

[Submit to Algorithms](#)[Review for Algorithms](#)[► Journal Menu](#)[► Journal Browser](#)

## Editorial Board

- [Editorial Board](#)
- [Algorithms for Databases, and Data Structures Section](#)
- [Combinatorial Optimization, Graph, and Network Algorithms Section](#)
- [Evolutionary Algorithms and Machine Learning Section](#)
- [Parallel and Distributed Algorithms Section](#)
- [Randomized, Online, and Approximation Algorithms Section](#)
- [Algorithms Analysis and Complexity Theory Section](#)
- [Algorithms for Multidisciplinary Applications Section](#)

## Editors (3)



**Prof. Dr. Frank Werner**

[Website SciProfiles](#)

*Editor-in-Chief*

Fakultät für Mathematik, Otto-von-Guericke-Universität, 39106 Magdeburg, Germany

**Interests:** discrete optimization; operations research; scheduling; graph theory; manufacturing systems

[Special Issues and Collections in MDPI journals](#)



**Dr. Dimitris Fotakis**

[Website](#)

*Associate Editor*

Computer Science Division, School of Electrical and Computer Engineering, National Technical University of Athens, 9, Iroon Polytechniou str., 15780 Athens, Greece

**Interests:** desing and analysis of algorithms; approximation algorithms; online algorithms; algorithmic game theory

---

**Prof. Dr. Alicia Cordero Barbero**

[Website](#)

*Associate Editor*

Department of Applied Mathematics and Institute for Multidisciplinary Mathematics, Universitat Politècnica de València, 46022 Valencia, Spain

**Interests:** iterative processes; numerical analysis; dynamic analysis

**Special Issues and Collections in MDPI journals**

## Editorial Board Members (93)

Filter Editorial Board Members

Filter

---

**Prof. Dr. Faisal N. Abu-Khzam**

[Website](#)

Department of Computer Science and Mathematics, Lebanese American University, Beirut, Lebanon

**Interests:** design and analysis of algorithms; algorithmic graph theory; fixed-parameter algorithms

**Special Issues and Collections in MDPI journals:**

Special Issue in [Algorithms: Reconfiguration Problems](#)



**Dr. Tatsuya Akutsu**

[Website](#)

Bioinformatics Center, Institute for Chemical Research, Kyoto University, Gokasho, Uji, Kyoto 611-0011, Japan

**Interests:** computational biology; string and tree algorithms; complex networks

**Special Issues and Collections in MDPI journals:**

Special Issue in [Algorithms: Biological Networks](#)

Special Issue in [Algorithms: Biological Networks II](#)

---

**Dr. Luca Becchetti**

[Website](#)

Department of Computer, Control and Management Engineering Antonio Ruberti, Sapienza University of Rome, via Ariosto 25, 00185 Rome, Italy

**Interests:** design and analysis of randomized algorithms and probabilistic analysis; spectral graph theory and graph clustering; algorithms for large scale data analysis; algorithmic modelling of complex systems

**Special Issues and Collections in MDPI journals:**

Special Issue in [Algorithms: Algorithms for Large Scale Data Analysis](#)

---

**Prof. Dr. Francesco Bergadano**

[Website](#)

Dipartimento di Informatica, Università degli Studi di Torino, corso Svizzera 185, 10149 Torino, Italy

**Interests:** security and privacy; identity management; log and security analytics; data analysis; anomaly detection



**Prof. Dr. Francesco Bergadano**

[Website](#)

Department of Computer Science, University of Torino, Via Pessinetto 12, 10149 Torino, Italy

**Interests:** cyber security; security intelligence and analytics

**Special Issues and Collections in MDPI journals:**



**Prof. Dr. Philip Bille**

[Website](#)

Department of Applied Mathematics and Computer Science (DTU Compute), Technical University of Denmark, 2800 Kgs. Lyngby, Denmark

**Interests:** pattern matching; data compression; parallelism in modern computer architectures

**Special Issues and Collections in MDPI journals:**

Special Issue in **Algorithms: Data Compression Algorithms and their Applications**

---

**Prof. Dr. Vittorio Bilò**

[Website](#)

Department of Mathematics and Physics “Ennio De Giorgi”, University of Salento, Lecce, Italy

**Interests:** algorithmic game theory; computational social choice; computational complexity and design of efficient algorithms; interconnection networks

**Special Issues and Collections in MDPI journals:**

Special Issue in **Algorithms: Algorithmic Game Theory 2020**

---

**Prof. Dr. Paolo Boldi**

[Website](#)

Department of Computer Science, University of Milano, Milano 20135, Italy

**Interests:** algorithms; social network analysis; data mining; graph theory

---



**Dr. Tom Burr**

Statistical Sciences Group, Los Alamos National Laboratory Mail Stop F600, Los Alamos, NM 87545, USA

**Interests:** statistical learning; pattern recognition; multivariate calibration; measurement error modeling; time series analysis

---

**Prof. Dr. Costas Busch**

[Website](#)

Computer Science and Engineering Division, Louisiana State University, Baton Rouge, LA 70803, USA

**Interests:** distributed algorithms and data structures; communication algorithms; wireless and sensor networks; algorithmic game theory

**Special Issues and Collections in MDPI journals:**

Special Issue in **Algorithms: Sensor Algorithms**

Special Issue in **Sensors: Sensor Algorithms**

---

**Jobs at MDPI**

**For Publishers**

**For Societies**

**MDPI Initiatives**

**Institutional Open Access Program (IOAP)**

**Sciforum**

**Preprints**

**Scilit**

**SciProfiles**

**MDPI Books**

**Encyclopedia**

**MDPI Blog**

**Follow MDPI**

**LinkedIn**

**Facebook**

**Twitter**

**Subscribe to receive issue release  
notifications and newsletters from MDPI  
journals**

Select options ▼

Enter your email address...

**Subscribe**

© 1996-2020 MDPI (Basel, Switzerland) unless otherwise stated

**Disclaimer**

**Terms and Conditions**

**Privacy Policy**

[Submit to Algorithms](#)[Review for Algorithms](#)[► Journal Menu](#)[► Journal Browser](#)

## Table of Contents

### Algorithms, Volume 13, Issue 6 (June 2020) – 24 articles

- Issues are regarded as officially published after their release is announced to the [table of contents alert mailing list](#).
- You may [sign up for e-mail alerts](#) to receive table of contents of newly released issues.
- PDF is the official format for papers published in both, html and pdf forms. To view the papers in pdf format, click on the "PDF Full-text" link, and use the free [Adobe Reader](#) to open them.

Order results

Publication Date

Result details

Normal

[Show export options](#) ▼

Open Access

Article



## An Algorithm for Fuzzy Negations Based-Intuitionistic Fuzzy Copula Aggregation Operators in Multiple Attribute Decision Making

by [Stylianos Giakoumakis](#) and [Basil Papadopoulos](#)*Algorithms* **2020**, *13*(6), 154; <https://doi.org/10.3390/a13060154> - 26 Jun 2020

Viewed by 304

**Abstract** In this paper, we develop a novel computation model of Intuitionistic Fuzzy Values with the usage of fuzzy negations and Archimedean copulas. This novel computation model's structure is based on the extension of the existing operations of intuitionistic fuzzy values with some classes [...] [Read more](#).

[► Show Figures](#)

Open Access

Article



## Novel Graph Model for Solving Collision-Free Multiple-Vehicle Traveling Salesman Problem Using Ant Colony Optimization

by  **Anugrah K. Pamosoaji** and  **Djoko Budiyanto Setyohadi**

*Algorithms* **2020**, 13(6), 153; <https://doi.org/10.3390/a13060153> - 26 Jun 2020

Viewed by 377

**Abstract** In this paper, a novel graph model to figure Collision-Free Multiple Traveling Salesman Problem (CFMTSP) is proposed. In this problem, a group of vehicles start from different nodes in an undirected graph and must visit each node in the graph, following the well-known [...] [Read more.](#)

(This article belongs to the Section **Combinatorial Optimization, Graph, and Network Algorithms**)

► [Show Figures](#)

Open Access Article



## DS Evidence Theory-Based Energy Balanced Routing Algorithm for Network Lifetime Enhancement in WSN-Assisted IOT

by  **Liangrui Tang** and  **Zhilin Lu**

*Algorithms* **2020**, 13(6), 152; <https://doi.org/10.3390/a13060152> - 24 Jun 2020

Viewed by 278

**Abstract** Wireless sensor networks (WSNs) can provide data acquisition for long-term environment monitoring, which are important parts of Internet of Things (IoT). In the WSN-assisted IoT, energy efficient routing algorithms are required to maintain a long network lifetime. In this paper, a DS evidence [...] [Read more.](#)

(This article belongs to the Section **Combinatorial Optimization, Graph, and Network Algorithms**)

► [Show Figures](#)

Open Access Article



## Compression of Next-Generation Sequencing Data and of DNA Digital Files

by  **Bruno Carpentieri**

*Algorithms* **2020**, 13(6), 151; <https://doi.org/10.3390/a13060151> - 24 Jun 2020

Viewed by 276

**Abstract** The increase in memory and in network traffic used and caused by new sequenced biological data has recently deeply grown. Genomic projects such as HapMap and 1000 Genomes have contributed to the very large rise of databases and network traffic related to genomic [...] [Read more.](#)

(This article belongs to the Special Issue **2020 Selected Papers from Algorithms Editorial Board Members**)

► [Show Figures](#)

Open Access Article



## Fibers of Failure: Classifying Errors in Predictive Processes

by  **Leo S. Carlsson**,  **Mikael Vejdemo-Johansson**,  **Gunnar Carlsson** and  **Pär G. Jönsson**

*Algorithms* **2020**, 13(6), 150; <https://doi.org/10.3390/a13060150> - 23 Jun 2020

Viewed by 423

**Abstract** Predictive models are used in many different fields of science and engineering and are always prone to make faulty predictions. These faulty predictions can be more or less malignant depending on the model application. We describe fibers of failure (FiFa) [...] [Read more.](#)

(This article belongs to the Special Issue **Topological Data Analysis**)

► [Show Figures](#)

Open Access Article



## A Distributed Approach to the Evasion Problem

by  Denis Khryashchev ,  Jie Chu ,  Mikael Vejdemo-Johansson and  Ping Ji

*Algorithms* 2020, 13(6), 149; <https://doi.org/10.3390/a13060149> - 23 Jun 2020

Viewed by 373

**Abstract** The Evasion Problem is the question of whether—given a collection of sensors and a particular movement pattern over time—it is possible to stay undetected within the domain over the same stretch of time. It has been studied using topological techniques since 2006—with sufficient [...] [Read more](#).

(This article belongs to the Special Issue [Topological Data Analysis](#))

► [Show Figures](#)

Open Access

Article



## An Application of a Modified Gappy Proper Orthogonal Decomposition on Complexity Reduction of Allen-Cahn Equation

by  Chutipong Dechanubeksa and  Saifon Chaturantabut

*Algorithms* 2020, 13(6), 148; <https://doi.org/10.3390/a13060148> - 22 Jun 2020

Viewed by 303

**Abstract** This work considers model reduction techniques that can substantially decrease computational cost in simulating parametrized Allen–Cahn equation. We first employ the proper orthogonal decomposition (POD) approach to reduce the number of unknowns in the full-order discretized system. Since POD cannot reduce the computational [...] [Read more](#).

(This article belongs to the Section [Algorithms for Multidisciplinary Applications](#))

► [Show Figures](#)

Open Access

Article



## Local Comparison between Two Ninth Convergence Order Algorithms for Equations

by  Samundra Regmi ,  Ioannis K. Argyros and  Santhosh George

*Algorithms* 2020, 13(6), 147; <https://doi.org/10.3390/a13060147> - 20 Jun 2020

Viewed by 361

**Abstract** A local convergence comparison is presented between two ninth order algorithms for solving nonlinear equations. In earlier studies derivatives not appearing on the algorithms up to the 10th order were utilized to show convergence. Moreover, no error estimates, radius of convergence or results [...] [Read more](#).

Open Access

Article



## A Survey on Approximation in Parameterized Complexity: Hardness and Algorithms

by  Andreas Emil Feldmann ,  Karthik C. S. ,  Euiwoong Lee and  Pasin Manurangsi

*Algorithms* 2020, 13(6), 146; <https://doi.org/10.3390/a13060146> - 19 Jun 2020

Viewed by 411

**Abstract** Parameterization and approximation are two popular ways of coping with NP-hard problems. More recently, the two have also been combined to derive many interesting results. We survey developments in the area both from the algorithmic and hardness perspectives, with emphasis on new techniques [...] [Read more](#).

