Distributed coordination for collision avoidance of multiple ships considering ship maneuverability

Li, Shijie; Liu, Jialun; Negenborn, Rudy R.

DOI
10.1016/j.oceaneng.2019.03.054

Publication date
2019

Document Version
Accepted author manuscript

Published in
Ocean Engineering

Citation (APA)
Li, S., Liu, J., & Negenborn, R. R. (2019). Distributed coordination for collision avoidance of multiple ships considering ship maneuverability. *Ocean Engineering*, *181*, 212-226.
https://doi.org/10.1016/j.oceaneng.2019.03.054

Distributed coordination for collision avoidance of multiple ships considering ship maneuverability

Shijie Li[a], Jialun Liu[b,c,*], Rudy R. Negenborn[d]

[a] *School of Logistics Engineering, Wuhan University of Technology, No.1178 Heping Road, Wuhan, P. R. China*
[b] *Intelligent Transportation Systems Research Center (ITSC), Wuhan University of Technology, No.1178 Heping Road, Wuhan, P. R. China*
[c] *National Engineering Research Center for Water Transport Safety (WTSC), Wuhan University of Technology, No.1178 Heping Road, Wuhan, P. R. China*
[d] *Department of Maritime and Transport Technology, Delft University of Technology, Delft, The Netherlands*

Abstract

Over the past two decades, a number of methods have been proposed for solving maritime collision avoidance problems. Most of these works take a single ship's perspective and focus on one-to-one or one-to-many situations. To more complicated many-to-many situations, less attention has been paid. To deal with the many-to-many collision avoidance problem, this paper proposes a distributed coordination strategy which consists of two phases: firstly, predictions of ship trajectories are made based on ship dynamics, giving different candidate rudder angles, and potential collision risks that may be caused by each rudder angle selection are evaluated based on calculations of collision risk parameters; secondly, an optimization strategy is adopted to find the most efficient collision avoidance plan for the ships, namely, the rudder angles that each ship should take, and the corresponding operation time for rudder steering, with the overall objective to minimize the sum of time that each ship spends in avoiding collisions with the other ships. Simulation experiments are carried out to evaluate the effectiveness of the proposed method, as well as the corresponding communication and computation costs.

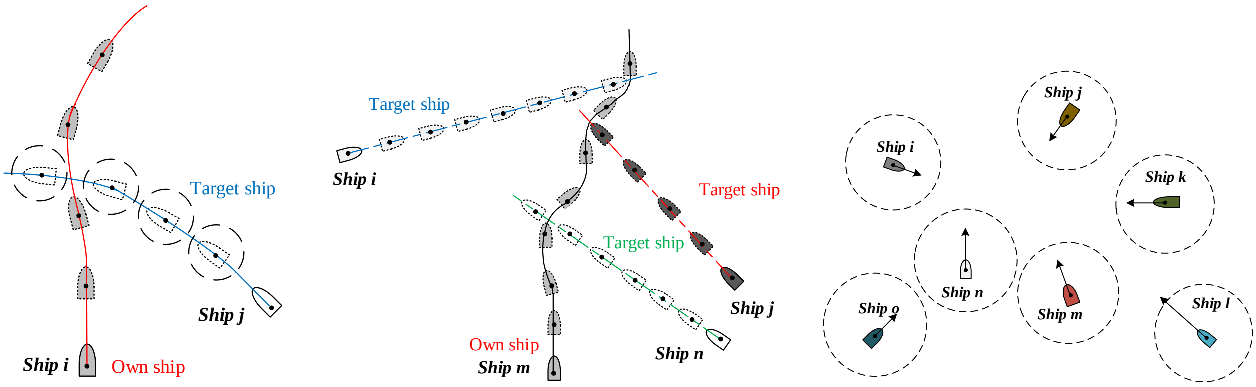
Keywords: collision avoidance, distributed coordination, ship maneuverability, decision making

