=> \overline{x_I} - \bar{x} = 3\varepsilon\left(\sigma_{I_l} + \sigma_r\right) where\, 0 < \varepsilon < 1$$

$$=> \sigma_{I_l} + \sigma_r = \left[\overline{x_I} - \bar{x} / 3\varepsilon\right]$$

$$Let\left(\overline{x_I} - \bar{x}\right) = \Delta\bar{x}$$

$$=> \sigma_{I_l} + \sigma_r = \left[\Delta\bar{x} / 3\varepsilon\right] \quad (10)$$

Squaring on both sides:

$$=> {\sigma_{I_l}}^2 + {\sigma_r}^2 + 2\sigma_{I_l}\sigma_r = \left[\left(\Delta\bar{x}\right)^2 / 9\varepsilon^2\right]$$

$$=> {\sigma_{I_l}}^2 + {\sigma_r}^2 = \left[\left(\Delta\bar{x}\right)^2 / 9\varepsilon^2\right] - 2\sigma_{I_l}\sigma_r \quad (11)$$

Substituting $\left(11\right)$ in $\left(9\right)$, we get:

$$LPI = exp\left[-\left(-\Delta\bar{x}\right)^2 / 2\left(\sigma_{I_l} - \sigma_r\right)^2\right]$$

$$=> LPI$$

$$= exp\left[-\left(\Delta\overline{x}\right)^2 / 2\left(\sigma_{I_l}{}^2 + \sigma_r{}^2 - 2\sigma_{I_l}\sigma_r\right)\right]$$

$$=> LPI = exp\left[\frac{-\left(\Delta\overline{x}\right)^2}{2\left(\frac{\left(\Delta\overline{x}\right)^2}{9\varepsilon^2} - 2\sigma_{I_l}\sigma_r - 2\sigma_{I_l}\sigma_r\right)}\right]$$

$$=> LPI = exp\left[\frac{-\left(\Delta\overline{x}\right)^2}{2\left(\frac{\left(\Delta\overline{x}\right)^2}{9\varepsilon^2} - 4\sigma_{I_l}\sigma_r\right)}\right] \quad (12)$$

Similarly we get the following equations for **Case II** and **Case III**:

Case II:

$$if\ \overline{x_I} - 3\sigma_{I_l} = \overline{x} + 3\sigma_r$$

$$LPI = exp\left[\frac{-\left(\Delta\overline{x}\right)^2}{2\left(\frac{\left(\Delta\overline{x}\right)^2}{9\varepsilon^2} - 4\sigma_{I_l}\sigma_r\right)}\right] \quad (13)$$

$$where\ \varepsilon = 1$$

Case III:

$$if\ \overline{x_I} - 3\sigma_{I_l} > \overline{x} + 3\sigma_r$$

$$LPI = exp\left[\frac{-\left(\Delta\overline{x}\right)^2}{2\left(\frac{\left(\Delta\overline{x}\right)^2}{9\varepsilon^2} - 4\sigma_{I_l}\sigma_r\right)}\right] \quad (14)$$

$$where\ \varepsilon > 1$$

Box 2.

$$LPI = \begin{cases} & if\ \ \overline{x_I} - 3\sigma_{I_l} < \overline{x} + 3\sigma_r\ \ then\, choose\ \ 0 < \varepsilon < 1 \\ exp\left[\frac{-\left(\Delta\overline{x}\right)^2}{2\left(\frac{\left(\Delta\overline{x}\right)^2}{9\varepsilon^2} - 4\sigma_{I_l}\sigma_r\right)}\right] & if\ \ \overline{x_I} - 3\sigma_{I_l} = \overline{x} + 3\sigma_r\ \ then\, choose\ \ \varepsilon = 1 \\ & if\ \ \overline{x_I} - 3\sigma_{I_l} > \overline{x} + 3\sigma_r\ then\, choose\ \ \varepsilon > 1 \end{cases} \quad (15)$$

ILLUSTRATIVE EXAMPLE

The fuzzy shortest path for the following network shown in Figure 3 is determined using link preference index.

The following steps are applied on the network shown in Figure 3:

Step 1: Construct a network $G = (V, E)$.

Figure 3 shows a network G with $|V| = 6 and |E| = 8$ where $V and E$ are the set of vertices and set of edges respectively. The edge weights in the form of QGFN are as follows:

$$S = gfn^{*}[9,3,2]$$

$$Q = gfn^{*}[20,6,4]$$

$$R = gfn^{*}[10,2,2]$$

$$U = gfn^{*}[15,5,5]$$

$$V = gfn^{*}[14,4,5]$$

$$W = gfn^{*}[8,1,2]$$

$$Y = gfn^{*}[16,3,4]$$

$$G = gfn^{*}[20,6,4]$$

Step 2: Find all possible paths (P_I), from source vertex to the destination vertex.

In the considered network G four paths are possible from the source vertex 1 to the destination vertex 6 which are as follows:

$$P_1 = S + U + W + G$$

$$P_2 = S + U + Y$$

$$P_3 = S + R + V + G$$

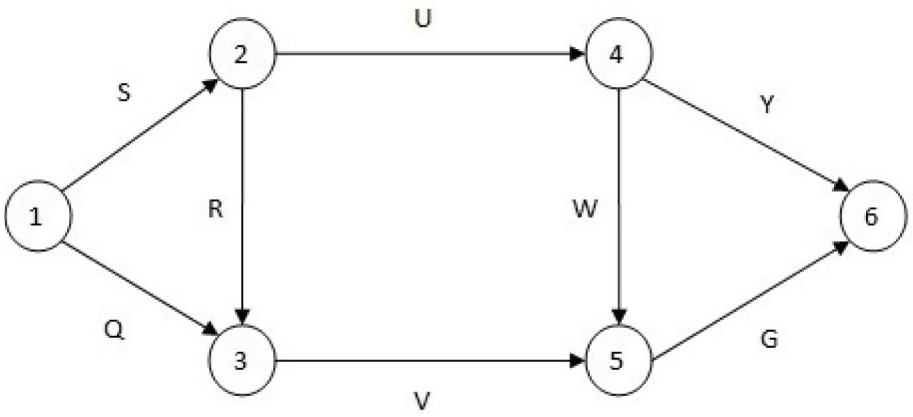
$$P_4 = Q + V + G$$

Step 3: Convert the weights (QGFN) of the edges present in the path (P_I) to DFN, using Definition 2.

The above stated edge weights are now converted into decomposed fuzzy numbers which are shown below.

Let the value of $K = 2$, and then the following intervals are generated:

Figure 3. Network



$$S = \left\{ [0,15]^0, [5.5,11.5]^{0.5}, [9,9]^1 \right\}$$
$$Q = \left\{ [2,32]^0, [13,25]^{0.5}, [20,20]^1 \right\}$$
$$R = \left\{ [4,16]^0, [7.5,12.5]^{0.5}, [10,10]^1 \right\}$$
$$U = \left\{ [0,30]^0, [9,21]^{0.5}, [15,15]^1 \right\}$$
$$V = \left\{ [2,29]^0, [9,20.5]^{0.5}, [14,14]^1 \right\}$$
$$W = \left\{ [4,14]^0, [6.5,10.5]^{0.5}, [8,8]^1 \right\}$$
$$Y = \left\{ [7,28]^0, [12.5,21]^{0.5}, [16,16]^1 \right\}$$
$$G = \left\{ [1,22]^0, [4.5,13]^{0.5}, [7,7]^1 \right\}$$

Step 4: Compute the path length (Z_I) for each of the paths (P_I) found using Definition 3 $(I = 1,2\ldots\ldots\ldots\ldots,n)$.

Let the path lengths be denoted by Z_1, Z_2, Z_3, Z_4 of paths P_1, P_2, P_3, P_4 respectively. As per the Definition 3, the following path lengths are generated:

$$Z_1 = \left\{ [5,81]^0, [25.5,56]^{0.5}, [39,39]^1 \right\}$$
$$Z_2 = \left\{ [7,73]^0, [27,53.5]^{0.5}, [40,40]^1 \right\}$$
$$Z_3 = \left\{ [7,82]^0, [26.5,57.5]^{0.5}, [40,40]^1 \right\}$$
$$Z_4 = \left\{ [5,83]^0, [26.5,58.5]^{0.5}, [41,41]^1 \right\}$$

Step 5: Compute the fuzzy shortest path length (Z_{min}) using Definition 4.

Using Definition 4, the FSPL(Z_{min}) is calculated considering $Z_1, Z_2, Z_3 \, and \, Z_4$:

$$Z_{min} = \left\{ [5,73]^0, [25.5,53.5]^{0.5}, [39,39]^1 \right\}$$

Step 6: Convert $Z_I \quad for \, I = 1,2,\ldots\ldots\ldots\ldots,n$ and Z_{min} to QGFN.

Now the path lengths $Z_1, Z_2, Z_3, Z_4 \, and \, Z_{min}$ are converted back from decomposed fuzzy numbers to QGFN:

$$Z_1 = gfn^* [39,11,15]$$
$$Z_2 = gfn^* [40,11,11]$$
$$Z_3 = gfn^* [40,10,14]$$
$$Z_4 = gfn^* [41,13,15]$$
$$Z_{min} = gfn^* [39,11,12]$$

Step 7: Calculate the LPI between Z_{min} and $Z_I \quad for \; I = 1,2,\ldots\ldots\ldots,n$ using Definition 5. Assign ranks to each of the paths (P_I).

Using equation 15 of Definition 5, we get the LPI and ranks seen in Table 1.

Step 8: Identify the fuzzy shortest path as the one with the highest LPI.

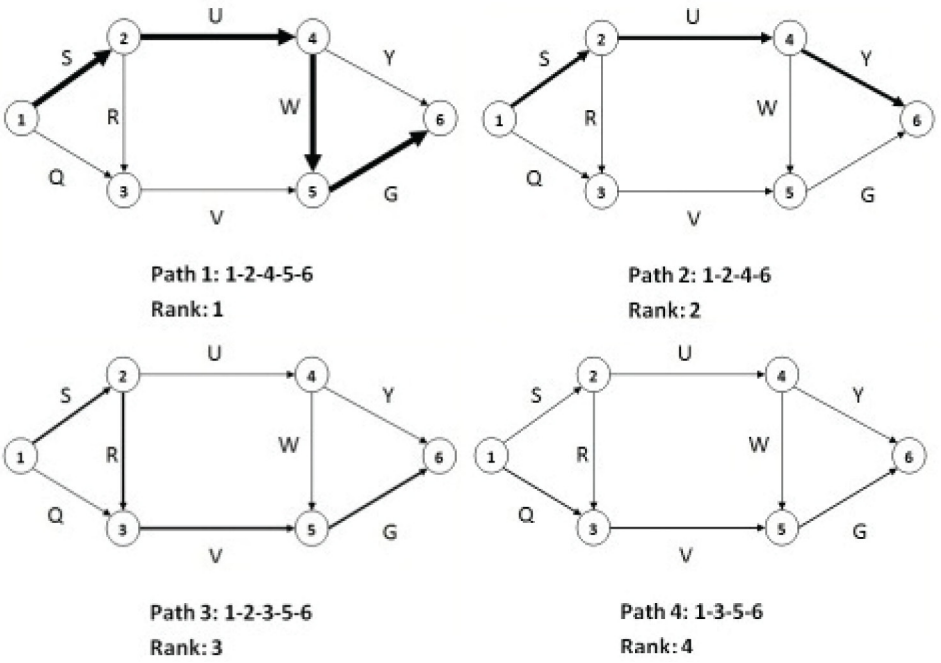
The different paths along with their ranks are shown in the following Figure 4.

The fuzzy shortest path in the network considered is P_1 which has the highest LPI value and hence the greatest rank 1.If all the weight spreads are set to zero, the QGFN is reduced to its crisp value and then it can be easily determined that the shortest path is P_1 which is the same as generated using LPI. The two path lengths Z_2 and Z_3 have the same modal values 40. Hence, using the centroid method, we cannot rank the two path lengths

Table 1. LPI value and ranks of different paths

Path	Equation	LPI Value	Rank
$P_1 : 1-2-4-5-6$	$LPI\left(Z_{min} < Z_1\right)$	1	1
$P_2 : 1-2-4-6$	$LPI\left(Z_{min} < Z_2\right)$	0.98	2
$P_3 : 1-2-3-5-6$	$LPI\left(Z_{min} < Z_3\right)$	0.97	3
$P_4 : 1-3-5-6$	$LPI\left(Z_{min} < Z_4\right)$	0.136	4

Figure 4. Ranking of paths in the network



but LPI has the advantage of ranking these two path lengths.

EXTENDED DIJKSTRA'S ALGORITHM USING FIBONACCI HEAP AND QUASI-GAUSSIAN MEMBERSHIP FUNCTIONS

The algorithm stated below is an extended version of the Dijkstra's algorithm that generates the shortest path for a source-destination pair.

If one requires all the shortest paths emanating from the source, the algorithm can be rerun using different targets or destination nodes. In steps 1-3, the array maintaining the value of LPI for each vertex is initialized with $-\infty$. In the next step, for the vertex chosen as source the value of LPI is replaced with a zero. In step 5, the ***INSERT ()*** function is used to initialize the priority queue Q with the LPI value of each vertex. In step 6, an array V_T, used to store the sequence of nodes in the shortest path is initialized. From steps 7-21, a while loop runs inside which u is a variable that holds the vertex with the maximum value of LPI in Q and ***DELETE_MAX ()*** is a function used to delete that vertex from Q and store it in variable u^* which is then placed in the array V_T. Then in step 14-15, for each neighbour v of vertex u, the new LPI value is calculated using the function ***LPI _ between***$\left(V_T, v\right)$ that uses equation 15. For the sequence of nodes in V_T, first the weights of the edges connecting them is added then that value is compared with the weights of the edges connecting the neighbouring vertices v of u. This new LPI value of vertex v is saved in the variable val. In steps 16-17, we compare to see if the new LPI value of v is greater than the previous one stored in the array LPI, if yes the value LPI [v] is updated and the ***INCREASE ()*** function is used to update the LPI value of vertex v in Q. In step 21 the while loop is ended when $u = t$ where t is the specified destination vertex. Finally, in step 22 the array V_T is returned which stores the shortest path.

When the priority queue Q is implemented as an ordinary array or a linked list, the running time of the algorithm is $O\left(\left|V\right|^2\right)$. When an adjacency list is used for storing a graph and a Fibonacci heap is used for implementing the priority queue Q, the running time of the algorithm is $O\left(\left|E\right| + \left|V\right| \log \left|V\right|\right)$ (Cormen et al., 1990). Also, since exponent is a monotonic function of its argument, the algorithm can still work if in step 15 of Algorithm 1, exponentiation is avoided and only the argument values are used.

CONCLUSION

In this paper we have suggested a method that determines the shortest path in the fuzzy environment using link preference index, the formula for which has been derived for the Quasi-Gaussian fuzzy numbers. The LPI assigns ranks to the different paths present in the network and the one with the highest LPI value is concluded as the shortest path which has been shown with an illustrative example. The method chosen for the intermediate fuzzy arithmetic operations lead to simpler calculations and satisfactory results. At the same time, the method can be applied in varied situations like when the value for k is small, the number of sub intervals are less but the evaluation is faster and can be implemented on a slower machine however for a faster machine by increasing the value of k we can get more and more accurate results. Also, an extended Dijkstra's algorithm has been proposed that can be implemented using a Fibonacci heap, deals with Quasi-Gaussian fuzzy numbers as weights and determines the fuzzy shortest path for a given network.

The LPI proposed in this paper is a monotonic decreasing function of $\Delta\overline{x}$. Links closer to FSPL are preferred by the index. It can also take care of non symmetry in the membership function. It tends to increase if Z_{min} is long tailed towards right or Z_1 is long tailed towards left (i.e. the two functions lean towards each other). In the future, it will be interesting to try and simplify the LPI expression and generalize it to LR fuzzy numbers modelled as parametric Bezier curves. Finally, the technique can be extended to other graph problems like travelling salesman problem, minimum spanning tree problem etc where fuzzy numbers can be used to represent uncertain path lengths.

Algorithm 1. Fuzzy_Dijkstra $\left(\boldsymbol{G},\boldsymbol{s},\boldsymbol{t}\right)$

```
for each vertex v in G :
    LPI[v] ← −∞ ;          // Initialization of the array storing the
                              Link Preference Index (LPI).
end for
LPI[s] ← 0 ;               // s is the source vertex.
INSERT (Q, v, LPI[v]) ;
// Initialize the Priority Queue Q and function INSERT () is used to update
   the queue.
V_T ← ∅ ;                  // V_T is an array storing the sequence of
                              nodes in the shortest path.
while u ≠ t :              // t is the destination vertex.
    u ← vertex in Q with greatest LPI[ ] ;
        if LPI[u] = −∞
            break ;
        end if
    u* ← DELETE_MAX(Q) ;
// DELETE_MAX () is used to remove the vertex with the greatest LPI value from
   the priority queue Q.
    V_T ← V_T ⋃ {u*} ;
            for each neighbour v of u :
                val ← LPI _ between(V_T, v) ;
// LPI_between () is a function that determines Link Preference Index (LPI)
   using equation (15) .
                    if val > LPI[v]
                        LPI[v] ← val ;
                        INCREASE(Q, v, LPI[v]) ;
// INCREASE () function updates the value of LPI for vertex v in the priority queue Q.
                    end if
            end for
end while
return V_T      // the sequence of nodes in the shortest path is returned.
end Fuzzy_Dijkstra
```

REFERENCES

Blue, M., Bush, B., & Puckett, J. (2002). Unified approach to fuzzy graph problems. *Fuzzy Sets and Systems*, *125*, 355–368. doi:10.1016/S0165-0114(01)00011-2.

Chuang, T. N., & Kung, J. Y. (2005). The fuzzy shortest path length and the corresponding shortest path in a network. *Computers & Operations Research*, *32*, 1409–1428. doi:10.1016/j.cor.2003.11.011.

Chuang, T. N., & Kung, J. Y. (2006). A new algorithm for the discrete fuzzy shortest path problem in a network. *Applied Mathematics and Computation*, *174*, 1660–1668. doi:10.1016/j.amc.2005.04.097.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (1990). *Introduction to algorithms*. New York, NY: McGraw Hill.

Dongrui, W. (2012). Twelve considerations in choosing between Gaussian and trapezoidal membership functions in interval type-2 fuzzy logic controllers. In *Proceedings of 2012 IEEE International Conference on Fuzzy Systems*.

Dubois, D., & Prade, H. (1978). Operations on fuzzy numbers. *International Journal of Systems Science*, *9*, 613–626. doi:10.1080/00207727808941724.

Dubois, D., & Prade, H. (1979). Fuzzy real algebra: Some results. *Fuzzy Sets and Systems*, *2*, 327–348. doi:10.1016/0165-0114(79)90005-8.

Dubois, D., & Prade, H. (1980). *Fuzzy sets and systems: Theory and applications*. New York, NY: Academic Press.

Ebrahimnejad, A. (2012). Cost efficiency measures with trapezoidal fuzzy numbers in data envelopment analysis based on ranking functions: Application in insurance organization and hospital. *International Journal of Fuzzy System Applications*, *2*(3), 51–68. doi:10.4018/ijfsa.2012070104.

Elizabeth, S., & Sujatha, L. (2011). Fuzzy shortest path problem based on index ranking. *Journal of Mathematics Research*, *3*(4). doi:10.5539/jmr.v3n4p80.

Garibaldi, J. M., & John, R. I. (2003). Choosing membership functions of linguistic terms. *2003 IEEE International Conference on Fuzzy Systems, 1*, 578-583.

Grauel, A., & Ludwig, L. A. (1999). Construction of differentiable membership functions. *Fuzzy Sets and Systems*, *101*(2), 219–225. doi:10.1016/S0165-0114(98)00165-1.

Hanss, M. (1999). On the implementation of fuzzy arithmetical operations for engineering problems. In *Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society* (pp. 462-466).

Hanss, M. (2005). *Applied fuzzy arithmetic: An introduction with engineering applications*. Berlin, Germany: Springer.

Hanss, M., & Willner, K. (2000). A fuzzy arithmetical approach to the solution of finite element problems with uncertain parameters. *Mechanics Research Communications*, *27*, 257–272. doi:10.1016/S0093-6413(00)00091-4.

Hernandes, F., Lamata, M. T., Verdegay, J. S., & Yamakami, A. (2007). The shortest path problem on networks with fuzzy parameters. *Fuzzy Sets and Systems*, *158*, 1561–1570. doi:10.1016/j.fss.2007.02.022.

Kashtiban, M. M., Khoei, A., & Hadidi, K. (2008). Optimization of rational-powered membership functions using extended Kalman filter. *Fuzzy Sets and Systems*, *159*(23), 3232–3244. doi:10.1016/j.fss.2008.06.021.

Klein, C. M. (1991). Fuzzy shortest paths. *Fuzzy Sets and Systems*, *39*, 27–41. doi:10.1016/0165-0114(91)90063-V.

Klir, G. J., & Yuan, B. (1997). *Fuzzy sets and fuzzy logic theory and applications*. New Delhi, India: Prentice-Hall of India Private Limited.

Kreinovich, V., Quintana, C., & Reznik, L. (1992). Gaussian membership functions are most adequate in representing uncertainty in measurements. In *Proceedings of NAFIPS'92: North American Fuzzy Information Processing Society Conference.*

Kung, J. Y., Chuang, T. N., & Lin, C. T. (2006). Decision making on network problem with fuzzy arc lengths. In *Proceedings of the IMACS Multiconference on Computational Engineering in Systems Applications* (pp. 578-580).

Moore, R. E. (1966). *Interval analysis*. Englewood Cliffs, NJ: Prentice-Hall.

Ramli, N., & Mohamad, D. (2009). A comparative analysis of Centroid methods in ranking fuzzy numbers. *European Journal of Scientific Research*, *28*(3), 492–501.

Sireesha, V., Ravishankar, N., Srinivasa Rao, K., & Phani Bhusan Rao, P. (2012). On the latest times and float times of activities in a fuzzy project network with LR fuzzy numbers. *International Journal of Fuzzy System Applications*, *2*(2), 91–101. doi:10.4018/ijfsa.2012040105.

Sujatha, L., & Elizabeth, S. (2011). Fuzzy shortest path problem based on similarity degree. *Applied Mathematical Sciences*, *5*(66), 3263–3276.

Zadeh, L. A. (1978). Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, *1*, 3–28. doi:10.1016/0165-0114(78)90029-5.

Madhushi Verma received the BE Degree in Computer Engineering from Jaipur Engineering College, Rajasthan University. She worked as a lecturer in Vivekananda Institute of Technology, Rajasthan Technical University. Now she is pursuing PhD in Computer Engineering from IIT-BHU, Varanasi.

K. K. Shukla is professor of Computer Engineering at Indian Institute of Technology, BHU, India. He has 30 years of research and teaching experience. Professor Shukla has published more than 120 research papers in reputed journals and conferences and has more than 90 citations. 15 PhDs have been awarded under his supervision so far. Professor Shukla has to his credit, many projects of national importance at BHU, Hindustan Aeronautics and Smiths Aerospace U.K. Presently he has research collaboration with Space Applications Center, ISRO, Tata Consultancy Services, Institut National de Recherche en Informatique et en Automatique (INRIA), France and École de Technologie Supérieure (ÉTS), Canada. He has written 4 books on Neuro-computers, Real Time Task Scheduling, Fuzzy modeling, Image Compression and has contributed chapters to 3 books published in the U.S. Professor Shukla is a Fellow of the Institution of Engineers, Fellow of the Institution of Electronics and Telecommunications Engineers, Senior Member, ISTE and the Senior Member, Computer Society of India.

Hindawi Publishing Corporation
Advances in Fuzzy Systems
Volume 2015, Article ID 569248, 12 pages
http://dx.doi.org/10.1155/2015/569248

Research Article

Application of Fuzzy Optimization to the Orienteering Problem

Madhushi Verma and K. K. Shukla

Department of Computer Science and Engineering, IIT (BHU), Varanasi 221005, India

Correspondence should be addressed to Madhushi Verma; madhushi.rs.cse@itbhu.ac.in

Received 5 February 2015; Accepted 18 June 2015

Academic Editor: Katsuhiro Honda

This paper deals with the orienteering problem (OP) which is a combination of two well-known problems (i.e., travelling salesman problem and the knapsack problem). OP is an NP-hard problem and is useful in appropriately modeling several challenging applications. As the parameters involved in these applications cannot be measured precisely, depicting them using crisp numbers is unrealistic. Further, the decision maker may be satisfied with graded satisfaction levels of solutions, which cannot be formulated using a crisp program. To deal with the above-stated two issues, we formulate the *fuzzy* orienteering problem (FOP) and provide a method to solve it. Here we state the two necessary conditions of OP of maximizing the total collected score and minimizing the time taken to traverse a path (within the specified time bound) as fuzzy goals and the remaining necessary conditions as crisp constraints. Using the max-min formulation of the fuzzy sets obtained from the fuzzy goals, we calculate the fuzzy decision sets (Z and Z^*) that contain the feasible paths and the desirable paths, respectively, along with the degrees to which they are acceptable. To efficiently solve large instances of FOP, we also present a parallel algorithm on CREW PRAM model.

1. Introduction

The orienteering problem (OP) is an NP-hard problem, derived from the game of orienteering where the player is required to start from the initial control point and arrive at the final control point within the specified time limit, at the same time collecting the rewards (Score) assigned to each of the checkpoints that link the initial control point to the final control point. All those players who arrive at the final control point after the time expires are disqualified and the player reaching the final control point within the allotted time and with the maximum collected score is declared as the winner of the game. This game reflects many real life situations related to the field of logistics, tourism, building telecommunication networks, home delivery systems, and so forth, which can be modelled as OP [1]. There are four types of OP which include the simple orienteering problem, team orienteering problem, orienteering problem with time window, and team orienteering problem with time window. The OP can be observed as a combination of two well-known problems, that is, travelling salesman problem (TSP) and the knapsack problem (KP), where the objective of maximizing the score is derived from KP and the objective of minimizing the time taken to travel from the initial control point and final control point is similar to the objective of TSP with the difference that in TSP all the vertices connecting the source to the target should be visited once, but in OP it is not necessary to visit all the intermediate checkpoints [1]. One real life application of OP is in the tourism industry where tourists come to visit the cities which are rich in culture and traditions and have some historical significance. The aim is to visit as many locations (including palaces, museums, monuments, and restaurants) as possible within their duration of stay. Due to the restriction of time it is not possible to visit all the places, so depending upon the choice and taste of the tourist, the guide can assign priorities to the various locations to be visited and then explore as many of them as possible within the time frame available so as to make their trip as economical and as beneficial as possible [2]. As seen in the stated example, the two parameters involved, that is, time and score, cannot be determined precisely as the priorities differ from one tourist to another and the time taken to travel from one location to another cannot be predicted exactly. So, the uncertainty in both the parameters can be modelled in a better way by representing them using fuzzy numbers instead of crisp or real numbers.

Since the orienteering problem is NP-hard, several heuristic algorithms and a few approximation algorithms are available in the literature. Some of the first heuristics for OP were the stochastic (S-algorithm) and deterministic (D-algorithm) ones suggested by Tsiligirides in 1984 [3]. In 1987, a center-of-gravity heuristic was proposed by Golden et al. [4]. A four-phase heuristic was introduced by Ramesh and Brown in 1991 [5]. An artificial neural network based approach was proposed by Wang et al. in 1995 [6] and, in 1996, a five-step heuristic was suggested by Chao et al. [7]. A branch-and-cut heuristic for OP was stated by Fischetti et al. in 1998 [8]. The first genetic algorithm for OP was developed by Tasgetiren in 2001 [9]. In 2002, Liang et al. proposed and compared a tabu search and an ant colony optimization approach for OP [10]. The approximations known for Prize Collecting Travelling Salesman Problem can be easily extended for the unrooted version of OP [11, 12] and, for the rooted version, a constant factor approximation was suggested by Blum et al. [13]. A heuristic method based on the greedy randomized adaptive search procedure and the path relinking methodologies was proposed in 2013 by Campos et al. [14].

This paper reports the first use of fuzzy logic to tackle the uncertainty involved in the two parameters (score and time). We introduce the fuzzy orienteering problem (FOP) and model the parameters involved as trapezoidal fuzzy numbers (TFNs). Several types of fuzzy numbers exist like Gaussian fuzzy numbers, exponential fuzzy numbers, quadratic fuzzy numbers, triangular fuzzy numbers, and trapezoidal fuzzy numbers, but here we prefer TFN because its computation is simple; it has a linear membership function and is the most generic class, so it is widely used in the applied engineering problems and in the scientific field.

In Section 2, we state necessary definitions and Section 3 provides the explanation of fuzzy optimization. In Section 4, the problem definition is stated. Section 5 explains the fuzzy formulation of OP along with the steps to deal with it and obtain a solution. The proposed method of solving FOP is demonstrated in Section 6 through an illustrative example. In Section 7, the parallel algorithm for FOP is presented and, finally, the paper is concluded in Section 8.

2. Preliminaries

Definition 1 (trapezoidal fuzzy numbers (TFNs)). A fuzzy number is an extension of the crisp or real number which refers to a set of possible values represented by a membership function, each of which is assigned a weight called its grade of membership between 0 and 1 whereas the crisp number is associated with a single value.

For any crisp set X which is a subset of the universal set U, the membership function is denoted by μ_X which assigns a value within $[0, 1]$ called the grade of membership to every element of X. So, the set $\widetilde{X} = \{(x, \mu_{\widetilde{X}}(x)) : x \in U\}$ is called a fuzzy set. A fuzzy set $\widetilde{X}$, whose membership function possess the properties stated below, is called a fuzzy number [15]:

(a) $\mu_{\widetilde{X}}$ is piecewise continuous,

(b) $\widetilde{X}$ is convex ($\mu_{\widetilde{X}}(\alpha x_1 + (1 - \alpha)x_2) \geq \min(\mu_{\widetilde{X}}(x_1), \mu_{\widetilde{X}}(x_2))$ $\forall x_1, x_2 \in \mathbb{R}, \forall \alpha \in [0, 1]$),

(c) $\widetilde{X}$ is normal ($\mu_{\widetilde{X}}(x_0) = 1$, $x_0 \in \mathbb{R}$), where $\mathbb{R}$ is the universal set of real numbers.