(This article belongs to the Special Issue [New Frontiers in Parameterized Complexity and Algorithms](#))

Open Access

Article



## Binary Time Series Classification with Bayesian Convolutional Neural Networks When Monitoring for Marine Gas Discharges

by  Kristian Gundersen ,  Guttorm Alendal ,  Anna Oleynik and  Nello Blaser

*Algorithms* 2020, 13(6), 145; <https://doi.org/10.3390/a13060145> - 19 Jun 2020

Viewed by 484

**Abstract** The world's oceans are under stress from climate change, acidification and other human activities, and the UN has declared 2021–2030 as the decade for marine science. To monitor the marine waters, with the purpose of detecting discharges of tracers from unknown locations, large [...] [Read more.](#)  
(This article belongs to the Special Issue [Methods and Applications of Uncertainty Quantification in Engineering and Science](#))

► [Show Figures](#)

Open Access Article



## Efficient Probabilistic Joint Inversion of Direct Current Resistivity and Small-Loop Electromagnetic Data

by [Christin Bobe](#) , [Daan Hanssens](#) , [Thomas Hermans](#) and [Ellen Van De Vijver](#)

*Algorithms* **2020**, 13(6), 144; <https://doi.org/10.3390/a13060144> - 18 Jun 2020

Viewed by 350

**Abstract** Often, multiple geophysical measurements are sensitive to the same subsurface parameters. In this case, joint inversions are mostly preferred over two (or more) separate inversions of the geophysical data sets due to the expected reduction of the non-uniqueness in the joint inverse solution. [...] [Read more.](#)

► [Show Figures](#)

Open Access Article



## Numerically Efficient Fuzzy MPC Algorithm with Advanced Generation of Prediction—Application to a Chemical Reactor

by [Piotr M. Marusak](#)

*Algorithms* **2020**, 13(6), 143; <https://doi.org/10.3390/a13060143> - 14 Jun 2020

Viewed by 522

**Abstract** In Model Predictive Control (MPC) algorithms, control signals are generated after solving optimization problems. If the model used for prediction is linear then the optimization problem is a standard, easy to solve, quadratic programming problem with linear constraints. However, such an algorithm may [...] [Read more.](#)

(This article belongs to the Special Issue [Model Predictive Control: Algorithms and Applications](#))

► [Show Figures](#)

Open Access Article



## Parallelized Swarm Intelligence Approach for Solving TSP and JSSP Problems

by [Piotr Jedrzejowicz](#) and [Izabela Wierzbowska](#)

*Algorithms* **2020**, 13(6), 142; <https://doi.org/10.3390/a13060142> - 12 Jun 2020

Viewed by 414

**Abstract** One of the possible approaches to solving difficult optimization problems is applying population-based metaheuristics. Among such metaheuristics, there is a special class where searching for the best solution is based on the collective behavior of decentralized, self-organized agents. This study proposes an approach [...] [Read more.](#)

(This article belongs to the Special Issue [Swarm Intelligence Applications for NP Hard Challenges](#))

► [Show Figures](#)

Open Access Article



## Dynamic Ring Exploration with ( $H,S$ ) View

by [Tsuyoshi Gotoh](#) , [Yuichi Sudo](#) , [Fukuhito Ooshita](#) and [Toshimitsu Masuzawa](#)

*Algorithms* **2020**, 13(6), 141; <https://doi.org/10.3390/a13060141> - 12 Jun 2020

Viewed by 425



**Abstract** The researches about a mobile entity (called agent) on dynamic networks have attracted a lot of attention in recent years. Exploration which requires an agent to visit all the nodes in the network is one of the most fundamental problems. While the exploration [...] [Read more.](#)

(This article belongs to the Special Issue [Distributed Computing Theory, Systems, Algorithms, and Data Structures](#))

► [Show Figures](#)

Open Access

Editorial



## Special Issue on Ensemble Learning and Applications

by [Panagiotis Pintelas](#) and [Ioannis E. Livieris](#)

*Algorithms* **2020**, *13*(6), 140; <https://doi.org/10.3390/a13060140> - 11 Jun 2020

Viewed by 441

**Abstract** During the last decades, in the area of machine learning and data mining, the development of ensemble methods has gained a significant attention from the scientific community. Machine learning ensemble methods combine multiple learning algorithms to obtain better predictive performance than could be [...] [Read more.](#)

(This article belongs to the Special Issue [Ensemble Algorithms and Their Applications](#))

Open Access

Article



## Optimization Algorithms for Detection of Social Interactions

by [Vincenzo Cutello](#), [Georgia Fargetta](#), [Mario Pavone](#) and [Rocco A. Scollo](#)

*Algorithms* **2020**, *13*(6), 139; <https://doi.org/10.3390/a13060139> - 11 Jun 2020

Viewed by 556

**Abstract** Community detection is one of the most challenging and interesting problems in many research areas. Being able to detect highly linked communities in a network can lead to many benefits, such as understanding relationships between entities or interactions between biological genes, for instance. [...] [Read more.](#)

(This article belongs to the Special Issue [Algorithms for Graphs and Networks](#))

► [Show Figures](#)

Open Access

Article



## Late Acceptance Hill-Climbing Matheuristic for the General Lot Sizing and Scheduling Problem with Rich Constraints

by [Andreas Goerler](#), [Eduardo Lalla-Ruiz](#) and [Stefan Voß](#)

*Algorithms* **2020**, *13*(6), 138; <https://doi.org/10.3390/a13060138> - 09 Jun 2020

Viewed by 486

**Abstract** This paper considers the general lot sizing and scheduling problem with rich constraints exemplified by means of rework and lifetime constraints for defective items (GLSP-RP), which finds numerous applications in industrial settings, for example, the food processing industry and the pharmaceutical industry. To [...] [Read more.](#)

(This article belongs to the Special Issue [Optimization Algorithms for Allocation Problems](#))

► [Show Figures](#)

Open Access

Article



## Sparse Logistic Regression: Comparison of Regularization and Bayesian Implementations

by [Mattia Zanon](#), [Giuliano Zambonin](#), [Gian Antonio Susto](#) and [Seán McLoone](#)

*Algorithms* **2020**, *13*(6), 137; <https://doi.org/10.3390/a13060137> - 08 Jun 2020

Viewed by 458

**Abstract** In knowledge-based systems, besides obtaining good output prediction accuracy, it is crucial to understand the subset of input variables that have most influence on the output, with the goal of gaining deeper insight into the underlying process. These requirements call for logistic model [...] [Read more.](#)

(This article belongs to the Special Issue **Classification and Regression in Machine Learning**)

► **Show Figures**

Open Access Article



## Improved Convergence Speed of a DCD-Based Algorithm for Sparse Solutions

by **Zhi Quan** and **Shuhua Lv**

*Algorithms* **2020**, *13*(6), 136; <https://doi.org/10.3390/a13060136> - 04 Jun 2020

Viewed by 452

**Abstract** To solve a system of equations that needs few updates, such as sparse systems, the leading dichotomous coordinate descent (DCD) algorithm is better than the cyclic DCD algorithm because of its fast speed of convergence. In the case of sparse systems requiring a [...] **Read more.**

► **Show Figures**

Open Access Feature Paper Article



## A Recursive Least-Squares Algorithm for the Identification of Trilinear Forms

by **Camelia Elisei-Iliescu**, **Laura-Maria Dogariu**, **Constantin Paleologu**, **Jacob Benesty**, **Andrei-Alexandru Enescu** and **Silviu Ciochină**

*Algorithms* **2020**, *13*(6), 135; <https://doi.org/10.3390/a13060135> - 01 Jun 2020

Viewed by 519

**Abstract** High-dimensional system identification problems can be efficiently addressed based on tensor decompositions and modelling. In this paper, we design a recursive least-squares (RLS) algorithm tailored for the identification of trilinear forms, namely RLS-TF. In our framework, the trilinear form is related to the [...] **Read more.**

► **Show Figures**

Open Access Article



## Study of Quasi-Static Magnetization with the Random-Field Ising Model

by **Roman Gozdur**

*Algorithms* **2020**, *13*(6), 134; <https://doi.org/10.3390/a13060134> - 29 May 2020

Viewed by 533

**Abstract** The topic of this paper is modeling based on Hamiltonian spin interactions. Preliminary studies on the identification of quasi-static magnetizing field in a magnetic system were presented. The random-field Ising model was then used to simulate the simplified ferromagnetic structure. The validation of [...] **Read more.**

(This article belongs to the Special Issue **Algorithms for Diagnostics and Nondestructive Testing**)

► **Show Figures**

Open Access Article



## Metric Embedding Learning on Multi-Directional Projections

by **Gábor Kertész**

*Algorithms* **2020**, *13*(6), 133; <https://doi.org/10.3390/a13060133> - 29 May 2020

Viewed by 565

**Abstract** Image based instance recognition is a difficult problem, in some cases even for the human eye. While latest developments in computer vision—mostly driven by deep learning—have shown that high performance models for classification or categorization can be engineered, the problem of discriminating similar [...] **Read more.**

(This article belongs to the Special Issue **Bio-Inspired Algorithms for Image Processing**)

► **Show Figures**

## Short-Term Wind Speed Forecasting Using Statistical and Machine Learning Methods

by  **Lucky O. Daniel** ,  **Caston Sigauke** ,  **Colin Chibaya** and  **Rendani Mbuva**

*Algorithms* **2020**, *13*(6), 132; <https://doi.org/10.3390/a13060132> - 26 May 2020

Viewed by 636

**Abstract** Wind offers an environmentally sustainable energy resource that has seen increasing global adoption in recent years. However, its intermittent, unstable and stochastic nature hampers its representation among other renewable energy sources. This work addresses the forecasting of wind speed, a primary input needed [...] [Read more](#).

► [Show Figures](#)

## Unsupervised Text Feature Selection Using Memetic Dichotomous Differential Evolution

by  **Ibraheem Al-Jadir** ,  **Kok Wai Wong** ,  **Chun Che Fung** and  **Hong Xie**

*Algorithms* **2020**, *13*(6), 131; <https://doi.org/10.3390/a13060131> - 26 May 2020

Viewed by 585

**Abstract** Feature Selection (FS) methods have been studied extensively in the literature, and there are a crucial component in machine learning techniques. However, unsupervised text feature selection has not been well studied in document clustering problems. Feature selection could be modelled as an optimization [...] [Read more](#).

(This article belongs to the Special Issue [Memetic Algorithms for Solving Very Complex Optimization Problems](#))

► [Show Figures](#)

[Show export options](#) ▼

Displaying articles 1-24

Previous Issue  
[Volume 13, May](#)

Next Issue  
[Volume 13, July](#)

### Further Information

[Article Processing Charges](#)

[Pay an Invoice](#)

[Open Access Policy](#)

[Contact MDPI](#)

[Jobs at MDPI](#)

### MDPI Initiatives

[Institutional Open Access Program \(IOAP\)](#)

[Sciforum](#)

[Preprints](#)

[Scilit](#)

### Guidelines

[For Authors](#)

[For Reviewers](#)

[For Editors](#)

[For Librarians](#)

[For Publishers](#)

[For Societies](#)

### Follow MDPI

[LinkedIn](#)

[Facebook](#)

[Twitter](#)

[SciProfiles](#)

[MDPI Books](#)

[Encyclopedia](#)

[MDPI Blog](#)

Subscribe to receive issue release  
notifications and newsletters from MDPI  
journals

Acoustics  
Actuators  
Administrative Sciences  
Aerospace  
Agriculture

Enter your email address...

**Subscribe**

© 1996-2020 MDPI (Basel, Switzerland) unless otherwise stated

[Disclaimer](#)

[Terms and Conditions](#)

[Privacy Policy](#)

## Article

# Novel Graph Model for Solving Collision-Free Multiple-Vehicle Traveling Salesman Problem Using Ant Colony Optimization

Anugrah K. Pamosoaji <sup>1,\*</sup> and Djoko Budiyo Setyohadi <sup>2</sup><sup>1</sup> Fakultas Teknologi Industri, Universitas Atma Jaya Yogyakarta, Yogyakarta 55281, Indonesia<sup>2</sup> Magister Informatika, Universitas Atma Jaya Yogyakarta, Yogyakarta 55281, Indonesia; djoko.bdy@gmail.com

\* Correspondence: kusumo\_pamosoaji@mail.uajy.ac.id; Tel.: +62-274-487711 (ext. 3240)

Received: 28 April 2020; Accepted: 23 June 2020; Published: 26 June 2020

**Abstract:** In this paper, a novel graph model to figure Collision-Free Multiple Traveling Salesman Problem (CFMTSP) is proposed. In this problem, a group of vehicles start from different nodes in an undirected graph and must visit each node in the graph, following the well-known Traveling Salesman Problem (TSP) fashion without any collision. This paper's main objective is to obtain free-collision routes for each vehicle while minimizing the traveling time of the slowest vehicle. This problem can be approached by applying speed to each vehicle, and a novel augmented graph model can perform it. This approach accommodates not only the position of nodes and inter-node distances, but also the speed of all the vehicles is proposed. The proposed augmented graph should be able to be used to perform optimal trajectories, i.e., routes and speeds, for all vehicles. An ant colony optimization (ACO) algorithm is used on the proposed augmented graph. Simulations show that the algorithm can satisfy the main objective. Considered factors, such as limitation of the mission successfulness, i.e., the inter-vehicle arrival time on a node, the number of vehicles, and the numbers of vehicles and edges of the graph are also discussed.