1. Introduction

Ship collisions can cause great losses of lives and property, and yield negative impacts to the maritime environment. While advanced assistant systems, such as GPS (Global Positioning System), ARPA (Automatic Radar Plotting Aid), AIS (Automatic Identification System), and ECDIS (Electronic Chart Display and Information System), have been developed and installed on ships, collision accidents still happen every now and then. In addition, the development of larger ships nowadays also sets higher requirements on collision avoidance methods. Therefore, it is important to improve the intelligence and autonomy in assisting ships to make safe and quick decisions to avoid potential collisions. Additionally, more and more attention has been paid to smart ships and, moreover, autonomous ships. The ability of autonomous collision avoidance is a critical prerequisite to fulfill the idea of autonomous navigation.

In practice, several methods have been adopted to avoid ship collisions, such as lookouts, radar and VHF radio. The 1972 International Regulations for Preventing Collision at Sea (COLREGs), proposed by the International Maritime Organization (IMO), is supposed to be obeyed by all ships. On the one hand, it describes potential collision scenarios between encountering ships and provides a set of guidelines for safe maneuvering at sea. On the other hand, it could also be hard to describe all possible conditions in the form of rules due to the complexity of actual maritime environment. A number of maritime collision avoidance methods, such as ship domain, fuzzy theory, evolutionary algorithms, and real-time control algorithms, have been proposed over the last two decades. Relevant literature reviews can be found in (Tam et al., 2009; Johansen et al., 2016).

*Corresponding author: Jialun Liu, jialunliu@whut.edu.cn.



(a) One-to-one situation. (b) One-to-many situation. (c) Many-to-many situation.

Figure 1: Examples of encountering situations.

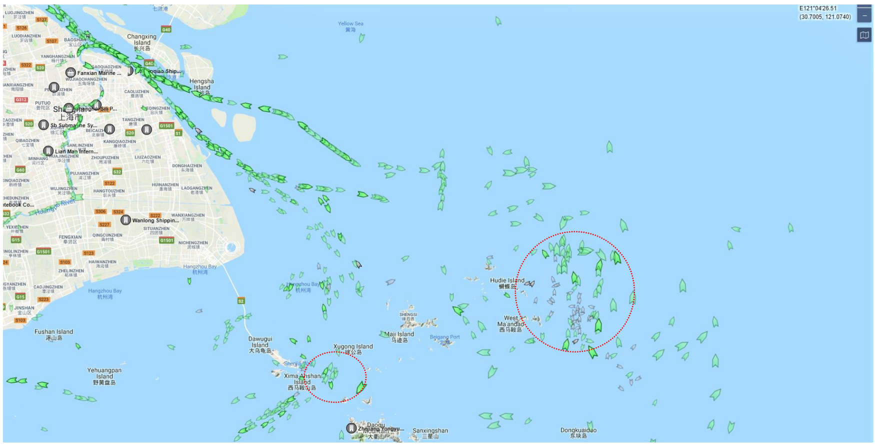


Figure 2: The maritime traffic near Yangshan Port in Shanghai, P. R. China (MarineTraffic, 2018).

Figure 1 gives examples of ships' encountering situations, including a) one-to-one, b) one-to-many and c) many-to-many situations. In one-to-one and one-to-many situations, two types of ships are considered, namely the own ship and the target ship. In one-to-one situation, the own ship only considers one target ship at one time, while in one-to-many situation, the own ship needs to consider a set of upcoming target ships, usually in a one-by-one manner. Both situations take the perspective of own ship to avoid the collision with target ships, and the main anti-collision decisions are made for the own ship, based on the prediction and estimation of the movements of target ships. It is assumed that the own ship has a detection range, and that it can exchange messages with target ships within the detection range. The own ship should always keep a safety domain, which is a circle with a certain radius between itself and a target ship, otherwise collision accidents may happen. In a one-to-many situation as shown in Figure 1(b), only the potential collision risks between the own ship m and the target ships i, j and n are considered, while the risks between ships i, j and n themselves are usually ignored in the collision avoidance problem of own ship m.