Fuzzy numbers are used to represent those parameters which are imprecise and, in several real life applications, parameters like cost, time, demand, capacity, and so forth are involved for which it is not possible to specify exact values. In these situations, it is appropriate to use a fuzzy number instead of a crisp number as they provide a more practical presentation of the real world. There are several types of fuzzy numbers, each of which can be defined by a membership function. In this paper, to represent the two parameters time and score as fuzzy numbers, we use TFN. For a trapezoidal fuzzy number represented as $\widetilde{A} = (a_1, a_2, a_3, a_4)$, the membership function is as follows [15]:

$$\mu_{\widetilde{A}}(x) = \begin{cases} 0, & x < a_1 \\ \dfrac{x - a_1}{a_2 - a_1}, & a_1 < x \leq a_2 \\ 1, & a_2 < x < a_3 \\ \dfrac{a_4 - x}{a_4 - a_3}, & a_3 \leq x < a_4 \\ 0, & x > a_4, \end{cases} \tag{1}$$

where $a_1 \leq a_2 \leq a_3 \leq a_4$.

Definition 2 (addition of TFN). In applications where the goal is to maximize or minimize some fuzzy quantity, some intermediate mathematical operations are to be performed which mostly include addition. For the problem stated here, we need to maximize the score acquired within a given time bound and for this we need to add the weights assigned to the vertices to calculate the score and the weights assigned to the edges to determine the time taken by a path. As stated the parameters here are considered to be TFNs and the mathematical operations performed on fuzzy numbers (like addition) differ from the usual crisp case.

Two TFNs represented as $\widetilde{A} = (a_1, a_2, a_3, a_4)$ and $\widetilde{B} = (b_1, b_2, b_3, b_4)$ can be added using the following formula [15]:

$$\begin{aligned} \widetilde{A} + \widetilde{B} &= (a_1, a_2, a_3, a_4) + (b_1, b_2, b_3, b_4) \\ &= (a_1 + b_1, a_2 + b_2, a_3 + b_3, a_4 + b_4). \end{aligned} \tag{2}$$

Definition 3 (ranking of TFN). In case of crisp numbers, there exists a natural ordering which clearly states that a real number is greater/smaller than other real numbers, but the absence of such a natural ordering in case of fuzzy numbers leads to the requirement of formulating a ranking method, using which the fuzzy numbers can be ordered. Here we use the circumcenter of centroids (COC) method for ranking TFN. Several methods are available in the literature for ranking of TFN, most of which use the centroid of the trapezoid for ranking considering it to be a balancing point, but in the method suggested by Rao and Shankar, the COC value is used for the ranking purpose. The COC is considered to be a better point of reference as it is obtained by calculating the circumcenter of the triangle formed by joining

the centroids of the three subparts of the trapezoid which include two triangles and one rectangle [16].

As suggested by Rao and Shankar [16], if $\widetilde{A} = (a_1, a_2, a_3, a_4)$, then

$$\mathrm{COC}\left(\widetilde{A}\right) = (x, y) = \left(\frac{a_1 + 2a_2 + 2a_3 + a_4}{6}, \frac{\left(2a_1 + a_2 - 3a_3\right)\left(2a_4 + a_3 - 3a_2\right) + 5}{12}\right). \tag{3}$$

Using this value of $\mathrm{COC}(\widetilde{A})$, the rank of the TFN $\widetilde{A}$ is obtained using the following formula:

$$R\left(\widetilde{A}\right) = \sqrt{x^2 + y^2}. \tag{4}$$

If we have two fuzzy numbers $\widetilde{A}$ and $\widetilde{B}$, then $\widetilde{A} > \widetilde{B}$ if $R(\widetilde{A}) > R(\widetilde{B})$ and vice versa. When $R(\widetilde{A}) = R(\widetilde{B})$, it is difficult to determine the ordering and some additional information is used as suggested by Rao and Shankar to tackle the situation.

Index of Optimism. The degree of optimism of the decision maker was taken care of by this index as stated by Rao and Shankar in [16]. So the decision maker's view point was also considered and, for any TFN $\widetilde{A} = (a_1, a_2, a_3, a_4)$ and $\mathrm{COC}(\widetilde{A})$, the formula for the index is as follows [16]:

$$I_\delta\left(\widetilde{A}\right) = \delta y + (1 - \delta)\, x \quad \text{where } \delta \in [0, 1]. \tag{5}$$

Larger value of δ shows that the decision maker is optimistic, whereas smaller values depict the situation where the decision maker is pessimistic.

Index of Modality. Another index was introduced by Rao and Shankar to tackle the situation where the rank of two TFNs becomes equal because the index of optimism alone may not be sufficient in such a case to determine the ranking as it considers only the importance of extreme values. The index of modality considers the importance of central value as well as the extreme values and the formula to determine the index is as follows [16]:

$$I_{\delta,\gamma}\left(\widetilde{A}\right) = \gamma\left(\frac{(x + y)}{2}\right) + (1 - \gamma)\, I_\delta\left(\widetilde{A}\right), \quad \text{where } \gamma \in [0, 1]. \tag{6}$$

If we have two TFNs $\widetilde{A}$ and $\widetilde{B}$, with $R(\widetilde{A}) = R(\widetilde{B})$, then $\widetilde{A} > \widetilde{B}$ when $I_{\delta,\gamma}(\widetilde{A}) > I_{\delta,\gamma}(\widetilde{B})$ and vice versa.

Definition 4 (expected value of trapezoidal fuzzy number (EV)). To determine the grade of membership of a TFN, that is, to find out the degree to which the TFN satisfies the specified requirement, we need to calculate its expected value and then use Definition 1. The formula for expected value as used by Jimenez et al. in [17] is as follows.

For a given TFN $\widetilde{A} = (a_1, a_2, a_3, a_4)$,

$$\mathrm{EV}\left(\widetilde{A}\right) = \frac{1}{4}\left(a_1 + a_2 + a_3 + a_4\right). \tag{7}$$

This defuzzification technique is used as it is the simplest possible and gives satisfactory results for the fuzzy orienteering problem. It will be interesting to study other defuzzification techniques although the general conclusions of the paper will remain unaffected.

Definition 5 (fuzzy decision set (Z)). Fuzzy decision set is a set of elements providing a feasible solution to the stated problem. The fuzzy linear programming problem can have several goals, each represented by a membership function and a fuzzy set (F_l) containing the elements along with their grades of membership obtained using the membership function [17].

We define the fuzzy decision as

$$Z = F_1 \cap F_2 \cap \cdots \cap F_l \quad \text{i.e. } \mu_Z(x) = \mu_{F_1}(x) * \mu_{F_2}(x) * \cdots * \mu_{F_l}(x), \tag{8}$$

where $*$ represents a t-norm which can be any operation like *minimum*, *algebraic product*, and so forth. In case of OP, $*$ represents *minimum* operation. To obtain the most desirable solution, we determine the element with the highest membership degree in the fuzzy decision set Z, that is, the value of x that maximizes the membership function of fuzzy decision Z denoted by x^*, using the equation stated below:

$$\mu_{Z^*}(x^*) = \max\left\{\mu_Z(x)\right\}, \tag{9}$$

where Z^* denotes the fuzzy set of the most desirable solutions.

3. Fuzzy Optimization

In engineering design or decision making problems, a large number of feasible solutions are available and to choose the solution which is the best one from this set, we need to concentrate on the uncertainty associated with the variables that lead to the optimal solution. Probabilistic concepts can take care of the randomness that arises due to natural fluctuation or natural variations, but the uncertainty that comes into existence due to qualitative statements, vague statements, vague nature of the objective, and linguistic statements showing the willingness of the decision maker (like the solution is acceptable, low, satisfactory, etc.) cannot be addressed through probabilistic concepts, so we introduce the concept of fuzzy logic in solving the optimization problems. In the crisp definition of optimization problems, we have crisp conditions where solutions violating the constraints or not satisfying the objective function are completely unacceptable, but, in fuzzy optimization, the concept of degree is introduced. The solution becomes a matter of degree; that is, degree of acceptability or degree of satisfaction is associated with the constraints and the objective functions, and this way we provide a latitude to the acceptability of a solution. This degree of acceptability linked with the objective functions and constraints can be reflected through fuzzy membership functions.

To deal with the situations where several stakeholders vaguely state their preferences as constraints or objective functions using linguistic statements, we convert these statements into fuzzy sets or fuzzy membership functions and

then using some technique find out the best "compromise solution." In fuzzy optimization, we do not distinguish between the objective functions and the constraints; instead we refer to them as fuzzy goals, represented in the form of fuzzy sets defined by their respective membership functions. So, the latitude or uncertainty present in the decision making is tackled through these membership functions. In addition to the fuzzy goals, we can also have crisp constraints modelling the physical conditions or technological feasibility that have to be met in a particular solution.

The whole idea of fuzzy optimization is to allow for latitude in the constraints and flexibility in the objective function. Instead of a 0-1 type solution, we allow for some violation of the original constraints to some degree, set certain limit for the objective function, and accept solutions on both sides of the limit to different degrees. The objective function and the set of constraints are converted into fuzzy sets; their associated membership functions are defined and then all the membership functions are combined to determine the fuzzy decision. Through fuzzy optimization, the preferences of the decision maker are quantified and the uncertainty due to vagueness, imprecision, and so forth, which is common in decision making problems, is tackled using membership functions [18–20].

4. Problem Definition

The orienteering problem can be represented by a completely connected undirected graph $G(V, E)$, where $V = \{v_1, \ldots, v_N\}$ is the set of vertices and E is the set of edges. A score S_i is associated with every vertex $v_i \in V$ and the time taken to traverse each edge $e_{ij} \in E$ is denoted by t_{ij}. The goal here is to determine a path P connecting any subset of V that necessarily includes the start vertex (v_1) and the end vertex (v_N), satisfies the time bound $T_{\max}$, and also maximizes the total collected score [1].

In this paper, we present the fuzzy orienteering problem (FOP) where the two quantities involved, that is, time and score, are considered to be fuzzy numbers. The reason to introduce fuzziness into the formulation of OP is that the crisp mathematical formulation is very strict in three ways: (1) the objective function should be either maximized or minimized, (2) none of the constraints should be violated as it leads to an infeasible solution, and (3) all constraints are given equal importance. However, these three necessary requirements lead to an unrealistic representation of the real world. By partly relaxing these using fuzzy logic, we can model the physical world in a more realistic manner. Several situations might exist in real life applications that can be easily represented in the fuzzy environment which may include the following [21].

(1) The decision maker is not willing to maximize or minimize the objective function; instead he wants to reach some aspiration level like "improving the present fuel consumption situation to some extent" in transportation problems which cannot even be defined or stated in the crisp case.

(2) Maybe the decision maker is willing to accept small violations in the constraints especially when the objective function deals with the aspiration levels and the less than or greater than relation is not required to be followed in the strict mathematical sense. Moreover, the parameters or variables involved may have vagueness which cannot be expressed in the crisp formulation.

(3) In the crisp representation, all constraints are of equal importance but maybe, for the decision maker, different constraints have different importance and small violations of different constraints may be acceptable to different degrees.

In this fuzzy formulation of OP, we have recognized that the parameters time and score are fuzzy in nature and in the fuzzy version we provide latitude to the desired solution by relaxing the constraints to some extent and stating the degree up to which they are feasible. Here we do not distinguish between objective function and constraints; instead the two necessary conditions of maximizing the total collected score and following the time bound are represented as two goals which are conflicting as one is to be maximized and the other one is to be minimized. These are represented as linear membership functions and the rest of the constraints are considered to be crisp. The detailed explanation of the fuzzy formulation is presented in the next section.

5. Fuzzy Formulation of OP

The fuzzy version can be represented in the following way showing by tilde the parameters that have a fuzzy character:

$$\sum_{i=1}^{N-1}\sum_{j=2}^{N}\widetilde{S_i}x_{ij} \gtrsim S_{\min}, \tag{10}$$

$$\sum_{j=2}^{N}x_{1j} = 1, \qquad \sum_{i=1}^{N-1}x_{iN} = 1, \tag{11}$$

$$\sum_{i=1}^{N-1}x_{ik} \leq 1 \quad \forall k = 2, \ldots, N-1, \tag{12}$$

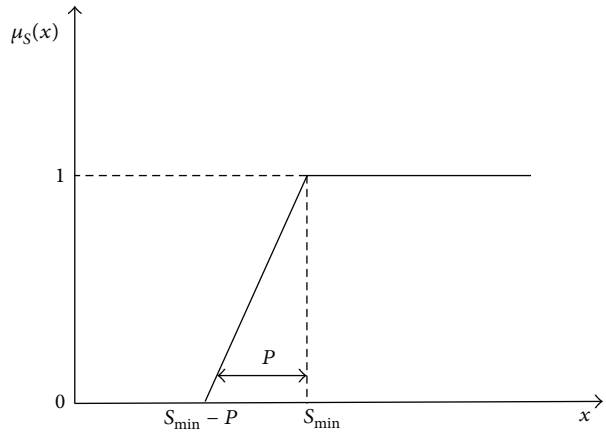
$$\sum_{j=2}^{N}x_{kj} \leq 1 \quad \forall k = 2, \ldots, N-1, \tag{13}$$

$$\sum_{i=1}^{N-1}\sum_{j=2}^{N}\widetilde{t_{ij}}x_{ij} \lesssim T_{\max}, \tag{14}$$

$$2 \leq u_i \leq N \quad \forall i = 2, \ldots, N, \tag{15}$$

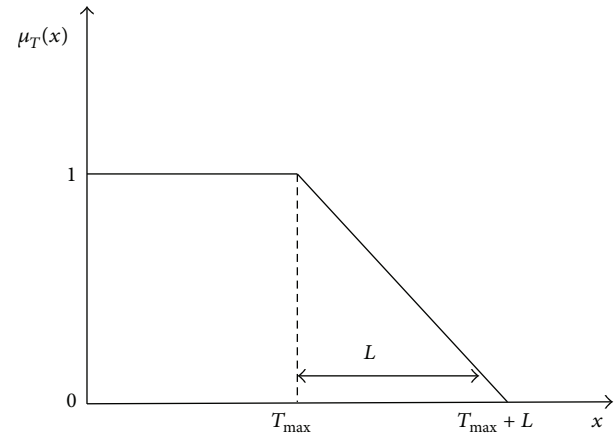
$$u_i - u_j + 1 \leq (N-1)\left(1 - x_{ij}\right) \quad \forall i, j = 2, \ldots, N, \tag{16}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j = 1, \ldots, N. \tag{17}$$



$$\mu_S(x) = \begin{cases} 0 & \text{if } x \le S_{\min} - P \\ \dfrac{x - S_{\min} + P}{P} & \text{if } S_{\min} - P < x < S_{\min} \\ 1 & \text{if } x \ge S_{\min} \end{cases}$$

Figure 1: Membership function for total collected score of a path.



$$\mu_T(x) = \begin{cases} 0 & \text{if } x \ge T_{\max} + L \\ \dfrac{T_{\max} - x + L}{L} & \text{if } T_{\max} < x < T_{\max} + L \\ 1 & \text{if } x \le T_{\max} \end{cases}$$

Figure 2: Membership function for total time taken to traverse a path.

As stated above, the constraints represented by (10) and (14) are considered to be fuzzy and represent the two conflicting goals of maximizing the total collected score and minimizing the time taken to traverse a path, respectively. The symbols $\widetilde{S_i}$ and $\widetilde{t_{ij}}$ in (10) and (14) represent the fuzzy score and fuzzy time values, respectively. The remaining constraints represented by (11), (12), (13), (15), (16), and (17) remain crisp. The variable u_i denotes the position of vertex v_i in the path and if vertex v_j is visited after vertex v_i, then $x_{ij} = 1$; else $x_{ij} = 0$. The necessary condition that each path starts in v_1 and ends in v_N is ensured by (11). The constraint that each path remains connected and no vertex is visited more than once in a path is taken care of by (12)-(13) and the requirement of eliminating subtours is implemented by (15)-(16) [1]. The symbols "$\gtrsim$" and "$\lesssim$" are the fuzzy version of "$\ge$" and "$\le$" representing the "fuzzy greater than or equal to" and "fuzzy less than or equal to," respectively. The meaning of these symbols is that the constraints can be violated to some extent and, depending on the importance of the constraint, this violation leads to different degree of acceptance [22]. For example, in (10), the symbol "$\gtrsim$" indicates that the total collected score of a particular path should be greater than $S_{\min}$ (most desirable case), but those paths which have their total collected score slightly less than $S_{\min}$, that is, up to $S_{\min} - P$, are also acceptable but to a lesser degree. Similarly, in (14), the total time taken to traverse a path should be less than $T_{\max}$ (most acceptable case), but paths having their total time greater than $T_{\max}$ are also acceptable to different degrees up to the limit of $T_{\max} + L$ as signified by the symbol "$\lesssim$." The two fuzzy goals can be represented by their membership functions as shown in Figures 1 and 2. The fuzzy decision sets Z and Z^* are depicted in Figure 3. It denotes the best "compromise solution" which is obtained by the following max-min formulation:

$$\begin{aligned} Z &= F_1 \cap F_2, \\ \mu_Z(x) &= \min\left\{\mu_T(x), \mu_S(x)\right\}, \\ \mu_{Z^*}(x^*) &= \max\left\{\mu_Z(x)\right\} \\ &= \max\left\{\min\left\{\mu_T(x), \mu_S(x)\right\}\right\}. \end{aligned} \tag{18}$$

The intersection of the two fuzzy sets (F_1 and F_2) is the minimum value of the two fuzzy sets for each x, which forms the fuzzy decision set Z (as shown by dark straight lines) and

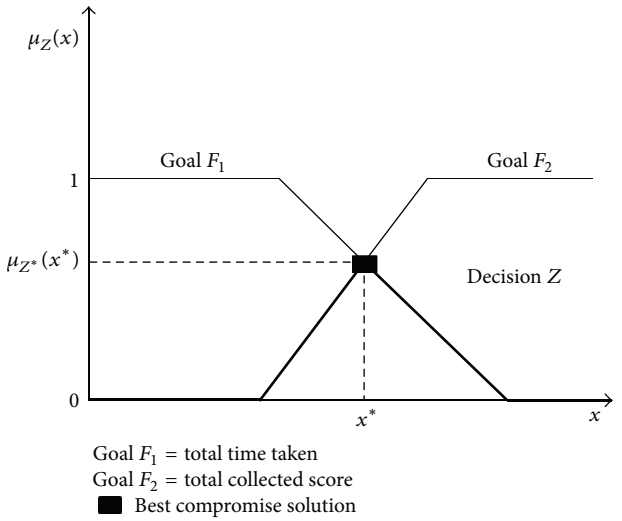


Figure 3: Fuzzy decision sets Z and Z^*.

the maximum value of this decision set Z forms the other set Z^* which holds the most desirable solution (as shown by the dashed line) of Figure 3.

FOP Algorithm. The following are the steps to determine the path that maximizes the total collected score within the specified time limit for a given $G(V, E)$ with N nodes.

(1) Determine all the possible paths (W_m) connecting v_1 and v_N (using (11), (12), (13), (15), (16), and (17)).

(2) For each possible path,

(a) calculate the total time taken to traverse the path and the total collected score (using Definition 2 and (2));

(b) calculate the expected value of the total time taken to traverse the path and the total collected score (using Definition 4 and (7));

(c) calculate its membership degree for the membership function of time (using (14) and Figure 2); let it be denoted by fuzzy set F_1;

(d) calculate its membership degree for the membership function of score (using (10) and Figure 1); let it be denoted by fuzzy set F_2.

(3) Determine the feasible paths denoted by the fuzzy decision set Z obtained using Definition 5 and (8).

(4) The most desirable path (final solution) is denoted by the fuzzy decision set Z^* obtained using Definition 5 and (9).

(5) If the fuzzy decision set Z^* contains more than one path, the total collected score (S_i) for each of the paths in Z^* can be ranked (using Definition 3 and (3), (4), (5), and (6)) to determine the path that maximizes the total collected score.

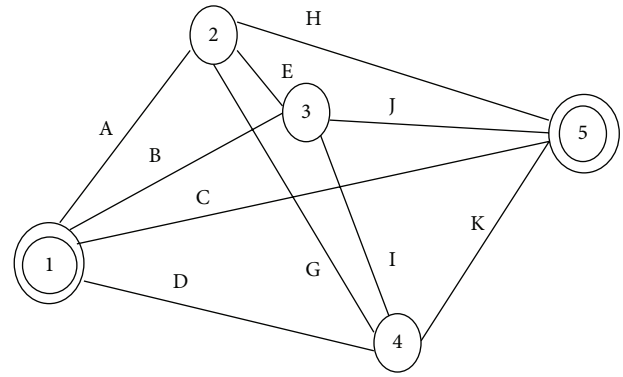


Figure 4: Input graph $G(V, E)$ with source vertex = 1 and destination vertex = 5.

Table 1: The values of edge weights (time taken to travel from one node to another).

Edge	Label	Fuzzy time values
e_{12}	A	(1, 4, 6, 9)
e_{13}	B	(6, 9, 13, 16)
e_{14}	D	(14, 15, 21, 22)
e_{15}	C	(4, 6, 8, 10)
e_{23}	E	(2, 3, 3, 4)
e_{24}	G	(5, 8, 8, 11)
e_{25}	H	(14, 18, 22, 26)
e_{34}	I	(5, 8, 12, 15)
e_{35}	J	(0, 2, 10, 12)
e_{45}	K	(1, 2, 2, 3)
	$T_{\max} = 20$, $L = 15$	

Table 2: The values of node weights (score values).

Node	Label	Fuzzy score values
v_1	1	(1, 2, 8, 9)
v_2	2	(8, 9, 11, 12)
v_3	3	(3, 5, 9, 11)
v_4	4	(17, 20, 24, 27)
v_5	5	(1, 2, 4, 5)
	$S_{\min} = 25$, $P = 13$	

6. Illustrative Example

Consider the network $G(V, E)$ in Figure 4 with total number of nodes $N = 5$ and the value for edge weights and node weights of G are stated in Tables 1 and 2, respectively.

Step 1. All possible paths connecting v_1 and v_5 are found that satisfy (11), (12), (13), (15), (16), and (17) and are as follows:

$$\begin{aligned}
&W_1\text{: 1-5,}\\
&W_2\text{: 1-2-5,}\\
&W_3\text{: 1-3-5,}\\
&W_4\text{: 1-4-5,}\\
&W_5\text{: 1-2-3-5,}\\
&W_6\text{: 1-3-2-5,}\\
&W_7\text{: 1-2-4-5,}\\
&W_8\text{: 1-4-2-5,}\\
&W_9\text{: 1-3-4-5,}\\
&W_{10}\text{: 1-4-3-5,}\\
&W_{11}\text{: 1-2-3-4-5,}\\
&W_{12}\text{: 1-2-4-3-5,}\\
&W_{13}\text{: 1-3-4-2-5,}\\
&W_{14}\text{: 1-4-3-2-5,}\\
&W_{15}\text{: 1-4-2-3-5,}\\
&W_{16}\text{: 1-3-2-4-5.}
\end{aligned} \tag{19}$$

Step 2. (a) Table 3 shows the total time taken to traverse and total collected score for each of the above stated paths which is calculated using Definition 2 and (2).

(b) Next we calculate the expected value of the total time taken to traverse the path and the total collected score using Definition 4 and (7) for each of the paths W_1 to W_{16} as shown in Table 4.

(c) and (d) The grade of membership of each possible path from W_1 to W_{16} is determined for both the membership functions (time and score) using (10) and (14) and Figures 1 and 2 and their values are shown in Table 5.

So, the two fuzzy sets F_1 and F_2 for the total time taken and total collected score, respectively, are as follows:

$$\begin{aligned}
F_1 &= \{W_1/1, W_2/0.67, W_3/1, W_4/1, W_5/1, W_6/0.06,\\
&\quad W_7/1, W_8/0, W_9/0.8, W_{10}/0.06, W_{11}/1, W_{12}/0.4,\\
&\quad W_{13}/0, W_{14}/0, W_{15}/0, P_{16}/0.73\},\\
F_2 &= \{W_1/0, W_2/0.23, W_3/0, W_4/1, W_5/0.77,\\
&\quad W_6/0.77, W_7/1, W_8/1, W_9/1, W_{10}/1, W_{11}/1, W_{12}/1,\\
&\quad W_{13}/1, W_{14}/1, W_{15}/1, W_{16}/1\}.
\end{aligned} \tag{20}$$

Step 3. Now the fuzzy decision set Z is obtained using Definition 5 and (8) to determine all the feasible paths along with the membership grades stating the degree up to which each of the feasible paths is acceptable.

Table 3: The total time taken and total collected score values for each of the paths.

Path	Total time taken to traverse	Total collected score
W_1	(4, 6, 8, 10)	(1, 2, 8, 9)
W_2	(15, 22, 28, 35)	(9, 11, 19, 21)
W_3	(6, 11, 23, 28)	(4, 7, 17, 20)
W_4	(15, 17, 23, 25)	(18, 22, 32, 36)
W_5	(3, 9, 19, 25)	(12, 16, 28, 32)
W_6	(22, 30, 38, 46)	(12, 16, 28, 32)
W_7	(7, 14, 16, 23)	(26, 31, 43, 48)
W_8	(33, 41, 51, 59)	(26, 31, 43, 48)
W_9	(12, 19, 27, 34)	(21, 27, 41, 47)
W_{10}	(19, 25, 43, 49)	(21, 27, 41, 47)
W_{11}	(9, 17, 23, 31)	(29, 36, 52, 59)
W_{12}	(11, 22, 36, 47)	(29, 36, 52, 59)
W_{13}	(30, 43, 55, 68)	(29, 36, 52, 59)
W_{14}	(35, 44, 58, 67)	(29, 36, 52, 59)
W_{15}	(21, 28, 42, 49)	(29, 36, 52, 59)
W_{16}	(14, 22, 26, 34)	(29, 36, 52, 59)

Table 4: The expected value of the total time taken to traverse the path and the total collected score for each of the paths.

Path	EV (total time taken)	EV (total collected score)
W_1	7	5
W_2	25	15
W_3	17	12
W_4	20	27
W_5	14	22
W_6	34	22
W_7	15	37
W_8	46	37
W_9	23	34
W_{10}	34	34
W_{11}	20	44
W_{12}	29	44
W_{13}	49	44
W_{14}	51	44
W_{15}	35	44
W_{16}	24	44

Here, $\mu_Z(x) = \min\{\mu_T(x), \mu_S(x)\}$, where $x = W_1$ to W_{16} is used to calculate Z:

$$\begin{aligned}
Z &= \{W_1/0, W_2/0.23, W_3/0, W_4/1, W_5/0.77, W_6/0.06,\\
&\quad W_7/1, W_8/0, W_9/0.8, W_{10}/0.06, W_{11}/1, W_{12}/0.4,\\
&\quad W_{13}/0, W_{14}/0, W_{15}/0, P_{16}/0.73\}.
\end{aligned} \tag{21}$$

Step 4. Finally, the fuzzy decision set Z^* is obtained using Definition 5 and (9) that contains the desirable path:

$$Z^* = \{W_4/1, W_7/1, W_{11}/1\}. \tag{22}$$

Step 5. As can be observed from Step 4, there are three paths that are desirable as they fulfil all the crisp constraints and

Table 5: The grade of membership of each possible path for the membership functions of both time and score.

Path (W_m)	$\mu_T(W_m)$	$\mu_S(W_m)$
W_1	1	0
W_2	0.67	0.23
W_3	1	0
W_4	1	1
W_5	1	0.77
W_6	0.06	0.77
W_7	1	1
W_8	0	1
W_9	0.8	1
W_{10}	0.06	1
W_{11}	1	1
W_{12}	0.4	1
W_{13}	0	1
W_{14}	0	1
W_{15}	0	1
W_{16}	0.73	1

the two fuzzy goals. To conclude with one most desirable path among the three available in Z^*, we use the ranking method described in Definition 3 and use (3), (4), (5), and (6) to rank the score of each path in Z^* and determine the best one. The values for which are shown in Table 6.

So, the most desirable path is W_{11} as it has the highest rank.

7. Parallel Formulation of FOP

To solve the FOP more efficiently and to yield a better and improved performance when applied on large instances, we present a parallel algorithm for FOP that uses the CREW PRAM (Concurrent Read Exclusive Write Parallel Random Access Machine) model. In a PRAM model, several processors are connected to a common block of shared memory. This shared memory can be accessed by all the processors and these processors communicate with each other by reading from or by writing to the shared memory. In a PRAM model, each processor is like a sequential RAM and can perform the arithmetic operations, assignment, comparison, memory access, and so forth in unit time. In CREW PRAM, more than one processor can read concurrently from the same cell of the shared memory but cannot write into the same cell concurrently; that is, write operation has to be exclusive [23, 24]. For each parallel algorithm, the following terms are computed to determine whether it is work optimal or not [24]:

$$\begin{aligned}
&\text{(a) speed up} = \frac{\text{runtime of the best known sequential algorithm}}{\text{runtime of the parallel algorithm for a } p\text{-processor machine}}\\
&\text{(b) total work done} = p * \text{runtime of the parallel algorithm for a } p\text{-processor machine}\\
&\text{(c) efficiency} = \frac{\text{runtime of the best known sequential algorithm}}{\text{total work done}}.
\end{aligned} \tag{23}$$

Any parallel algorithm is said to be work optimal if it has a linear speedup and an efficiency of 1.

If we have a graph with n nodes, then the maximum number of edges possible in the path connecting the source vertex (v_1) and the destination vertex (v_n) is $n-1$, so the total number of possible paths can be calculated using the following formula:

$$\text{Total no. of paths } (p) = \sum_{i=1}^{n-1} \frac{(n-2)!}{[n-(i+1)]!}. \tag{24}$$

Figure 5 shows the several steps of the proposed parallel algorithm for FOP.

In the parallel formulation of FOP, the module SEQ-NOP computes the total number of possible paths (p) for a given input graph $\{G, v_1, v_n\}$ using (24) sequentially. We assume that the number of processors is equal to the total number of paths (p). This value is utilized by the following parallel modules to output the path in G connecting v_1 and v_n that maximizes the total collected score and minimizes the total travel time. If the module PAR-MAX returns more than one path with the same maximum value of membership value (MV), then the module SEQ-RDP sequentially ranks the total collected score (TS) of the paths with the same maximum value of MV using the COC method of ranking fuzzy numbers and using Definition 3 and (3), (4), (5), and (6). Finally, the path with the highest rank is returned as the *most desirable path*.

The pseudocode for each of the modules is as follows.

PAR-PATH. See Pseudocode 1. The p valid paths can be computed in $O(1)$ time using p CREW PRAM processors.

So,

$$\begin{aligned}
\text{speed up} &= \frac{O(p)}{O(1)} = O(p),\\
\text{total work done} &= O(1) * p = O(p),\\
\text{efficiency} &= \frac{O(p)}{O(p)} = 1.
\end{aligned} \tag{25}$$

Therefore, this parallel module is work optimal.