**Keywords:** Traveling salesman problem; collision-free trajectory; augmented graph; augmented edge; Ant colony optimization; multiple-vehicles system.

## 1. Introduction

Distribution systems involving autonomous vehicles, such as automated guided vehicles (AGV), are an interesting research topic that continuously grows in the operational research area. Routing and scheduling have been dominating issues to explore. Particularly, in applications involving multiple vehicles, vehicle routing problem (VRP) and its variants are well-known approach to solve client services. Such problems are usually generalized by the traveling salesman problem (TSP)[1–5]. Many approaches to solve complexity in TSP are explored, such as min-k-SCCP [6–9].

In general, many scholars are focused on the minimization of cost, especially with regard to the required processing time [1,10–15]. However, in the modern distribution systems, other issues, such as collision, appear [11,14]. Particularly, for instance, in warehouses, cross-sections always exist in their layout. This condition leads to an unsafe situation caused by the high possibility of collision. Numerous researches have been reported, such as conflict-free routing for material handling vehicles [10–14] and distribution system [15–18]. Thus, this issue emerges in studies on avoiding collision scheduling. Mostly, artificial intelligence (AI)-based methods were proposed to solve the problem [1,14–18].

Based on our investigation, less of the publications address a problem of asymmetric TSP (ATSP). In this problem, an individual (vehicle, human, etc.) is required to travel from node-to-node on a not-fully connected graph, where each node can be visited only once by minimizing a given cost function. This type of problem was initially analyzed by a simple problem such as TSP, which has been explored for decades [15,16,18–24]. At the first appearance, this problem was overcome by simple computational techniques, such as the ascent method [19] or taboo parallel search method [20]. However, as more variations of TSP appeared, the use of these computational techniques is inadequate, especially because of great computational efforts. A particular complex variant of TSP is Multiple TSP (MTSP), which is proven as NP-hard [10,25–27]. In the MTSP, each node is allowed to be visited only once. Furthermore, the order of the routes of each agent is free of constraint. Consequently, this scenario leads to split routes, i.e., there is no intercepted route among agents. Several methods were proposed, i.e., Ant Colony Optimization (ACO) [17,27,28], Bee Colony Optimization (BCO) [18], neural network [21], branch-and-bound algorithm [13], and so forth.

We introduce a variant of MTSP called Collision-Free Multiple Traveling Salesman Problem (CFMTSP). Unlike the typical MTSP, this problem considers a particular case where more than one vehicle can visit the same node to establish some activities. Similar to single TSP, each vehicle has to visit all nodes only once. However, unlike the typical MTSP, each node is allowed to be visited by each vehicle only once, and for a time interval, only a vehicle can occupy the node. In CFMTSP, we consider the mission that all the vehicles must accomplish their own TSP. Our objective is to minimize the completion time of the slowest vehicle. A side-effect problem arises, i.e., collision avoidance at each visited node, especially which has more than one edge. Collision issues have appeared in numerous works [29–36]. The typical definition of collision considers only vehicle-to-vehicle distance [10–14,29]. Here, each vehicle has a “safety area” around its body that cannot be occupied by other vehicles. However, under particular motion directions, the safety area cannot guarantee the safety, since the vehicle moves faster than the minimum allowable speed toward another vehicle. This problem was coined in the work of Fraichard and Asama [30] by introducing an “inevitable collision states”, i.e., a set of configurations (position and direction) of a vehicle that makes the vehicle fail to avoid collision with any object. A successful development of the framework of predicting the collision was introduced in Reference [31], and is called the “reciprocal velocity obstacle”. This work defines “obstacle” in a more precise manner, by considering not only the position of the obstacle but also the velocity of the vehicle and the obstacle. Therefore, the collision time can be predicted [32–36].

The typical MTSP is mostly solved by utilizing a conventional graph model consisting of a set of nodes and a set of edges. However, such a graph model cannot be used to handle the issue of the collision, since the model only describes the information of position. In our work, we propose a novel graph model that contains information on position and speed options. Speed, together with node-to-node distance, is useful to determine the minimum arrival time difference between any two vehicles on a node, which in turn, can be used as a collision indicator. We assume that the safety area on each node corresponds to its minimum allowable arrival time difference. In order to find the optimal solution, we utilize an Ant Colony Optimization (ACO) algorithm on the proposed graph model, such that the solution is not only a sequence of visited nodes but also the speed that is applied for the node-to-node trip. The typical algorithm uses a single species of ant to find the optimal single vehicle’s route search. However, it is inadequate to use that single species to solve CFMTSP. Therefore, we propose multiple species of ants, as each ant represents a specific vehicle. The main feature of such an ACO algorithm is in the assumption that each ant only recognizes phomone trails from other ants in the same species.

Based on the previous study, the problem that is similar to the CFMTSP has never been explored yet. The closest research was reported in References [13,37,38]. The research in Reference [13] focused on finding the route and speed for the Vehicle Routing Problem (VRP). However, this method uses an assumption of independent speed choice between any two edges, which is mechanically unrealistic because of the existence of acceleration constraints (see [36]). The issue of collision in MTSP appeared in References [37–39]. In Reference [37], a fleet of vehicles serves all

nodes in an overlapped time-based batch. Accordingly, this problem may contain the possibility of collision, as well, since any two vehicles can visit the same node in an overlapped time. In Reference [38], a variant of TSP, namely Close Enough Traveling Salesmen Problem (CETSP), is presented. This work considers the node more as an area with a predetermined radius rather than the nodes as points. Moreover, the area is allowed to be occupied by only one vehicle at any time. However, similar to Reference [13], the node-to-node travel time in both articles is assumed to be predetermined. This assumption is practically not realistic, because a single (autonomous) vehicle is controlled automatically by a computer-based control system [39]. The control system has to send some speed instructions to the AGV while traveling some paths [35]. Several published algorithms did not consider speed selection, because they were assumed that some constant speeds are applied to the paths [25–27].

The organization of this paper is described as follows. Section 2 introduces several terminologies needed to support the analysis and discussions. Section 3 describes the problem statement. Section 4 explains the proposed methods, including the proposed augmented graph and the application of ACO on the graph. Section 5 reveals the simulation results and discussions. Finally, Section 6 concludes the overall work and describes future works.

## 2. Preliminaries

Let  $G(V, E)$  be an undirected graph, where  $V = \{v_i\}_{i=1}^{N_v}$  denotes a set of  $N_v$  nodes, and  $E = \{e_{i,j}\}$  is a set of edges connecting two nodes  $v_i$  and  $v_j$ , i.e.,  $e_{i,j} = \{(v_i, v_j) | i, j \in \{1, 2, \dots, N_v\}\}$ . In addition, let  $B = \{b_{i,j}\} \in \{0, 1\}^{N_v \times N_v}$  be an adjacency matrix of  $G$ , i.e., the matrix that describes the connectivity of any pair of nodes in  $V$ , where  $b_{i,j}$  is assigned to one if  $v_i$  and  $v_j$ ,  $i \neq j$ , are connected and zero elsewhere. Let  $S = \{s_l | l \in \{1, 2, \dots, N_s\}\}$  be a set of  $N_s$  speed options  $s_l$  to be applied to all vehicles at any node in  $G$ . Suppose that there exists a group of  $N_u$  vehicles that are assigned to visit each node in  $G$ .

**Definition 1.** *Route and sub-route.* A sub-route from  $v_i$  to  $v_j$  is defined as  $e_{i,j}$ , and a route is defined as a sequence of sub-routes, i.e.,  $\Xi = \{e_{i,j} | i, j \in \{1, 2, \dots, N_v\}\}$  that begins from the start node  $v_{i,start} \in V$  to the end node  $v_{i,end} \in V$ . The sub-route and route that are traveled by the  $k$ -th vehicle,  $k \in \{1, \dots, N_u\}$ , are denoted as  $e_{i,j}^k \in E$  and  $\Xi^k$ , respectively.

**Definition 2.** *Trajectory and sub-trajectory.* A sub-trajectory  $\psi_{i,j,l}$  is defined as a pair of sub-routes and speed option  $(e_{i,j}, s_l)$ , or in other words, the sub-route from the  $i$ -th node to the  $j$ -th node by applying the  $l$ -th speed option. A trajectory is a sequence of sub-trajectories, denoted as  $\Psi = \{\psi_{i,j,l}\}$  connecting the start node  $v_{i,start} \in V$  to the end node  $v_{i,end} \in V$ . The sub-trajectory and trajectory that are traveled by the  $k$ -th vehicle are denoted as  $\psi_{i,j,l}^k$  and  $\Psi^k$ , respectively.

**Definition 3.** *Arrival time.* Arrival time of the  $k$ -th vehicle to the  $i$ -th node, denoted as  $t_i^k$ , is defined as the time when the vehicle starts to enter the node.

**Definition 4.** *Operational time.* The operational time of the  $k$ -th vehicle on the  $i$ -th node, denoted as  $t_{op}$ , is defined as the difference between the times the vehicle leaves and enters the node.

**Definition 5.** *Completion time.* A completion time,  $t_c^k$ , is the time required by the  $k$ -th vehicle to visit all nodes in  $G$ .

**Definition 6.** *Collision.* Any two vehicles are said to have not collided at the  $i$ -th node if, and only if, the arrival time difference  $|t_i^{k_1} - t_i^{k_2}| > t_{op}$ ,  $k_1 \neq k_2$ , and during the time interval each vehicle is out in the inevitable collision states defined in References [21,22].

### 3. Problem Statement

In this study, we enhanced the problem of the typical TSP (see [6,12]) to collision-free multiple TSP (CFMTSP). In this problem, each vehicle attempts to establish its individual TSP mission. The CFMTSP is described as follows. First, we need to find a complete trajectory for all vehicles, i.e.,  $\{\psi_{i,j,l}^k\}$ , such that the following function is minimized:

$$t_{\max} = \max(t_c^1, t_c^2, \dots, t_c^{N_u}), \quad (1)$$

subject to the following:

1. There is no collision between any vehicles (see Definition 6) at each node.
2. The start times,  $t_{\text{start}}^k$ , of all vehicles are zero, i.e.,  $t_{\text{start}}^1 = t_{\text{start}}^2 = \dots = t_{\text{start}}^{N_u} = 0$ .
3.  $s_l > 0$ .

Note that the second constraint is designated to be a collision indicator. If the constraint is violated, then the vehicles have collided with each other. The third constraint emphasizes that there is no delay among the start times of the vehicles.

**Assumption 1.** All vehicles start from different nodes.

**Assumption 2.** The operational times,  $t_{op}$ , for all nodes in  $G$  are assumed to be constant.

**Assumption 3.** The number of speed options, i.e.,  $N_s$ , is the same for all vehicles.

**Assumption 4.** Collisions are considered only at the nodes. The edges are assumed to be sufficiently broad, so that any vehicles passing through the same edge will not collide with each other.

**Assumption 5.** The graph  $G(V, E)$  is not a multi-graph, i.e., the number of edges between any two nodes is exactly one.

### 4. Proposed Methods

#### 4.1. Augmented Graph

##### 4.1.1. Graph Model

Solving the problem described in (1) is difficult by using typical graph  $G$ , since there is no information about the arrival time of each vehicle at each node. Consequently, the inter-vehicle collision problem is unable to solve. To address such a problem, instead of using typical graph  $G$ , we developed a novel structure of the graph, which is called an *augmented graph*, denoted by  $G^a$ . The augmented node can be constructed from the typical graph  $G$ . Let  $G^a(V^a, E^a)$  be an *augmented graph*, where  $V^a = \{\psi_{i,j,l}\}$ , as long as  $b_{i,j} = 1$ ,  $i, j \in \{1, \dots, N_v\}$  and  $l \in \{1, 2, \dots, N_s\}$ , is defined as a set of *augmented nodes* and  $E^a = \{\xi_{i_1,j_1,l_1}^{i_2,j_2,l_2}\}$ , where  $\xi_{i_1,j_1,l_1}^{i_2,j_2,l_2} = (\psi_{i_1,j_1,l_1}, \psi_{i_2,j_2,l_2})$  is *augmented edge*, i.e., start and end connected sub-trajectories pairs  $\psi_{i_1,j_1,l_1}$  and  $\psi_{i_2,j_2,l_2}$ . Note that the notation  $\xi_{i_1,j_1,l_1}^{i_2,j_2,l_2}$  implies that  $\psi_{i_2,j_2,l_2}$  must be a successor of  $\psi_{i_1,j_1,l_1}$ . Therefore, it is required that  $i_2 = j_1$ .

Figure 1 visualizes the proposed  $G^a$ ,  $V^a$ , and  $E^a$ .  $G^a$  expands the typical  $G$  from the node-to-node relation into transition-to-transition relation. In the typical graph  $G$ , the edges are weighted by node-to-node distance  $L_{i,j}$  while in the augmented graph  $G^a$ , the augmented edges are weighted by acceleration, whose formulation is conducted using start and final speeds the node-to-node distance (see Equation (5)).