As the anti-collision decisions of the involved ships are highly related, as a small change in the course of one ship may affect the future decisions of the other ships, it is important to consider the impacts of all the encountering ships' collision avoidance decisions. Therefore, in many-to-many situations, it is assumed that the ships are able to exchange information regarding their intended courses during the collision avoidance procedure via a communication and coordination structure. Figure 2 gives an example that shows the real-time traffic density of ships near Yangshan Port in Shanghai, P. R. China. It can be seen that in practice, multiple ships encountering situations happen frequently, especially in the open sea near a seaport.

When multiple ships encounter one another, the decision for each ship to maneuver for avoiding the collision depends on many factors, such as ship speed, course, relative position, and ship maneuverability. In general, ship course alteration is more effective than ship speed alteration as it leads to faster effects. Meanwhile, the alteration of a ship's course is also easier to be observed both visually and on radar by surrounding ships (Wang et al., 2017). In practice, a ship's rudder angle affects its rudder forces and moments, therewith leading to changes in its course. Therefore, this paper considers rudder angle alteration as the main collision avoidance decision.

To optimize the collision avoidance operation of multiple ships, it is important to make sure that the ships choose suitable rudder angles. This paper aims to find optimal rudder angles and operation time of rudder steering for multiple encountering ships with collision risks using a distributed coordination scheme. The motivation for adopting a distributed scheme is threefold: firstly, it is natural to model their interactions in a distributed way, as they are independent parties; secondly, the peer-to-peer style of a distributed scheme is more tolerant to message loss, delay, and asynchronous information; thirdly, distributing the search for a consistent collision avoidance plan over a set of ships reduces the computational burden and reliance on a single coordinator or controller.

Precise predictions of potential collision risks between multiple ships, as well as accurate estimation of the effects that ships' anti-collision decisions lead to, form the basis of efficient collision avoidance coordination. Therefore, it is important to consider the impacts of ship dynamics, steering and propulsion system. Meanwhile, integrated and intelligent optimization strategies are required to ensure the efficiency of collision avoidance decisions. This paper proposes a distributed coordination approach for solving the multiple ships collision avoidance problem considering both the maneuverability of individual ships and mutually-affected anti-collision decisions. It is assumed that each ship can exchange information regarding its position, course, and speed with the other ships.

The coordination strategy consists of two phases: firstly, each ship makes predictions regarding its potential trajectories with different rudder angles and dynamically calculating collision risk parameters; secondly, a coordination scheme based on distributed constraint optimization (DCOP) is formulated, and typical DCOP algorithms are adopted to search for globally optimal solutions for multiple ships. In order to evaluate the computation and communication costs of the distributed coordination strategy, experiments are carried out. This results in insight into the type, amount, and size of messages involved when multiple ships exchange information to find globally optimal solutions.

1.1. Related work

1.1.1. Collision risk evaluation

Collision risk evaluation is the basis of collision avoidance decision making. The most commonly used parameters are the Distance at the Closest Point of Approach (DCPA) and the Time to the Closest Point of Approach (TCPA), for more details we refer the readers to a survey on ship collision risk evaluation in (Xu and Wang, 2014). In recent years, new collision risk evaluation methods have been proposed. A real-time multi-ship collision evaluation is proposed in (Zhen et al., 2017), consisting of a spatial clustering process (DBSCAN) for detecting clusters of encounter ships, and a multi-ship collision risk index model for encounter ships within each cluster. A new analytic formula for domain-based collision risk parameters is introduced in (Szlapczynski and Szlapczynska, 2016), including the degree of domain violation (DDV) and the time to domain violation (TDV) as indices.