PAR-TOT. See Pseudocode 2. The values for TTT and TS for each of the paths in D_k can be obtained in $O(1)$ time using p CREW PRAM processors.

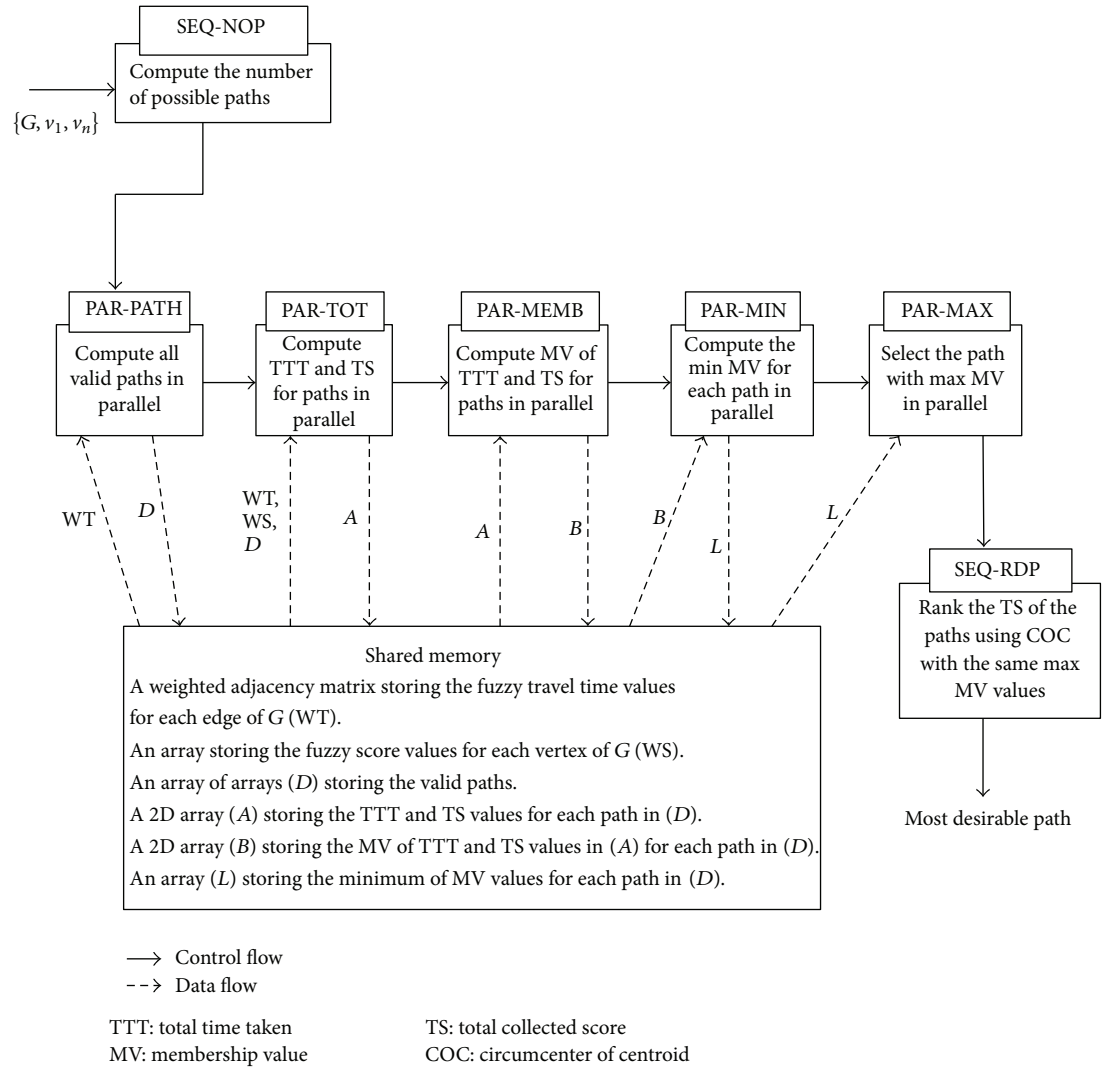


Figure 5: Several steps of the parallel formulation of FOP.

```
Processor k (in parallel for 1 ≤ k ≤ p) does:
{
cp := compute a unique path connecting v1 and vn;
   if (cp == VALID) then          // A path is valid if it satisfies the constraints stated in (11), (12), (13), (15), (16) and (17).
      Dk := cp;
   else
      DISCARD cp;
}
```

Pseudocode 1

Table 6: The ranks of the desirable paths.

Path	Score	Rank
W_4	(18, 22, 32, 36)	3
W_7	(26, 31, 43, 48)	2
$\mathbf{W_{11}}$	(**29**, **36**, **52**, **59**)	**1**

So,

$$\text{speed up} = \frac{O(p)}{O(1)} = O(p),$$
$$\text{total work done} = O(1) * p = O(p), \tag{26}$$
$$\text{efficiency} = \frac{O(p)}{O(p)} = 1.$$

Therefore, this parallel module is work optimal.

```
Processor k (in parallel for 1 ≤ k ≤ p) does:
{
    for (j = 0) do:
      {
        A_kj := Calculate the TTT for each VALID path in D_k;
        // TTT (Total Time Taken) for each VALID path is calculated using Definition 2 and (2).
      }
    for (j = 1) do:
      {
        A_kj := Calculate the TS for each VALID path in D_k;
        // TS (Total Collected Score) for each VALID path is calculated using Definition 2 and (2).
      }
}
```

Pseudocode 2

```
Processor k (in parallel for 1 ≤ k ≤ p) does:
{
    for (j = 0) do:
      {
        B_kj := Calculate the MV for TTT in A_kj for each VALID path in D_k;
        // MV (Membership Value) for TTT (Total Time Taken) is calculated using Definition 4 and (7), (14) and Figure 2.
      }
    for (j = 1) do:
      {
        B_kj := Calculate the MV for TS in A_kj for each VALID path in D_k;
        // MV (Membership Value) for TS (Total Collected Score) is calculated using Definition 4 and (7), (10) and Figure 1.
      }
}
```

Pseudocode 3

PAR-MEMB. See Pseudocode 3. The membership values for TTT and TS for each of the paths in D_k can be calculated in $O(1)$ time using p CREW PRAM processors.

So,

$$\text{speed up} = \frac{O(p)}{O(1)} = O(p),$$

$$\text{total work done} = O(1) * p = O(p), \tag{27}$$

$$\text{efficiency} = \frac{O(p)}{O(p)} = 1.$$

Therefore, this parallel module is work optimal.

PAR-MIN. See Pseudocode 4. The minimum of the two membership values for TTT and TS for each of the paths in D_k can be computed in $O(1)$ time using p CREW PRAM processors.

So,

$$\text{speed up} = \frac{O(p)}{O(1)} = O(p),$$

$$\text{total work done} = O(1) * p = O(p), \tag{28}$$

$$\text{efficiency} = \frac{O(p)}{O(p)} = 1.$$

Therefore, this parallel module is work optimal.

PAR-MAX. See Pseudocode 5. The maximum value amongst the values present in the array L can be determined in $O(p)$ time using p CREW PRAM processors.

So,

$$\text{speed up} = \frac{O(p^2)}{O(p)} = O(p),$$

$$\text{total work done} = O(p) * p = O(p^2), \tag{29}$$

$$\text{efficiency} = \frac{O(p^2)}{O(p^2)} = 1.$$

Therefore, this parallel module is work optimal.

```
Processor k (in parallel for 1 ≤ k ≤ p) does:
{
    for (j = 0) do:
      {
        L[k] := MIN(B[k][j], B[k][j + 1])
        \\ Compute the minimum of the two values using Definition 5 and (8).
      }
}
```

Pseudocode 4

```
Processor k (in parallel for 1 ≤ k ≤ p) does:
{
  G[k] := GRADE of L[k];
  // GRADE of any element L[k] in L is equal to one plus the number of elements in L smaller than L[k].
  PRINT the path in D with the greatest GRADE in G.
}
```

Pseudocode 5

8. Conclusion

In this paper, we deal with the orienteering problem which is an NP-hard problem and suggest a method to solve the fuzzy orienteering problem. As stated before, in these types of problems, the parameters involved like time and score are naturally imprecise, so representing them using real or crisp numbers leads to an unrealistic presentation of the real world. To provide a realistic model, fuzzy numbers can be used as they are capable of modelling the uncertainty present in the parameters and here we use trapezoidal fuzzy numbers to represent the values of the parameters involved (i.e., time and score). In the literature, the integer program formulation of OP has been stated. For these kinds of formulations, the objective function (either maximization or minimization) and the constraints are to be followed strictly in mathematical sense. However, this may not be the case in real life application where the decision maker may be willing to relax the constraints to some extent to generate a solution that can achieve a level of satisfaction which cannot be represented by the crisp formulation, but the fuzzy version can take care of these requirements. In our work, the two necessary conditions of maximizing the total collected score and that too within the specified time bound is represented by two fuzzy goals and the other requirements like each vertex is visited once, no sub tours are formed etc. are represented as crisp constraints. By representing the two necessary conditions in the form of fuzzy goals we provide a latitude to the desired solution by relaxing the constraints as can be observed in Figures 1 and 2 where we have fixed a time bound $T_{\max}$ and a minimum score $S_{\min}$ for the desired solution, but we accept solution up to $T_{\max} + L$ and $S_{\min} - P$ up to different degrees and this helps the decision maker to achieve the level he wants in his solution. This approach requires that the shapes and parameters for the membership functions be decided depending on the application at hand taking into account the extent of uncertainty prevailing. To make the proposed method applicable to large instances, we also present a work optimal parallel formulation of FOP. When provided with several graphs as input, further speedup can be expected since the stages will form a 5-stage linear pipeline providing a speedup of 5 ideally when all the stages are busy. In this way the method suggested above provides a more practical illustration of the physical world and generates a solution that is more appropriate and realistic.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The first author would like to acknowledge the financial support by IIT (BHU) in terms of teaching assistantship.

References

[1] P. Vansteenwegen, W. Souffriau, and D. van Oudheusden, "The orienteering problem: a survey," *European Journal of Operational Research*, vol. 209, no. 1, pp. 1–10, 2011.

[2] M. Schilde, K. F. Doerner, R. F. Hartl, and G. Kiechle, "Metaheuristics for the bi-objective orienteering problem," *Swarm Intelligence*, vol. 3, no. 3, pp. 179–201, 2009.

[3] T. Tsiligirides, "Heuristic methods applied to orienteering," *Journal of the Operational Research Society*, vol. 35, no. 9, pp. 797–809, 1984.

[4] B. L. Golden, L. Levy, and R. Vohra, "The orienteering problem," *Naval Research Logistics*, vol. 34, no. 3, pp. 307–318, 1987.

[5] R. Ramesh and K. M. Brown, "An efficient four-phase heuristic for the generalized orienteering problem," *Computers & Operations Research*, vol. 18, no. 2, pp. 151–165, 1991.

[6] Q. Wang, X. Sun, B. L. Golden, and J. Jia, "Using artificial neural networks to solve the orienteering problem," *Annals of Operations Research*, vol. 61, no. 1, pp. 111–120, 1995.

[7] I. Chao, B. Golden, and E. Wasil, "Theory and methodology—a fast and effective heuristic for the orienteering problem," *European Journal of Operational Research*, vol. 88, pp. 475–489, 1996.

[8] M. Fischetti, J. J. S. González, and P. Toth, "Solving the orienteering problem through branch-and-cut," *INFORMS Journal on Computing*, vol. 10, no. 2, pp. 133–148, 1998.

[9] M. Tasgetiren, "A genetic algorithm with an adaptive penalty function for the orienteering problem," *Journal of Economic and Social Research*, vol. 4, pp. 1–26, 2001.

[10] Y.-C. Liang, S. Kulturel-Konak, and A. E. Smith, "Meta heuristics for the orienteering problem," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, pp. 384–389, Honolulu, Hawaii, USA, May 2002.

[11] B. Awerbuch, Y. Azar, A. Blum, and S. Vempala, "Improved approximation guarantees for minimum-weight k-trees and prize-collecting salesmen," *SIAM Journal on Computing*, vol. 28, pp. 254–262, 1999.

[12] D. Johnson, M. Minkoff, and S. Phillips, "The prize collecting steiner tree problem: theory and practice," in *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 760–769, 2000.

[13] A. Blum, S. Chawla, D. R. Karger, T. Lane, A. Meyerson, and M. Minkoft, "Approximation algorithms for orienteering and discounted-reward TSP," in *Proceedings 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS '03)*, pp. 1–10, October 2003.

[14] V. Campos, R. Martí, J. Sánchez-Oro, and A. Duarte, "GRASP with path relinking for the orienteering problem," *Journal of the Operational Research Society*, vol. 65, no. 12, pp. 1800–1813, 2013.

[15] A. Bansal, "Trapezoidal fuzzy numbers (a, b, c, d): arithmetic behavior," *International Journal of Physical and Mathematical Sciences*, vol. 2, no. 1, pp. 39–44, 2011.

[16] P. P. Rao and N. R. Shankar, "Ranking fuzzy numbers with a distance method using circumcenter of centroids and an index of modality," *Advances in Fuzzy Systems*, vol. 2011, Article ID 178308, 7 pages, 2011.

[17] M. Jimenez, M. Arenas, A. Bilbao, and M. V. Rodriguez, "Linear programming with fuzzy parameters: an interactive method resolution," *European Journal of Operational Research*, vol. 177, no. 3, pp. 1599–1609, 2007.

[18] U. Kaymak and J. M. Sousa, *Weighted Constraints in Fuzzy Optimization*, Report Series Research in Management, ERIM Research Program: "Business Processes, Logistics and Information Systems", 2001, http://repub.eur.nl/res/pub/85/erimrs20010403174009.pdf.

[19] J. Ramik, *Soft Computing: Overview and Recent Developments in Fuzzy Optimization*, 2001, http://irafm.osu.cz/research_report/118_softco01.pdf.

[20] D. P. Loucks and E. V. Beek, *Water Resources Systems Planning and Management: An Introduction to Methods, Models and Applications*, UNESCO Publishing, 2005.

[21] H. J. Zimmermann, "Fuzzy set theory," *WIREs Computational Statistics*, vol. 2, pp. 317–332, 2010.

[22] F. Jarray, "Discrete tomography and fuzzy integer programming," *Iranian Journal of Fuzzy Systems*, vol. 8, pp. 41–48, 2011.

[23] G. E. Blelloch and B. M. Maggs, "Parallel Algorithms," http://homes.cs.washington.edu/~arvind/cs424/readings/BlellochM96b.pdf.

[24] E. Horowitz, S. Sahni, and S. Rajasekaran, *Computer Algorithms/C++*, Galgotia Publications, New Delhi, India, 1999.

ISSN 2277-3061

Roulette Wheel Selection based Heuristic Algorithm for the Orienteering Problem

Madhushi Verma*, Mukul Gupta, Bijeeta Pal, K. K. Shukla

Department of Computer Science and Engineering, IIT (BHU), Varanasi- 221005, India
madhushi.rs.cse@itbhu.ac.in
Department of Computer Science and Engineering, IIT (BHU), Varanasi- 221005, India
mukul.gupta.cse10@itbhu.ac.in
Department of Computer Science and Engineering, IIT (BHU), Varanasi- 221005, India
bijeeta.pal.cse10@itbhu.ac.in
Department of Computer Science and Engineering, IIT (BHU), Varanasi- 221005, India
kkshukla.cse@itbhu.ac.in

ABSTRACT

Orienteering problem (OP) is an NP-Hard graph problem. The nodes of the graph are associated with scores or rewards and the edges with time delays. The goal is to obtain a Hamiltonian path connecting the two necessary check points, i.e. the source and the target along with a set of control points such that the total collected score is maximized within a specified time limit. OP finds application in several fields like logistics, transportation networks, tourism industry, etc. Most of the existing algorithms for OP can only be applied on complete graphs that satisfy the triangle inequality. Real-life scenario does not guarantee that there exists a direct link between all control point pairs or the triangle inequality is satisfied. To provide a more practical solution, we propose a stochastic greedy algorithm (RWS_OP) that uses the roulette wheel selection method, does not require that the triangle inequality condition is satisfied and is capable of handling both complete as well as incomplete graphs. Based on several experiments on standard benchmark data we show that RWS_OP is faster, more efficient in terms of time budget utilization and achieves a better performance in terms of the total collected score as compared to a recently reported algorithm for incomplete graphs.

Keywords

Orienteering problem; complete graphs; incomplete graphs; roulette wheel selection method; Heuristic algorithm.

1 INTRODUCTION

Orienteering problem (OP) is a challenging NP-Hard combinatorial optimization graph problem that originated from a water sport where players need to visit a set of control points, starting at the source and reach the destination within a fixed time frame with an objective of collecting maximum possible rewards (associated with each control point). This problem is a combination of the travelling salesman problem (TSP) and the Knapsack problem (KP). The goal of maximizing the total collected score in OP is derived from KP and that of minimizing the total travel time from TSP. Unlike TSP, in OP it is not mandatory to visit each and every node of the network (Vansteenwegen et al., 2011). Many real-life situations can be modeled as OP like the transportation networks, home delivery system, telecommunication networks, logistics, trip planning in tourism industry, robot path planning, etc. In disaster management, a robot can be used with a map of its surrounding area to supply the necessary items like food, medicines, etc. to the affected locations. However, because the robot is battery operated, it cannot visit all the locations and the need here is to determine a path connecting the maximum number of locations (which are highly affected and requires immediate attention) that can be visited within the time bound (battery life of the robot) (Blum et al., 2003). Another example is that of a travelling salesman who does not have enough time to visit all the cities, but has some knowledge about the cities where maximum sales can take place. Therefore, the goal here is to maximize the total sales, but within the time constraint of a day or a week (Tsiligirides, 1984). In each of the stated applications, the task is to determine a solution, i.e. a path that collects the maximum rewards possible and also satisfies the time bound (Vansteenwegen et al., 2011; Schilde et al., 2009). Several variants of OP exist, which include team orienteering problem, orienteering and team orienteering with time window and the generalized orienteering problem (Vansteenwegen et al., 2011).

As stated earlier, OP is an NP-Hard problem so an exact algorithm is not practically feasible in terms of execution time for larger instances. However, a few exact algorithms were suggested by Laporte et al and Hayes et al based on the concepts of linear programming and dynamic programming (Laporte and Martello, 1990; Hayes and Norman, 1984). Other methods available in the literature include heuristic and approximation algorithms for OP. In 1984, one of the first heuristics for OP was proposed by Tsiligirides (1984). A four-phase heuristic and a centre-of-gravity heuristic was suggested by Ramesh and Brown and Golden et al respectively (Ramesh and Brown, 1991; Golden et al., 1987). Later, Fischetti et al stated a branch and cut heuristic for OP (Fischetti et al., 1998). Several other methods like artificial neural network, tabu search, pareto ant colony optimization, pareto variable neighbourhood search, etc. were introduced to deal with OP (Schilde et al., 2009; Wang et al., 1995; Gendreau et al., 1998). In 2013, Campos et al presented a Greedy Randomized Adaptive Search Procedure and the Path Relinking approach to solve OP (Campos et al., 2013). An approximation for the un-rooted version of OP was suggested (Awerbuch et al., 1999; Johnson et al., 2000) and Blum et al proposed an approximation algorithm for the rooted version of OP (Blum et al., 2003). Fomin et al, introduced an approximation algorithm for the time-dependent variant of OP (Fomin and Lingas, 2002). Most of the suggested algorithms for OP can be applied on complete graphs only but considering the practical applications of OP, it can be observed that many situations cannot be modeled only through complete graphs. The first genetic algorithm (GA) for OP was proposed by Tasgetiren (Tasgetiren, 2001) but Ostrowski et al presented a genetic algorithm for OP that can be solved for both complete as well as incomplete graphs (Ostrowski and Koszelew, 2011).

Here, we propose a stochastic greedy heuristic for OP (RWS_OP) that uses the roulette wheel selection method for determining the path that maximizes the total collected score within the specified time frame (T_{max}). The algorithm is guaranteed to reach the destination node (v_N) since the starting node (v_1) and the final node (v_N) are always the end points for all the generated paths. One necessary condition of OP is to ensure that a node is visited at most once, which in our algorithm is implemented by removing the explored nodes from the set of available nodes. This algorithm can be applied on both complete as well as incomplete graphs. We also compare our results with those reported by Ostrowski et al for large instances to show that RWS_OP executes more efficiently. Ostrowski et al proposed two algorithms one for incomplete graphs (IG) and the other that requires conversion of IG to complete graphs (CG) before OP can be solved. Few drawbacks of their paper are: (1) The paper does not categorically provide conclusive evidence about which of these algorithms is better. (2) Further, the authors allow each vertex to be visited more than once, but the reward is collected only on the first visit. This strategy is not only disadvantageous from the time budget point of view, but also produces invalid results (i.e. Non-Hamiltonian path). (3) In the second version of their algorithm, virtual edges need to be added using Dijkstra's algorithm, which results in unnecessary complication. (4) Their strategy requires application-specific complex crossover and mutation operations that often produce infeasible partial solutions that require additional correction/repair operations. In this paper, we propose a stochastic greedy algorithm that uses roulette wheel selection to avoid the search getting trapped in local maxima and removes all the disadvantages of Ostrowski's method mentioned above. It also outperforms Ostrowski's algorithm by improving the score and utilizing the time budget up to almost 99%. The objective function used in our method is motivated by greedy adaptive search procedure and path relinking (GRASP) technique suggested by Campos et al (Campos et al 2013). We show that roulette wheel selection with our new instance coding technique outperforms full GA implementation by Ostrowski et al in terms of both quality of solution and search time and space. In our method of candidate representation, we start with the shortest path between v_1 and v_N and using the roulette wheel proportionate selection scheme, keep adding nodes until T_{max} is reached. At each step, the probability of selecting a node is proportionate to its fitness defined in equation 9. Thus, without using crossover and mutation and without the need to maintain a population of possible solution, this algorithm is able to outperform GA based technique reported by Ostrowski et al. In section 2 of this paper, the problem formulation of OP is explained. Section 3 presents the various steps of the proposed algorithm. The observations based on the experiments performed on standard benchmarks are explained in section 4 and 5. Finally, the paper is concluded in section 6.

ISSN 2277-3061

2 PROBLEM FORMULATION

The orienteering problem can be modelled using a (complete or incomplete) weighted undirected graph $G(V,E)$ where $V = \{v_1, \dots\dots, v_N\}$ denotes the set of vertices and E denotes the set of edges. Let the time function on edges be $t: E \to \Re^+$ and a score function on nodes be $S: V \to \Re^+$. So if $V' \subseteq V$ then $S(V') = \sum_{v \in V'} S_v$ and if $E' \subseteq E$ then $t(E') = \sum_{e \in E'} t(e)$. The task is to compute a Hamiltonian path P, within the stated time bound (T_{max}) that connects the specified source (v_1), target (v_N) and also includes a subset (V') of V such that the total collected score is maximized (Vansteenwegen et al., 2011). To achieve the stated goal, here we use roulette wheel selection for exploring the various possible paths and return one that maximizes the total collected score within the specified time limit (T_{max}).

The OP can be presented as an integer programming problem and the formulation for the same is as follows (Vansteenwegen et al., 2011):

$$Max \sum_{i=1}^{N-1} \sum_{j=2}^{N} S_i x_{ij} \quad (1)$$

$$\sum_{j=2}^{N} x_{1j} = 1 \quad , \quad \sum_{i=1}^{N-1} x_{iN} = 1 \quad (2)$$

$$\sum_{i=1}^{N-1} x_{ik} \leq 1 \quad \forall k = 2, \dots\dots, N-1 \quad (3)$$

$$\sum_{j=2}^{N} x_{kj} \leq 1 \quad \forall k = 2, \dots\dots, N-1 \quad (4)$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} t_{ij} x_{ij} \leq T_{max} \quad (5)$$

$$2 \leq u_i \leq N \quad \forall i = 2, \dots\dots, N \quad (6)$$

$$u_i - u_j + 1 \leq (N-1)(1 - x_{ij}) \quad \forall i,j = 2, \dots\dots\dots, N \quad (7)$$

$$x_{ij} \in \{0,1\} \quad \forall i,j = 1, \dots\dots, N \quad (8)$$

Here, the variable u_i denotes the position of vertex v_i in the path and if a vertex v_j is visited after vertex v_i , then x_{ij} = 1 else x_{ij} = 0. Equation 1 ensures that the objective of OP, which is maximization of the total collected score is fulfilled. The necessary condition that each path starts in v_1 and ends in v_N is ensured by equation 2. The constraint that each path remains connected and no vertex is visited more than once in a path is taken care by equations 3-4. The total time taken to traverse a path is within the specified time limit is ensured by equation 5. The requirement of eliminating sub tours is implemented by equation 6-7.

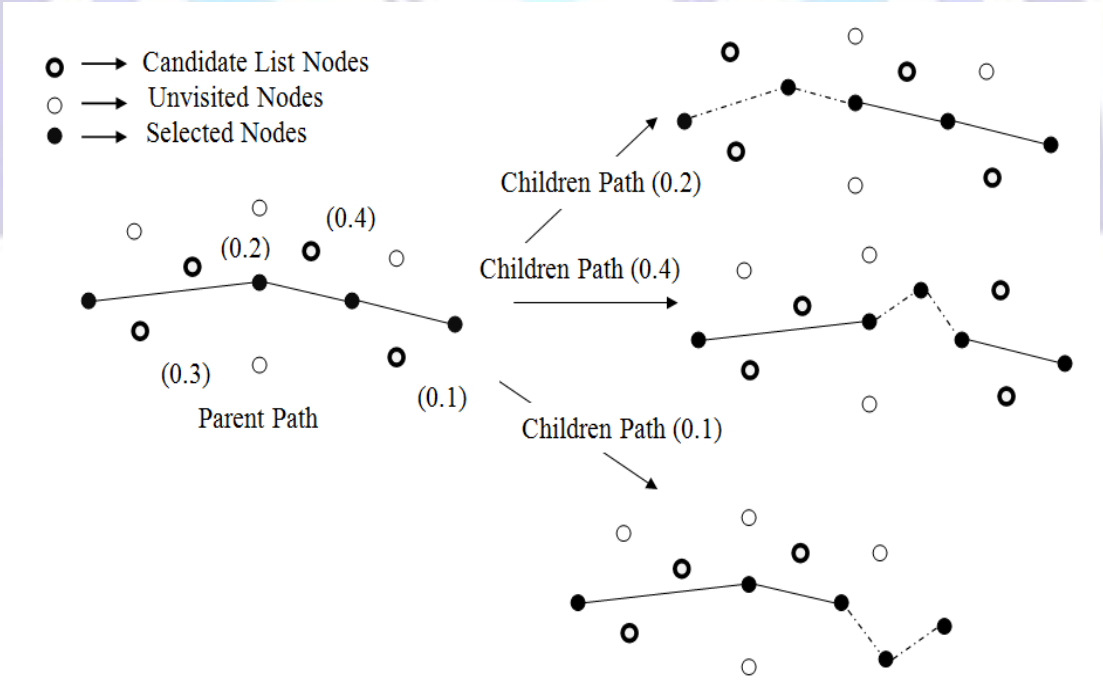


Fig 1: The process of selecting a path using roulette wheel selection function where the number in () denotes the probability of node selection.

ISSN 2277-3061

3 ALGORITHM RWS_OP

Input: A graph $G(V,E)$ with t_{ij}(time taken to traverse) value of each edge (e_{ij}) connecting vertex v_i and $v_j \in V$, S_i (Score) value of each vertex $v_i \in V$ and the $PathListSize$(maximum number of paths considered at each level).

Output: A Hamiltonian path with the highest possible collected score such that total travel time is within the specified time budget.

$Path$ is an array of nodes or vertices, which is a sequence connecting the source and the target.

$PathList$, $NewPathList$ and $ChildPathList$, are all queues and each of its element is a $Path$. $PathListSize$ is a constant whose value defines the maximum number of paths to be considered after each iteration.

$RWS\_OP(G, PathListSize, T_{max})$

1. ***Create*** $PathList$; // Array of paths which is initially empty.
2. $PathList \leftarrow \emptyset$;
3. $Path$ P $\leftarrow Dijkstra(v_1, v_n)$; // Shortest path between source (v_1) and target (v_N).
4. ***Insert*** P *to* $PathList$;
5. ***return*** $Generation(PathList)$;

$Generation(PathList)$

1. ***Create*** $NewPathList$,$Childpath List$; // Queues storing paths.
   $NewPathList \leftarrow \emptyset$;
   $ChildPathList \leftarrow \emptyset$;
2. ***for*** $i \leftarrow 0$ *to* $|PathList|$
   $a.Selection(PathList[i], ChildPathList)$; // $ChildPathList$ will contain children generated from each path in $PathList$.
3. ***for*** $i \leftarrow 0$ *to* $PathListSize$ // Selecting best $PathListSize$ children for next generation.
   $a.ChildPath = BestPath(ChildPathList)$; // The path with the maximum total score.
   Remove $ChildPath$ *from* $ChildPathList$;
   Insert $ChildPath$ *to* $NewPathList$;
   $b.$***if*** $|ChildPathList| == 0$
   break;
4. ***if*** $NewPathList == PathList$ // Terminate if no new child is generated and return the $BestPath$.
   return $BestPath(PathList)$;
5. ***return*** $Generation(NewPathList)$;

$Selection(Path\ P, ChildPathList)$ //$Path$ is an array of nodes that forms a path.

1. $q_{max} \leftarrow 0$;
2. ***for*** $i \leftarrow 0$ *to* $|V|$
3. $a.$***if*** $i \in P$ // If a node is already present in the path then ignore it.
   continue;
   $b.$***Calculate*** Δt_i ; // The time increment due to insertion of v_i at its best position.

$$c.q_i = \begin{cases} S_i/|\Delta t_i|, & \Delta t_i \geq 1 \\ S_i, & -1 \leq \Delta t_i < 1 \\ S_i * |\Delta t_i|, & \Delta t_i < -1 \end{cases};$$

$d.$ ***if*** $q_{max} \leq q_i$

$q_{max} \leftarrow q_i$

4. $Create\ CandidateList;$ // Array of candidate nodes used for roulette wheel selection.
5. $CandidateList \leftarrow \emptyset;$
6. ***for*** $i \leftarrow 0$ *to* $|V|$
7. $a.$***if*** $i \in P$

   continue;

   $b.$***if*** $(q_i \geq \alpha\ q_{max})\&\&\ (t_{parent} + \Delta t_i \leq T_{max})$ // α is the greediness parameter that decides which node should participate in roulette wheel selection.