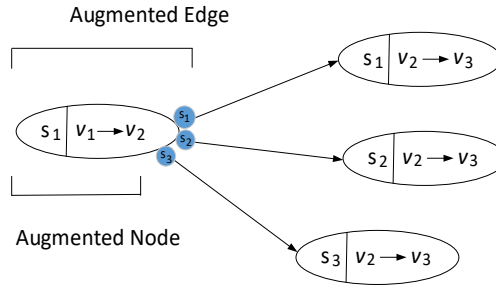


Figure 1. Augmented nodes and edges.

Figure 2 shows that, for a node-to-node trajectory, there are some speed alternatives to apply. Therefore, the augmented edge between any pair of augmented nodes can represent information of acceleration and traveling time, for instance, the transition from  $\psi_{1,2,1}$  to  $\psi_{2,3,2}$ . The applied initial and target speeds at the endpoint of  $\psi_{1,2,1}$  are  $s_1$  and  $s_2$ , respectively. Therefore, the uniform acceleration  $a_{1,2}$  along the augmented edge is formulated as following:

$$a_{i,j} = \frac{(s_j)^2 - (s_i)^2}{2L_{i,j}}, \quad (2)$$

where  $L_{i,j}$  is the length of the edge  $e_{1,2}$ , and it is plotted to the transition from  $\psi_{1,2,1}$  to  $\psi_{2,3,2}$ .

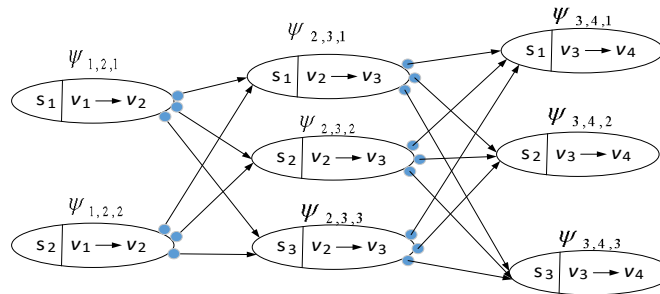


Figure 2. The proposed augmented graph.

Furthermore, the traveling time,  $t$ , related to the acceleration in Equation (2) is formulated as follows:

$$t = \begin{cases} \frac{L_{i,j}}{s_i}, & \text{if } a = 0, \\ \frac{-s_i + \sqrt{(s_i)^2 + 2aL_{i,j}}}{a_{i,j}}, & \text{otherwise.} \end{cases} \quad (3)$$

#### 4.1.2. Additional Adjacency Matrix

We introduced some supporting matrices to support the proposed algorithm. First of all, we introduced *edge matrix*, denoted as  ${}^eB = \{{}^eb_{i,j} | i, j \in \{1, 2, \dots, N_v\}\}$ , whose dimension is the same with the adjacency matrix  $B$ . Let  ${}^qe$ , where  $q$  is the identifier of an edge whose value can be determined by Algorithm 1, be defined as the edge identifier of each element of  $B$ , i.e.,  $b_{i,j}$  that has the value of 1. Therefore, we get the following:

$${}^eb_{i,j} = \begin{cases} {}^qe, & \text{if } b_{i,j} > 1, \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

where  $i, j \in \{1, 2, \dots, N_v\}$ .

---

**Algorithm 1** Determine the index of edge matrix

```

1:  Input:  $B$ 
2:   $q = 0$ ;
3:  For  $i = 1$  to  $N_v$ 
4:      For  $j = 1$  to  $N_v$ 
5:          If  $i \neq j$  and  $b_{i,j} = 1$ , then  $q = q + 1$  and  ${}^q e = q$ .
6:      End
7:  End
8:  Output: all  $\{{}^q e\}$ .

```

For instance, suppose that we have the following adjacency matrix of a graph  $G$  whose number of nodes is 5, as follows.

$$B = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} \quad (5)$$

Therefore, by using (4), we have the following:

$${}^eB = \begin{bmatrix} 0 & {}^1e & 0 & {}^2e & 0 \\ 0 & 0 & {}^3e & 0 & 0 \\ {}^4e & 0 & 0 & {}^5e & 0 \\ 0 & {}^6e & {}^7e & 0 & 0 \\ 0 & {}^8e & {}^9e & {}^{10}e & 0 \end{bmatrix}. \quad (6)$$

Furthermore, we use the index of  ${}^qe$ , i.e.,  $q$  as the index of row of another adjacency matrix named *trajectory adjacency matrix*, denoted as  ${}^{\psi}B$ . Each row and each column of this matrix represent edge  ${}^qe$  and speed  $s_l$ , respectively. Let  ${}^p\psi$  be the element of  ${}^{\psi}B$ , where we have the following:

$$p(q, l) = (q - 1)N_s + l, \quad (7)$$

$l \in \{1, 2, \dots, N_s\}$  and  $q$  are defined in (4). Suppose that we apply three-speed options; therefore,  $N_s = 3$ . From the adjacency matrix example in Equation (5) and its respective edge matrix in (6), we have the elements of  $T$  as follows: For edge  ${}^1e$ , we have three trajectories, i.e.,  ${}^1\psi$ ,  ${}^2\psi$ , and  ${}^3\psi$ . Since from Equations (5) to (7),  ${}^1e$  represents  $e_{1,2}$ , then  ${}^1\psi$ ,  ${}^2\psi$ , and  ${}^3\psi$  represent  $\psi_{1,2,1}$ ,  $\psi_{1,2,2}$ , and  $\psi_{1,2,3}$ , respectively. Similarly,  ${}^5e$  represents  $e_{3,4}$ . By using (10), we have trajectories  ${}^{13}\psi$ ,  ${}^{14}\psi$ , and  ${}^{15}\psi$ . Therefore, according to Equations (5)–(7),  ${}^{13}\psi$ ,  ${}^{14}\psi$ , and  ${}^{15}\psi$  represent  $\psi_{3,4,1}$ ,  $\psi_{3,4,2}$ , and  $\psi_{3,4,3}$ , respectively.

The last adjacency matrix is the *augmented-edge adjacency matrix*, denoted by  ${}^{\xi}B$ , where the row and column are indexed by the index of  ${}^p\psi$ , i.e.,  $p$ . Let  ${}^h\xi$  be the element of  ${}^{\xi}B$ , whose value is not zero and be associated to  $\xi_{i_1,j_1,l_1}^{i_2,j_2,l_2}$  by applying Algorithm 2.

---

**Algorithm 2** Determining the index of  $\xi$ :

```

1:  Inputs:  $\psi_B, p_1, p_2$ 
2:  Output:  $\xi_B, {}^h\xi$ 
3:   $h=0$ 
4:  For  $i=1, \dots, |\psi_B|$  // loop for all rows
5:      For  $j=1, \dots, |\psi_B|$  // loop for all columns
6:           $c_1 = \text{row}_{\psi}({}^i\psi); c_2 = \text{row}_{\psi}({}^j\psi)$ 
7:          If  $c_1 \neq c_2$ 
8:              If  $\text{end}_e({}^{c_1}e) = \text{start}_e({}^{c_1}e)$ 
9:                   $h=h+1$ 
10:                      $\xi_{b_{i,j}} = h$ 

```

---

```

11:            ${}^h\xi = h$ 
12:       Else
13:            ${}^\xi b_{i,j} = 0$ 
14:       End
15:   Else
16:        ${}^\xi b_{i,j} = 0$ 
17:   End if
18: End for
19: End for

```

---

The functions  $row_\psi(x)$ ,  $start_e(x)$ , and  $end_e(x)$  return the row index belonged to the row in  ${}^\psi B$  containing  $x$ , the indexes of start and end node in  ${}^e B$ , respectively. In the previous example, there are ten edges and three speed options. The number of trajectories is 30 and the dimension of  ${}^\xi B = \mathbb{Z}^{30 \times 30}$ . Until this step, we have the following adjacency matrices:  ${}^e B$ ,  ${}^\psi B$ , and  ${}^\xi B$ . Suppose that we are given  ${}^h\xi$ ; then, by using the map described in  ${}^\xi B$ , we obtain  ${}^{p_1}\psi$  and  ${}^{p_2}\psi$ , where  $p_1$  and  $p_2$  are the indexes of the row and column of  ${}^\xi B$  associated to  ${}^h\xi$ . After that, we check the map in  ${}^\psi B$ , and we obtain the edge  ${}^qe$  and speed option  $s_l$  for each  ${}^{p_1}\psi$  and  ${}^{p_2}\psi$ .

#### 4.2. Ant Colony Optimization

Ant Colony Optimization (ACO) was first introduced in Reference [28]. This algorithm is powerful for routing problems such as traveling salesman problem (TSP), vehicle routing problem (VRP), and their variants [17,27]. The algorithm mimics the behavior of a colony of ants in foraging activities. Suppose that, at the beginning, the colony has no information about the location of the food source. An ant system (AS) consists of a set of artificial ants which perform foraging activities, from their nest to a food source. Some ants begin to move randomly to any direction and deposit a chemical called “pheromone”, whose trails will be traced by other ants. This process is iterated such that the optimal route is that with the maximum number of pheromone trails.

In this study, we used an ACO algorithm for finding trajectories for a multiple-vehicle system. The algorithm is different from the typical ones. It involves more than one species of ants whose pheromone the others cannot detect trails of. In the cases of CFMTSP, a species represents a vehicle. We develop an algorithm such that each species performs a collision-free trajectory, i.e., all species can prevent collision with each other. In this study, we assume that any two species are not colliding with each other if the difference of their arrival times exceeds a minimum allowable value.

Suppose that there exist  $N_u$  ant species. Each species consists of  $N_m$  ants. The  $r$ -th ant of the  $k$ -th species, denoted by  $\theta^{k,r}$ ,  $k \in \{1, 2, \dots, N_u\}$ ,  $r \in \{1, 2, \dots, N_m\}$ , represents the  $k$ -th vehicle. Each ant is assumed to be able to recognize only the trails produced by ants of the same species. Therefore, if there exists a large concentration of pheromone in a sub-route, if it comes from different species, then it is less possible for the ant to choose that sub-route as a choice. As the typical ACO algorithms, as one ant passes a sub-route, it leaves pheromone trails along the sub-routes. From now on, the others will smell the trails, and based on the largest amount of pheromones, it will choose the sub-route. Even though there are pheromones produced by other different ant species, the ant cannot recognize them. This behavior is similar to the behavior of some colonies of ants in the real world, that is, they cannot recognize the trail of other different ant species, as reported in Reference [40]. Researchers in that study have discovered that a species of ants, i.e., *Lasius nigers* (*La. nigers*), is unable to recognize the pheromone trails produced by *Novomessor cockerelli* (*N. cockerelli*) and *Linepithema humilis* (*Li. humilis*). The reason is that their pheromone trails are constructed by different chemicals.

In this model, we applied the amount of pheromone applied to the proposed augmented graph  $G^a$  to augmented edges  ${}^h\xi$ . Define  $n$ ,  ${}^h\tau^k$ , and  $\Delta {}^h\tau^{k,r}$  as the iteration, the total amount of pheromone left by the  $k$ -th ant species on augmented edges  ${}^h\xi$ , and the increase of pheromone amount left by each  $r$ -th ant of the  $k$ -th ant species, respectively. We formulate  ${}^h\tau^k$  as follows.

$$h_{\tau^k}(n) = h_{\tau^k}(n-1) + \sum_{r=1}^{N_m} \Delta h_{\tau^{k,r}}(n), \quad (8)$$

if a solution is found and the  $i$ -th and  $j$ -th nodes are the part of the augmented edge passed through by the  $k$ -th ant species, and

$$h_{\tau^k}(n) = \rho h_{\tau^k}(n-1), \quad (9)$$

where  $\rho \in (0,1]$  is defined as evaporation rate, if the search fails to find a solution. Furthermore,  $\Delta h_{\tau^{k,r}}$  is formulated as follows:

$$\Delta h_{\tau^{k,r}} = \frac{Q}{d^{k,r}}, \quad (10)$$

where  $d^{k,r}$  is the total distance traveled by the  $r$ -th ant from the  $k$ -th species, and  $Q$  is a positive constant.

We apply the probability of selecting  $h_{\xi}$ , i.e.,  $Pr(h_{\xi})$  as follows:

$$Pr(h_{\xi}) = \frac{(h_{\eta})^{\beta} (h_{\tau^k})^{\gamma}}{\sum_{all \ h_{i\xi} \in Neighbors(h_{\xi})} (h_{i\eta})^{\beta} (h_{i\tau^k})^{\gamma}}, \quad (11)$$

Note that the selection of trajectories of more than one vehicle leads to a consequence of inter-vehicle collision-checking. Therefore, a procedure for checking the collision is developed.