1.1.2. Decision making in multiple ship encounter situations

To eliminate the insufficiency of ignoring ship maneuverability in the process of avoiding collisions, a ship maneuverability-based collision avoidance support system in close-quarter situations is proposed in (Wang et al., 2017). Characteristics of ship dynamics to calculate collision avoidance parameters and a PID controller in ship maneuvering are considered. The work in (Johansen et al., 2016) presents a concept for ship collision avoidance based on model predictive control. A finite set of alternative control behaviors is generated by varying offsets to the guidance course angle commanded to the autopilot, and changes to the propulsion command ranging from nominal speed to full reverse. Using simulated ship trajectory predictions, each alternative control behavior is evaluated with respect to its compliance with the COLREGS'72 and associated collision risks, in order to find the optimal control behavior. These works take a single ship perspective and mainly consider one-to-one ship encountering situations.

Regarding multi-ship encounter situations, a Distributed Local Search Algorithm (DLSA) and a Distributed Tabu Search Algorithm (DTSA) have been adopted in (Kim et al., 2015) to find optimal courses for each involved ship.

More specifically, when multiple ships encounter one another, it is assumed that the highest priority to choose the next course will be given to the ship that can reduce collision risk most significantly with course alteration. Each individual ship computes its collision risk based on the information on current courses that it receives from the neighboring ships. This process is repeated until the collision risk disappears. Later on, to reduce the communication burden and shorten the computation time, this approach has been extended by introducing a Distributed Stochastic Search Algorithm (DSSA), which allows each ship to change its intention in a stochastic manner immediately after receiving all of the intentions from the other ships in (Kim et al., 2017). While these works consider a distributed coordination structure involving multiple ships, the maneuverability of the ships has not been taken into account. As the main collision avoidance decision is course alteration, ignoring the characteristics of ship dynamics may cause situations in which the ships cannot change their course as fast as planned.

A decision support system for ship collision avoidance is proposed for Istanbul Strait in (Perera et al., 2015). The system uses manually controlled and reciprocally passing ships' data to train artificial neural networks (ANN), with the aim to make predictions of ships' future locations three minutes in advance. If collision risks exist, warnings would be sent to the Vessel Traffic Services (VTS) center and to the ships' personnel. In (Zhang et al., 2015), a distributed and real-time multi-ship anti-collision decision support formulation is presented that considers both course alteration and speed alteration. The decision-making process is distributed: each ship makes decisions based on its own judgment according to a set of pre-defined rules, while keeping on monitoring and receiving information from other ships.

1.1.3. Anti-collision trajectory planning

A number of optimization methods have been proposed to assist ships in finding collision-free and optimal trajectories in literature. Most of of these collision avoidance approaches are limited to one-to-one and one-to-many situations. Existing methods include Genetic Algorithm (Tam and Bucknall, 2010), Fuzzy Logic (Perera et al., 2011), Branch and Bound (Mohamed-Seghir, 2012), A* Algorithm (Naeem et al., 2012), Ant Colony Optimization (Escario et al., 2012; Lazarowska, 2014), Cooperative Path Planning (Tam and Bucknall, 2013), Neural Network (Simsir et al., 2014), Fast Marching Method (Liu and Bucknall, 2015), Multi-criteria Optimization (Lazarowska, 2017a), etc. This paper mainly discusses the most recent work on trajectory planning methods in many-to-many situations, in which the objective is to optimize all the trajectories of the involved ships.

Szlapczynski (2011) proposed an evolutionary algorithm-based safe trajectory planning for ships. Later on, Szlapczynski (2013a,b) extend this approach to make it applicable within Traffic Separation Scheme (TSS) governed by the IMO. These changes include detecting and penalizing TSS violations using a specialized fitness function, and the genration of safe trajectories that are already partially valid within a TSS. In (Szlapczynski, 2015), further improvements are made on these approaches for ship trajectories in restricted visibility according to the rule 19 of COLREGs.