   $c.$***Insert*** v_i *to* $CandidateList;$
8. ***if*** $|CandidateList| == 0$

   Insert P *to* $ChildPathList;$ // If no new nodes are added then insert the parent path P.

   Return;
9. ***for*** $i \leftarrow 0$ *to* $PathListSize$ // Generates $PathListSize$ number of children paths from path P.
10. $a.ChildNode \leftarrow RouleteWheel(CandidateList)$

    Remove $ChildNode\ from\ CandidateList;$

Insert $ChildNode$ *to Path P and* ***Insert*** *Path P to ChildPathList.*

Remove $ChildNode$ *From Path P;*

$b.$***if*** $|CandidateList| = 0$

break;

$RouleteWheel(CandidateList)$

1. $sum \leftarrow 0;$
2. ***for*** $i \leftarrow 0$ *to* $|CandidateList|$

   $sum += fitness(i);$ //$fitness(i) \leftarrow q_i$, where $q_i = \begin{cases} S_i/|\Delta t_i|, & \Delta t_i \geq 1 \\ S_i, & -1 \leq \Delta t_i < 1 \\ S_i * |\Delta t_i|, & \Delta t_i < -1 \end{cases}$
3. $RandomNumber \leftarrow Rand()\%(sum + 1)$
4. $PartialSum \leftarrow 0;$
5. $Selected \leftarrow 0;$
6. ***while*** $(RandomNumber > PartialSum)$

   $a.Selected++;$

   $b.PartialSum += fitness(selected);$
7. ***return*** $selected;$

$BestPath(PathList)$

1. $max \leftarrow 0; bestpath \leftarrow 0;$

2. $for\ i \leftarrow 0\ to\ |PathList|$

   a. $if\ \left(Score\left(PathList(i)\right)\right) > maxScore$ // *Score* ($Path\ P$) is the sum of the rewards associated with each node of

   $Path\ P$.

   $b. maxScore \leftarrow Score\left(PathList(i)\right); bestpath \leftarrow i;$

3. $return\ PathList(bestpath);$

The main aim of RWS_OP algorithm is to connect the source and the destination vertex and in between visit as many nodes as possible to collect the highest possible score within the given time bound. In lines 1-5 of $RWS\_OP()$, Dijkstra's algorithm is applied to find the shortest path connecting the source and target, and the path obtained is stored in $Path$. A queue of $Paths$, i.e. $PathList$ is maintained that contains the list of $Paths$ obtained after each iteration and is initialized with the shortest path P.

Lines 1-5 of $Generation()$ denotes that in each iteration (i.e. each run of), $Generation$ function inserts a node into the $Path$ obtained by the previous iteration. So, an iteration of $Generation$ function accepts a list of $Paths$ in the form of $PathList$ and for each $Path$ in the $PathList$, it calls the $Selection$ function. Another queue of $Paths$, i.e. $ChildPathList$ is maintained, which is initially empty. It stores the child paths generated using the $Selection$ function for each $Path$ of $PathList$. After each iteration, we consider only $PathListSize$ number of $Paths$ at a time. So $PathListSize$ number of $BestPaths$($BestPath$ is the one which has the highest value for total collected score) from the $ChildPathList$ are inserted into another queue of $Paths$, i.e. $NewPathList$. Then the termination condition is checked to determine whether a new child is generated by the $Generation$ function or not. If no new child is generated (i.e. all paths in the queue, $NewPathList$ is the same as that of $PathList$), then the $Generation$ function will return the $BestPath$ from the $PathList$ else, the $Generation$ function is recursively called for $NewPathList$.

The $Selection$ function is used for generating $ChildPaths$ from $Path\ P$. This function generates a $ChildPath$ which contains one node more than that already present in $Path$ P. This function decides the node to be inserted and its location in $Path\ P$. As shown in line 3 of $Selection()$, for each vertex that has not yet been inserted in $Path\ P$, its best position (one that leads to smallest increment in time, Δt_i) is evaluated and then the ratio q_i is computed. The ratio q_{max} is calculated in the following way:

$$(q_{max}) = max_{i \in (V \setminus P)}(q_i)$$

In lines 4-7 of $Selection()$, another list of nodes, named $CandidateList$ is created which is initially empty. All those nodes which satisfy the following inequality are inserted in the $CandidateList$:

$$CandidateList = \{i \in (V \setminus P) : (q_i \geq \alpha\ q_{\max}) \ \&\& \ (t_{parent} + \Delta t_i \leq T_{max})\}$$

Here α is the greediness parameter. Greater value of α denotes a more greedy solution and smaller value of α denotes a more random solution. As the value of α decreases, more nodes will be selected for the $CandidateList$. As stated in lines 8-10 of $Selection()$, if the $CandidateList$ is empty, $Path\ P$ is inserted in the queue $ChildPathList$, else $RouleteWheel$ selection method is used to choose a node from the $CandidateList$. The chosen node is removed from the $CandidateList$ and inserted at its best position in $Path\ P$ to generate a $ChildPath$ which is then en-queued into $ChildPathList$. This process of extracting nodes from the $CandidateList$ is repeated until a constant ($PathListSize$) number of $ChildPaths$ are generated or $CandidateList$ becomes empty.

Roulette wheel selection is a population-based selection method used in genetic algorithms that stochastically picks out a node based on their fitness value i.e. a node having greater fitness value has more chances of getting selected, although nodes with lower fitness also have a nonzero probability of selection. This assists the search in escaping local maxima. It is conceptually equal to giving each individual option, a portion of a circular roulette wheel proportional in area to the individual's fitness value (Zhang et al., 2012). As it can be seen, lines 1-7 of the $RouleteWheel\ ()$ selects a node from the $CandidateList$ using $FitnessProportionateSelection$ approach where an element is randomly chosen, but the probability of choosing an element with higher fitness is greater than choosing an element with lower fitness value. In our selection process, the fitness of an element of $CandidateList$ is computed using the following equation:

$$fitness(i) \leftarrow q_i, \text{ where } q_i = \begin{cases} S_i/|\Delta t_i|, & \Delta t_i \geq 1 \\ S_i, & -1 \leq \Delta t_i < 1 \\ S_i * |\Delta t_i|, & \Delta t_i < -1 \end{cases} \quad (9)$$

Therefore, nodes having a higher value of S_i and lower value of time increment (Δt), have a greater probability of getting selected and also the nodes having low S_i and high (Δt) values can be selected but with a lesser probability (> 0).

The time complexity of $Dijkstra's(v_1, v_N)$ algorithm is $O(|V|^2)$ where $|V|$ is the number of vertices in the Graph. In the $RouleteWheel$ function, there exists an iterative loop that runs $|CandidateList|$ times and because the $CandidateList$ can contain at most $(|V|-2)$ nodes, therefore the time complexity of $RouleteWheel$ will be $O(|V|)$.

In the selection function, line 2 is an iterative loop running for all the vertices of G i.e. $|V|$times and in each iteration, the best position of node i in the current $Path\ P$ is found which takes $O(|V|)$ time. Line 6 represents another iterative loop, which takes $O(|V|)$ time as it also runs for all the vertices of G and $Insert$ operation in the $CandidateList$ is similar to simple insertion in an array and takes $O(1)$time. $Remove$ operation in $CandidateList$ will take $O(|CandidateList|)$ or $O(|V|)$ time. $Remove$ operation in $ChildPathList$ takes $O(|V|)$ time as $O(|V|)$ time is required to bring the element to the front of $ChildPathList$ queue and $O(1)$ to dequeue it. $Insert$operation in $Path\ P$ takes $O(|V|)$ time. The iteration of line 9 takes $O(PathListSize * |V|)$ time as it runs for $PathListSize$ times, and each iteration takes $O(|V|)$ time. So the overall time complexity of $Selection$ function is $\left(O(|V|^2) + O(|V|) + O(PathListSize * |V|)\right)$, which is equivalent to $O(|V|^2)$ as $PathListSize \leq |V|$.

For the $Generation$ function, time complexity of line 2 which is an iterative loop is $O(PathListSize * |V|^2)$ as it runs for all the paths in $PathList$ and there can be at most $PathListSize\ Paths$ in this queue. In each iteration, it calls the $Selection$ function that takes $O(|V|^2)$ time. The $BestPath$ function will take $O(PathListSize)$ time. Since it is a recursive function, after each recursion, the $Paths$ contained in $PathList$ will increase by 1. The function will terminate when no new node is available that can be inserted in the $Path$. So, the recurrence relation can be written as:

$$T(M) = T(M-1) + PathListSize * |V|^2$$

Here, $T(M)$ is the time complexity for the $Generation$ function where, M is the number of nodes that can be added to the $Paths$ contained in $PathList$ and after one iteration the function calls itself with $M-1$ number of nodes that can be added to the $Paths$ of $NewPathList$. Solving the above recurrence, we get the overall complexity as $O(|V|^3)$.

The main function, i.e. RWS_OP uses the Dijkstra's algorithm ($O(|V|^2)$ time), $Selection$ function ($O(|V|^2)$ time) and $Generation$ function ($O(|V|^3)$ time). Therefore, the overall time complexity of RWS_OP is $O(|V|^3)$. The memory consumption of RWS_OP is $|PathListSize|^2 * N$ as opposed to Ostrowski_CG and Ostrowski_IG methods that occupies $PopulationSize * N$ memory where $N = Number\ of\ nodes\ in\ the\ graph$.

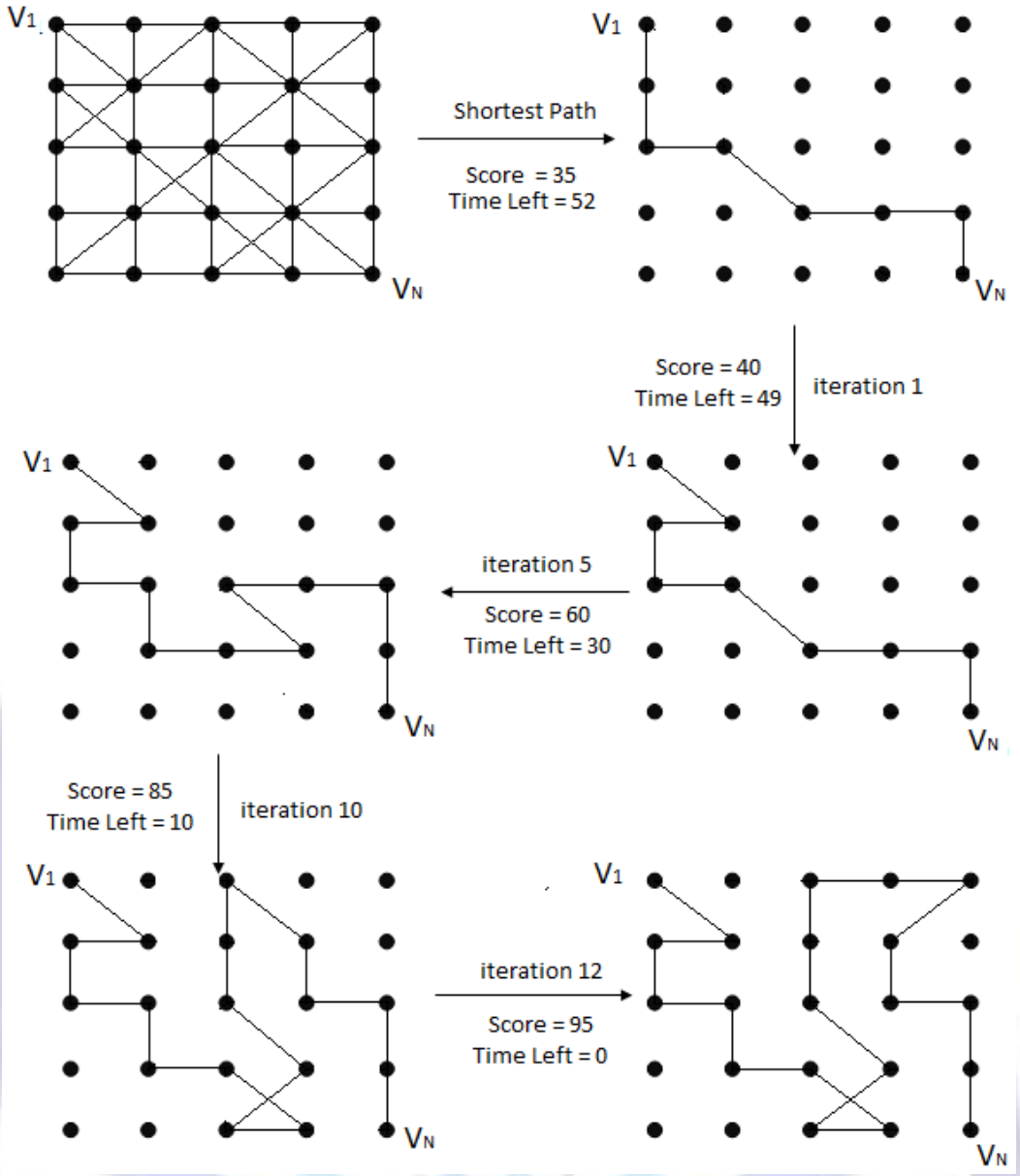


Fig 2: Progression of RWS_OP algorithm for a graph with 25 nodes with source $(V_1) = 1$, destination $(V_N) = 25$ and $T_{max} = 70$.

4 EXPERIMENTAL ANALYSIS

Most of the standard benchmark instances available for the orienteering problem include Set 1, Set 2 and Set 3 given by Tsiligirides, Set 64 and Set 66 given by Chao etc. (www.mech.kuleuven.be/en/cib/op/). In each of the available instances, only scores and coordinates of each vertex are specified leading to the formation of complete graphs satisfying the triangular inequality. However, in the real world, there is no guarantee that two nodes will be connected through a direct path. To deal with such cases, graphs are usually complemented with fictional edges by running Dijkstra's algorithm for every node before applying the classic OP algorithms available for complete graphs, but this results in a considerable increase in the search space size. Our method, works on complete as well as incomplete graphs without the need to insert fictional edges.

We have implemented our code in C++ using CodeBlocks on an Intel Core i5 650 at 2.20 GHz. As stated before, RWS_OP is capable of handling both complete as well as incomplete graphs, and here we present a few observations by applying RWS_OP on a real road network data based on 160 and 306 cities of Poland (http://piwonska.pl/p/research/). It consists of two text files -cities.txt and distances.txt. The file cities.txt specifies the names of the cities and the score of each, which is assigned based on the number of inhabitants in that city, i.e. $score = inhabitants/10000$. The distances.txt file was created from the real map of Poland representing the roads as edges in the graph stating for a particular city, its adjacent cities and their corresponding edge lengths.

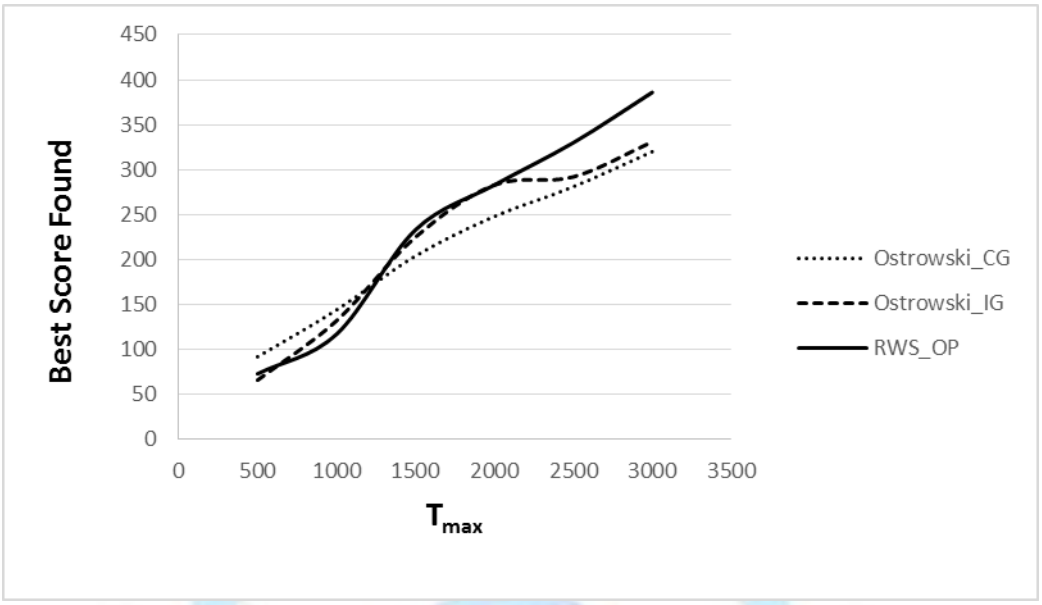
ISSN 2277-3061

5 DISCUSSION

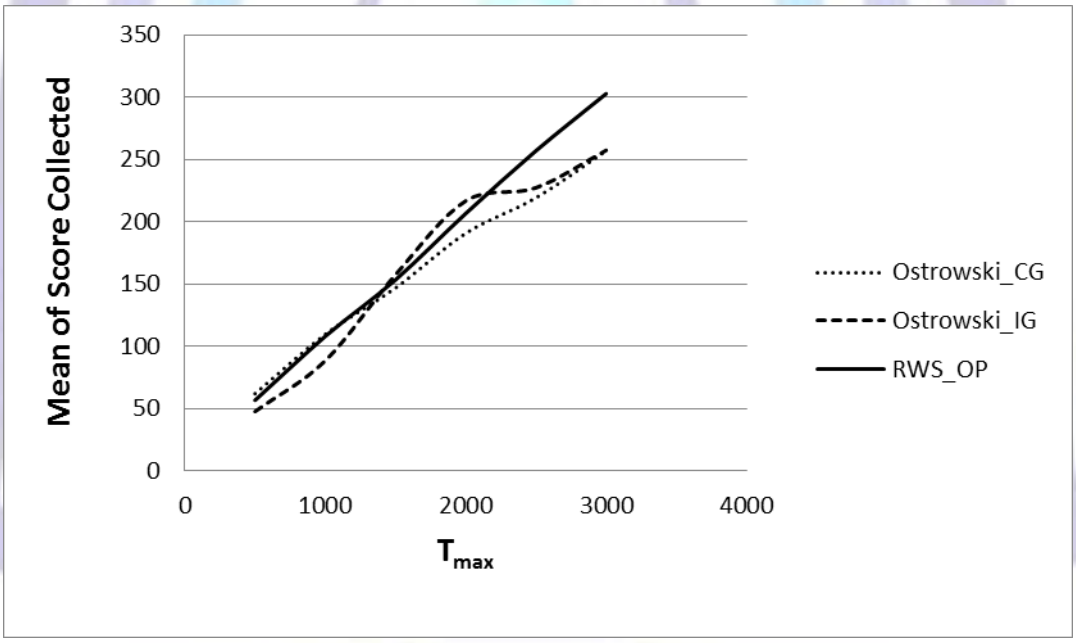
The following tables and plots show a comparison of our results with those obtained by applying the genetic algorithm suggested by Ostrowski et al on the same instances (Ostrowski and Koszelew, 2011). As can be seen in Table 1 and Fig. 3, it is observed that RWS_OP performs better for larger T_{max} values (taking the same first and last node i.e. source = destination) than the one proposed by Ostrowski et al because the highest total collected score attained for a graph with 306 nodes is greater in our case. Furthermore, the mean score and 95% confidence interval (CI) of mean for 30 runs $\left(i.e\ CI = \frac{1.96 * standard\ deviation\ of\ the\ number\ of\ runs}{\sqrt{number\ of\ runs}}\right)$ are higher in our case when compared to the values obtained by the genetic algorithm. Fig. 4 shows that for the same T_{max} value, there is a significant decrease in the execution time of RWS_OP as compared to the CG (Complete Graph version) and IG (Incomplete Graph version) of the genetic algorithm proposed by Ostrowski et al. It is also seen that the execution time increases linearly with the increase in T_{max} because increase in T_{max} leads to the exploration of more number of nodes. Fig. 5 presents the effect on scores by varying the value of the greediness parameter (α), which balances the degree of randomness and greediness. Increasing the value of the greediness parameter (α) makes the algorithm more greedy and the value of score obtained as a result of several executions of our code is the same most of the times i.e. lesser variation in the score, whereas decreasing the value of (α) leads to a situation with greater randomness, i.e. more variation in the scores obtained as shown through smaller and larger boxes respectively. It is also observed that the maximum score achieved increases with the increase in (α) however, for a large number of executions, a greater maximum score can be obtained even for smaller values of (α). RWS_OP is also efficient in terms of the time utilization (as shown in Table 2) as almost 99% of the specified time budget (T_{max}) is utilized, which helps in determining a better path that covers almost 70% of the cities, thus leading to greater total collected score. As the algorithm progresses, one node is added to the final path after each iteration, which results in an increase in the total collected score and decrease in the given time budget as shown in Fig. 6. Most of the experiments are performed at $\alpha = 0.6$ because at 0.6, both randomness and greediness come into play and as stated before, decreasing the value of α makes the algorithm more random and increasing it makes RWS_OP greedier. RWS_OP uses roulette wheel selection for choosing a node to be added in the path so different runs of the same algorithm with the same input parameters may result in selection of different nodes for being added in the final path and the trend in the utilization of time budget and an increase in the total collected score for three different runs at $\alpha = 0.2$ and $T_{max} = 1500$ for 160 cities instance is shown in Fig. 7. Fig. 8 shows how the time budget is utilized, and the total collected score increases with each iteration of RWS_OP for different values of α at $T_{max} = 1500$. The proposed heuristic is capable of exploring almost 70% of the search space as shown in Fig. 9 (a) and 9 (b) is a box plot showing the percentage of nodes explored and unexplored at $T_{max} = 7000$ for different values of α. As can be observed in 9 (b), the percentage of nodes explored (NE) is higher than the percentage of nodes unexplored (NU). Furthermore, the percentage of score collected for the explored nodes is higher than the percentage of score left out (score that could not be collected) for lower values of α, as a lower value of α induces more randomness into RWS_OP. RWS_OP is efficient both in terms of time and space complexity when compared to Ostrowski_CG and Ostrowski_IG methods, therefore, RWS_OP can be implemented for larger values of T_{max} which helps in achieving a higher total collected score for the considered instances as shown in Table 3 and Fig. 10.

Table 1. Comparison of maximum, mean and confidence Interval (CI) for mean of scores obtained by RWS_OP (keeping $v_1 = v_N$ i.e. $v_1 = v_N = 1$) with those obtained by executing the Ostrowski'salgorithm (Please refer Ostrowski and Koszelew, 2011, their Table 5 for Ostrowski_CG and Table 7 for Ostrowski_IG) on Real Road Network database with 306 cities of Poland.

T_{max}	RWS_OP(α=0.6)			Ostrowski_CG (highest fitness/travelTime)			Ostrowski_IG (fitness-Gain^2/travelTimeIncrease)		
	Mean	CI for Mean	Maximum	Mean	CI for Mean	Maximum	Mean	CI for Mean	Maximum
500	56.56	±3.5	73	61.9	±3.5	92	47.5	±3.3	66
1000	107.9	±2.64	117	109.5	±5.4	144	88.3	±6.0	132
1500	153.3	±9.29	233	146.7	±8.8	204	157.4	±13.5	225
2000	206.5	±13.1	283	190.6	±9.7	248	216.6	±12.8	283
2500	256.9	±13.07	330	219.1	±10.7	281	227.2	±13.9	292
3000	302.7	±13.4	386	256.9	±11.4	320	257.3	±15.2	331



(a)



(b)

Fig 3: Comparison of (a) maximum score and (b) mean score of each method with respect to time budget (T_{max}) based on 30 runs at $\alpha = 0.6$ for Real Road Network database with 306 cities of Poland.

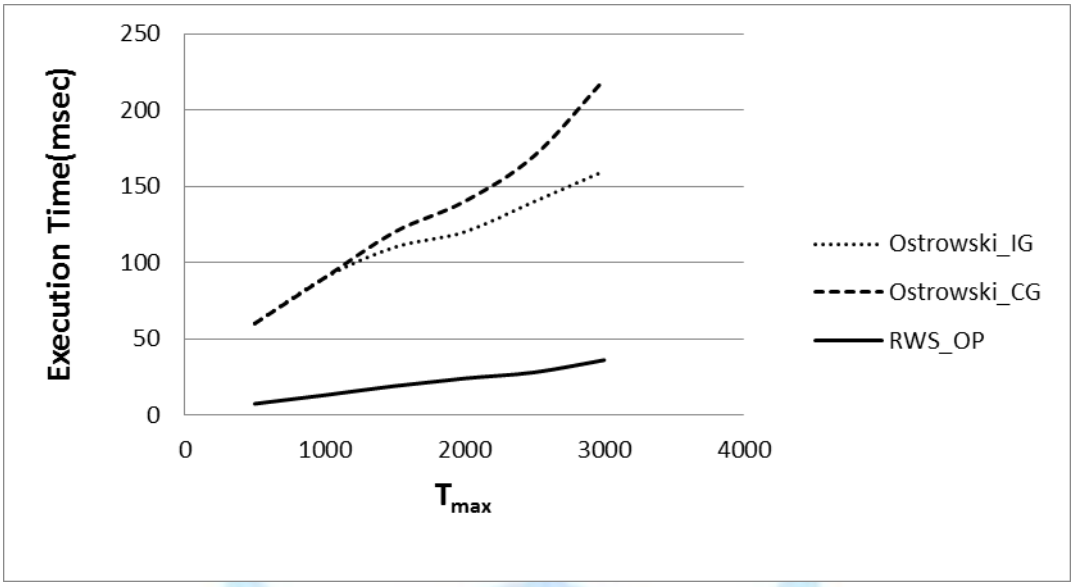
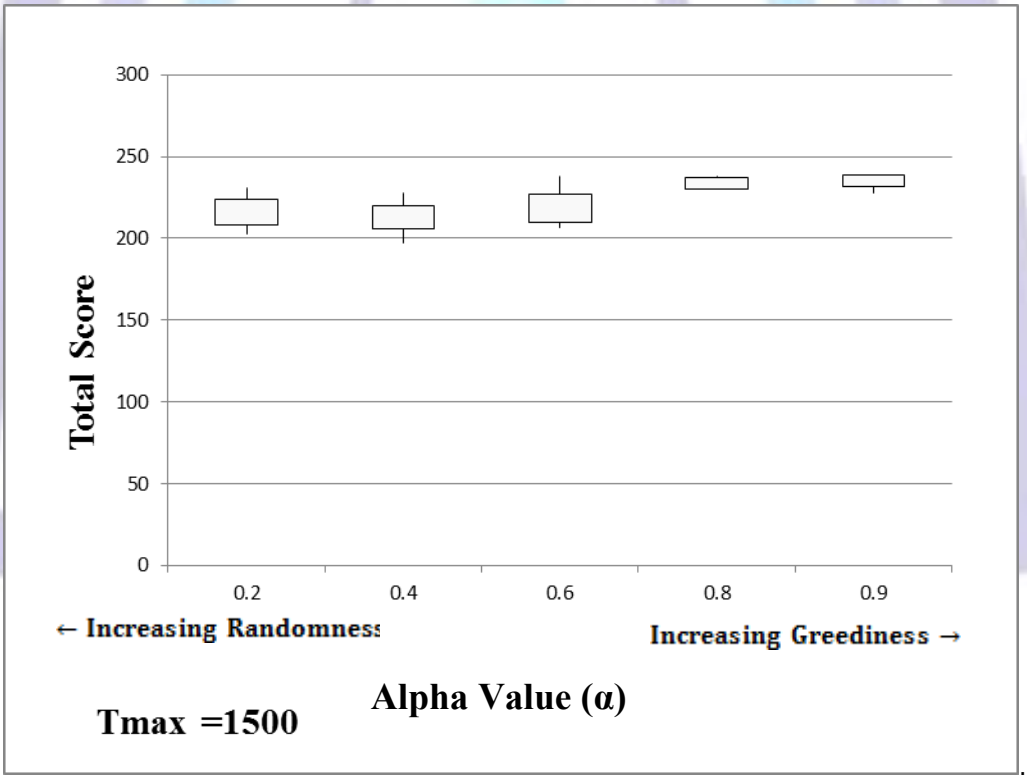
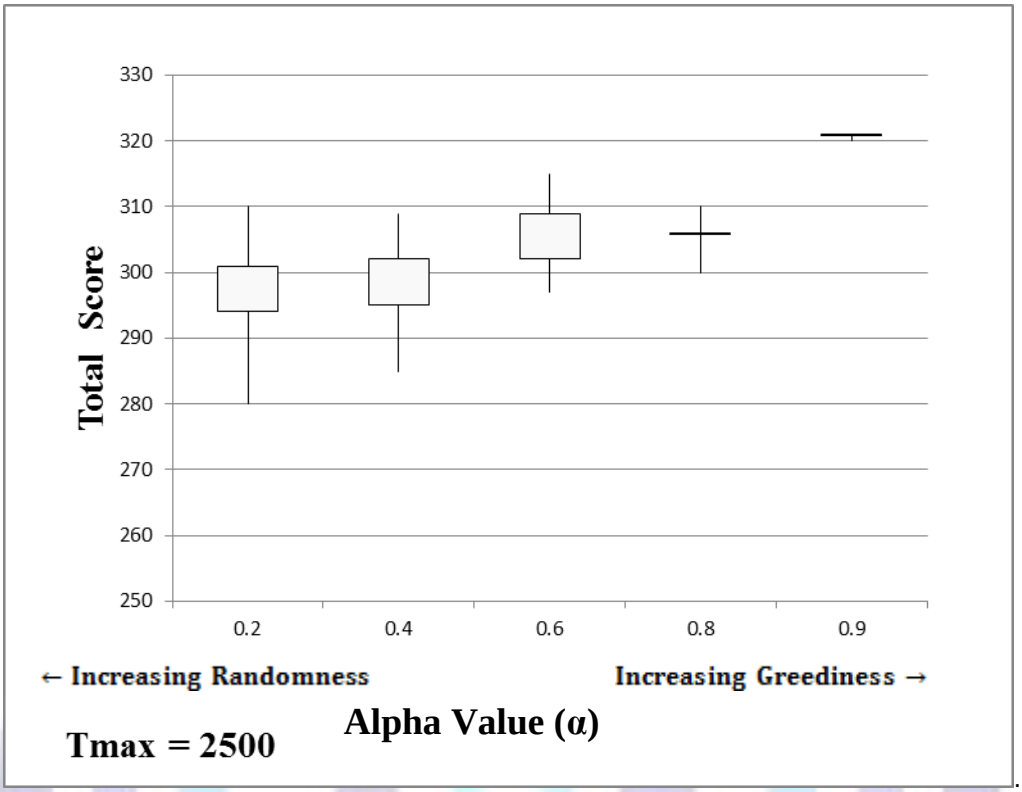


Fig 4: Comparison of execution time of each method with respect to time budget (T_{max}) based on 30 runs at $\alpha = 0.6$ for Real Road Network database with 306 cities of Poland.