Before describing the main algorithm, we define a number of variables:  $v_{i,start}^{k,r}$  and  $s_{start}^{k,r}$  are the initial position and speed, respectively.  $L_{unv}^k$ ,  $L_e^k$ ,  $L_{\psi}^k$ ,  $L_{\xi}^k$ , and  $L_{\xi,sel}^k$  are the lists of unvisited nodes, collection of edges, collection of trajectories, and collection of augmented-edges, the sequence of selected augmented-edges from the start to end nodes, respectively.  $L_{vis}^i$  is a collection for each node  $i \in \{1,2, \dots, N_v\}$  that stores information of the  $k$ -th vehicle that has visited the node and its associated arrival time,  $t_i^k \cdot \Psi_{best}^k$  is the best trajectory performed by the  $k$ -th species.

In Line 1 of Algorithm 3, the input is the graph  $G$ , initial positions  $v_{i,start}^{k,r}$  and speed  $s_{start}^{k,r}$  of each  $r$ -ant of the  $k$ -th ant species. Note that  $r$  also represents the index of iteration. Consequently,  $N_m$  represents the number of iterations. In Lines 3 and 4, all the required adjacency matrices are constructed, and  $L_{unv}^k$  for all ant species is set to be empty.

The searching process starts from Line 5. For each iteration, trajectories are constructed for each species. In Line 7, some required initializations are performed, such as the initial amount of pheromones. The values are set randomly small to prevent division-by-zero at the beginning at the process. The next processes are focused on identifying the augmented edge of the current occupied node,  $v_{curr}^k$ .

The function  $CreateEdgeList(v_{curr}^k)$  is purposed to extract all edges  ${}^qe$  in  $G$  whose start nodes are  $v_{curr}^k$ . In this process, the edge matrix  ${}^eB$  is used. The resulted  ${}^qe$  is then pushed to  $L_e^k$ .

The next step is to extract the sub-trajectories,  ${}^p\psi$ , whose edges are  ${}^qe$ . However, two conditions make the extraction fail. First, it is possible that the end node of the edge  ${}^qe$ , i.e.,  $end_e({}^qe)$ , was visited. Therefore, the availability of the end node of the edge must be checked (Line 13). Second, even though  $end_e({}^qe)$  is available, if there exists another ant occupying the node such that the second constraint is violated, then the process is continued to the next edge. This process is revealed in Lines 14–16.

If the end node of the edge  ${}^qe$  has not been visited yet and has no collision issue, then the function  $CreateTrajectoriesList(L_e^k, {}^qe)$  is executed (Line 18). The  $CreateTrajectoriesList(L_e^k, {}^qe)$  uses trajectory adjacency matrix  ${}^pB$  as reference to find the correct trajectories that are spanned by the  $q$ -th edge in  $L_e^k$ . If the end node of the edge  ${}^qe$  has been visited previously, the process will check the other edges. If there is no edge available, then it can be concluded that a complete trajectory is failed to found. Therefore, the process is continued to new iteration (Lines 29–32). In Line 30, the amount of pheromone trails is reduced by calling  $ReducePheromoneTrailsAmount(L_{\xi}^k, h_{\tau^k})$ , which applies Equation (12).

**Algorithm 3** Main Algorithm

```

1: Inputs:  $G$ ,  $v_{i,start}^{k,r}$ ,  $s_{start}^{k,r}$  for all  $k \in \{1,2, \dots, N_u\}, r \in \{1,2, \dots, N_m\}$ ,
2: Perform  ${}^eB$ ,  ${}^\psi B$ , and  ${}^\xi B$ .
3:  $\Psi_{best}^k \leftarrow \{\}$ , for all  $k \in \{1,2, \dots, N_u\}$ .
4:  $L_{unv}^k \leftarrow \{\}_{i=1}^{N_v}$ , for all  $k \in \{1,2, \dots, N_u\}$ .
5: For  $r=1$  to  $N_m$ 
6:    $v_{curr}^k = v_{i,start}^{k,r}$ 
7:   Initialize( $v_{curr}^k, s_{start}^{k,r}$ );
8:    $L_{unv}^k \leftarrow \{v_i\}_{i=1}^{N_v}$ ,  $L_e^k \leftarrow \{\}$ ,  $L_\psi^k \leftarrow \{\}$ ,  $L_\xi^k \leftarrow \{\}$ ,  $L_{\xi,sel}^k \leftarrow \{\}$ ,  $L_{vis}^i \leftarrow \{\}$ .
9:   For each  $k$ -th ant species
10:    While  $L_{unv}^k \neq \{\}$ 
11:       $L_e^k \leftarrow CreateEdgeList(v_{curr}^k)$ .
12:      For each edge in  $L_e^k$ , i.e.,  ${}^qe$ ,
13:        If  $end_e({}^qe) \in L_{unv}^k$ ,
14:          If  $IsCollided(L_{vis}^i, end_e({}^qe))$  is TRUE
15:            continue;
16:          End
17:           ${}^p\psi = CreateSubTrajectoriesList(L_e^k, {}^qe)$ .
18:           $L_\psi^k \leftarrow {}^p\psi$ .
19:          For each  $p$ -th trajectory in  $L_\psi^k$ , i.e.,  ${}^p\psi$ ,
20:             $L_\xi^k \leftarrow CreateAugmentedEdgesList(L_\psi^k, {}^p\psi)$ .
21:          End
22:           $L_{\xi,sel}^k \leftarrow SelectAugmentedEdge(L_\xi^k)$ .
23:           $\{t_{i,max}^k\} \leftarrow CalculateMaxArrivalTime(L_{\xi,sel}^k)$ .
24:           $Remove(L_{unv}^k, end_e({}^qe))$ .
25:        Else
26:          continue.
27:        End
28:      End
29:      If  $L_\psi^k == \{\}$ 
30:         ${}^{h\tau^k} = ReducePheromoneTrailsAmount(L_{\xi,sel}^k, {}^{h\tau^k})$ 
31:        go to Line 5.
32:      End
33:       $L_\psi^k \leftarrow \{\}$ .
34:    End //end while
35:  End // end for
36:   ${}^{h\tau^k} = CalculatePheromoneTrailsAmount(L_{\xi,sel}^k, {}^{h\tau^k})$ , for all  $k \in \{1,2, \dots, N_u\}$ .
37:   $Pr(L_{\xi,sel}^k) = CalculateProbability(L_{\xi,sel}^k, {}^{h\tau^k})$ , for all  $k \in \{1,2, \dots, N_u\}$ .
38:   $t_{best}^k = \min(\{t_{i,max}^k\})$ 
39:   $L_{unv}^k \leftarrow \{\}_{i=1}^{N_v}$ 
40: End
41: Outputs:  $\psi_{best}^k$  with  $t_{i,max}^k = t_{best}^k$ , for all  $k \in \{1,2, \dots, N_u\}$ .

```

In Lines 19–21, the function  $CreateAugmentedEdgesList(L_\psi^k, {}^p\psi)$  is used to extract the augmented edges that is spanned by the  $p$ -th trajectory in  $L_\psi^k$ . This function uses augmented-edge adjacency matrix  ${}^\xi B$  as a reference to find the correct augmented edge that is spanned by the  $p$ -th trajectory in  $L_\psi^k$ . The result is  ${}^{h\xi}$  and is pushed to  $L_\xi^k$ , and we select the augmented edge, whose probability, calculated using Equation (14), is the largest. The selected augmented edge is then pushed into  $L_{\xi,sel}^k$  (Line 22). In addition, the maximum traveling time,  $t_{i,max}^k$ , is calculated in Line 23. Here, the

final value of  $t_{i,max}^k$  is the traveling time of the slowest ant species. Moreover, the member of  $L_{unv}^k$  that is the same to  $end_e(^qe)$  is removed (Line 24), since the  $end_e(^qe)$  is selected to be visited.

After all ants from all species complete their routes, the pheromone trails on all augmented edges are updated in Line 31 by using Equation (11). By using the current pheromone trails, the probability of all augmented edges in  $L_{\xi}^k$ ,  $k \in \{1, 2, \dots, N_u\}$  is calculated by using  $CalculateProbability(L_{\xi,sel}^k, ^h\tau^k)$  in Line 37. The global minimum,  $t_{best}^k$  is then calculated in Line 38. Finally, the output is  $\psi_{best}^k$  with  $t_{i,max}^k = t_{best}^k$  for all  $k \in \{1, 2, \dots, N_u\}$ .

## 5. Results and Discussions

We tested the performance of the proposed algorithm by conducting simulations involving graphs with various numbers of nodes, i.e., 10, 15, and 20 nodes, as shown in Figure 3–5, respectively. For each graph, four variations of connectivity were simulated to show the success of finding solutions. We denoted the graph, together with the connectivity of variations, by using the code “CONFIG A-B”, where A is the number of nodes and B is the label of a variation. For instance, the first variation of the graphs with 10, 15, and 20 nodes were denoted as CONFIG 10-1, CONFIG 15-1, and CONFIG 20-1, respectively. Note that all edges were bi-directional.

Three significant aspects were evaluated. The first aspect was the influence of minimum allowable arrival time on the solution’s existence and the convergence of the solutions. The second was the correlation between the average degrees of all nodes, the number of vehicles, and the number of nodes to the success of finding solutions. The last aspect was the accuracy of the resulted minimum traveling time of the slowest vehicle, according to the variation of the evaporating constant and its effect on the convergence of the search results.

For evaluating the first and third aspects, we established simulations for three vehicles, where the 1st, 2nd, and 3rd vehicles started from Nodes 4, 1, and 3, respectively. Each vehicle had four speed options, i.e., 0.1, 0.5, 1, and 1.5 m/s. Meanwhile, for evaluating the second aspect, we applied two until seven vehicles on each connectivity configuration. Vehicles 1 until 7 were started from Nodes 4, 1, 3, 5, 10, 7, and 8, respectively. For one simulation, we applied 3000 iterations of searching the optimal solution.

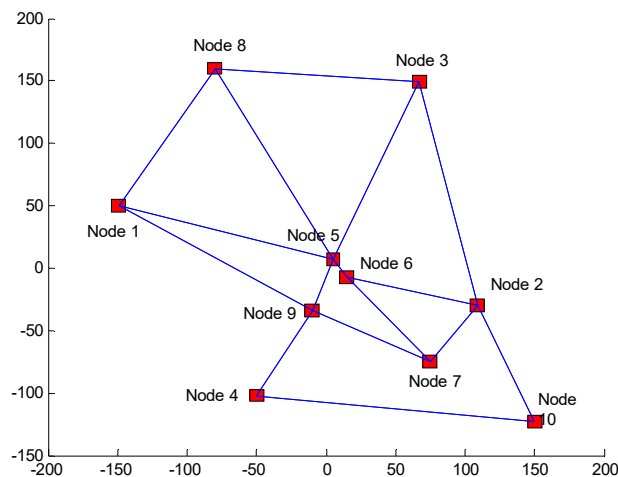


Figure 3. Graph with CONFIG 10-1 configuration (10 nodes).

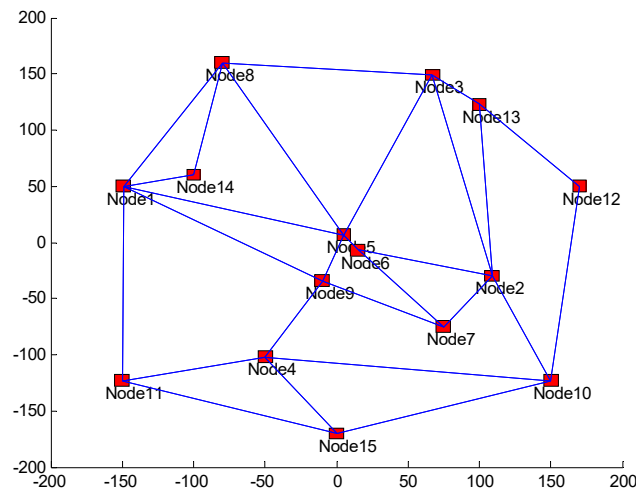


Figure 4. Graph with CONFIG 15-1 configuration (15 nodes).

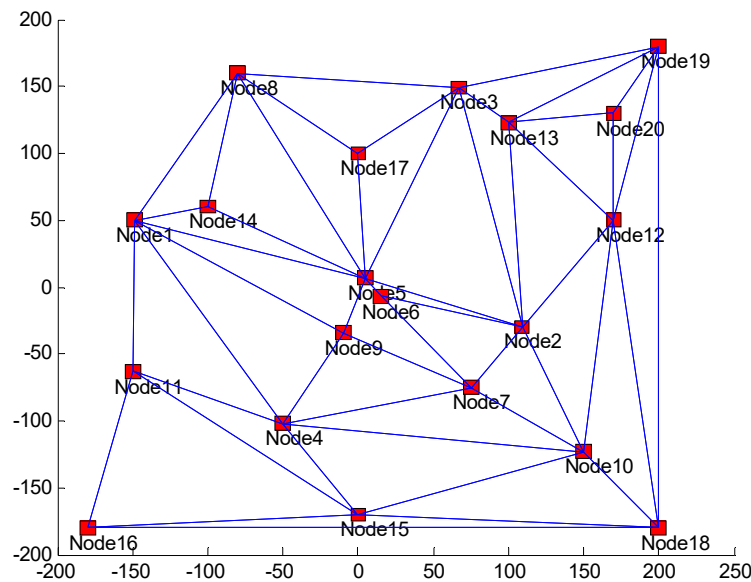


Figure 5. Graph with CONFIG 20-1 configuration (20 nodes).