Tam and Bucknall (2013) developed a deterministic collision avoidance path planning algorithm to provide collision-free path for all involved ships. The algorithm plans navigation paths for all encountering ships in a cooperative mode, from a multi-ship perspective. Hornauer et al. (2015) proposed a partly-cooperative decentralized trajectory optimization algorithm to avoid collision between ships. The movement for non-cooperative ships is computed by a Bayesian model using the data from AIS. The probability of the estimated position for a passive ship that predicts the trajectories by historic probabilistic models is accurately computed. In (Szlapczynska, 2015), a ship maneuver auto-negotiation system is proposed, in which the ships can communicate with each other and negotiate their maneuvers via the designed negotiation procedure while assuring compliance with COLREGs. Lazarowska (2017b) proposes a decision support system to plan a safe, optimal path for a ship where both static and dynamic obstacles are both considered. It utilizes the idea of a trajectories database, which is later on searched through to find the best solutions.

1.2. Contribution

The main contribution of this paper are:

- Proposal of a distributed coordination mechanism with guarantees on solution optimality. The distributed mechanism does not require any instructions from a centralized system, which means that the ships can find optimal collision avoidance decisions by themselves, with peer-to-peer communication. This ensures the robustness and autonomy of the proposed approach. In addition, the optimality of solutions is also guaranteed because of the distributed constraint optimization algorithms used for solving the collision avoidance problem.

- Dynamic calculation of collision risk parameters using accurate maneuverability models. When a ship starts maneuvering to avoid collisions with other ships, the collision risks between itself and the other ships are also changing. Without accurate predictions on the changes of collision risks over time when the involved ship starts maneuvering, the generated collision avoidance decisions may not be accurate. Therefore, this paper considers an empirical maneuvering model of ships (Liu et al., 2017), so as the calculate collision risk among multiple ships in a dynamic way.

- Investigation and analysis of the communication costs of the distributed coordination mechanism. While a distributed mechanism is not rare in the existing literature, the actual communication costs incurred by ship-to-ship interactions have been rarely studied. For this, the incurred communication costs including the types, numbers, and sizes of messages exchanged between ships during the distributed coordination process are studied in this paper. This can give practitioners insights regarding the communication requirements for implementing the proposed approach.

1.3. Outline

This remainder of the paper is organized as follows. Section 2 introduces the structure of the proposed approach. Detailed descriptions of the two phases part of approach are given in Section 3 and Section 4. Experimental results are presented in Section 5. Conclusions and future work are given in Section 6.

2. Structure of the proposed approach

Figure 3 shows the structure of the proposed approach that consists of two phases. Initially, information regarding the current states of the ships within a certain detection range and the maritime environment are considered as known. Phase 1 concerns the dynamic calculation of collision risk parameters based on ship maneuvering model. Given the current ship states and environmental information, for each candidate rudder angle, predictions of ship's future trajectories are made based on ship maneuvering model. According to the coordinates and ship states on the predicted trajectories, collision risk parameters can be obtained. Based on collision risk parameters including time to the closest point of approach (TCPA) and distance at the closest point of approach (DCPA), Phase 2 concerns a distributed coordination model of multiple ships. Distributed constraint optimization-based methods are incorporated to search for optimal solutions, with which optimal rudder angle and the rudder steering time that each ship should take are determined.

3. Dynamic calculation of collision risk parameters considering ship maneuverability

In order to construct the distributed coordination model for collision avoidance, collision risks among multiple ships need to be quantified via collision risk parameters. As expressed in Section 1.1.1, distance at the closest point of approach (DCPA) and time to the closest point of approach (TCPA) between two ships are commonly used in literature to evaluate the potential risks between any two ships. This paper adopts the same parameters here. In practice, ships' states such as speeds, coordinates, and courses are changing over time: the DCPA and TCPA between any two ships are therefore also changing over time. Hence, this paper considers a dynamic way of calculating the DCPA and TCPA between any two ships.