(a)



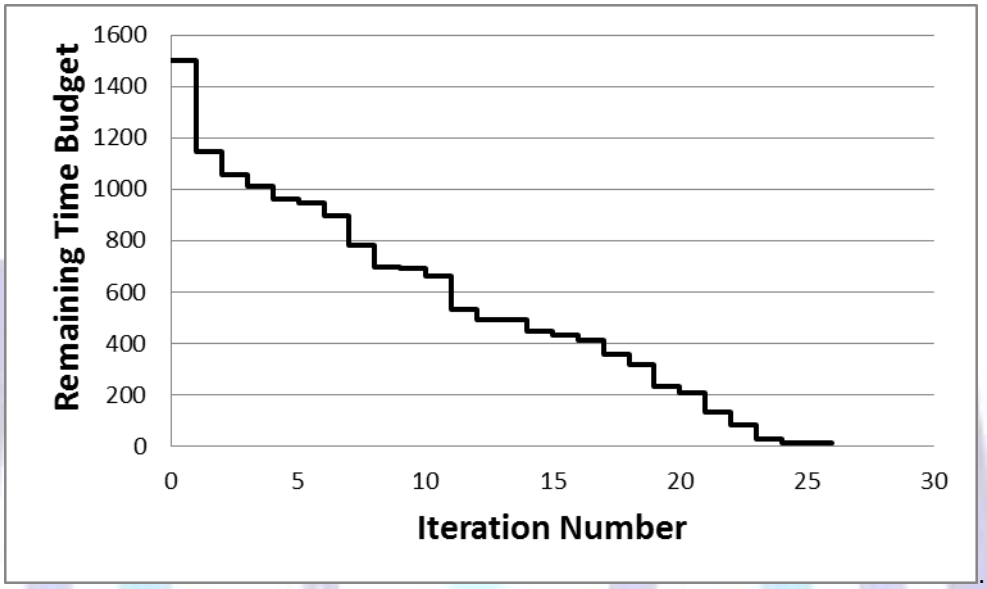
(b)

Fig 5: Comparison of score with respect to α for (a) $T_{max} = 1500$ and (b) $T_{max} = 2500$ for a Real Road Network database with 306 cities of Poland.

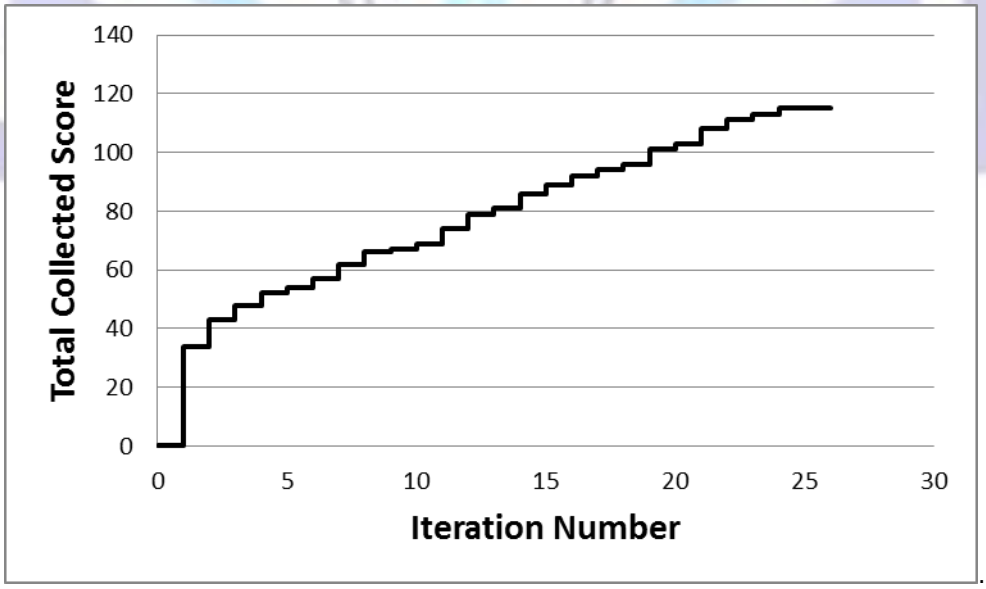
Table 2. The Highest Score Collected, Mean of Score Collected, Mean Time to Traverse the Path and % of Time Budget Utilized values obtained by RWS_OP at $\alpha = 0.6$ (keeping $v_1 \neq v_N$ i.e. $v_1 = 1$ and $v_N = 306$ for 306 cities and $v_1 = 1$ and $v_N = 160$ for 160 cities) when implemented on a Real Road Network database with 306 cities and 160 cities of Poland.

T_{max}	306 cities (α=0.6)				160 cities (α=0.6)			
	Highest Score Collected	Mean of Score Collected	Mean Time to Traverse the Path	% of Time Budget Utilized	Highest Score Collected	Mean of Score Collected	Mean Time to Traverse the Path	% of Time Budget Utilized
500	122	120.1	495.1	99.02	49	48.83	496.67	99.34
750	154	150.3	743.4	99.12	67	65.1	747.07	99.6
1000	177	174.3	995.1	99.51	88	76.8	990.8	99.08
1250	210	195.1	1246.53	99.72	104	102.3	1230.3	98.42
1500	243	220.4	1495.33	99.69	117	115	1488.37	99.25
1750	261	243.6	1742.72	99.58	129	127.9	1739.1	99.38
2000	286	270	1993.4	99.67	145	142.6	1993.63	99.68
2250	310	291	2244.8	99.77	162	156.1	2242.4	99.66
2500	324	312.4	2493.76	99.75	185	176.4	2485.8	99.43
2750	345	332.5	2744.8	99.81	202	193.2	2736.4	99.5

3000	371	349.6	2993.1	99.77	214	205.1	2980.3	99.34
3500	411	392.1	3495.47	99.87	250	243.6	3488.9	99.68
4000	451	436.9	3995	99.88	271	265.1	3982.7	99.57
4500	502	487.1	4494.3	99.87	306	290.23	4479.4	99.54
5000	537	524.5	4995.2	99.90	314	307.53	4960.9	99.22
5500	574	561	5491.7	99.85	322	316.9	5319.9	96.73
6000	620	593.3	5990.7	99.85	322	314.7	5336	88.93

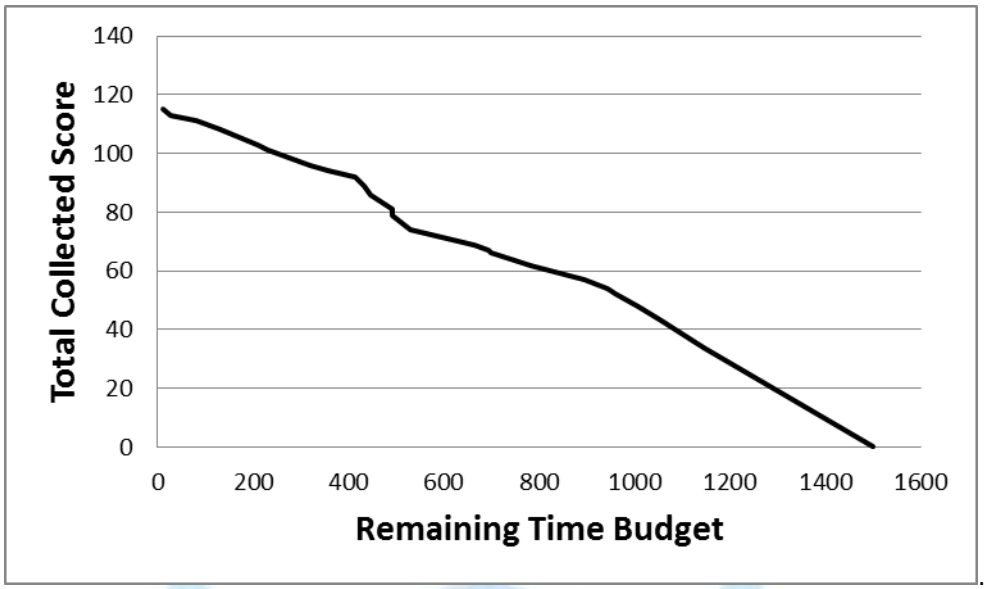


(a)



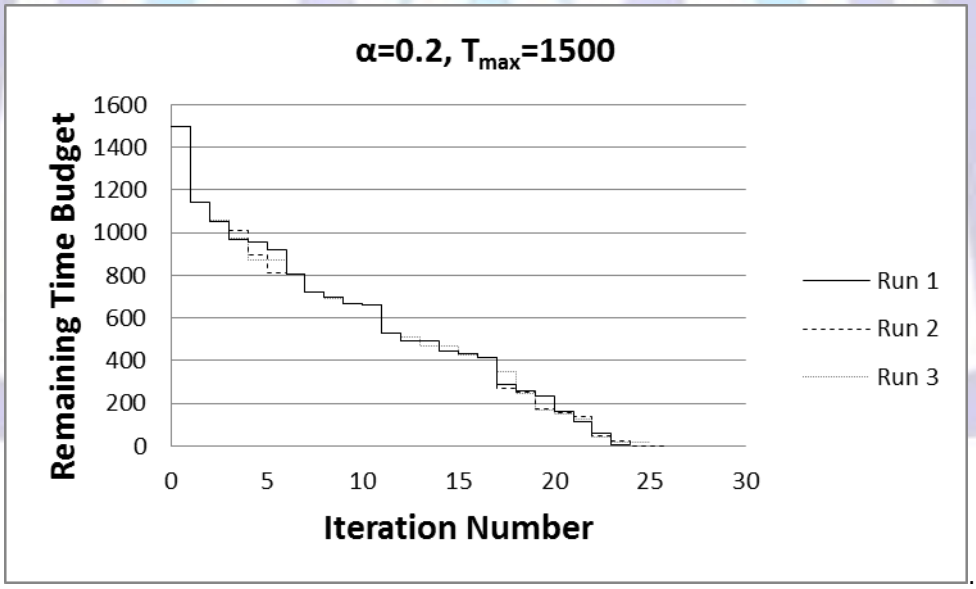
(b)

ISSN 2277-3061



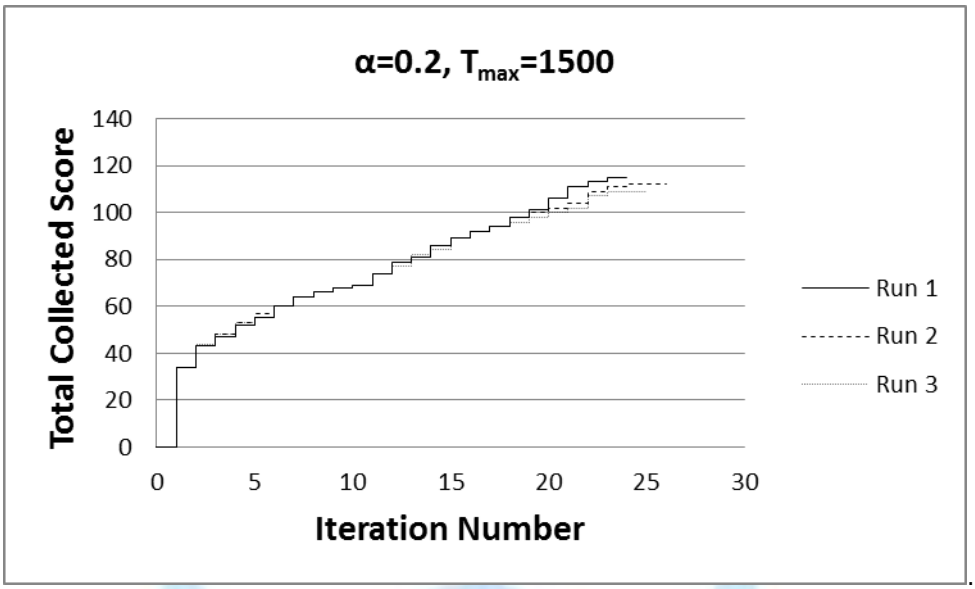
(c)

Fig 6: Plots showing (a) utilization of the time budget and (b) increase in the total collected score at $\alpha = 0.6$ and $T_{max} = 1500$ for a Real Road Network database with 160 cities of Poland. As the RWS_OP algorithm progresses, with each iteration, a node is added to the final path which results in an increase in the total collected score and decrease in the time budget as shown in (c).



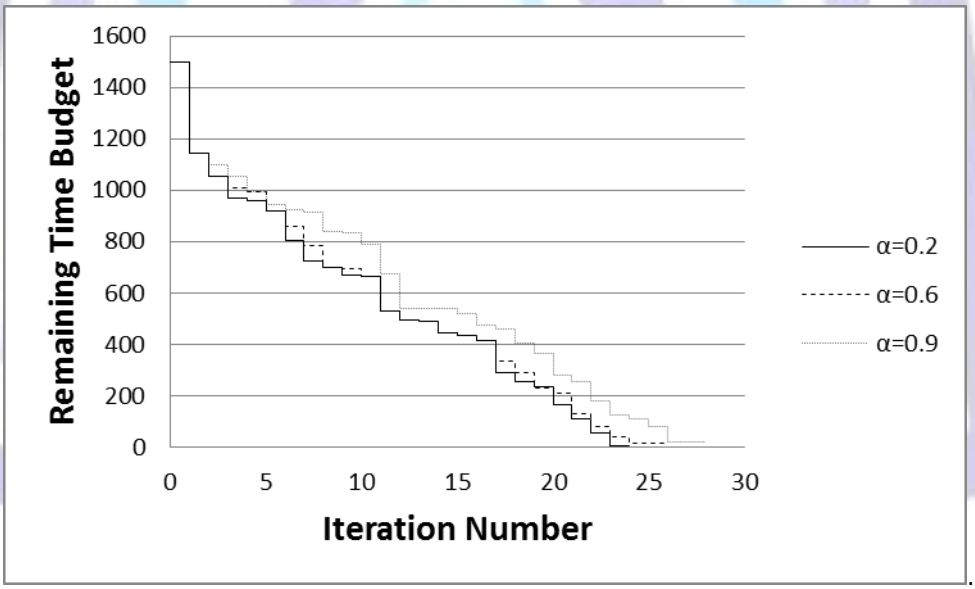
(a)

ISSN 2277-3061



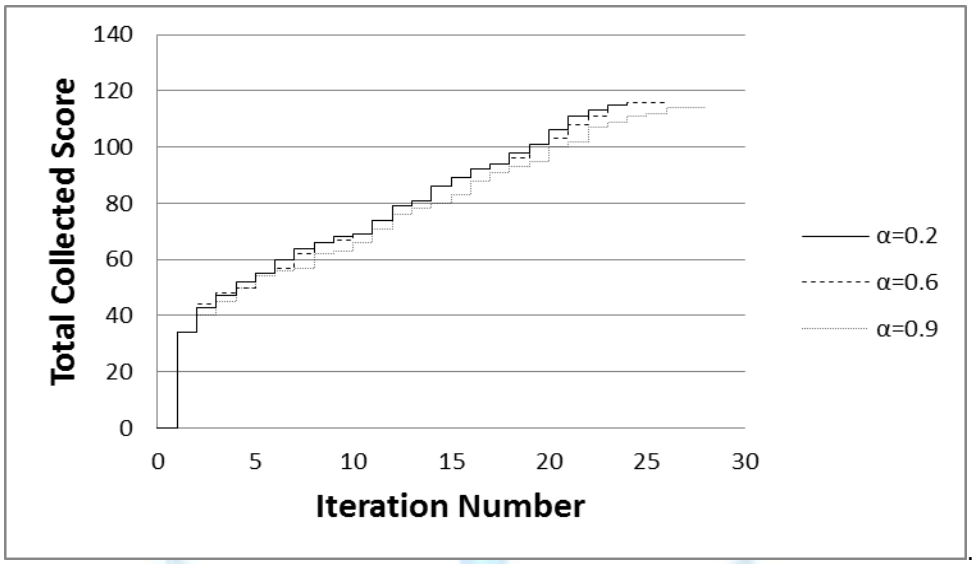
(b)

Fig 7: Plots showing the observation of three different runs of RWS_OP with $\alpha = 0.2$ and $T_{max} = 1500$ for a Real Road Network database with 160 cities of Poland. As the algorithm progresses, it results in (a) decrease in the time budget and (b) increase in the total collected score as shown above.



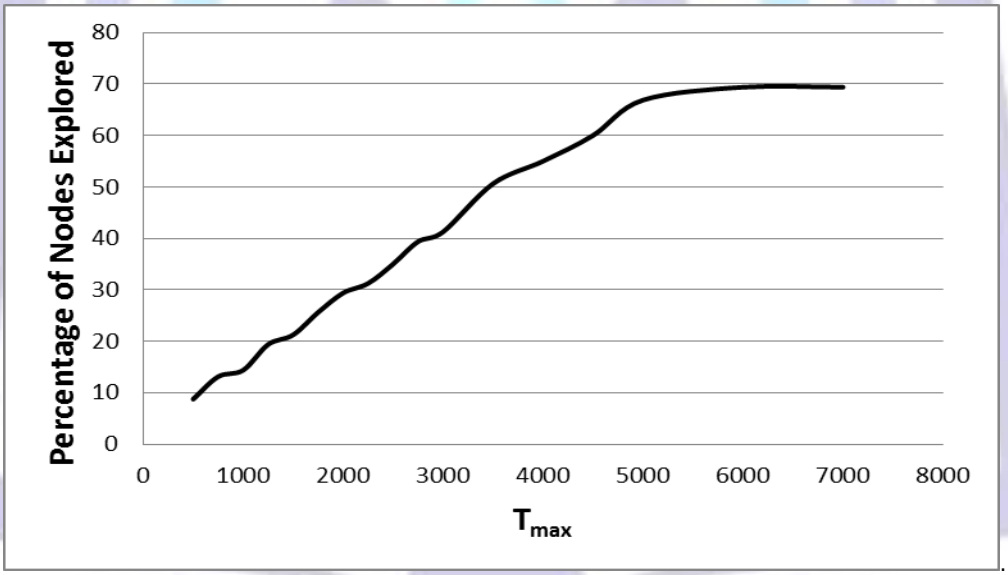
(a)

ISSN 2277-3061

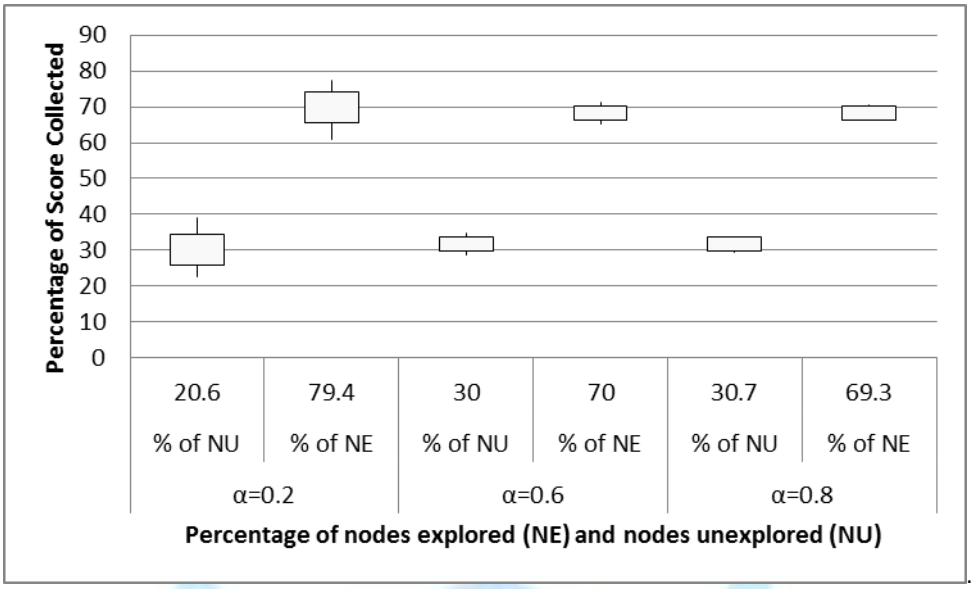


(b)

Fig 8: Plots showing (a) utilization of the time budget and (b) increase in the total collected score for three different α values at $T_{max} = 1500$ for a Real Road Network database with 160 cities of Poland.



(a)



(b)

Fig 9: Plots showing (a) the percentage of nodes explored with the increase in T_{max} values at $\alpha = 0.6$ and (b) percentage of nodes explored and unexplored for different values of α at $T_{max} = 7000$ for a Real Road Network database with 160 cities of Poland for 30 runs.

Table 3. The Highest Score Collected, Mean of Score Collected and confidence interval (CI) for Mean of Score Collected obtained by RWS_OP when implemented on a Real Road Network database with 306 cities of Poland for different T_{max} values at $\alpha = 0.6$ (keeping $v_1 = v_N$ i.e. $v_1 = v_N = 1$).

T_{max}	RWS_OP(α=0.6)		
	Mean of Score Collected	CI for Mean of Score Collected	Highest Score Collected
500	56.56	±3.5	73
1000	107.9	±2.64	117
s1500	153.3	±9.29	233
2000	206.5	±13.1	283
2500	256.9	±13.07	330
3000	302.7	±13.4	386
3500	353.16	±14.9	430
4000	427.8	±12.54	460
4500	466.7	±13.5	508
5000	506.1	±12.2	548
5500	553.2	±11.12	593
6000	595.2	±6.4	645
7000	653.1	±6.64	686
8000	718.2	±4.23	743
9000	767.8	±4.75	784
10000	769	±8.04	792
11000	769	±7.33	793

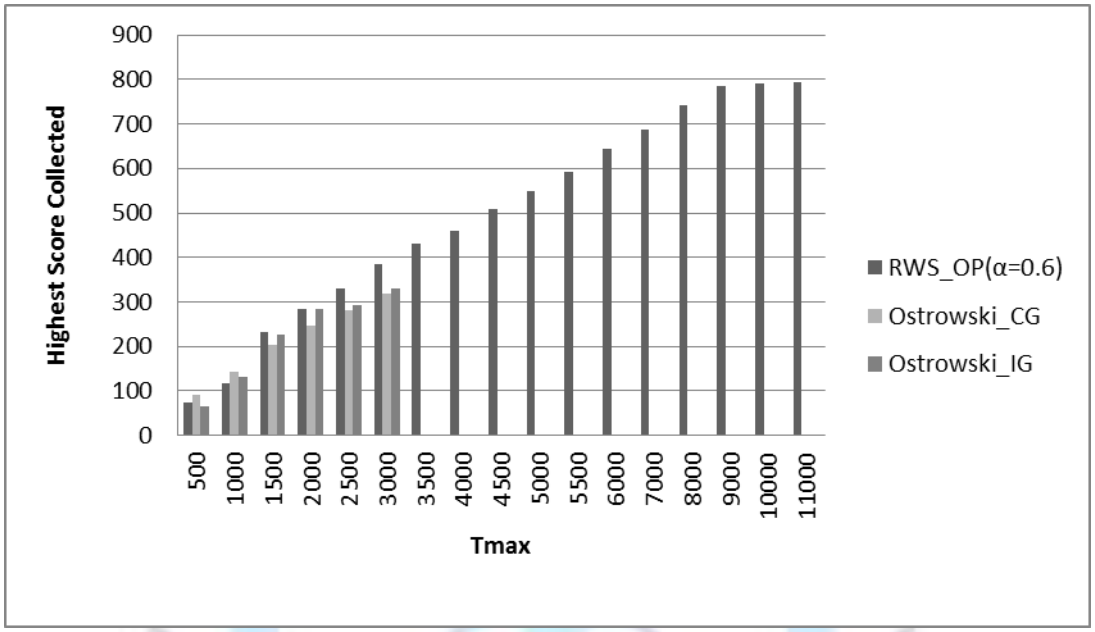


Fig 10: Plot showing that RWS_OP can achieve higher total collected score for larger T_{max} values as compared to Ostrowski_CG and Ostrowski_IG methods when implemented on a Real Road Network database with 306 cities of Poland at $\alpha = 0.6$.

6 CONCLUSION

In this paper, we considered the orienteering problem which is a NP-Hard combinatorial optimization problem and suggested a heuristic that uses the roulette wheel selection process for obtaining a Hamiltonian path that satisfies the time bound and helps in maximizing the total collected score. RWS_OP differs from the other techniques available in the literature as it can be applied on both complete as well as incomplete graphs whereas most of the existing algorithms can only be applied on complete graphs. Through experimental analysis, we showed that RWS_OP is more efficient than the previously suggested method by Ostrowski et al for incomplete graphs in terms of execution time. For a particular time bound, the proposed heuristic (RWS_OP) achieves a higher total collected score than the genetic algorithm of Ostrowski et al, utilizes almost 99% of the given time budget and is capable of exploring 70% of the considered search space. It is expected that when our algorithm is augmented with path relinking meta-heuristic (Glover and Laguna, 2000) its effectiveness will be further enhanced. In the future, we plan to integrate elite sub-paths found at earlier stages of the algorithm to produce new near optimal solutions. It will also be interesting to study the effect of incorporating adaptive or incremental beam search with the heuristic used in this paper.

REFERENCES

[1] Awerbuch, B., Azar, Y., Blum, A. and Vempala, S. 1999. Improved approximation guarantees for minimum-weight k-trees and prize-collecting salesmen, *Siam J. Computing,* Vol. 28, pp. 254–262.

[2] Blum, A., Chawla, S., Karger, D. R., Lane, T., Meyerson, A. and Minkoff, M. 2003. Approximation Algorithms for Orienteering and Discounted-Reward TSP, *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS'03)*, pp. 1-10.

[3] Campos, V., Marti, R., Sanchez-Oro, J. and Duarte, A. 2013. GRASP with Path Relinking for the Orienteering Problem. http://www.uv.es/rmarti/paper/docs/routing7.pdf.

[4] Fischetti, M., Salazar, J. and Toth, P. 1998. Solving the orienteering problem through branch-and-cut, *INFORMS Journal on Computing*, Vol. 10, pp. 133–148.

[5] Fomin, F. V. and Lingas, A. 2002. Approximation algorithms for time-dependent orienteering, *Information Processing Letters*, Vol. 83, pp. 57–62.

[6] Gendreau, M., Laporte, G. and Semet, F. 1998. A tabu search heuristic for the undirected selective travelling salesman problem, *European Journal of Operational Research*, Vol. 106, pp. 539–545.

[7] Glover, F. and Laguna, M. 2000. Fundamentals of scatter search and path relinking, *Control Cybern*, Vol. 29 No. 3, pp. 653–684.

[8] Golden, B., Levy, L. and Vohra, R. 1987. The orienteering problem, *Naval Research Logistics*, Vol. 34, pp. 307–318.

[9] Hayes, M. and Norman, J.M. 1984. Dynamic Programming in Orienteering: Route Choice and the Siting of Controls, *Journal of the Operational Research Society*, Vol. 35 No. 9, pp. 791-796.

[10] Johnson, D., Minkoff, M. and Phillips. S. 2000. The prize collecting steiner tree problem: Theory and practice, *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 760–769.

[11] Laporte, G. and Martello, S. 1990. The Selective Traveling Salesman Problem, *Discrete Applied Mathematics*, Vol. 26, pp. 193-207.

[12] Ostrowski, K. and Koszelew, J. 2011. The Comparison of Genetic Algorithms which Solve Orienteering Problem using Complete and Incomplete Graph, *Informatyka*, Vol. 8, pp. 61-77.

[13] Ramesh, R. and Brown, K. 1991. An efficient four-phase heuristic for the generalized orienteering problem, *Computers and Operations Research*, Vol. 18, pp. 151–165.

[14] Schilde, M., Doerner, K. F., Hartl, R. F. and Kiechle, G. 2009. Metaheuristics for the bi-objective orienteering problem, *Swarm Intelligence*, Vol. 3, pp. 179-201.

[15] Tasgetiren, M. 2001. A genetic algorithm with an adaptive penalty function for the orienteering problem, *Journal of Economic and Social Research*, Vol. 4 No. 2, pp. 1–26.

[16] Tsiligirides, T. 1984. Heuristic methods applied to orienteering, *Journal of the Operational Research Society*, Vol. 35, pp. 797–809.

[17] Vansteenwegen, P., Souffriau, W. and Oudheusden, D. V. 2011. The orienteering problem: A survey, *European Journal of Operational Research*, Vol. 209, pp. 1–10.

[18] Wang, Q., X. Sun, B. Golden, and J. Jia. 1995. Using artificial neural networks to solve the orienteering problem. *Annals of Operations Research* 61:111–120.

[19] Zhang, L., H. Chang, and R. Xu. 2012. Equal-width Partitioning Roulette Wheel Selection in Genetic Algorithm. In: *Proceedings of the Conference on Technologies and Applications of Artificial Intelligence*, pp. 62-67.

Proceedings of the Second International Conference on
"Emerging Research in Computing, Information, Communication and Applications"
ERCICA 2014
ISBN: 9789351072638

ELSEVIER

Evaluation of Ranking Methods for the Constrained Fuzzy Shortest Path Problem

Madhushi Verma[1,*] and K. K. Shukla[2]

[1]*Research Scholar,* [2]*Professor*
Department of Computer Science and Engineering,
Indian Institute of Technology (BHU), Varanasi 221 005, India.
e-mail: madhushi.rs.cse@iitbhu.ac.in

Abstract. This paper deals with the constrained shortest path problem (CSPP) which gives the cheapest path whose end-to-end delay consumption is bounded by a given constant. It is an NP-Complete problem that finds application in various fields which include multimedia, scheduling, networking, wireless sensor networks etc. where managing the uncertainty in the network is an important task in order to provide the assurance of Quality of Service (QoS).The best way to tackle this uncertainty is the use of fuzzy numbers for representing the parameters. Here the algorithm proposed by Chen *et al.* is extended to deal with the fuzzy environment and generate a solution for the constrained fuzzy shortest path problem (CFSPP) by representing the parameter cost using trapezoidal fuzzy numbers and applying a ranking procedure to determine the ordering of the fuzzy numbers. Three popular techniques for ranking of fuzzy numbers were implemented and their performances were compared on power law random graphs generated using gengraph-win. Based on a large number of experiments on random graphs we show that Weighting Function method is best in terms of execution time and cost whereas the Circumcenter of Centroids method gives minimum path discretization error.

Keywords: Constrained shortest path problem, Constrained fuzzy shortest path problem, Ranking, Circumcenter of centroids, Maximizing set and minimizing set, Weighting function.