### 5.1. The Effect of *Minimum Allowable Arrival Time Difference* $t_{occ}$

The first simulation set was established to evaluate the relation between the minimum allowable arrival time difference,  $t_{occ}$ , and the solution's existence. The purpose of the simulations is to confirm the hypotheses. In such simulations, we used  $\alpha = 1$  and  $\beta = 1$  and evaporate rate  $\rho = 0.95$ . We applied five values of  $t_{occ}$ , i.e., 10, 50, 100, 150, and 300 s.

Table 1; Table 2 show the results of 10 trials for varying allowable minimum occupation time  $t_{occ}$  and varying evaporate constant  $\rho$ , respectively. In each table, as shown in Column (1), for each occupation time, we established five trials. Column (1) reveals the values of  $t_{occ}$ . Column (2) is the success rate of finding a complete solution. Columns (3) and (4) show the average and standard

deviation of the maximum traveling times, respectively. Column (5) shows the minimum value of maximal traveling times.

The application of  $t_{occ}$  is evaluated for its effect on the percentage of successful trials from 10 trials, i.e.,  $n_{success}$ , average traveling time of the slowest vehicle,  $T_{avg}$ , the standard deviation of  $T_{avg}$ , i.e.,  $\sigma_T$ , and the minimum traveling time that is ever found from the ten trials, i.e.,  $T_{min}$ . For this purpose, we use a statistical correlation technique. The correlations between  $t_{occ}$  and  $n_{success}$ ,  $T_{avg}$ ,  $\sigma_T$ , and  $T_{min}$  are  $-0.99$ ,  $0.83$ ,  $0.75$ , and  $0.88$ , respectively. It can be concluded that there is a strong negative relationship between  $t_{occ}$  and  $n_{success}$  and almost-strong positive relation between  $t_{occ}$  and the other three variables. In addition, the tendency of failure can be identified by a parameter  $\sigma_T/t_{occ}$ . We found from the simulation that  $\sigma_T/t_{occ}$  is directly proportional to the probability of success in finding a solution, i.e.,  $n_{success}$ .

**Table 1.** Simulation results for various  $t_{occ}$  for three vehicles.

(1)	(2)	(3)	(4)	(5)
$t_{occ}(s)$	$n_{success}$	$T_{avg}(s)$	$\sigma_T(s)$	$T_{min}(s)$
10.00	10	1754.16	320.69	1482.50
50.00	9	2648.04	663.91	1896.30
100.00	8	3412.84	1191.16	1961.10
150.00	8	3314.01	841.83	1684.20
300.00	5	3735.32	1180.45	2851.00

**Table 2.** Simulation results for various evaporate constants at  $t_{occ} > 150$  s.

(1)	(2)	(3)	(4)	(5)
$\rho$	$n_{success}$	$T_{avg}(s)$	$\sigma_T(s)$	$T_{min}$
0.1	9	2977.72	732.50	1949.00
0.2	7	3317.33	636.00	2189.30
0.3	6	2269.77	403.38	1744.00
0.4	8	3213.81	777.59	1966.40
0.5	9	2929.99	609.94	2064.70

## 5.2. Successfulness of Finding a Solution

Since the graph is not fully connected, it is possible that there exists a situation such that any single solution is failed to find as the effect of the number of nodes, the average degree of all nodes increase, and the number of vehicles. The degree of a node is defined as the number of edges that connect to the node. We established simulations for 10, 15, and 20 nodes under  $t_{occ} = 10$  s.

### 5.2.1. Simulations for 10 Nodes

For the ten nodes cases, we applied four different configurations of connectivity, as shown in Table 3, i.e., CONFIGs 10-1, 10-2, 10-3, and 10-4. CONFIG 10-1 is the graph with the connectivity visualized in Figure 3. CONFIG 10-2 is constructed from CONFIG 10-1 with the elimination of the edge connecting Nodes 5 to 8. CONFIG 10-3 is constructed from CONFIG 10-3, with the elimination of the edge connecting Nodes 5 to 9. CONFIG 10-4 is constructed from the CONFIG 10-1, with the addition of the edges connecting Nodes 1 to 4, 3 to 6, and 4 to 7. Table 3 describes the degree of each node. The averages of the degree of nodes for CONFIG 10-1, 10-2, 10-3, and 10-4 are 3.20, 3.00, 2.80, and 4.00.



**Table 3.** Simulation results for 10 nodes.

Number of Degree of Nodes									
Node	CONFIG 10-1		CONFIG 10-2		CONFIG 10-3		CONFIG 10-4		
1	3		3		3		4		
2	4		4		4		4		
3	3		3		3		4		
4	2		2		2		4		
5	5		4		3		5		
6	3		3		3		4		
7	3		3		3		5		
8	3		2		2		3		
9	4		4		3		4		
10	2		2		2		3		
Average	3.20		3.00		2.80		4.00		
Number of Vehicles	$\varphi$	Success Rate (%)	$\varphi$	Success Rate (%)	$\varphi$	Success Rate (%)	$\varphi$	Success Rate (%)	
	3	0.09	100.00	0.10	100.00	0.11	100.00	0.08	100.00
	4	0.13	90.00	0.13	100.00	0.14	100.00	0.10	100.00
	5	0.16	100.00	0.17	100.00	0.18	100.00	0.13	100.00
	6	0.19	90.00	0.20	90.00	0.21	0.00	0.15	100.00
	7	0.22	0.00	0.23	0.00	0.25	0.00	0.18	90.00

We investigate the relationship between the nodes' average degrees and the number of vehicles to the possibility of successfulness of finding a solution. Here, we introduce a parameter  $\phi$  defined as follows:

$$\phi = \frac{N_u}{r_{avg} N_v}, \quad (12)$$

where  $r_{avg}$  is defined as the average degree of all nodes. From Table 3, it can be seen that the solution can be found for  $\phi \leq 0.19$ . It can be concluded that, from the 10 trials, at the range of the  $\phi$ , there exists at least one successful trial.

### 5.2.2. Simulations for 15 Nodes

In this simulation, we added five nodes, as follows. Nodes 11, 12, 13, 14, and 15 are located at (−150 m, −123 m), (170 m, 50 m), (100 m, 123 m), (−100 m, 60 m), and (0 m, −170 m), respectively. Moreover, four edges configurations in the new graph were established, called CONFIG 15-1, CONFIG 15-2, CONFIG 15-3, and CONFIG 15-4. The CONFIG 15-1 connectivity configuration is described in Figure 4. CONFIG 15-2 is constructed by adding an edge connecting Nodes 1 and 5 on the CONFIG 15-1. The CONFIG 15-3 configuration is performed by adding edges connecting Nodes 1 to 4, 2 to 12, and 7 to 10 to CONFIG 15-2. The CONFIG 15-4 configuration is performed by adding edges connecting nodes 4 to 7 and 3 to 6 to CONFIG 15-3. The degree of each edge in each connectivity configuration is described in Table 4.

From Table 4, it can be seen that the solution can be found for  $\phi \leq 0.08$ , which means that, at the range of the  $\phi$ , it can be concluded that, from the 10 trials, there always exists at least one successful trial.

**Table 4.** Simulation results for 15 nodes.

Node	Number of Degree of Nodes							
	CONFIG 15-1	CONFIG 15-2	CONFIG 15-3	CONFIG 15-4				
1	5	5	6	6				
2	5	5	6	6				
3	4	4	4	5				
4	4	4	5	6				
5	5	6	6	6				
6	3	3	3	4				
7	3	3	4	5				
8	4	4	4	4				
9	4	4	4	4				
10	4	4	5	5				
11	3	3	3	3				
12	2	2	3	3				
13	3	3	3	3				
14	2	3	3	3				
15	3	3	3	3				
Average	3.60	3.73	4.13	4.40				
Number of Vehicles	$\varphi$	Success Rate (%)	$\varphi$	Success Rate (%)	$\varphi$	Success Rate (%)	$\varphi$	Success Rate (%)
3	0.06	100.00	0.05	100.00	0.05	90.00	0.05	90.00
4	0.07	90.00	0.07	90.00	0.06	90.00	0.06	100.00
5	0.09	10.00	0.09	20.00	0.08	10.00	0.08	50.00
6	0.11	0.00	0.11	0.00	0.10	0.00	0.09	0.00
7	0.13	0.00	0.13	0.00	0.11	0.00	0.11	0.00

### 5.2.3. Simulations for 20 Nodes

In this simulation, we added five nodes from the CONFIG 15-1, as follows. Nodes 16, 17, 18, 19, and 20 are located at (−100 m, −180 m), (0 m, 100 m), (200 m, −180 m), (200 m, 180 m), and (170 m, 130 m), respectively. Similar to the simulation for 15 nodes, we define four edge configurations in the new graph, namely CONFIG 20-1, CONFIG 20-2, CONFIG 20-3, and CONFIG 20-4. The CONFIG 20-1 connectivity configuration is described in Figure 5. CONFIG 20-2 is constructed by modifying CONFIG 20-1, i.e., adding edges connecting Nodes 1 to 16, 4 to 16, 2 to 20, 6 to 9, 8 to 19, and 12 to 17. CONFIG 20-3 is performed by adding Edges 2 to 17, 3 to 5 to 9, 7 to 15, and 9 to 15 and eliminating Edges 4 to 16, 4 to 10, and 14 to 17 on CONFIG 20-2. CONFIG 20-4 is performed by the addition of Edges 3 to 17, 4 to 16, 4 to 10, and 5 to 14 to CONFIG 20-3.

Table 5 reveals the simulation results of such connectivity configurations. It can be shown that the solution can be found for  $\phi \leq 0.02$ . It means that, from the 10 trials, it can be concluded that there always exists at least one successful trial at the range of the  $\phi$ .

**Table 5.** Simulation results for 20 nodes.

Node	Number of Degree of Nodes			
	CONFIG 20-1	CONFIG 20-2	CONFIG 20-3	CONFIG 20-4
1	6	7	7	7
2	7	8	9	9
3	6	6	6	7
4	6	7	5	7
5	8	8	9	10
6	3	4	4	4
7	5	4	5	5

8	5	5	5	5				
9	4	5	6	6				
10	6	6	5	6				
11	4	4	4	4				
12	6	7	8	8				
13	5	5	6	6				
14	3	4	4	5				
15	5	5	7	7				
16	3	5	4	5				
17	3	4	5	6				
18	5	5	5	5				
19	5	6	6	6				
20	3	4	5	5				
Average	4.90	5.45	5.75	6.15				
Number of Vehicles	$\varphi$	Success Rate (%)	$\varphi$	Success Rate (%)	$\varphi$	Success Rate (%)	$\varphi$	Success Rate (%)
2	0.02	10.00	0.02	100.00	0.02	100.00	0.02	100.00
3	0.03	0.00	0.03	90.00	0.03	100.00	0.02	90.00
4	0.04	0.00	0.04	0.00	0.03	80.00	0.03	90.00
5	0.05	0.00	0.05	0.00	0.04	0.00	0.04	0.00
6	0.06	0.00	0.06	0.00	0.05	0.00	0.05	0.00
7	0.07	0.00	0.06	0.00	0.06	0.00	0.06	0.00

### 5.3. Analysis of Accuracy

From the simulations established in Section 5.3, we can analyze the accuracy of the result, i.e., minimum traveling time of the slowest vehicle. The standard deviation  $\sigma_T$  can identify the accuracy of the simulation. As shown in Tables 1 and 2, the averages of  $\sigma_T$  for  $t_{occ} > 10$  s and  $t_{occ} > 150$  s are 373.48 s and 631.2 s. It indicates that the results in lower  $t_{occ}$  are more accurate than the higher ones.

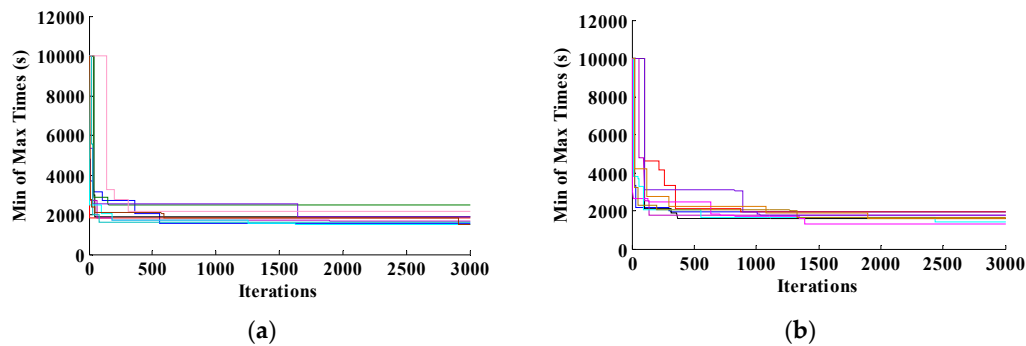
In addition, we analyzed the relationship between the average  $T_{avg}$  and  $\sigma_T$  by using the same technique as the one used in Section 5.2., i.e., statistical correlation. The results are described as follows. The correlations between  $T_{avg}$  and  $\sigma_T$  for  $t_{occ} > 10$  s and  $t_{occ} > 150$  s are 0.92 and 0.84, respectively. It indicates that there is a relatively strong positive relation between  $T_{avg}$  and  $\sigma_T$ : larger  $T_{avg}$  tends to lead to a larger  $\sigma_T$ , or, in other words, a larger  $T_{avg}$  tends to lead to lower accuracy.