Table 1 gives the symbols and the corresponding definitions used in this paper. Ship set V, prediction time $T^{\text{prediction}}$, and minimum safety distance D^{safe} are known parameters. Based on ship dynamics, ship states including surge and sway speed $(u_i(t), v_i(t))$, velocity $V_i(t)$, heading angle $\varphi_i(t)$, coordinates $(x_i(t), y_i(t))$ are determined. Then, the relative distance $R_{ij}(t)$, relative velocity $V_{ij}^{\text{R}}(t)$, relative bearing angle $\beta_{ij}^{\text{R}}(t)$ and relative heading angle $\varphi_{ij}^{\text{R}}(t)$ between any two ships are determined, in order to calculate collision risk parameters $D_{ij}^{\text{CPA}}(t)$ and $T_{ij}^{\text{CPA}}(t)$. Rudder angle δ_i and rudder steering time T_i^{steering} are the main decision variables. Considering typical ship maneuvering, this paper assumes that the rudder angle ranges from -30° on the port side to +30° on the starboard side (±30°) in steps of 10°. Therefore, rudder angle variable $\delta_i \in D_i = \{-30°, -20°, -10°, 0°, 10°, 20°, 30°\}$.

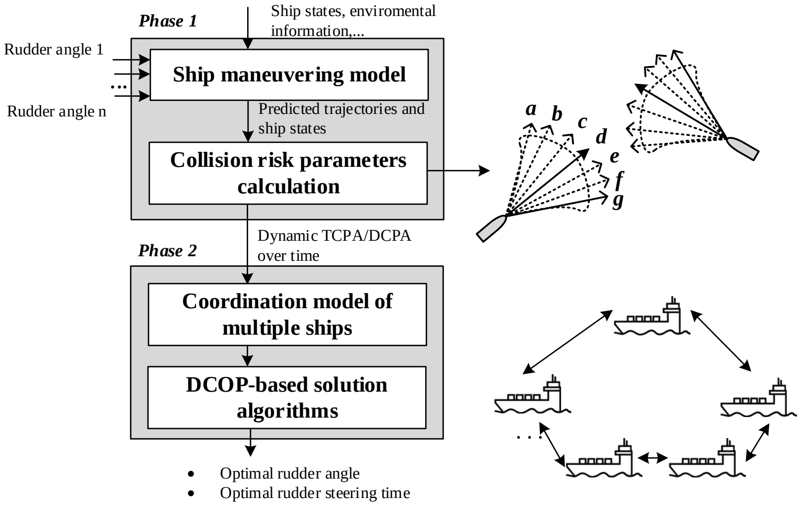


Figure 3: Structure of the proposed approach

Table 1: Symbols and definitions.

Symbols	Definitions
V	a set of encountering ships
$T^{\text{prediction}}$	length of trajectory prediction time
D^{safe}	minimum safe distance that any two ships should keep to avoid collision
$(u_i(t), v_i(t))$	forward speed u and sway speed v of ship i at time t
$V_i(t)$	velocity of ship i at time t
$\varphi_i(t)$	heading angle of ship i at time t
$(x_i(t), y_i(t))$	coordinates of ship i on x axis and y axis at time t
$R_{ij}(t)$	relative distance between ships i and j at time t
$V_{ij}^{\text{R}}(t)$	relative velocity between ships i and j at time t
$\beta_{ij}^{\text{R}}(t)$	relative bearing angle between ships i and j at time t
$\varphi_{ij}^{\text{R}}(t)$	relative heading angle between ships i and j at time t
$D_{ij}^{\text{CPA}}(t)$	distance between ships i and j at their closest point of approach (CPA) at time t
$T_{ij}^{\text{CPA}}(t)$	traveling time from ship i's position to its CPA with ship j at time t
δ_i	rudder angle that ship i takes
D_i	domain of variable δ_i
T_{ij}^{m}	shortest rudder steering time for ship i to avoid collision with ship j
D_{ij}^{m}	domain of variable T_{ij}^{m}
T_i^{steering}	shortest rudder steering time for ship i to avoid collisions with other ships

In order to evaluate the effects of the chosen rudder angles on the collision risks, trajectories of ships are predicted based on ship dynamics. Firstly, for each ship $i \in V$, it takes a value for its rudder angle δ_i from its variable domain D_i, and calculates the trajectories with different rudder angles in a prediction time $T^{\text{prediction}}$ based on on a ship maneuvering model. A ship's trajectory consists of a series of ship coordinates over time t.