1. Introduction

The shortest path problem (SPP) is a well-known and well-studied combinatorial optimization problem, the solution for which can be generated in polynomial time. The goal in SPP is to find a path between two points i.e. the source and the target with the smallest possible length. The domain of the problem is a graph with a real number representing the weights assigned to each of the edges of the graph and the shortest path is a sequence of edges connecting the source and the target such that the sum of their weights is minimized [1,2]. Adding additional requirements in SPP, of fulfilling a set of constraints that can be minimum bandwidth requirement, inclusion/exclusion of nodes, end to end delay etc. makes it intractable [1]. One such variation of SPP is the Constrained Shortest Path Problem (CSPP) which is a well-known NP-Complete Problem and in CSPP the shortest path generated must obey a set of constraints [1]. CSPP finds many real life applications like traffic engineering, communication networks, multimedia applications like voice/video calls, web broadcasting, cable television networks, ATM circuit routing, computer networks etc. [2,3]. The obstacle in such problems is to tackle the uncertainty involved in the network of the stated applications and at the same time provide Quality of Service (QoS) assurance. In such a network environment, the uncertainty is due to the presence of parameters like delay, cost, time, energy etc. which are not naturally precise and can be appropriately modelled using Fuzzy numbers leading to an extension of CSPP i.e. Constrained Fuzzy Shortest Path Problem (CFSPP). Fuzzy version of SPP has been studied by a number of researchers but as far as we know very few papers report the constrained version of fuzzy CSP [4–6]. Joksh in [7] proposed one of the first algorithms for CSPP. As stated earlier CSPP is a NP-Complete problem and the methods suggested in the literature to tackle such problems is to use Heuristic or

*Corresponding author

Approximation algorithms. So most of the algorithms available for CSPP fall in either of the two stated categories and incorporate the strategies of Dynamic Programming (DP) or Lagrangian Relaxation (LR) [1,2,8–17].

In this paper we consider the problem of determining the delay constrained least cost shortest path. Two parameters are involved i.e. delay and cost. The exact values of these parameters are usually not available. Statistical methods can model the uncertainty due to noise with a given probability distribution function. The uncertainty due to lack of precision can be tackled using fuzzy memberships. We represent one of the parameters i.e. cost in the form of fuzzy numbers to model and tackle the non-statistical uncertainty involved in the network. Various types of fuzzy numbers exist which include Exponential fuzzy number, Gaussian fuzzy numbers, Triangular fuzzy numbers, Quasi-Gaussian fuzzy numbers, Quadratic fuzzy numbers, Trapezoidal fuzzy numbers (TFN) etc. We have used TFN for representing the parameter cost as fuzzy number for two reasons: one it has a linear membership function and out of all the types of fuzzy numbers, TFN is the most generic class so it can be widely applied in the scientific field and applied engineering problems for modelling the uncertainty. The second reason is that its concept is easy to understand and the simplicity of its computation makes it more popular and so TFN is more widely used in solving problems like CSPP [18].

One requirement in problems like CSPP is the ordering of fuzzy numbers when the parameters involved are represented in the form of fuzzy numbers. In such problems, the task is to determine the shortest path i.e. the one with the least cost fulfilling certain constraints which in our case is the delay requirement. The problem that arises in ranking the fuzzy numbers is the absence of the natural ordering that is present in the case of real numbers. A number of techniques have been suggested in the literature to rank or determine the ordering of fuzzy numbers but most of them use some strategy of converting the fuzzy number to real number and then it is analyzed. However, in this process, significant information is lost hence several researchers have attempted to design strategies that can rank the fuzzy numbers such that the loss of information is minimized. The idea of ranking fuzzy numbers was first proposed by Jain [19] and since then many methods and techniques have been developed for ranking. According to Wang and Kerre [20,21], all the strategies available for ranking fuzzy numbers can be divided into three categories based on fuzzy scoring, fuzzy mean and spread and preference relation. In this paper we have fuzzified the algorithm suggested by Chen *et al.* [1] for CSPP to tackle the uncertainty in the environment and also used three different methods for ranking the fuzzy numbers based on Circumcenter of Centroids, Weighting functions and Maximizing set and Minimizing set proposed by Rao and Shanker [22], A. Saeidifar [23] and Chou *et al.* [24] respectively with the aim to determine the method that can generate the best result.

2. Pre-requisites

2.1 *CFSPP problem*

Given a graph $G(V, E)$ with a fuzzy cost and a crisp delay associated with each edge, find the cheapest path that satisfies the given delay constraint.

2.2 *Trapezoidal fuzzy numbers (TFN)*

Unlike Boolean logic, fuzzy logic considers the set of values between true (1) and false (0) and a fuzzy number is represented by a membership function whose domain is this set of values between 0 and 1. So, all types of fuzzy numbers can be defined by a membership function which represent the real world in a more realistic manner and to every possible value of the universal set, a value between 0 and 1 is assigned depending on the membership function which is called its 'grade of membership'. The membership function of a trapezoidal fuzzy number represented as $A = (a_1, a_2, a_3, a_4)$ is as follows [18]:

$$\mu_A(x) = \begin{cases} 0, & x < a_1 \\ \frac{x-a_1}{a_2-a_1}, & a_1 < x \leq a_2 \\ 1, & a_2 < x < a_3 \\ \frac{a_4-x}{a_4-a_3}, & a_3 \leq x < a_4 \\ 0, & x > a_4 \end{cases} \tag{1}$$

where $a_1 \leq a_2 \leq a_3 \leq a_4$.

2.3 *Addition of TFN*

In problems where the goal is to determine the shortest path we need to find out the sum of the weights assigned to the edges and when the weights are fuzzy numbers, the procedure is not the same as that for real numbers. As stated in Section 2.2, each TFN is represented by a set of four values and if we have two such TFN represented as $A = (a_1, a_2, a_3, a_4)$ and $B = (b_1, b_2, b_3, b_4)$, then the formula for their addition is as follows [18]:

$$A + B = (a_1, a_2, a_3, a_4) + (b_1, b_2, b_3, b_4) = (a_1 + b_1, a_2 + b_2, a_3 + b_3, a_4 + b_4). \tag{2}$$

2.4 *Ranking of TFN*

As discussed earlier, one important property that exists in case of real numbers is that of natural ordering which is lacking in case of fuzzy numbers. Several methods have been suggested for ranking TFN. In the literature, these methods are usually compared against each other using small handpicked examples. Three recently introduced techniques of ranking TFN are discussed below and their behaviour is compared on a non-trivial constrained fuzzy shortest path problem in subsequent sections.

Definition 1: Circumcenter of centroids (COC)

Many of the methods proposed use the centroid of the trapezoid for ranking of TFN as it can be considered the balancing point but the technique suggested by Rao and Shanker [22] considers the COC to be a better reference point as it is determined by joining the centroids of the three sub parts of the trapezoid i.e. two triangle and one rectangle. We get another triangle on joining the three centroids and the circumcenter of this resultant triangle when used as a reference point for ranking of TFN, gives a more accurate result. For details see figure 2.

According to Rao and Shanker, the following formula can be used to determine the COC of any given TFN. If $A = (a_1, a_2, a_3, a_4)$, then

$$COC(A) = (x, y) = \left(\frac{a_1 + 2a_2 + 2a_3 + a_4}{6}, \frac{(2a_1 + a_2 - 3a_3)(2a_4 + a_3 - 3a_2) + 5}{12} \right) \tag{3}$$

This value of $COC(A)$ is then used to determine the rank of fuzzy number A using the following formula:

$$R(A) = \sqrt{x^2 + y^2} \tag{4}$$

For two given fuzzy numbers A and B, $R(A) > R(B)$ implies $A > B$ and $R(A) < R(B)$ implies $A < B$. In the third case, when $R(A) = R(B)$, we need some additional information to determine the rank of A and B and for that we use the following procedure:

Index of optimism

An index was defined by Rao and Shanker [22] that relates to the decision maker's view point by considering his degree of optimism. If we have a fuzzy number $A = (a_1, a_2, a_3, a_4)$ and $COC(A)$ then the index of optimism can be stated as:

$$I_\delta(A) = \delta\, y + (1 - \delta)x \quad \text{where } \delta \in [0, 1] \tag{5}$$

If the value of δ is large, it shows that the decision maker is more optimistic and vice versa.

Index of modality

Another index called index of modality was proposed by Rao and Shanker [22] for ranking of fuzzy numbers A and B which considers the importance of both the central value as well as the extreme values. The index of optimism was alone not sufficient enough for ranking as it considered only the extreme values of *COC*. The index of modality can be represented as:

$$I_{\delta,\gamma}(A) = \gamma\,((x + y)/2) + (1 - \gamma)I_\delta(A) \quad \text{where } \gamma \in [0, 1] \tag{6}$$

Following the index stated above we can tackle the situation where $R(A) = R(B)$ by calculating $I_{\delta,\gamma}(A)$ and $I_{\delta,\gamma}(B)$. If $I_{\delta,\gamma}(A) > I_{\delta,\gamma}(B)$ it implies $A > B$ and vice versa.

Definition 2: Maximizing set (MAS) and minimizing set (MIS)

A concept of total utility(combination of left and right utility values) using MAS and MIS was proposed by Chen [25] for ranking fuzzy numbers but some shortcomings were discovered in this method like the incapability of ranking those fuzzy numbers which had the same left, right or total utility values. To overcome these limitations, Chou *et al.* [24] suggested a revised ranking approach in which two left and right utilities were considered instead of just single left and right utility value while calculating the total utility. In addition, the revised approach also considers the degree of optimism of the decision maker and has the capability of differentiating between various types of fuzzy numbers.

In [24], for a TFN $= (a_1, a_2, a_3, a_4)$, f_A^L and f_A^R are used to represent the left and right membership functions, which according to equation 1 are as follows:

$$f_A^L(x) = \frac{x - a_1}{a_2 - a_1} \quad \text{and} \quad f_A^R(x) = \frac{a_4 - x}{a_4 - a_3} \tag{7}$$

If n trapezoidal fuzzy numbers are considered i.e. $A_i, i = 1, 2, \ldots, n$ with membership function μ_{A_i} then the $MAS(M)$ and $MIS(G)$ with membership functions f_M and f_G respectively are represented as follows:

$$f_M(x) = \begin{cases} [(x - x_{\min})/(x_{\max} - x_{\min})]^k, & x_{\min} \le x \le x_{\max} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

$$f_G(x) = \begin{cases} [(x_{\max} - x)/(x_{\max} - x_{\min})]^k, & x_{\min} \le x \le x_{\max} \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where $x_{\min} = \inf S$, $x_{\max} = \sup S$, $S = \cup_{i=1}^{n} S_i$, $S_i = \{x/\mu_{A_i}(x) > 0\}$ and $k = 1$.

As stated earlier, in this revised approach suggested by Chou *et al.*, pair wise computation is performed and for each A_i, two left utilities and two right utilities are calculated. The formulas for utilities are stated below:

Right utilities

$$U_{M_{i1}} = \sup_x (f_M(x) \wedge f_{A_i}^R(x)), i = 1, 2. \tag{10}$$

$$U_{G_{i2}} = \sup_x (f_G(x) \wedge f_{A_i}^R(x)), i = 1, 2. \tag{11}$$

Left utilities

$$U_{G_{i1}} = \sup_x (f_G(x) \wedge f_{A_i}^L(x)), i = 1, 2. \tag{12}$$

$$U_{M_{i2}} = \sup_x (f_M(x) \wedge f_{A_i}^L(x)), i = 1, 2. \tag{13}$$

The formula for total utility which also considers the decision maker's degree of optimism by using the parameter α is as follows:

$$U_T^{\alpha}(i) = \left\{ \alpha[U_{M_{i1}}(i) + 1 - U_{G_{i2}}(i)] + (1 - \alpha)[U_{M_{i2}}(i) + 1 - U_{G_{i1}}(i)] \right\} \Big/ 2 \quad i = 1, 2. \tag{14}$$

The functions f_M and f_G of $MAS(M)$ and $MIS(G)$ respectively intersects the left membership function f_A^L and the right membership function f_A^R of the fuzzy number A. The points of intersection for a TFN $A_i = (a_{i_1}, a_{i_2}, a_{i_3}, a_{i_4})$ can be denoted as M_{i1}, M_{i2} and G_{i1}, G_{i2} and the following equations can be used to determine their values:

$$x_{M_{i1}} = \frac{a_{i_4}\, x_{\max} - a_{i_3}\, x_{\min}}{(a_{i_4} - a_{i_3}) + (x_{\max} - x_{\min})} \tag{15}$$

$$U_{M_{i1}} = \frac{a_{i_4} - x_{M_i}}{(a_{i_4} - a_{i_3})} = \frac{a_{i_4} - x_{\min}}{(a_{i_4} - a_{i_3}) + (x_{\max} - x_{\min})} \tag{16}$$

$$x_{G_{i1}} = \frac{a_{i_2}\, x_{\max} - a_{i_1} x_{\min}}{(a_{i_2} - a_{i_1}) + (x_{\max} - x_{\min})} \tag{17}$$

$$U_{G_{i1}} = \frac{x_{G_i} - a_{i_1}}{(a_{i_2} - a_{i_1})} = \frac{x_{\max} - a_{i_1}}{(a_{i_2} - a_{i_1}) + (x_{\max} - x_{\min})} \tag{18}$$

So for each fuzzy number A_i, the total utility value is given by:

$$U_T^{\alpha}(i) = \frac{1}{2}\left(\alpha\left[\frac{a_{i_4} - x_{\min}}{(a_{i_4} - a_{i_3}) + (x_{\max} - x_{\min})} + \frac{a_{i_3} - x_{\min}}{(a_{i_3} - a_{i_4}) + (x_{\max} - x_{\min})}\right] + (1-\alpha)\left[\frac{a_{i_1} - x_{\min}}{(a_{i_1} - a_{i_2}) + (x_{\max} - x_{\min})} + \frac{a_{i_2} - x_{\min}}{(a_{i_2} - a_{i_1}) + (x_{\max} - x_{\min})}\right]\right), \quad i = 1, 2. \tag{19}$$

For two given fuzzy numbers A_1 and A_2, if $U_T^{\alpha}(A_1) > U_T^{\alpha}(A_2)$ then $A_1 > A_2$ and vice versa.

Definition 3: Weighting function (WF)

Another method of ranking fuzzy numbers was proposed by Saeidifar [23]. The interesting fact about this ranking technique is that it uses two things i.e. fuzzy numbers are defuzzified and a weighting function is applied. In this approach a weighted distance measure is defined on fuzzy numbers and then this distance is minimized to obtain the point approximations and the weighted interval of fuzzy numbers.

According to Saeidifar [23], a few definitions for a fuzzy number A with $[A]_\beta = [\underline{a}(\beta), \overline{a}(\beta)]$ and a weighting function $f(\beta) = (\underline{f}(\beta), \overline{f}(\beta))$ are as follows:

(i) Nearest $f = (\underline{f}, \overline{f})$ – weighted interval approximation (NWIA):

$$NWIA_f(A) = \left[\int_0^1 \underline{f}(\beta)\underline{a}(\beta)\, d\beta, \int_0^1 \overline{f}(\beta)\overline{a}(\beta)\ d\beta\right] \tag{20}$$

(ii) f-weighted mean:

$$\overline{L_f}(A) = \frac{\int_0^1 \underline{f}(\beta)\underline{a}(\beta) + \overline{f}(\beta)\overline{a}(\beta)}{2}\, d\beta \tag{21}$$

Therefore, for the fuzzy number, the weighting mean $\overline{L_f}(A)$ is denoted as:

$$\overline{L_f}(A) = \frac{1}{2}\int_0^1 \underline{f}(\beta)\underline{a}(\beta)d\beta + \overline{f}(\beta)\overline{a}(\beta)d\beta = \frac{\int_0^1 \underline{f}(\beta)\underline{a}(\beta)d\beta + \int_0^1 \overline{f}(\beta)\overline{a}(\beta)d\beta}{\int_0^1 \underline{f}(\beta)d(\beta) + \int_0^1 \overline{f}(\beta)d(\beta)} \tag{22}$$

If we have two fuzzy numbers A and B and a WF (f) then the ranking can be determined in the following way:

$$A \prec B \text{ if and only if } \overline{L_f}(A) < \overline{L_f}(B),$$
$$A \sim B \text{ if and only if } \overline{L_f}(A) = \overline{L_f}(B),$$
$$A \succ B \text{ if and only if } \overline{L_f}(A) > \overline{L_f}(B)$$

2.5 *Path delay discretization*

CSPP is a NP-Complete problem and techniques have been proposed to reduce the problem into a polynomial time solvable one by using the method of discretization. However, the effectiveness of such techniques is dependent on the amount of error induced due to discretization. Chen *et al.* [1] suggested two techniques for discretization which include path delay discretization and randomized discretization considering two important factors i.e. providing more efficient network functions and optimum usage of limited resources.

In this paper we prefer the path delay discretization method for solving CFSPP because the problem of error accumulation that occurs in case of randomized discretization is absent here as the interval partitioning method is used to perform discretization on path delays [1].

For any path, with delay $d(P)$, delay requirement r and an integer λ that bounds the delay requirement, the discretized delay can be determined using the following equation:

$$d'(P) = \left[\frac{d(P)}{r}\lambda\right] \tag{23}$$

where $[X] = \text{floor}(X)$ is the largest integer not greater than X.

3. Problem Definition

A network can be represented by an ordered pair $G(V, E)$ where V denotes a set of nodes and E denotes a set of edges such that $|V| = n$ and $|E| = m$. Each edge is denoted by a pair of nodes as $(u, v) \in E$. The notation used to represent the delay consumption and cost values of an edge is $d(u, v)$ and $c(u, v)$ respectively and the same for path is denoted as $d(P)$ and $c(P)$ respectively such that

$$d(P) = \sum_{(u,v)\in P} d(u, v) \tag{24}$$

$$c(P) = \sum_{(u,v)\in P} c(u, v) \tag{25}$$

L represents the longest path present in the network and $l(P)$, the length (number of hops) of P.

Any path P is called feasible if $d(P) \leq r$ and the cheapest feasible path is one that obeys the condition $d(P) \leq r$ in terms of delay and has the minimum cost $c(P_{s,t})$ among all available paths linking the source and the target nodes denoted by s and t respectively [1]. As in this paper we are dealing with the fuzzy version of the problem (CFSPP), the cost $c(u, v)$ is represented as TFN so to find out the value of $c(P)$, we use equation 2 of Section 2.3 and to decide the cheapest feasible least cost path, each of the three methods specified in Section 2.4 is used with the aim to determine the method that is more efficient and accurate. The fuzzy version of CSPP algorithm is given below:

CFSPP algorithm

```
Get_abcd( alpha, stretch, cost)
1. a = cost − stretch
2. b = a + alpha
3. d = cost + stretch
4. c = d − alpha
Initialize (V, s, λ)
5. for each vertex v ∈ V, each i ∈ [0, ..... , λ]
6. Get_abcd( alpha, stretch, cost)
7. w[v, i] := ∞, π[v, i] := NIL, z[v, i] := ∞
8.    w[s, 0] := 0, z[s, i] := 0
9.  end for
Relax_FPDA(u, v, i, λ)
10.  i' := floor((z[u,i] + d(u,v))/r λ)
11.  Get_abcd( alpha, stretch, cost)
12.  if i' ≤ λ and w[v, i'] > w[u, i] + c(u, v)
// Compare 1 using Definition 1 of Section 2.4 and Equation 3, 4, 5 and 6.
// Compare 2 using Definition 2 of Section 2.4 and Equation 19.
// Compare 3 using Definition 3 of Section 2.4 and Equation 22.
13.         w[v, i'] := w[u, i] + c(u, v)
14.         π[v, i'] := u
15.     z[v, i'] := min{z[v, i'], z[u, i] + d(u, v)}
16.  end if
FPDA_Dijkstra (G, s, λ)
17. Initialize (V, s, λ)
18.  for i = 0 to λ
19.       Q := V
20.       while Q ≠ φ
21.            u := Extract_Min(Q)
22.            if w[u, i] = ∞
23.              break out of the while loop
24.            end if
25.  Q := Q − { u }
26.  for every adjacent node v of u
27.      Relax_FPDA(u, v, i, λ)
28.  end for
29.         end while
30.  end for
FPDA(G, s)
31. λ := λ0
32.  do
33. λ := 2λ
34.         FPDA_Dijkstra (G, s, λ)
35. while ∃v ∈ V, d(P^v) > (1 + ε)r // where P^v is the path with min[w[v, i] | i ∈ [0 ..... λ]]
```

The stated algorithm is an extension of the Path Delay Discretization Algorithm (PDA), presented by Chen *et al.* [1]. The algorithm is capable of handling the fuzzy environment. Here we are dealing with two constraints i.e. cost and delay. Each edge of the graph has two values associated with it (cost and delay) and in the fuzzy version of the algorithm we consider one of the parameters i.e. cost as a fuzzy number and the other parameter which is delay is represented as a real number. Both the values are initially generated randomly using r and (). From these crisp values the trapezoidal fuzzy number for each cost value is generated using Get_abcd(). alpha and stretch are the two constant values supplied by the user for the formulation of TFN. $w[v, i]$, $\pi[v, i]$ and $z[v, i]$ are the three two dimensional arrays used which are all initialized by the function Initialize (V, s, λ). The cost of the cheapest path P connecting s and v with $d^r(P) = i$ is stored in the array $w[v, i]$, $\forall v \in V$ and $i \in [0, \ldots, \lambda]$. $z[v, i]$ is an array that keeps a track of the minimum delay of paths connecting s and v for which discretized delays are i. $\pi[v, i]$ is another array storing the last link of the path. *FPDA_Dijkstra*(G, s, λ) evaluates $w[v, i]$ and $\pi[v, i]$ for any target v and any given λ. This function determines the cheapest path among different delay paths, $d^r(P) \in [0, \ldots, \lambda]$. Let the cheapest path be denoted as P^v. The function *FPDA*(G, s) calls *FPDA_Dijkstra*(G, s, λ) iteratively with increasing value of λ (as can be observed in the lines 31–34 of CFSPP algorithm) till $d(P^v)$ is less than $(1+ \in)r$ for all $v \in V$.

For a detailed explanation of the algorithm, the readers are suggested to refer to [26]. Here it uses three different methods for ranking of fuzzy numbers with the aim to compare their performances. The experimental analysis along with the results are presented in the next section of this paper. The time complexity of the algorithm remains the same as specified in[1], i.e. $O((m + n \log n)L/\varepsilon)$ since the number of arithmetic operations in the fuzzy version increases only by a constant factor.

4. Experimental Analysis

We implemented CFSPP algorithm in C language and compiled using CodeBlocks for running on an i5 based system running at 3.20 GHz with 3 GB RAM. Three versions of the program were written with the three different ranking methods as per equations 3, 4, 5, 6, 19 and 22. These programs were verified by comparison with hand calculations on small graphs. Also, it was verified that for large graphs when the width of fuzzy numbers is set to zero the path obtained agrees with algorithm for the crisp case as given by [1].

Many real networks like the World Wide Web links, biological networks, and social networks are known to be *scale-free*, i.e. the fraction of nodes in the network having a given degree follows a power law. Therefore, we created test cases using power law random graph model. The test graphs generated using *gengraph-win*. The parameters used for the random graph generation were (*n, alpha, min, max, z*), where *n* is an integer denoting the count of degree number or the number of nodes, *min* is the minimum degree, *max* is the maximum degree and *alpha* is the exponent of the power law distribution which is a random number between 1-2.5. The command "*distrib n alpha min max z*" was used to generate a sample of *n* integers in the range specified by *min-max* from a heavy-tailed distribution of exponent *alpha* and average *z*. The experiments were repeated sufficient number of times with $n = 250$, $alpha = 2.5$, $min = 25$, $max = 75$ and $z = 45$ and conclusions are drawn based on the average behaviour. The graph generator *gengraph-win* gave un-weighted power law graphs with random structure. Random weights were assigned to the edges using uniformly distributed random numbers in the range 10 to 100 obtained using the *C rand*() function.

The results are shown in the following figures. In figure 1(a), the defuzzified cost of the shortest path found is plotted against the delay requirement. It is observed that as we relax the delay requirement, the cost of the shortest path found by the algorithm decreases and finally at very large values of delay requirement, the delay constraint becomes insignificant and the algorithm runs like the classical shortest path algorithm i.e. Dijkstra's algorithm giving the same result for both the algorithms. In terms of cost, MAS/MIS and WF methods provide better results than COC. In figure 1(b), the path discretization error that comes into existence due to discretization is shown for all the three methods and it is observed that in terms of path discretization error, COC shows better performance than the other two methods. This may be due to the fact that COC is equidistant from each vertex (i.e. centroids of the two triangles and central rectangle) as shown in figure 2. Figure 1(c) shows how the change in delay requirement, affects the CPU execution time of the three methods and from the graph it is visible that WF is better than the other two techniques in this aspect. We also notice that the CPU execution time generally increases when the delay requirement is relaxed (i.e. increased). This is because with relaxed delay constraint there is an increase in number of feasible paths thereby increasing the size of search space that has to be explored by the algorithm.

As the delay requirement is relaxed, the cost of cheapest path found decreases and for very high delay requirement value, the cost of shortest path reduces and tends to Dijkstra's values as shown by the blue part of the plot in figure 3. In the same figure, the green and red regions of the surface demonstrate the effect of the proposed algorithm and shows

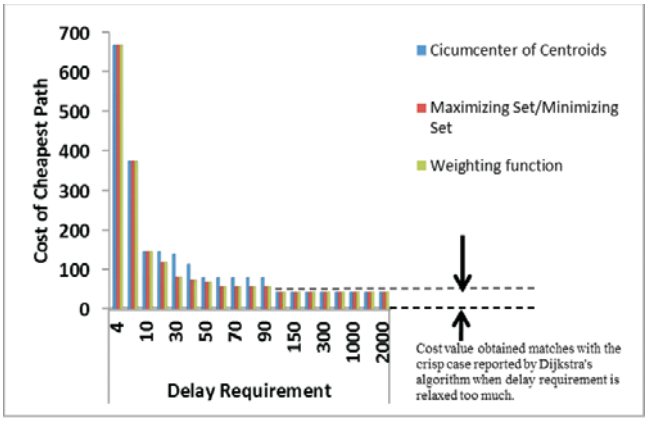


Figure 1a. Behavior in terms of cost of shortest path shown by different ranking methods by varying the delay requirement on a random graph with 250 nodes generated by gengraph-win. The solution tends to the unconstrained crisp case solved by Dijkstra's algorithm.

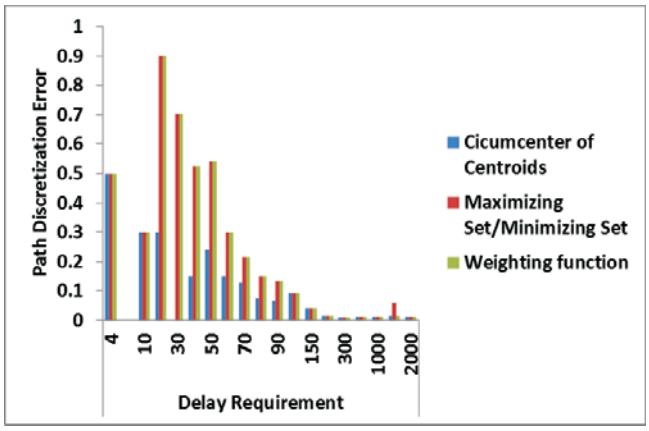


Figure 1b. Behavior in terms of path discretization error shown by different ranking methods by varying the delay requirement on a random graph with 250 nodes generated by gengraph-win.

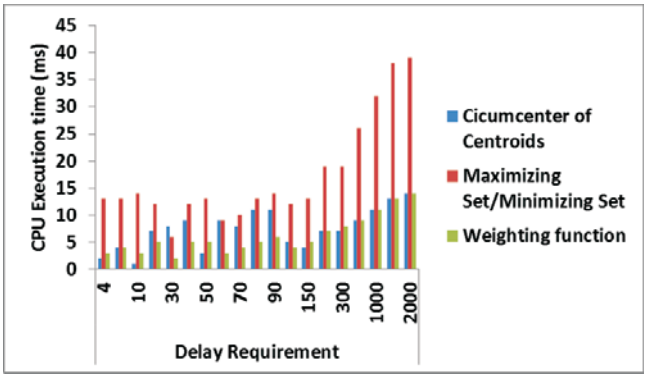


Figure 1c. Behavior in terms of CPU Execution time shown by different ranking methods by varying the delay requirement on a random graph with 250 nodes generated by gengraph-win.

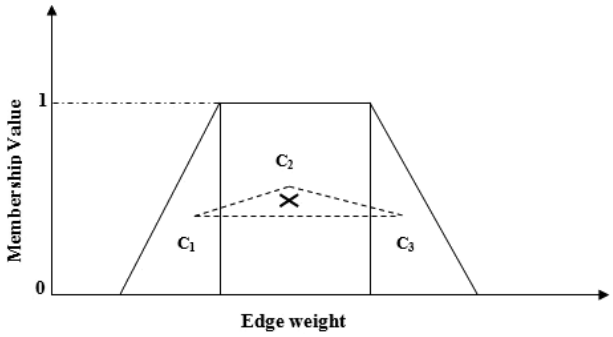


Figure 2. The point considered as Circumcenter of Centroid (COC) is shown by X.

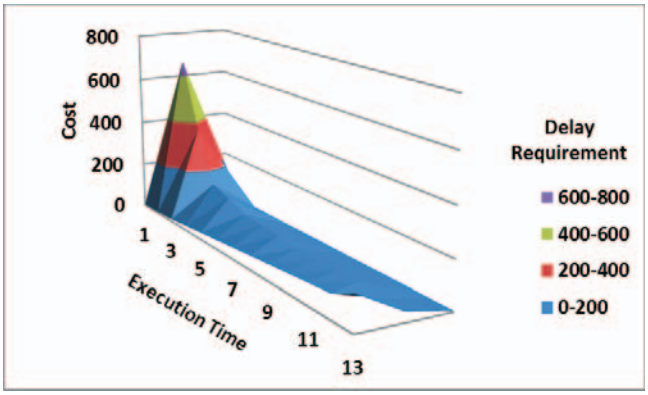


Figure 3. A surface plot with the delay requirement, cost and CPU execution time of WF.