Figure 6; Figure 7 show the searching progression for  $t_{occ} > 10$  s and  $t_{occ} > 150$  s, respectively, each with  $\rho = 0.1$  and  $\rho = 0.5$ . It can be concluded that the algorithm's ability to converge is better for  $t_{occ} > 10$  s than for  $t_{occ} > 150$  s. It can be concluded from the final values until the 3000th iteration. Moreover, Figure 6; Figure 7 confirm the accuracy conclusions conducted from the statistical correlation analysis. Accuracy analysis can infer other behavior, that is, the ability to reach  $T_{avg}$ . Figure 7 shows that, for  $t_{occ} > 10$  s, the algorithm's ability to reach  $T_{avg}$  is better than that for  $t_{occ} > 150$  s. Furthermore, it can be seen that, for  $t_{occ} > 150$  s, even until the 1500th iteration, the algorithm still can update the minimum of maximum times. However, it cannot go further to  $T_{avg}$ . The interesting situation occurs for  $\rho = 0.5$  in the same  $t_{occ}$ . In this situation, mostly, once a solution is found, the algorithm cannot find another better solution.

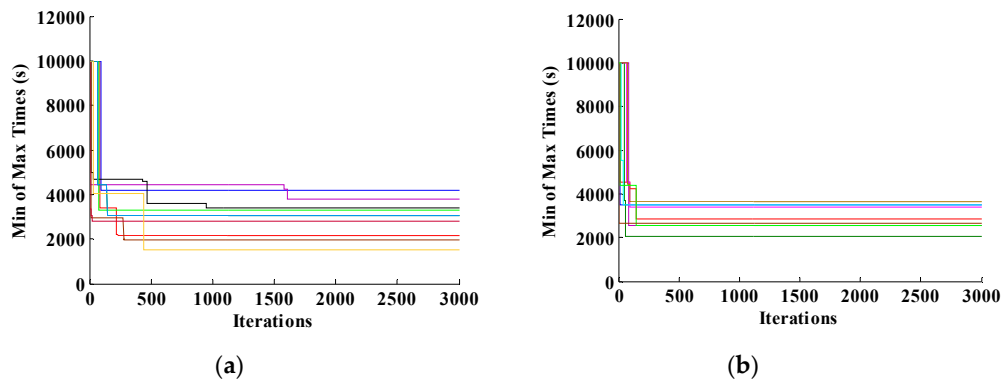
### 5.4. The Near-Optimal Trajectories

We show two samples of near-optimal trajectories obtained from the simulations to verify that any point in the graph  $G$  is visited by different vehicles with the arrival time difference between any vehicles exceeds  $t_{occ}$ . Tables 6 and 7 show the optimal trajectories for  $t_{occ} > 10$  s and  $t_{occ} > 150$  s, respectively. To verify that the arrival time differences exceed  $t_{occ}$ , we provide Table 8. According to

Table 8, for  $t_{occ} > 10$  s, the smallest arrival time difference is 11.7 s, i.e., between Vehicles 2 and 3 at Node 2. For  $t_{occ} > 150$  s, the smallest arrival time difference is 165.4 s, i.e., between Vehicles 2 and 3 at Node 2.



**Figure 6.** The progression of searching the optimal solution for  $t_{occ} > 10$  s: (a) for  $\rho = 0.1$ ; (b)  $\rho = 0.5$ .



**Figure 7.** The progression of searching the optimal solution for  $t_{occ} > 150$  s: (a) for  $\rho = 0.1$ ; (b)  $\rho = 0.5$ .

**Table 6.** most near-optimal trajectories for  $t_{occ} > 10$  s.

	Routes	4	10	2	6	7	9	1	5	3	8
<b>Vehicle 1</b>	Arrival Time (s)	0	365.6	446.9	511.5	584.0	659.5	767.8	874.4	998.3	1194.9
	Applied Speed (m/s)	0.1	1	1.5	1.5	1	1.5	1.5	1.5	1	1.5
	Routes	1	5	6	7	9	4	10	2	3	8
<b>Vehicle 2</b>	Arrival Time (s)	0	290.7	313.6	404.3	467.3	565.9	817.2	944.3	1174.1	1292.0
	Applied Speed (m/s)	1	1	0.5	1.5	1.5	0.1	1.5	0.1	1.5	1
	Routes	3	8	5	1	9	7	6	2	10	4
<b>Vehicle 3</b>	Arrival Time (s)	0	184.3	403.0	602.9	711.2	786.7	859.2	956.0	1091.5	1292.6
	Applied Speed (m/s)	0.1	1.5	0.1	1.5	1.5	1	1.5	0.5	1	1

**Table 7.** most near-optimal trajectories for  $t_{occ} > 150$  s.

	Routes	4	10	2	6	5	3	8	1	9	7
<b>Vehicle 1</b>	Arrival Time (s)	0	670.3	805.8	981.8	1003.3	1106.6	1290.9	1723.7	1886.1	1949.0
	Applied Speed (m/s)	0.1	0.5	1	0.1	1.5	1.5	0.1	0.5	1.5	1.5
	Routes	1	8	5	3	2	10	4	9	7	6
<b>Vehicle 2</b>	Arrival Time (s)	0	432.8	666.2	872.8	1240.5	1342.2	1476.2	1539.3	1710.9	1875.8
	Applied Speed (m/s)	0.1	0.5	1	0.5	0.5	1.5	1.5	1	0.1	1

	Routes	3	8	5	1	9	4	10	2	6	7
<b>Vehicle 3</b>	Arrival Time (s)	0	184.3	324.3	537.5	754.0	832.9	993.8	1075.1	1171.9	1262.6
	Applied Speed (m/s)	0.1	1.5	1	0.5	1	1	1.5	1	1	1

**Table 8.** arrival time difference for  $t_{occ} > 10$  s and  $t_{occ} > 150$  s.

Nodes	For $t_{occ} > 10$ s			For $t_{occ} > 150$ s		
	Vehicles 1 and 2	Vehicles 1 and 3	Vehicles 2 and 3	Vehicles 1 and 2	Vehicles 1 and 3	Vehicles 2 and 3
1	767.8	364.8	403	1723.7	1186.2	537.5
2	497.4	509.1	11.7	434.7	269.3	165.4
3	175.8	998.3	1174.1	233.8	1106.6	872.8
4	565.9	1292.6	726.7	1476.2	832.9	643.3
5	583.7	471.4	112.3	337.1	679	341.9
6	197.9	347.7	545.6	894	190.1	703.9
7	179.7	202.7	382.4	238.1	686.4	448.3
8	97.1	1010.6	1107.7	858.1	1106.6	248.5
9	192.2	51.7	243.9	346.8	1132.1	785.3
10	451.6	725.9	274.3	671.9	323.5	348.4

## 6. Conclusions

A method to solve the Collision-Free Multiple Traveling Salesman Problem (CFMTSP) applied to multiple agents was proposed. In this problem, each agent must visit all nodes in a provided graph, following the provided edges. The graph is modified into an augmented graph such that the information of nodes' position and speed options can be accommodated, and in turn, the arrival time to each node can be determined. According to the optimization of collision-free for each vehicle, an ACO involving multiple ant species is utilized on the augmented graph. The pheromone trails are not left on the edges (such as in typical graph model), but the augmented edges. As a consequence, the probability of selection is assigned to those augmented edges, as well. As a result, the solution is not the sequence of routes but also the trajectory (routes and selected speeds). The algorithm has to guarantee that the resulted trajectories are collision-free.

Simulations were established with no violation of minimum allowable arrival time difference. In addition, the simulations have shown three behaviors of the resulted solutions. First, the increase of minimum allowable arrival time difference leads to the decrease of the number of successful trials, almost-strongly increases of the average, the standard deviation, and the ever-found minimum traveling times. Second, it can be concluded that there exists a threshold of the ratio  $\phi$  such that the solution for all vehicles is unable to find. It was demonstrated that the threshold of  $\phi$  is lower as the number of nodes increases. Moreover, for the same number of nodes, the increase of the average of nodes' degree leads to the number of vehicles that are able to find solutions successfully. Third, it is concluded that the increase in the average of minimum traveling time tends to decrease the algorithm's accuracy. Third, the evaporate rate in the algorithm has a weak influence on the average, the standard deviation, and the ever-found minimum traveling times.

Future works will focus on some issues, such as the continuation of speed options and the consequences of modifying the augmented graph. Another issue is in the performance of the ACO algorithm, such as the convergence of the optimal trajectories, the decrease of the standard deviation of the slowest agent's traveling time in high minimum allowable arrival time difference.

**Author Contributions:** A.K.P. contributed to writing the original draft, formal analysis, conceptualization, methodology, and visualization; D.B.S. contributed to validation, review and editing of the manuscript, and supervision; conceptualization, methodology, and other aspects were agreed on by all authors. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received internal funding from Universitas Atma Jaya Yogyakarta, Indonesia.

**Acknowledgments:** This research was supported by Universitas Atma Jaya Yogyakarta. We thank Universitas Atma Jaya for providing the funding and tools required for this research.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

- Li, Y.; Soleimani, H.; Zohal, M. An improved ant colony optimization algorithm for the multi-depot green vehicle routing problem with multiple objectives. *J. Clean. Prod.* **2019**, *227*, 1161–1172.
- Bae, H.; Moon, I. Multi-depot vehicle routing problem with time windows considering delivery and installation vehicles. *Appl. Math. Model.* **2016**, *40*, 13–14.
- Mandziuk, J.; Swiechowski, M. UCT in capacitated vehicle routing problem with traffic jams. *Inf. Sci.* **2017**, *406–407*, 42–56.
- Bektas, T. The multiple traveling salesman problem: An overview of formulations and solution procedures. *Omega* **2006**, *34*, 209–219.
- Lawler, E.L. *The Travelling Salesman Problem: A Guided Tour of Combinatorial Optimization*; John Wiley and Sons: Hoboken, NJ, USA, 1985.
- Golden, B.; Raghavan, S.; Wasil, E.A. The vehicle routing problem: Latest advances and new challenges, operations research. In *Computer Science Interfaces Series*; Springer: Berlin/Heidelberg, Germany, 2008, Volume 43.
- Khachay, M.; Neznakhina, K. Polynomial time solvable subclass of the generalized traveling salesman problem on grid clusters. In *Proceeding of the International Conference on Analysis of Images, Social Networks and Texts*, Moscow, Russia, 5–7 July 2017; pp. 346–355.
- Chalarux, T.; Sripratak, P. Worst case analyses of nearest neighbor heuristic for finding the minimum weight K-cycle. *Curr. Appl. Sci. Technol.* **2020**, *20*, 178–185.
- Khachay, M.; Neznakhina, K. Approximability of the minimum-weight k-size cycle cover problem. *J. Glob. Optim.* **2016**, *66*, 65–82.
- Fazlollahtabar, H.; Saidi-Mehrabad, M. Mathematical model for deadlock resolution in multiple AGV scheduling and routing network: A case study. *Ind. Robot.* **2015**, *42*, 252–263.
- Vivaldini, K.; Rocha, L.F.; Martarelli, N.J.; Becker, M.; Moreira, A.P. Integrated tasks assignment and routing for the estimation of the optimal number of AGVS. *Int. J. Adv. Manuf. Technol.* **2016**, *82*, 719–736.
- Hu, W.; Mao, J.; Wei, K. Energy-efficient rail guided vehicle routing for two-sided loading/unloading automated freight handling systems. *Eur. J. Oper. Res.* **2017**, *258*, 943–957.
- Adamo, T.; Bektas, T.; Ghiani, G.; Guerriero, E.; Manni, E. Path and speed optimization for conflict-free pickup and delivery under time windows. *Trans. Sci.* **2018**, *52*, 739–755.
- Zhong, M.; Yang, Y.; Dessouky, Y.; Postolache, O. Multi-AGV scheduling for conflict-free path planning in automated container terminals. *Comput. Ind. Eng.* **2020**, *142*, 106371.
- Ali, I.M.; Essam, D.; Kasmarik, K. A novel design of differential evolution for solving discrete traveling salesman problems. *Swarm Evol. Comput.* **2020**, *53*, 100607.
- Baniasadi, P.; Foumani, M.; Smith-Miles, K.; Ejov, V. A transformation technique for the clustered generalized traveling salesman problem with application to logistics. *Eur. J. Oper. Res.* **2020**, *2*, 444–457, doi:10.1016/j.ejor.2020.01.053.
- Pacheco-Valencia, V.; Hernandez, J.A.; Sigaretta, J.M.; Vakhania, N. Simple constructive, insertion, and improvement heuristics based on the girding polygon for the Euclidean traveling salesman problem. *Algorithms* **2020**, *13*, 5.
- Pandiri, V.; Singh, A. An artificial bee colony algorithm with variable degree of perturbation for the generalized covering traveling salesman. *Appl. Soft Comput.* **2019**, *78*, 481–495.
- Held, M.; Karp, R.M. The traveling salesman problem and minimum spanning tree. *Oper. Res.* **1970**, *18*, 1138–1162.
- Fiechter, C.-N. A parallel tabu search algorithm for large traveling salesman problem. *Discret. Appl. Math.* **1994**, *51*, 243–267.
- Siqueira, P.H.; Steiner, M.T.A.; Scheer, S. A new approach to solve the traveling salesman problem. *Neurocomputing* **2007**, *70*, 1013–1021.
- Rego, C.; Gamboa, D.; Glover, F.; Osterman, C. Traveling salesman problem heuristics: Leading methods, implementations and latest advances. *Eur. J. Oper. Res.* **2011**, *211*, 427–441.