The maneuvering model for a ship i is expressed as follows:

$$\begin{cases} (m+m_x)\dot{u} - (m+m_y)vr - x_G m r^2 = X_H + X_P + X_R \\ (m+m_y)\dot{v} + (m+m_x)ur + x_G m\dot{r} = Y_H + Y_P + Y_R \\ (I_z + x_G^2 m + J_z)\dot{r} + x_G m(\dot{v} + ur) = N_H + N_P + N_R, \end{cases} \tag{1}$$

where subscripts H,P,R represent the hull, the propeller, and the rudder, respectively; m, m_x, and m_y are ship mass, added mass in x-direction, and added mass in y-direction, respectively; I_z and J_z are moment of inertia and added moment of inertia around the z-axis, u and v are ship longitudinal and lateral speed, respectively; r is ship yaw rate around midship, and the dot notation of u, v and r represents the derivative of each parameter. For more details regarding the model, we refer the readers to (Liu et al., 2017). Given different rudder angles, the hydrodynamic force X_R due to rudder acting on midship in $x-$ direction is determined, thereby the forward speed u and acceleration $\dot{u}$ in $x-$axis, as well as sway speed v and acceleration $\dot{v}$ in $y-$axis are also determined. Based on the ship motion variables (u, v) and $(\dot{u}, \dot{v})$, coordinates $(x(t), y(t))$ of the ship on x-axis and y-axis at time t can be calculated.

Secondly, based on the ship coordinates and speeds over time t, this paper calculates the distance at the closest point of approach $D_{ij}^{\text{CPA}}(t)$ and time to the closest point of approach $T_{ij}^{\text{CPA}}(t)$ between any two ships i and j over time t. As an example, Figure 4 shows the relative position of two encountering ships based on the applied coordinate systems and relevant symbols. The ships are assumed to sail in the earth-fixed coordinate system $o_0 - x_0 y_0 z_0$ with a body-fixed coordinate system $o - xyz$ on the midship point. The predicted trajectory is the path of the center of gravity O in the $o_0 - x_0 y_0 z_0$ system. In addition, speed (u, v), acceleration $(\dot{u}, \dot{v})$, force (X, Y), and moment N, are also defined on midship O.

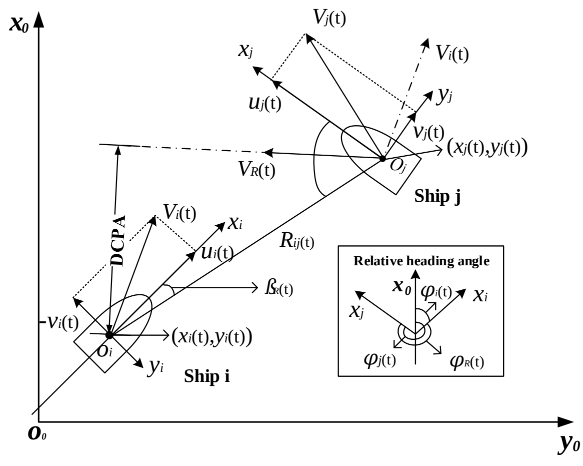


Figure 4: The relative position between two encountering ships based on earth-fixed coordinate system.