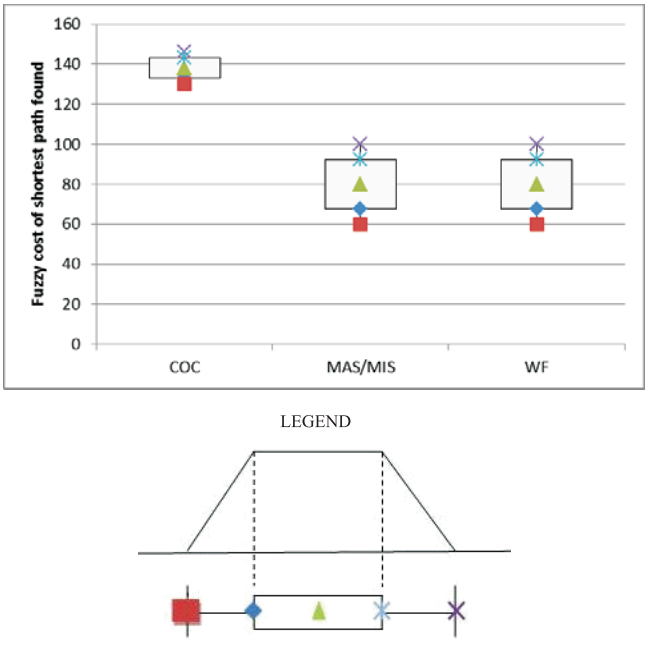


Figure 4. Box and Whisker plot showing the fuzzy cost as a TFN with four parameters at the delay requirement = 30 units.

how the cost of shortest path found by the algorithm increases with stricter delay constraint. The behaviour of the proposed algorithm shows a marked deviation from Dijkstra's algorithm when delay constraint is strict (i.e. small), as is clearly seen in figure 3. The cost obtained by all the three methods in the form of a trapezoidal fuzzy numbers with respect to a constant delay constraint of 30 is shown using a box and whisker plot in figure 4. As visible in figure 4, the cost obtained by COC method of ranking is much higher than the cost obtained by the other two methods i.e. MAS/MIS and WF.

5. Conclusion and Future Work

In this paper, we have dealt with two aspects namely uncertainty and multiple delay constraints in the shortest path problem. The original PDA algorithm was suggested by Chen *et al.* [1] and its fuzzy version for CFSPP is capable of tackling the uncertainty by representing the parameter cost as a fuzzy number and the other parameter i.e. delay, is represented by the crisp value in the same manner as in [1]. To determine the ranking of the fuzzy numbers we have used three different techniques which include Circumcenter of Centroids, Maximizing Set/Minimizing Set and Weighting Function and by comparing different output values of cost, path discretization error and CPU execution time with respect to varying delay requirement, we have concluded that the most efficient method of them all is the WF technique. So, the shortest path between a specified source and target, taking into consideration the multiple delay constraints and the uncertainty, can be most efficiently and accurately determined using WF as it is the best method in terms of execution time and cost however the method that gives minimum path discretization error is Circumcenter of Centroid method. In future, we plan to study the effects of fuzziness in delay requirement with crisp cost as well as the case when both the parameters are fuzzy. In these cases it would be interesting to see which fuzzy number ranking method gives better results.

Acknowledgment

Programming efforts by Jahnavi Singhal, Kavita Gupta and Sonul Saxena are gratefully acknowledged. The first author would like to acknowledge the financial support by IIT (BHU) in terms of teaching assistantship.

References

[1] S. Chen, M. Song and S. Sahni, Two Techniques for Fast Computation of Constrained Shortest Paths. IEEE/ACM Transactions on Networking, vol. 16(1), pp. 105–115, (2008).
[2] Y. Xiao, K. Thulasiraman, G. Xue and A. Juttner, The Constrained Shortest Path Problem. Algorithmic Approaches and an Algebraic Study with Generalization, http://www.cs.elte.hu/∼alpar/publications/jour/AKCE_October_25.pdf.
[3] Y. Dou, L. Zhu and H. S. Wang, Solving the Fuzzy Shortest Path Problem using Multi-Criteria Decision Method based on Vague Similarity Measure. Applied Soft Computing, vol. 12, pp. 1621–1631, (2012).
[4] X. Ji, K. Iwamura and Z. Shao, New Models for Shortest Path Problem with Fuzzy Arc Lengths. Applied Mathematical Modelling, vol. 31, pp. 259–269, (2007).
[5] J. Su and A. Li, Simulated Annealing Approach for the Constrained Shortest Path with Fuzzy Arc and Node Weights. In: World Congress on Computer Science and Information Engineering, vol. 4, pp. 754–758, (2009).
[6] J. Su and A. Li, Approach to the Shortest Path with Fuzzy Constraints by Simulated Annealing Algorithm. In: Global Congress on Intelligent System, vol. 1, pp. 516–520, (2009).
[7] H. Joksch, The Shortest Route Problem with Constraints. Journal of Mathematical Analysis and Applications, vol. 14(1), pp. 191–197, (1966).
[8] S. Sahni, General Techniques for Combinatorial Approximation. Operations Research, vol. 25, pp. 920–936, (1977).
[9] O. Ibarra and C. Kim, Fast Approximation Algorithms for the Knapsack and Sum of Subsets Problems. Journal of the Association for Computing Machinery, vol. 22, pp. 463–468, (1975).
[10] A. Warburton, Approximation of Pareto Optima in Multiple-objective Shortest Path Problems. Operations Research, vol. 35, pp. 70–79, (1987).
[11] R. Hassin, Approximation Schemes for the Restricted Shortest Path Problem. Mathematics of Operation Research, vol. 17(1), pp. 36–42, (1992).
[12] D. Lorenz and D. Raz, A Simple Efficient Approximation Scheme for the Restricted Shortest Paths Problem. Operations Research Letters, vol. 28, pp. 213–219, (2001).
[13] G. Luo, K. Huang, J. Wang, C. Hobbs and E. Munter, Multi-QoS Constraints based Routing for IP and ATM Networks. In: IEEE Workshop on QoS Support for Real-Time Internet Applications, Vancouver Canada, (1999).
[14] R. Ravindran, K. Thulasiraman, A. Das, K. Huang, G. Luo and G. Xue, Quality of Services Routing: Heuristics and Approximation Schemes with a Comparative Evaluation. In: IEEE International Symposium on Circuit and Systems, IEEE Press, Canada, pp. 775–778, (2002).
[15] T. Korkmaz and M. Krunz, Multi-constrained Optimal Path Selection. In: IEEE INFOCOM, USA, pp. 834–843, (2001).

[16] G. Xue, Primal-dual Algorithms for Computing Weight-constrained Shortest Paths and Weight-constrained Minimum Spanning Trees. In: 2000 IEEE International Performance, Computing, and Communications Conference, IEEE IPCCC '00, USA, pp. 271–277, (2000).
[17] G. Y. Handler and I. Zang, A Dual Algorithm for the Constrained Shortest Path Problem. Networks, vol. 10, pp. 293–310, (1980).
[18] A. Bansal, Trapezoidal Fuzzy Numbers (a, b, c, d): Arithmetic Behavior. International Journal of Physical and Mathematical Sciences, vol. 2(1), pp. 39–44, (2011).
[19] R. Jain, Decision Making in the Presence of Fuzzy Variables. IEEE Transactions on Systems, Man and Cybernetics, vol. 6(10), pp. 698–703, (1976).
[20] X. Wang and E. E. Kerre, Reasonable Properties for the Ordering of Fuzzy Quantities (I). Fuzzy Sets and Systems, vol. 118(3), pp. 375–385, (2001).
[21] X. Wang and E. E. Kerre, Reasonable Properties for the Ordering of Fuzzy Quantities (II). Fuzzy Sets and Systems, vol. 118(3), pp. 387–405, (2001).
[22] P. Phani Bushan Rao and N. Ravi Shankar, Ranking Fuzzy Numbers with a Distance Method using Circumcenter of Centroids and an Index of Modality. Advances in Fuzzy Systems, Hindawi Publishing Corporation, (2011). doi:10.1155/2011/178308.
[23] A. Saeidifar Application of Weighting Functions to the Ranking of Fuzzy Numbers. Computers and Mathematics with Applications, vol. 62, pp. 2246–2258, (2011).
[24] S. Y. Chou, L. Q. Dat and V. F. Yu, A Revised Method for Ranking Fuzzy Numbers using Maximizing Set and Minimizing Set. Computers & Industrial Engineering, vol. 61, pp. 1342–1348, (2011).
[25] S. H. Chen, Ranking Fuzzy Numbers with Maximizing Set and Minimizing Set. Fuzzy Sets and Systems, vol. 17, pp. 113–129, (1985).
[26] M. Verma and K. K. Shukla, Fuzzy Constrained Shortest Path Algorithm using Circumcenter of Centroids. In: 3rd IEEE International Advance Computing Conference, IEEE IACC'13, Ghaziabad, India, pp. 649–652, (2013).

A Stochastic Greedy Heuristic Algorithm for the Orienteering Problem

Madhushi Verma*
Department of Computer Science and Engineering
IIT (BHU)
Varanasi, India
madhushi.rs.cse@itbhu.ac.in
(*Corresponding Author)

Bijeeta Pal
Department of Computer Science and Engineering
IIT (BHU)
Varanasi, India
bijeeta.pal.cse10@itbhu.ac.in

Mukul Gupta
Department of Computer Science and Engineering
IIT (BHU)
Varanasi, India
mukul.gupta.cse10@itbhu.ac.in

K. K. Shukla
Department of Computer Science and Engineering
IIT (BHU)
Varanasi, India
kkshukla.cse@itbhu.ac.in

Abstract— **In this paper we introduce a new stochastic greedy heuristic algorithm for the orienteering problem (OP) which is an NP-Hard combinatorial optimization graph problem. The goal of OP is to determine a Hamiltonian path that connects the specified source and target, includes a set of control points and achieves the best possible total collected score within the fixed time frame. This problem finds application in several fields like logistics, telecommunication networks, tourism industry etc. In each of these applications, it is necessary to provide a practical modelling and the best way to tackle this situation is to use incomplete graphs. However, most of the techniques available in the literature can only be applied on complete graphs. Unlike other algorithms, our algorithm can be applied on both complete and incomplete graphs. We have implemented our algorithm using four different selection procedures and evaluated their performance on standard benchmarks. We find that the algorithm works best with the roulette wheel selection.**

Keywords- Orienteering problem, selection methods, incomplete graphs.

I. INTRODUCTION

Orienteering Problem (OP) which is a combination of two well-known problems i.e. travelling salesman problem (TSP) and Knapsack problem (KP) originated from a water sport where the goal is to determine a path connecting the source and the target and at the same time explore a set of control points. Not all control points can be visited because of the limit on available time (or total distance travelled) [1]. OP is an NP-Hard combinatorial optimization problem and finds application in various fields like the tourism industry, robot path planning in disaster management, home delivery system, telecommunication networks, logistics, transportation networks etc. Several variants of OP like team orienteering, orienteering and team orienteering with time window, generalized orienteering are defined in the literature which models different real life situations [1].

In 1990, Laporte et al suggested an exact algorithm for OP using the concept of linear programming [2]. Another exact algorithm was introduced by Hayes et al based on the concept of dynamic programming [3]. However, as OP is NP-Hard, it is not feasible in terms of execution time to implement exact algorithms for large instances. So, the efficient technique to solve OP is to use some heuristic or approximation algorithm and the first heuristic for OP was proposed by Tsiligirides in 1984 [4]. In 1991, a four-phase heuristic was suggested by Ramesh and Brown [5]. Golden et al introduced the centre-of-gravity heuristic in 1987 [6]. A well-known branch and cut method to solve OP was presented by Fischetti et al [7]. Later, methods like tabu search and artificial neural networks were proposed to solve OP [8-9]. In 2009, pareto ant colony optimization and pareto variable neighbourhood search techniques were suggested by Schilde et al to deal with OP [10]. Campos et al in 2013, introduced a method called Greedy Randomized Adaptive Search Procedure (GRASP) which is an efficient heuristic for tackling the orienteering problem [11]. For the time dependent variant of OP, an approximation algorithm was suggested by Fomin et al [12]. Other approximation algorithms for the un-rooted version and rooted version of OP were proposed by Awerbuch et al., Johnson et al and Blum et al respectively [13-15]. As stated earlier, OP finds several real life applications and most of the practical situations cannot be modelled through complete graphs only. A more realistic picture can be presented by modelling such situations using incomplete graphs. However, most of the algorithms available in the literature for OP can be applied on complete graphs only. Tasgetiren proposed the first genetic algorithm for OP but in 2011, Ostrowski et al presented another genetic algorithm that can be applied on both complete as well as incomplete graphs [16-17].

In this paper, we present a heuristic for OP (SEL_OP) which can be implemented on both complete as well as incomplete graphs. In our algorithm, we use different selection procedures and compare the results to determine the selection method that helps in achieving the best possible score for the orienteering problem.

978-1-4799-6758-2/14/$31.00 ©2014 IEEE

II. PRE-REQUISITES

A. Problem Definition

In OP the aim is to determine a Hamiltonian Path P that connects the stated source (v_1) and target (v_N), includes a subset (V') of the vertex set V such that the total collected score can be maximized within the stated time budget (T_{max}). OP can be represented by an undirected weighted graph (complete or incomplete) $G(V, E)$ where V is the set of vertices and E is the set of edges. We associate two functions i.e. a time function and a score function to the edges and vertices respectively. Let $t: E \rightarrow \Re^+$ denote the time function and $S: V \rightarrow \Re^+$ signify the score function. So, for a subset $V' of\ V$ and $E' of\ E$, we have $S(V') = \sum_{v \in V'} S_v$ and $t(E') = \sum_{e \in E'} t(e)$ respectively [1].

The OP can be stated as an integer programming problem which is as follows [1]:

$$Max \sum_{i=1}^{N-1} \sum_{j=2}^{N} S_i x_{ij} \quad (1)$$

$$\sum_{j=2}^{N} x_{1j} = 1 \ , \ \sum_{i=1}^{N-1} x_{iN} = 1 \quad (2)$$

$$\sum_{i=1}^{N-1} x_{ik} \leq 1 \quad \forall\ k = 2, \dots\dots, N-1 \quad (3)$$

$$\sum_{j=2}^{N} x_{kj} \leq 1 \quad \forall\ k = 2, \dots\dots, N-1 \quad (4)$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} t_{ij} x_{ij} \ \leq T_{max} \quad (5)$$

$$2 \leq u_i \leq N \quad \forall\ i = 2, \dots\dots, N \quad (6)$$

$$u_i - u_j + 1 \leq (N-1)\left(1 - x_{ij}\right) \quad \forall\ i, j = 2, \dots\dots\dots, N \quad (7)$$

$$x_{ij} \in \{0,1\} \quad \forall\ i, j = 1, \dots\dots, N \quad (8)$$

Equation 1 represents the objective function of OP i.e. maximization of the total collected score. Few other constraints like a path should have v_1 as its starting and v_N as its ending node is taken care by equation 2 and no vertex is visited more than once and the path remains connected is ensured by equation 3-4. The important condition that the path satisfies the time bound (T_{max}) is represented by equation 5. The need to eliminate sub tours is executed by equations 6 and 7. Variable u_i signifies the position of vertex v_i in the path and x_{ij} = 1 if v_j is visited after v_i otherwise, x_{ij} = 0.

B. Selection Methods

1) Tournament Selection

In this selection technique, one individual is chosen from a population of individuals. Several tournaments are run among the randomly selected individuals from a population and the best ones are copied to the next generation (those that win the tournament i.e. have the best fitness value). The process is repeatedly performed several times to saturate the population. The tournament size is denoted by q and the most common tournament size is q=2. By altering the tournament size, the selection pressure can be easily adjusted. In case the tournament size is large then weak individuals have a lesser chance of getting selected. Some advantages of this method are (1) it does not require any sorting mechanism and therefore can be implemented in O(n) time and (2) it supports parallel architecture [18-20].

2) (μ, λ) Selection

This selection technique was formulated with the idea of reducing the offspring population generated as a result of recombination and mutation. In this case, the offspring population of size $\lambda \geq \mu$ is reduced by selecting μ best offspring individuals as new parents for the next generation. It is an easy to implement technique, takes less time and practically follows the greedy approach therefore generates good results and helps in most of the cases [18-20].

3) Roulette Wheel Selection

This selection method is also called proportional selection method and uses the fitness proportionate approach. It selects an element randomly from a list on the basis of the fitness function. The elements having a higher fitness value have a greater probability of getting selected, however, even the elements with a lesser fitness value has a non-zero probability of getting selected [18-20].

4) Random Selection

It is a non-probabilistic and the simplest selection method. In this technique, any node is randomly selected from the set of available candidate nodes i.e. a node is selected only because it is readily available and can be conveniently used for further operation without performing any other processing. This method can explore a large search space and provides a wide range of answers. Also, if the technique is allowed to run large number of times, it may generate the best possible solution [18-20].

III. ALGORITHM

Input: A graph $G(V, E)$ with t_{ij} (time taken to traverse) value and S_i (Score) value of each edge and each vertex respectively. A constant $PathListSize$ value which denotes the maximum number of paths that are considered at each level.

Output: A Hamiltonian path with the best possible total collected score such that total travel time is within the stated time bound.

$SEL\_OP(G, PathListSize, T_{max})$

1. $\boldsymbol{Create}\ PathList;$
   // Array of paths which is initially empty.
2. $PathList \leftarrow \emptyset;$
3. $Path\ \mathrm{P} \leftarrow Dijkstra(v_1, v_n);$
   // Shortest path between source (v_1) and target (v_N).
4. $\boldsymbol{Insert}\ P\ to\ PathList;$
5. $\boldsymbol{return}\ Generation(PathList);$

$Generation(PathList)$

1. $\boldsymbol{Create}\ NewPathList\ , ChildpathList;$
   // Queues storing paths.
   $NewPathList \leftarrow \emptyset;$
   $ChildPathList \leftarrow \emptyset;$
2. $\boldsymbol{for}\ i \leftarrow 0\ to\ |PathList|$

$a.Selection(PathList[i], ChildPathList);$
//$ChildPathList$ will contain children generated from each path in $PathList$.

3. $\boldsymbol{for}\ i \leftarrow 0\ to\ PathListSize$
// Selecting best $PathListSize$ children for next generation.
$a.ChildPath = BestPath(ChildPathList);$
// The path with the maximum total score.
$\boldsymbol{Remove}\ ChildPath\ from\ ChildPathList;$
$\boldsymbol{Insert}\ ChildPath\ to\ NewPathList;$
$b.\boldsymbol{if}\ |ChildPathList| == 0$
$break;$
4. $\boldsymbol{if}\ NewPathList == PathList$
// Terminate if no new child is generated and return the $BestPath$.
$\boldsymbol{return}\ BestPath(PathList);$
5. $\boldsymbol{return}\ Generation(NewPathList);$

$Selection\ (Path\ P, ChildPathList)$
//$Path$ is an array of nodes that forms a path.

1. $q_{max} \leftarrow 0;$
2. $\boldsymbol{for}\ i \leftarrow 0\ to\ |V|$
3. $a.\boldsymbol{if}\ i \in P$
// If a node is already present in the path then ignore it.
$\boldsymbol{continue};$
$b.\boldsymbol{Calculate}\Delta t_i\ ;$
// The time increment due to insertion of v_i at its best position.

$$c.\ q_i = \begin{cases} S_i/|\Delta t_i|, & \Delta t_i \geq 1 \\ S_i, & -1 \leq \Delta t_i < 1 \\ S_i * |\Delta t_i|, & \Delta t_i < -1 \end{cases}$$

$d.\boldsymbol{if} q_{max} \leq q_i$
$q_{max} \leftarrow q_i$
4. $Create\ CandidateList\ ;$
// Array of candidate nodes used for selection.
5. $CandidateList \leftarrow \emptyset;$
6. $\boldsymbol{for}\ i \leftarrow 0\ to\ |V|$
7. $a.\boldsymbol{if}\ i \in P$
$\boldsymbol{continue};$
$b.\boldsymbol{if}\ (q_i \geq \alpha\ q_{max})\&\&\ (t_{parent} + \Delta t_i \leq T_{max})$
// α is the greediness parameter that decides which node should participate in selection process.
$c.\boldsymbol{Insert}\ v_i\ to\ CandidateList\ ;$
8. $\boldsymbol{if}|CandidateList| == 0$
$\boldsymbol{Insert}\ P\ to\ ChildPathList;$
// If no new nodes are added then insert the parent path P.
$\boldsymbol{Return};$
9. $\boldsymbol{for}\ i \leftarrow 0\ to\ PathListSize$
// Generates $PathListSize$ number of children paths from path P.
10. $a.ChildNode \leftarrow Compute(CandidateList)$

// Compute 1 using tournament selection.
// Compute 2 using (μ, λ) selection.
// Compute 3 using roulette wheel selection.
// Compute 4 using random selection.

$\boldsymbol{Remove}\ ChildNode\ from\ CandidateList;$
$\boldsymbol{Insert}\ ChildNode\ to\ Path\ P\ and\ \boldsymbol{Insert}\ Path\ P\ to\ ChildPathList.$
$\boldsymbol{Remove}\ ChildNode\ From\ Path\ P;$
$b.\boldsymbol{if}\ |CandidateList| = 0$
$\boldsymbol{break};$

$BestPath(PathList)$

1. $max \leftarrow 0; bestpath \leftarrow 0;$
2. $\boldsymbol{for}\ i \leftarrow 0\ to\ |PathList|$
a.$\boldsymbol{if}\ \big(Score\big(PathList(i)\big)\big) > maxScore$
// *Score* ($Path\ P$) is the sum of the rewards associated with each node of $ath\ P$.
$b.maxScore \leftarrow Score\big(PathList(i)\big);$
$bestpath \leftarrow i;$
3. $return\ PathList(bestpath);$

The aim of above stated algorithm is to determine a path that connects the source (v_1) and target (v_N) and a subset of vertices that maximize the total collected score and obeys the time limit. This algorithm can be implemented on both complete as well as incomplete graphs. To ensure that in a path, source and target is connected we implement the Dijkstra's algorithm as shown in step 3 of $SEL\_OP()$. To take care of the other constraint that no vertex is visited more than once, the explored nodes are removed from the set of available nodes that forms the candidate list for the selection process. In the $Selection\ ()$, four selection techniques are used and a comparison of their results is presented in the next section. The time complexity of SEL_OP is $O(|V|^3)$.

IV. Experimental Analysis

Our code was implemented in C++ and compiled using CodeBlocks on an Intel Core i5 650 running at 2.20 GHz. SEL_OP has the capability to tackle both complete as well as incomplete graphs and here we report the results for incomplete graphs by running the code on some real data. We considered two instances i.e. a real road network data of 160 and 306 cities of Poland. Each instance is associated with two files i.e. cities.txt and distances.txt. The names of the various cities and their scores are specified in cities.txt. The score of each city is calculated on the basis of the number of inhabitants using the formula $score = inhabitants/10000$. The other file distances.txt represents for each city, its adjacent city and their respective edge lengths. These edges of the graph correspond to the roads of the real map of Poland.

The algorithm SEL_OP was executed using four different selection methods defined in section II (B). In Table 1, the maximum and mean score value achieved by each selection technique for different T_{max} values is presented and in Figure 1, these values are plotted for the first instance with 160 cities. It was observed that roulette wheel selection method performs better than the other three

techniques in most of the cases and helps in obtaining a better total collected score when compared to the other techniques.

It was also seen that when the same algorithm was executed for the 306 cities instance, only the random selection method suffers and rest of the selection methods achieves the same maximum total collected score most of the times as shown in Table 2 and Figure 2. The possible reason for this observation might be the existence of big clusters in the 306 cities instance which is absent in case of the 160 cities instance as shown in Figure 3 (a) and 3 (b). Therefore, the selection techniques have more possible options available to be explored in case of 160 cities instance and do not get stuck within a cluster. So, it can be stated here that roulette wheel selection procedure performs better than the other methods when executed by SEL_OP algorithm.

V. CONCLUSION

In this paper, we introduce a new stochastic greedy heuristic approach (SEL_OP algorithm) for the orienteering problem which can be applied for both complete as well as incomplete graphs. It was implemented using four different selection methods and the results thus obtained were compared on standard benchmarks. We find that roulette wheel selection method performs better than the other three selection techniques and helps in achieving a better total collected score for the specified time budget.

ACKNOWLEDGMENT

The first author would like to acknowledge the financial support by IIT (BHU) in terms of teaching assistantship.

REFERENCES

[1] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, "The orienteering problem: A survey", European Journal of Operational Research, vol. 209, pp. 1–10, 2011.

[2] G. Laporte, and S. Martello, "The Selective Traveling Salesman Problem", Discrete Applied Mathematics, vol. 26, pp. 193-207, 1990.

[3] M. Hayes, and J. M. Norman, "Dynamic Programming in Orienteering: Route Choice and the Siting of Controls", Journal of the Operational Research Society, vol. 35 (9), pp. 791-796, 1984.

[4] T. Tsiligirides, "Heuristic methods applied to orienteering", Journal of the Operational Research Society, vol. 35, pp. 797–809, 1984.

[5] R. Ramesh, and K. Brown, "An efficient four-phase heuristic for the generalized orienteering problem", Computers and Operations Research, vol. 18, pp. 151–165, 1991.

[6] B. Golden, L. Levy, and R. Vohra, "The orienteering problem", Naval Research Logistics, vol. 34, pp. 307–318, 1987.

[7] M. Fischetti, J. Salazar, and P. Toth, "Solving the orienteering problem through branch-and-cut", INFORMS Journal on Computing, vol. 10, pp. 133–148, 1998.

[8] Q. Wang, X. Sun, B. L. Golden, and J. Jia, "Using artificial neural networks to solve the orienteering problem", Annals of Operations Research, vol. 61, pp. 111–120, 1995.

[9] M. Gendreau, G. Laporte, and F. Semet, "A tabu search heuristic for the undirected selective travelling salesman problem", European Journal of Operational Research, vol. 106, pp. 539–545, 1998.

[10] M. Schilde, K. F. Doerner, R. F. Hartl, and G. Kiechle, G. "Metaheuristics for the bi-objective orienteering problem", Swarm Intelligence, vol. 3, pp. 179-201, 2009.

[11] V. Campos, R. Marti, J. Sanchez-Oro, and A. Duarte, "GRASP with Path Relinking for the Orienteering Problem", Journal of the Operational Research Society (2013), pp. 1–14, 2013.

[12] F. V. Fomin, and A. Lingas, "Approximation algorithms for time-dependent orienteering", Information Processing Letters, vol. 83, pp. 57–62, 2002.

[13] B. Awerbuch, Y. Azar, A. Blum, and S. Vempala, "Improved approximation guarantees for minimum-weight k-trees and prize-collecting salesmen", Siam J. Computing, vol. 28, pp. 254–262, 1999.

[14] D. Johnson, M. Minkoff, and S. Phillips, "The prize collecting steiner tree problem: Theory and practice", Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms, 2000, pp. 760–769.

[15] A. Blum, S. Chawla, D. R. Karger, T. Lane, A. Meyerson and M. Minkoff, "Approximation Algorithms for Orienteering and Discounted-Reward TSP", Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS'03), 2003, pp. 1-10.

[16] M. Tasgetiren, "A genetic algorithm with an adaptive penalty function for the orienteering problem", Journal of Economic and Social Research, vol. 4(2), pp. 1–26, 2001.

[17] K. Ostrowski, and J. Koszelew, "The Comparison of Genetic Algorithms which Solve Orienteering Problem using Complete and Incomplete Graph", Informatyka, vol. 8, pp. 61-77, 2011.

[18] T. Back, "Selective Pressure in Evolutionary Algorithms: A Characterization of Selection Mechanisms", Proceedings of the First IEEE Conference on Evolutionary Computation, 1994, pp. 57-62.

[19] N. M. Razali, and J. Geraghty, "Genetic Algorithm Performance with Different Selection Strategies in Solving TSP", Proceedings of the World Congress on Engineering 2011, vol II WCE 2011, July 6 - 8, 2011, London, U.K. pp. 1134-1139.

[20] R.Sivaraj, and T.Ravichandran, "A Review of Selection Methods in Genetic Algorithm", International Journal of Engineering Science and Technology (IJEST), vol. 3(5), pp. 3792-3797, 2011.

TABLE I. Comparison of the mean and maximum value of the total collected score obtained by SEL_OP when executed with four different selection procedures for 160 cities.

T_{max}	(μ,λ)		Random		Tournament		Roulette Wheel	
	Max	Mean	Max	Mean	Max	Mean	Max	Mean
500	49	49	49	48	49	48	49	48
750	67	67	67	64	67	66	67	65
1000	83	83	88	77	88	84	88	79
1250	103	103	103	99	103	101	102	100
1500	118	118	116	114	119	116	116	114
1750	135	135	130	128	135	128	130	128
2000	145	145	145	143	145	143	148	143
2250	157	157	160	154	157	155	162	155
2500	179	179	184	175	179	176	187	174
2750	190	190	203	193	191	189	203	193
3000	206	206	214	205	206	202	214	206
3250	234	234	233	223	234	228	233	224
3500	248	248	249	242	249	245	251	244
3750	255	255	261	255	257	255	263	256
4000	269	269	271	264	272	268	270	267

TABLE II. Comparison of the mean and maximum value of the total collected score obtained by SEL_OP when executed with four different selection procedures for 306 cities.