23. Essani, F.H.; Haider, S. An algorithm for mapping the asymmetric multiple traveling salesman problem onto colored petri nets. *Algorithms* **2018**, *11*, 143.
24. Groba, C.; Sartal, A.; Vasquez, X.H. Integrating forecasting in metaheuristic methods to solve dynamic routing problems: Evidence from the logistic processes of tuna vessels. *Eng. Appl. Artif. Intel.* **2018**, *76*, 55–66.
25. Xu, W.; Guo, S.; Li, X.; Guo, C.; Wu, R.; Peng, Z. A dynamic scheduling method for logistics tasks oriented to intelligent manufacturing workshop. *Math. Probl. Eng.* **2019**, *2019*, 7237459.
26. Pandiri, V.; Singh, A. A swarm intelligence approach for the colored traveling salesman problem. *Appl. Intell.* **2018**, *48*, 4412–4428.
27. Lu, L.-C.; Yue, T.-W. Mission-oriented ant-team ACO for min–max MTSP. *Appl. Soft. Comput.* **2019**, *76*, 436–444.
28. Dorigo, M.; Gambardella, L. Ant colony: A cooperative learning approach to the travelling salesman problem. *IEEE Trans. Evol. Comput.* **1997**, *1*, 53–66.
29. Widyotriatmo, A.; Hong, K.-S. Navigation function-based control of multiple wheeled vehicles. *IEEE Trans. Ind. Electron.* **2011**, *58*, 1896–1906.
30. Fraichard, T.; Asama, H. Inevitable collision states—A step towards safer robots? *Adv. Robot.* **2004**, *18*, 1001–1024.
31. Snape, J.; Berg, van den Berg, J.; Guy, S.J.; Manocha, D. The hybrid reciprocal velocity obstacle. *IEEE Trans. Robot.* **2011**, *27*, 696–706.
32. Alonso-Mora, J.; Beardsley, P.; Siegwart, R. Cooperative collision avoidance for nonholonomic robots. *IEEE Trans. Robot.* **2018**, *34*, 404–420.
33. Makarem, L.; Gillet, D. Model predictive coordination of autonomous vehicles crossing intersections. In Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems (ITSC), Hague, The Netherlands, 6–9 October 2013.
34. Mahbub, A.M.I.; Malikopoulos, A.A.; Zhao, L. Decentralized optimal coordination of connected and automated vehicles for multiple traffic scenarios. *Automatica* **2020**, *117*, 108958.
35. Tan, C.Y.; Huang, S.; Tan, K.K.; Teo, R.S.H. Three-dimensional collision avoidance for multi unmanned aerial. Vehicles Using Velocity Obstacle. *J. Intell. Robot. Syst.* **2020**, *97*, 227–248.
36. Pamosoaji, A.K.; Piao, M.; Hong, K.-S. PSO-based minimum-time motion planning for multiple-vehicle systems considering acceleration and velocity limitations. *Int. J. Control Autom. Syst.* **2019**, *17*, 2610–2623.
37. Yu, M.; Qi, X. A Vehicle routing problem with multiple overlapped batches. *Trans. Res. E* **2014**, *61*, 40–55.
38. Baranwal, M.; Roehl, B.; Salapaka, S.M. Multiple traveling salesmen and related problems: A maximum-entropy principle-based approach. In Proceedings of the 2017 American Control Conference (ACC), Seattle, WA, USA, 24–26 May 2017.
39. Pamosoaji, A.K.; Hong, K.-S. Group-based particle swarm optimization for multiple-vehicles trajectory planning. In Proceedings of the 15th International Conference on Control, Automation, and Systems (ICCAS), Busan, Korea, 13–16 October 2015; pp. 862–867.
40. Chalisser, J.M.; Renyard, A.; Gries, R.; Hoefele, D.; Alamsetti, S.K.; Gries, G. Ant sense, and follow, trail pheromones of ant community members. *Insects* **2019**, *10*, 1–11.





SJR

Scimago Journal &amp; Country Rank

Enter Journal Title, ISSN or Publisher Name

[Home](#)[Journal Rankings](#)[Country Rankings](#)[Viz Tools](#)[Help](#)[About Us](#)

# Algorithms

**Country**[Switzerland](#) -  [SJR Ranking of Switzerland](#)**Subject Area and Category**[Computer Science](#)  
[Computational Theory and Mathematics](#)[Mathematics](#)  
[Computational Mathematics](#)  
[Numerical Analysis](#)  
[Theoretical Computer Science](#)**Publisher**[MDPI AG](#)**Publication type**[Journals](#)**ISSN**[19994893](#)**Coverage**[2008-2020](#)**Scope**

Algorithm engineering, Algorithmic game theory and mechanism design, Algorithms for databases, Algorithms for language processing, Algorithms in biology, chemistry, physics, Algorithms related to automata theory and formal languages, Approximation algorithms, Combinatorial optimization, mathematical programming, operations research, discrete mathematics and graph theory, Communication and data networks, Computational geometry, Data structures, Differential equations, Distributed and parallel algorithms, Eigenvalue problems, Image processing with applications, Interdisciplinary applications in other areas of mathematics and computer science, Iterative methods and algorithms, Machine learning, Markov chains and simulation, Metaheuristics and matheuristics, Numerical analysis, Parametrized algorithms, Performance and testing of algorithms, Production planning, scheduling, transport, and timetabling, Quantum algorithms, Randomized algorithms, Sorting and search algorithm, Theory of algorithms

[Homepage](#)[How to publish in this journal](#)[Contact](#)[Join the conversation about this journal](#)

# 27

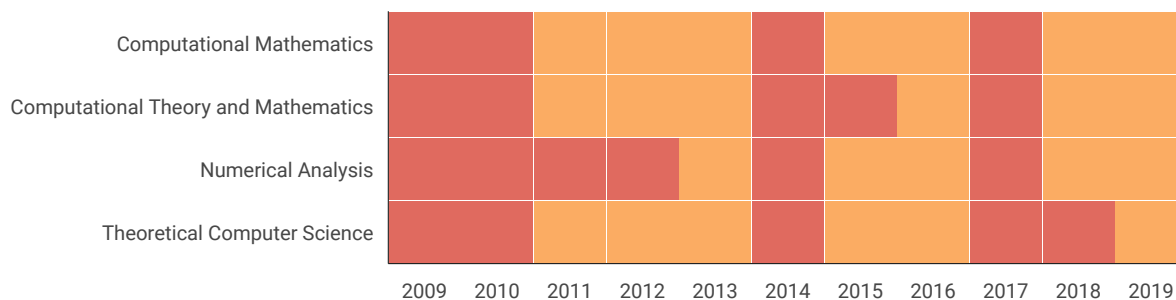
H Index



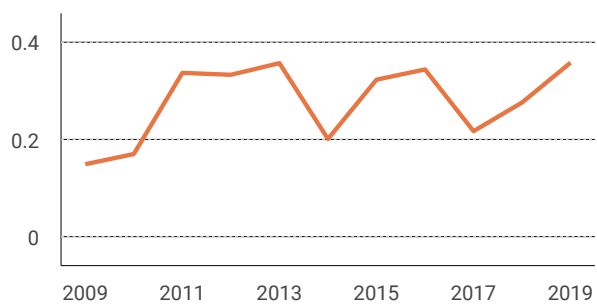


Product Sourcing Made Easy  
Alibaba.com

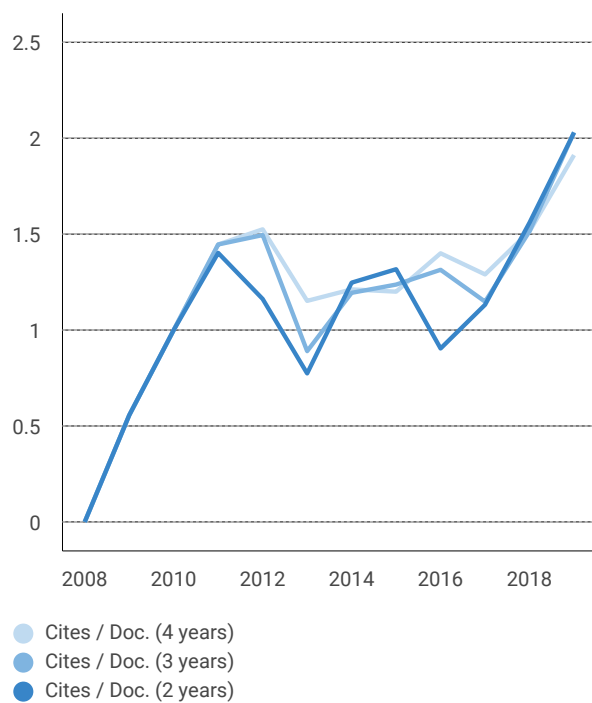
## Quartiles



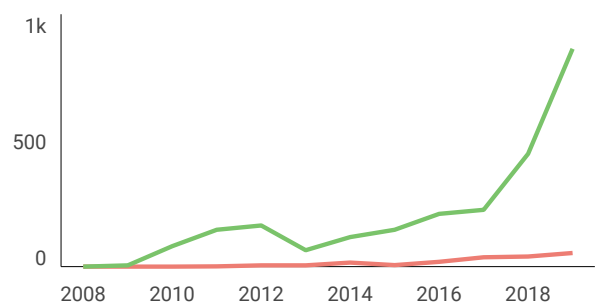
## SJR



## Citations per document



## Total Cites Self-Cites

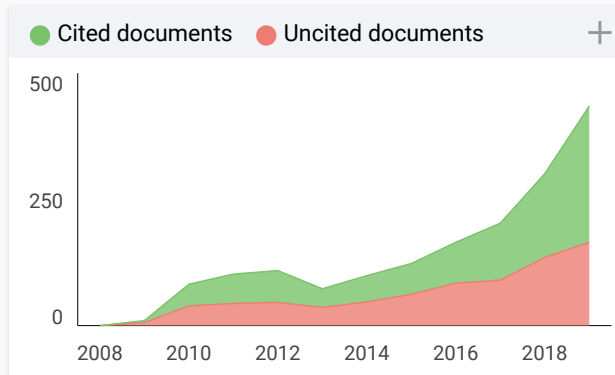
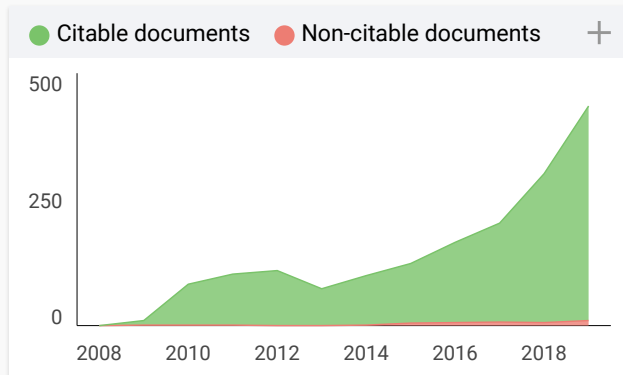


## External Cites per Doc Cites per Doc



## % International Collaboration





**Algorithms**

Q3 Computational Mathematics

best quartile

SJR 2019 0.36

powered by scimagojr.com

← Show this widget in your own website

Just copy the code below and paste within your html code:

```
<a href="https://www.sc
```



Loading comments...

Developed by:



Powered by:

Scopus

Follow us on @ScimagoJR

Scimago Lab, Copyright 2007-2020. Data Source: Scopus®

EST MODUS IN REBUS

Horatio (Satire 1,1,106)