T_{max}	(μ,λ)		Random		Tournament		Roulette Wheel	
	Max	Mean	Max	Mean	Max	Mean	Max	Mean
500	127	121	127	117	127	121	127	118
750	155	149	155	147	155	148	155	147
1000	178	173	177	168	178	172	178	169
1250	211	173	208	190	211	172	211	193
1500	236	213	234	218	236	219	236	221
1750	263	241	263	253	263	254	263	251
2000	282	280	279	273	282	277	282	273
2250	307	300	299	291	307	292	307	281
2500	325	323	323	307	325	309	325	306
2750	349	336	333	325	349	327	349	329
3000	366	355	366	346	366	345	366	344
3250	396	339	386	366	396	371	396	369
3500	420	394	414	384	420	393	420	390
3750	435	408	428	410	435	411	435	400
4000	459	430	448	429	459	437	459	440

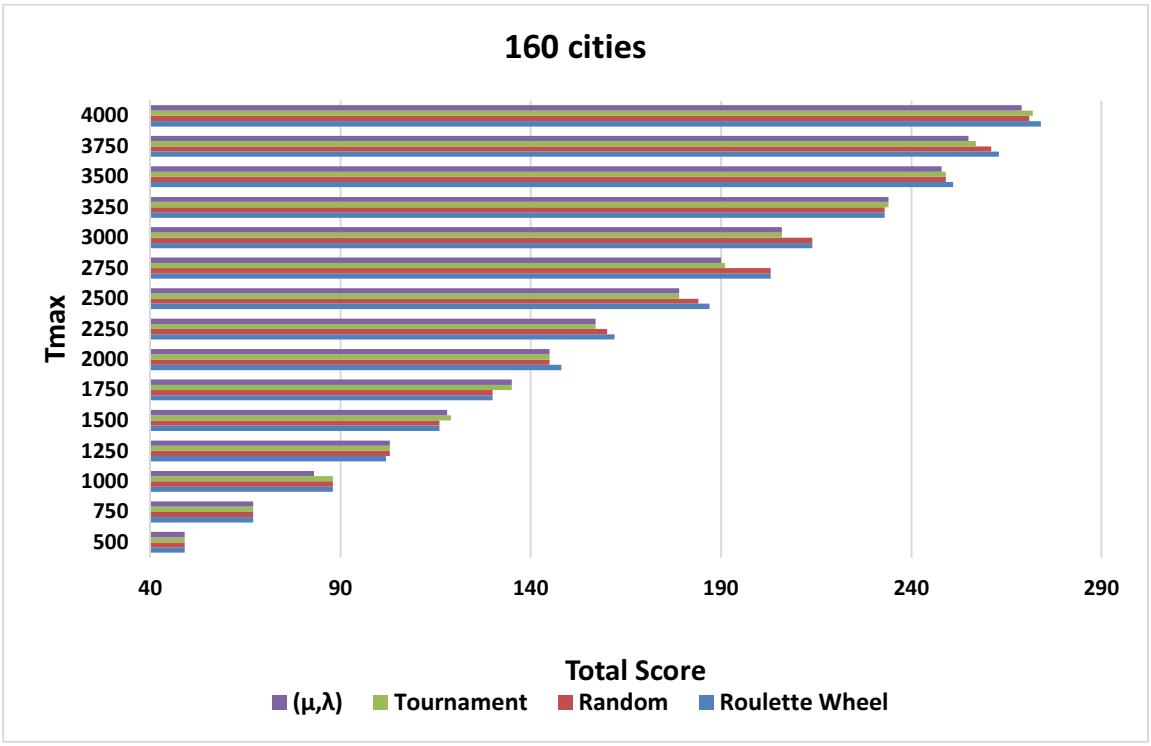


Figure 1. Comparison of the maximum value of the total collected score obtained by four different selection methods for different T_{max} values (160 cities).

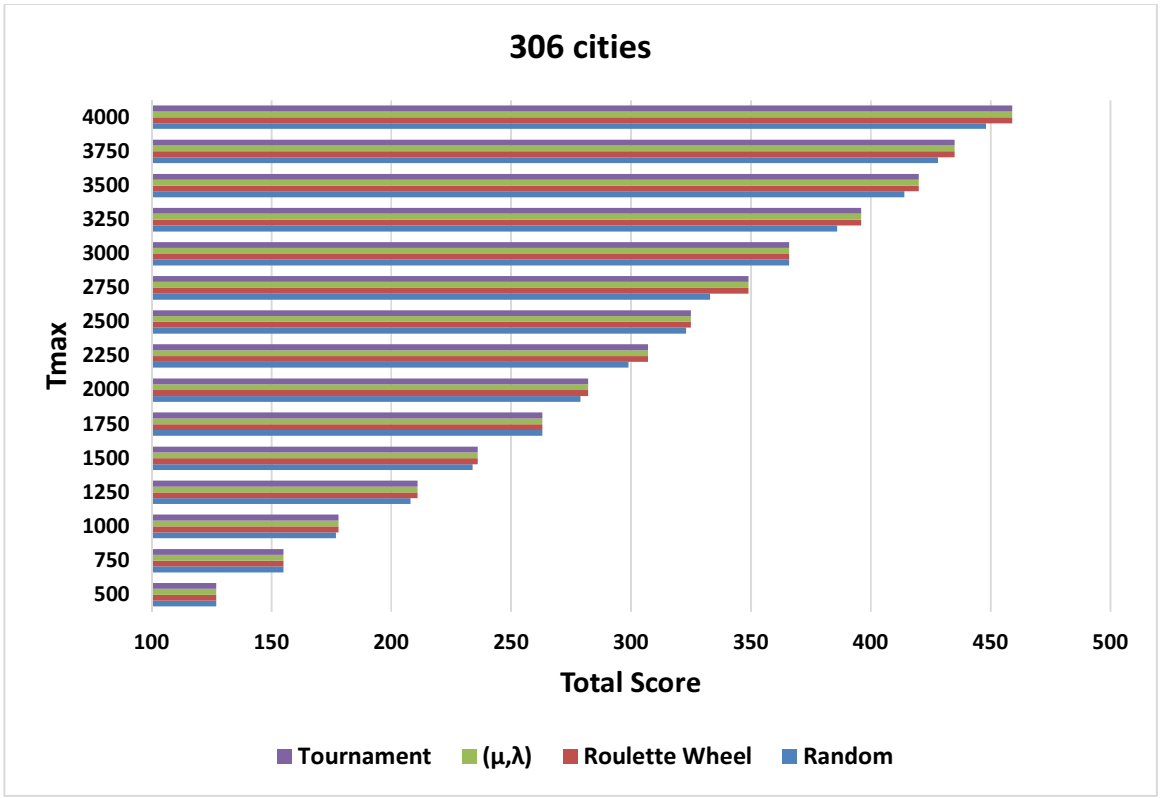
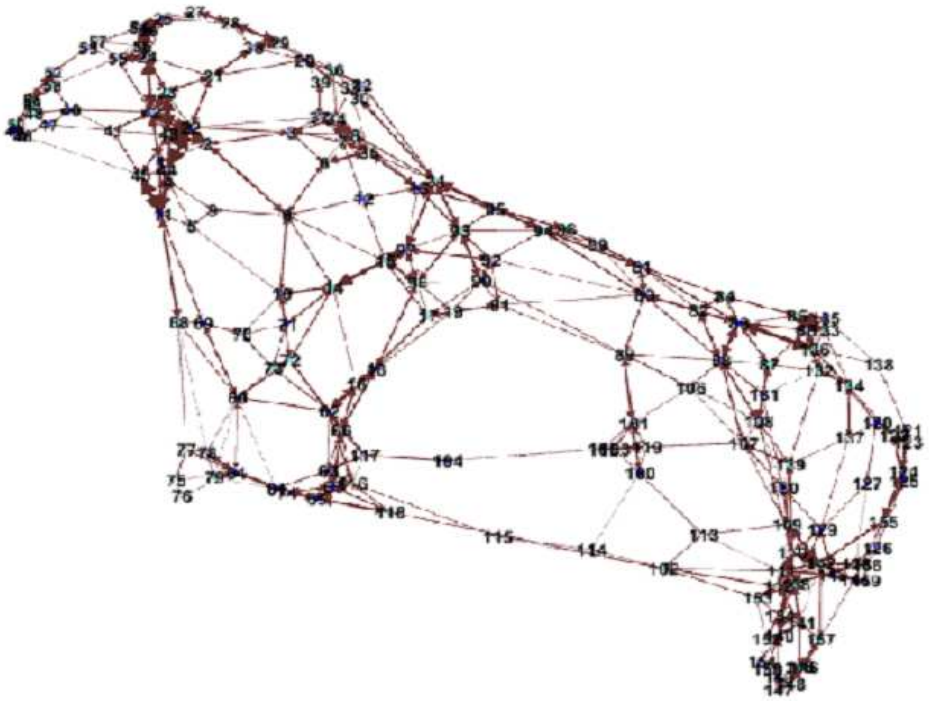
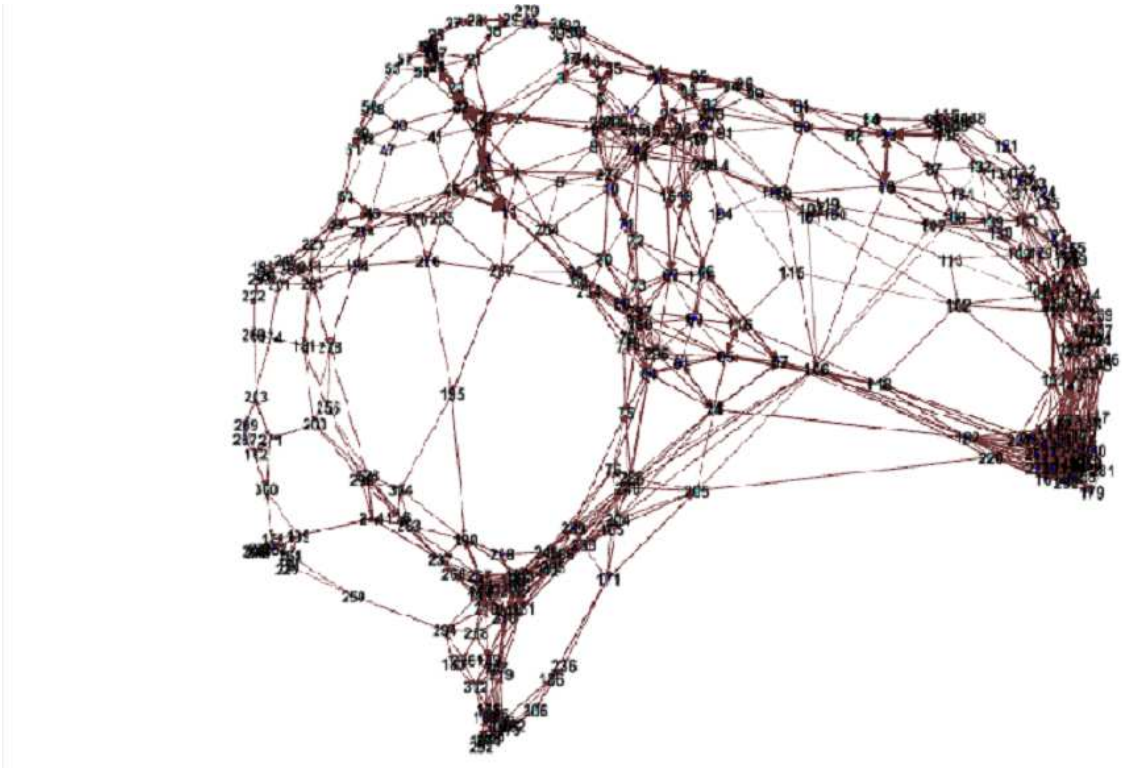


Figure 2. Comparison of the maximum value of the total collected score obtained by four different selection methods for different T_{max} values (306 cities).



(a)



(b)

Figure 3. Graph for (a) 160 cities and (b) 306 cities instance.

Fuzzy Constrained Shortest Path Algorithm using Circumcenter of Centroids

Madhushi Verma
Computer Engineering Department
IIT- BHU
Varanasi, India
madhushi.rs.cse@itbhu.ac.in

K. K. Shukla
Computer Engineering Department
IIT- BHU
Varanasi, India
kkshukla.cse@itbhu.ac.in

Abstract— **The constrained shortest path problem (CSPP) is a well known NP-Complete problem where the task is to determine the shortest path that satisfies certain constrains like delay and cost. Other than the straight-forward implementation in the area of networking, this problem also finds application in the field of multimedia, crew scheduling etc. In these kinds of applications it is important to take into consideration the uncertainty involved in the network environment to provide the Quality of Service (QoS) assurance which can be modeled by the use of fuzzy numbers for the representation of the parameters involved. In this paper, we extend the algorithm stated by Sahni et al to deal with the fuzzy constrained shortest path problem (FCSPP) by fuzzifying one of the constraints i.e. cost and representing it as a trapezoidal fuzzy number. Fuzzy numbers cannot be ordered like real numbers so the Circumcenter of Centroid method is used to rank the trapezoidal fuzzy numbers and determine the fuzzy constrained shortest path also taking into account the delay involved.**

Keywords—constrained shortest path, fuzzy constrained shortest path, trapezoidal fuzzy numbers, circumcenter of centroid method, path delay discretization.

I. INTRODUCTION

Shortest Path Problem (SPP) is one of the fundamental optimization problems in the field of Computer Science, Telecommunications and Operations Research that is polynomial time solvable. An extension of SPP is Constrained Shortest Path Problem (CSPP) where the shortest path computed is required to fulfill a set of constraints which leads to removal of those links from the graph that violates the constraints and then the Shortest Path algorithm is applied [1-2]. Adding one or more constraints to the SPP makes it intractable and CSPP is a well known NP-Complete Problem [1]. Applications of CSPP include computer networks, cable television networks, communication networks, transportation networks and multimedia applications like web broadcasting, video teleconferencing, remote diagnosis etc. [2-3]. To provide Quality of Service (QoS) assurance for the stated applications it is necessary to consider the uncertainty involved in the network environment as the parameters involved like cost, time, delay are not naturally precise and the uncertainty can be modeled by using Fuzzy numbers leading to Fuzzy Constrained Shortest Path Problem (FCSPP) [3].

One of the first algorithms, to solve CSPP was given by Joksh in [4]. CSPP is an NP-Complete problem so most of the algorithms suggested are either Heuristic or Approximation algorithms and the strategies used to provide the solution are mostly either Dynamic Programming (DP) or Lagrangian Relaxation (LR) [1-2]. Some initial techniques were suggested by Sahni [5] and Ibarra and Kim [6] that were later used by researchers to solve the CSPP using ε- approximation algorithms. Let m be the number of links and n be the number of nodes in the given network and $U\ and\ L$ be the upper bound and lower bound on the optimal cost respectively. In [7], Hassin suggested a Fully Polynomial Time Approximation Scheme (FPTAS) for CSPP based on the technique of DP and scaling / rounding with a time complexity of $O(\log\log(U/L)\,[mn\,\varepsilon^{-1} + \log\log(U/L)])$ and in [8], Juttner et al applied the technique of LR on the delay constrained least cost routing problem and solved it in $O(m^2 log^4 m)$ time. A heuristic algorithm was presented by Korkmaz and Krunz with the same time complexity as that of Dijkstra's algorithm [9]. Handler and Zang used the strategy of LR and applied it to the k-th shortest path problem algorithm, to solve CSPP [10].

Here we represent the parameters in the form of fuzzy numbers to model the uncertainty. Several types of fuzzy numbers exist like Gaussian fuzzy numbers, Exponential fuzzy numbers, Quadratic fuzzy numbers, Triangular fuzzy numbers, Trapezoidal fuzzy numbers etc but we prefer Trapezoidal fuzzy numbers for two important reasons. Firstly, it is the most generic class of fuzzy numbers and has a linear membership function. Therefore, it is widely used in modeling the uncertainty of the scientific and applied engineering problems. Secondly, its simplicity both in terms of concept and computation increases its use while dealing with problems like CSPP [11].

In problems like CSPP, it is important to rank the fuzzy numbers to determine their ordering as the requirement is to find the shortest path taking into consideration certain constraints. In case of real numbers, there exists a natural ordering which is absent for fuzzy numbers. So many ranking methods convert the fuzzy numbers to real numbers and then analyze them but in this procedure much of the information is lost. Attempts have been made to determine techniques that can rank the fuzzy numbers with the minimum loss of information. The concept of ranking fuzzy numbers was first suggested by Jain in [12]. Since then a lot of methods have been suggested by different researchers. Wang and Kerre in

978-1-4673-4529-3/12/$31.00 ©2012 IEEE

[13-14] categorized all the methods suggested for ranking the fuzzy numbers into three classes based on fuzzy mean and spread, fuzzy scoring and preference relation. Here we use the method based on Circumcenter of Centroids for ranking trapezoidal fuzzy numbers suggested by Rao and Shankar [15].

II. PRELIMINARIES

A. Trapezoidal fuzzy number(TFN)

A Trapezoidal Fuzzy Number $A = (a_1, a_2, a_3, a_4)$ can be represented by the following membership function where $a_1 \leq a_2 \leq a_3 \leq a_4$ [11].

$$\mu_A(x) = \begin{cases} 0, & x < a_1 \\ \frac{x-a_1}{a_2-a_1}, & a_1 < x \leq a_2 \\ 1, & a_2 < x < a_3 \\ \frac{a_4-x}{a_4-a_3}, & a_3 \leq x < a_4 \\ 0, & x > a_4 \end{cases} \quad (1)$$

B. Addition of TFN

If $A = (a_1, a_2, a_3, a_4)$ and $B = (b_1, b_2, b_3, b_4)$ be two trapezoidal fuzzy numbers then their sum is given by the following formula [11]:

$$A + B = (a_1, a_2, a_3, a_4) + (b_1, b_2, b_3, b_4) = (a_1 + b_1, a_2 + b_2, a_3 + b_3, a_4 + b_4) \quad (2)$$

C. Ranking of TFN

The natural ordering that exists for real numbers is missing in case of fuzzy numbers. In problems like FCSPP where weights are fuzzy numbers, it is required to rank the fuzzy numbers. Several methods have been proposed for ranking fuzzy numbers but here we use the Circumcenter of Centroids method. The centroid can be considered as the balancing point of the trapezoid but a better point of reference would be the Circumcenter of the Centroids as the trapezoid can be divided into three plane figures, one triangle, one rectangle and another triangle. The centroid of each of the plane figures when joined forms a triangle and then the Circumcenter is determined. This point is called the Circumcenter of Centroids (COC) and is equidistant from each of the vertex of the triangle formed by joining the Centroids of the plane figures and is considered to be a better point of reference.

If $A = (a_1, a_2, a_3, a_4)$ be a trapezoidal fuzzy number then COC is calculated using the following formula [15]:

$$COC(A) = (x, y) = \left(\frac{a_1+2a_2+2a_3+a_4}{6}, \frac{(2a_1+a_2-3a_3)(2a_4+a_3-3a_2)+5}{12}\right) \quad (3)$$

Now the rank of the fuzzy number A is defined as

$$R(A) = \sqrt{x^2 + y^2} \quad (4)$$

If we have two fuzzy numbers $A\ and\ B$ and if $R(A) > R(B)$ then $A > B$ and vice versa. If $R(A) = R(B)$ then we use the following method to determine the ranking:

Index of Optimism: For a given fuzzy number $A = (a_1, a_2, a_3, a_4)$ and $COC(A)$ an index related to ranking has been defined by Rao and Shanker in [15] which correspond to the decision maker's degree of optimism and can be represented as:

$$I_\delta(A) = \delta y + (1 - \delta)x \quad \text{where } \delta \in [0, 1] \quad (5)$$

The larger the value of δ the greater is the degree of optimism of the decision maker.

Index of Modality: As the index of optimism uses the extreme values of COC, it is not sufficient to rank the fuzzy numbers so another index called index of modality that considers the importance of central value along with the index of optimism was defined in [15] which can be stated as:

$$I_{\delta,\beta}(A) = \beta((x + y)/2) + (1 - \beta)I_\delta(A) \quad \text{where } \beta \in [0, 1] \quad (6)$$

So when $R(A) = R(B)$, we need to calculate $I_{\delta,\beta}(A)$ and $I_{\delta,\beta}(B)$.

If $I_{\delta,\beta}(A) > I_{\delta,\beta}(B)$ then $A > B$ else $B > A$.

D. Path Delay Discretization

As stated earlier CSPP is an NP-Complete problem. Techniques suggested to solve this problem reduce the number of different values (cost or delay) by using the process of discretization so that the problem can be reduced to a polynomial time solvable problem [1]. The amount of error introduced during the process of discretization determines the effectiveness of these techniques. Taking into consideration two factors, (1) the utilization of limited resources and (2) improving the efficiency of network functions, Sahni et al in [1] suggested two methods of discretization, randomized discretization and path-delay discretization.

Here we use the path-delay discretization technique to solve FCSPP. Instead of working with individual link delays as in case of randomized discretization which leads to the problem of error accumulation, path-delay discretization deals with path delays and discretization is performed using interval partitioning method so the problem of error accumulation is eliminated [1].

For a given path P,

$$d'(P) = \left\lfloor \frac{d(P)}{r} \lambda \right\rfloor \quad (7)$$

Where $d(P)$ denotes the delay of the path P, r is the given delay requirement and the delay requirement is bounded by an integer λ.

III. PROBLEM FORMULATION

Given network is denoted as $G(V, E)$ where V is a set of n nodes and E is a set of m links. Each edge $(u, v) \in E$ has a delay and a cost value associated with it denoted as $d(u, v)$ and $c(u, v)$ respectively. For a path P, the delay and cost of the path are denoted as $d(P)$ and $c(P)$ respectively where

$$d(P) = \sum_{(u,v) \in P} d(u, v) \quad (8)$$

$$c(P) = \sum_{(u,v) \in P} c(u, v) \quad (9)$$

$l(P)$ denotes the length (number of hops) of P and the length of the longest path in the network is denoted by L.

A path P is called a feasible path if $d(P) \leq r$ where r is the given delay requirement and the cheapest feasible path is one that not only satisfies the delay requirement but for which the cost $c(P_{s,t})$ is the minimum among all possible paths connecting the source node s and the destination node t [1].

In case of FCSPP, the cost values $c(u, v)$ are trapezoidal fuzzy numbers so the cost of the path $c(P)$ can be determined using Definition *B* of section II, equation 2 and the cheapest possible path can be determined using Definition *C* of section II and equation 3, 4 and 6.

IV. PROPOSED ALGORITHM

Get_abcd(alpha, stretch, cost)
1. $a = cost - stretch$
2. $b = a + alpha$
3. $d = cost + stretch$
4. $c = d - alpha$

Initialize (V, s, λ)
5. **for** each vertex $v \in V$, each $i \in [0, \ldots\ldots, \lambda]$
6. **Get_abcd(alpha, stretch, cost)**
7. $w[v, i] := \infty, \pi[v, i] := NIL, z[v, i] := \infty$
8. $w[s, 0] := 0, z[s, i] := 0$
9. **end for**

Relax_FPDA(u, v, i, λ)
10. $i' := \left\lfloor \frac{z[u,i] + d(u,v)}{r} \lambda \right\rfloor$
11. **Get_abcd(alpha, stretch, cost)**
12. **if** $i' \leq \lambda$ and $w[v, i'] > w[u, i] + c(u, v)$
// Compare using Definition *C* of section II and Equation 3, 4, 5 and 6.
13. $w[v, i'] := w[u, i] + c(u, v)$
14. $\pi[v, i'] := u$
15. $z[v, i'] := min\{z[v, i'],\ z[u, i] + d(u, v)\}$
16. **end if**

FPDA_Dijkstra(G, s, λ)
17. **Initialize** (V, s, λ)
18. **for** i = 0 to λ
19. $Q := V$
20. **while** $Q \neq \varphi$
21. $u := Extract\_Min(Q)$
22. **if** $w[u, i] = \infty$
23. **break** out of the **while** loop
24. **end if**
25. $Q := Q - \{u\}$
26. **for** every adjacent node v of u
27. **Relax_FPDA**(u, v, i, λ)
28. **end for**
29. **end while**
30. **end for**

FPDA(G, s)
31. $\lambda := \lambda_0$
32. **do**
33. $\lambda := 2\lambda$
34. **FPDA_Dijkstra**(G, s, λ)
35. **while** $\exists v \in d(P^v) > (1 + \epsilon)r$ // where P^v is the path with $min\{w[v, i] |\ i \in [0 \ldots\ldots \lambda]\}$

Path Delay Discretization Algorithm (PDA) was stated by Sahni et al in [1]. Here we have modified the algorithm to deal with CSPP in the fuzzy environment. Here, out of the two constraints cost and delay, we fuzzify the cost and use trapezoidal fuzzy numbers to represent the cost values. The delay values are represented as real numbers. We use a random graph generator to generate a graph and using the $rand()$ function, delay and cost values are generated randomly. Then from these randomly generated cost value, the trapezoidal fuzzy number is formed which is used in calculating the shortest path using the **FPDA_Dijkstra**() . The function **Get_abcd**() is used to determine the trapezoidal fuzzy numbers for the crisp cost values. Three two dimensional arrays are used $w[v, i], \pi[v, i]$ *and* $z[v, i]$. $w[v, i]$ stores the cost of the cheapest path P from s *to* v with $d^r(P) \leq i, \forall\ v \in V$ *and* $i \in [0, \ldots\ldots, \lambda]$. The last link of the path is stored in $\pi[v, i]$ and $z[v, i]$ is used to keep a record of the minimum delay of paths for which discretized delays are i from node s *to* v. The algorithm presented above has the same time complexity as that of the algorithm stated in [1] i.e. $O((m + n \log n)L/\varepsilon)$. We performed an experiment to determine the behavior of fuzzy path-delay discretization algorithm and observed that as the delay requirement and the network size increases, the average execution time increases which is shown in the following two figures. The reason for this behavior is that as the network size increases, the number of available path for the specified source and target increases and if the delay requirement increases, more of such paths can be explored to determine the feasible path.

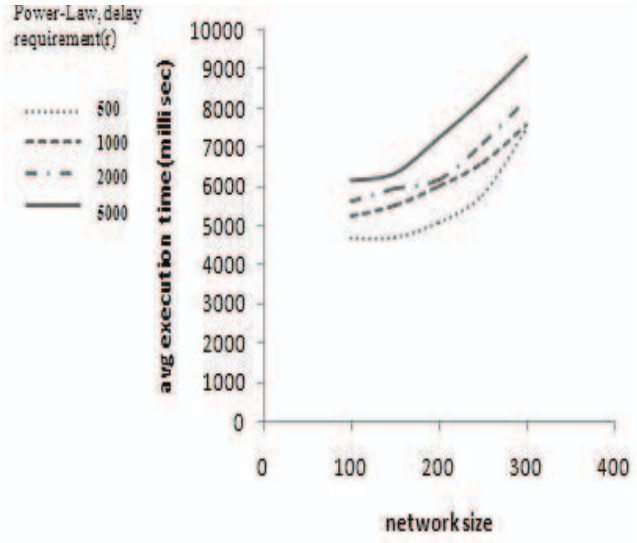


Fig. 1. Comparison of average execution time with different delay requirement.

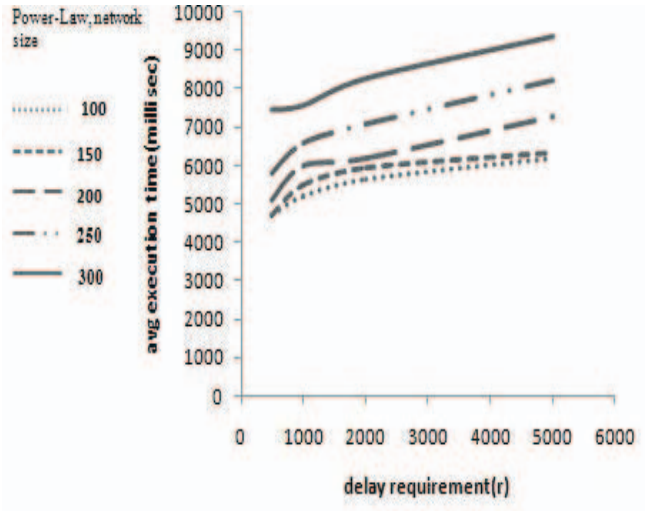


Fig. 2. Comparison of average execution time for different network size.

V. CONCLUSION

In this paper, the focus was on two aspects: uncertainty and multiple constraints. The uncertainty was taken care of by using fuzzy numbers for the representation of parameters and multiple constraints included delay and cost. An algorithm based on the path delay discretization technique suggested by Sahni et al in [1] is proposed that is capable of handling the uncertainty in the network environment by considering one of the constraints i.e. cost as a fuzzy number and delay is represented in the same manner as in [1] i.e. as a real number. The algorithm proposed follows the same steps as in PDA but with two differences: a function is used to create the trapezoidal fuzzy numbers that represent cost and the method Circumcenter of Centroid is used to rank the fuzzy numbers, with these differences the algorithm proposed is capable of determining the shortest path between a pair of source and target that satisfy the specified constraints and runs in $O((m + n\log n)L/\varepsilon)$ time.

ACKNOWLEDGMENT

The first author would like to acknowledge the IIT-BHU, Ph.D. scholarship.

REFERENCES

[1] S. Chen, M. Song and S. Sahni, "Two Techniques for Fast Computation of Constrained Shortest Paths", IEEE/ACM Transactions on Networking, Vol. 16, No. 1, February 2008.

[2] Y. Xiao, K. Thulasiraman, G. Xue and A. Juttner, "The Constrained Shortest Path Problem: Algorithmic Approaches and an Algebraic Study with Generalization". http://www.cs.elte.hu/~alpar/publications/jour/AKCE_October_25.pdf.

[3] Y. Dou, L. Zhu and H. S. Wang, "Solving the fuzzy shortest path problem using multi-criteria decision method based on vague similarity measure", Applied Soft Computing, Vol 12, pp 1621-1631, 2012.

[4] H. Joksch, "The shortest route problem with constraints", Journal of Mathematical Analysis and Applications, Vol 14, No. 1, pp 191-197, 1966.

[5] S. Sahni, "General Techniques for Combinatorial Approximation," Operations Research, Vol 25, 920-936, 1977.

[6] O. Ibarra and C. Kim, "Fast Approximation Algorithms for the Knapsack and Sum of Subsets Problems," Journal of the Association for Computing Machinery, Vol. 22, 463-468, October, 1975.

[7] R.Hassin, "Approximation Schemes for the Restricted Shortest Path Problem," Mathematics of Operation Research, Vol. 17(1), pp.36-42, 1992.

[8] A. Juttner, B. Szviatovszki, I. Mecs, and Z. Rajko, "Lagrange relaxation based method for the QoS routing problem," in Proc. IEEE INFOCOM, pp. 859–868, 2001.

[9] T. Korkmaz and M. Krunz, "Multi-constrained optimal path selection," in Proc. IEEE INFOCOM, pp. 834–843, 2001.

[10] G.Y. Handler and I. Zang. "A dual algorithm for the constrained shortest path problem". Networks, Vol. 10, pp 293–310, 1980.

[11] A. Bansal, "Trapezoidal Fuzzy Numbers (a, b, c, d): Arithmetic Behavior", International Journal of Physical and Mathematical Sciences, pp 39-44, 2011.

[12] R. Jain, "Decision making in the presence of fuzzy variables," IEEE Transactions on Systems, Man and Cybernetics, Vol. 6(10), pp. 698–703, 1976.

[13] X. Wang and E. E. Kerre, "Reasonable properties for the ordering of fuzzy quantities (I)," Fuzzy Sets and Systems, Vol. 118(3), pp. 375–385, 2001.

[14] X. Wang and E. E. Kerre, "Reasonable properties for the ordering of fuzzy quantities (II)," Fuzzy Sets and Systems, Vol. 118(3), pp. 387–405, 2001.

[15] P. Phani Bushan Rao and N. Ravi Shankar," Ranking Fuzzy Numbers with a Distance Method using Circumcenter of Centroids and an Index of Modality", Advances in Fuzzy Systems, Hindawi Publishing Corporation, 2011. doi:10.1155/2011/178308.