This phase determines the TCPA and DCPA of each ship at time t, represented as T_{ij}^{CPA} and D_{ij}^{DCPA} within the trajectory prediction time $T^{\text{prediction}}$. The relative distance $R_{ij}(t)$, relative bearing angle $\beta_{ij}^{\text{R}}(t)$, relative speed $V_{ij}^{\text{R}}(t)$ and relative heading angle $\varphi_{ij}^{\text{R}}(t)$ between ships i and j are calculated as follows:

$$R_{ij}(t) = \sqrt{(x_j(t) - x_i(t))^2 + (y_j(t) - y_i(t))^2} \tag{2}$$

$$\beta_{ij}^{\mathrm{R}}(t) = \arctan\left(\frac{x_j(t) - x_i(t)}{y_j(t) - y_i(t)}\right) + \Delta\beta_{ij}^{\mathrm{R}}(t), \qquad \text{where,} \qquad \Delta\beta_{ij}^{\mathrm{R}}(t) = \begin{cases} 0, & x_j(t) \geq x_i(t), y_j(t) \geq y_i(t) \\ 2\pi, & x_j(t) < x_i(t), y_j(t) \geq y_i(t) \\ \pi. & others \end{cases} \tag{3}$$

$$V_{ij}^{\mathrm{R}}(t) = \sqrt{(V_j(t)\sin\varphi_j(t) - V_i(t)\sin\varphi_i(t))^2 + (V_j(t)\cos\varphi_j(t) - V_i(t)\cos\varphi_i(t))^2} \tag{4}$$

$$\varphi_{ij}^{\mathrm{R}}(t) = \arctan\left(\frac{V_j(t)\sin\varphi_j(t) - V_i(t)\sin\varphi_i(t)}{V_j(t)\cos\varphi_j(t) - V_i(t)\cos\varphi_i(t)}\right) + \Delta\varphi_{ij}^{\mathrm{R}}(t) \tag{5}$$

$$\text{where,} \qquad \Delta\varphi_{ij}^{\mathrm{R}}(t) = \begin{cases} 0, & V_j(t)\sin\varphi_j(t) \geq V_i(t)\sin\varphi_i(t), V_j(t)\cos\varphi_j(t) \geq V_i(t)\cos\varphi_i(t) \\ 2\pi, & V_j(t)\sin\varphi_j(t) < V_i(t)\sin\varphi_i(t), V_j(t)\cos\varphi_j(t) \geq V_i(t)\cos\varphi_i(t) \\ \pi. & others \end{cases} \tag{6}$$

$$D_{ij}^{\mathrm{CPA}}(t) = R_{ij}(t)\sin\left(\varphi_{ij}^{\mathrm{R}}(t) - \beta_{ij}^{\mathrm{R}}(t) - \pi\right) \tag{7}$$

$$T_{ij}^{\mathrm{CPA}}(t) = \frac{R_{ij}(t)\cos\left(\varphi_{ij}^{\mathrm{R}}(t) - \beta_{ij}^{\mathrm{R}}(t) - \pi\right)}{|V_{ij}^{\mathrm{R}}(t)|}. \tag{8}$$

After collision risk parameters T_{ij}^{CPA} and D_{ij}^{DCPA} are known for ship i and j, it is important to identify when the collision risks among these two ships have been eliminated. When no collision risk exists between one ship and the rest of ships, this ship can terminate rudder steering and keep its current states. After the ship has passed its closest points of approach with the other ships, it can switch back to its original course.

For the two ships in Figure 4, maneuvering time T_{ij}^{m} needs to be determined for ship i to avoid collisions with ship j. Time T_{ij}^{m} represents the earliest time at which the closest distance between them is larger than the minimum safe distance $D_{ij}^{\mathrm{CPA}} \geq D^{\mathrm{safe}}$. This implies that at time T_{ij}^{m}, the ships have both reached relatively safe states because they are able to keep enough safe distance between one another afterwards. In addition, to avoid situations in which the close quarters situation happen long before the closest point of approach is actually reached, the relative distance between them should always be larger than the minimum safe distance D^{safe} during time T_{ij}^{m}, otherwise $T_{ij}^{m} = +\infty$. In other words, if any two ships i and j have already reached their closest point before the time when their $D_{ij}^{\mathrm{CPA}} \geq D^{\mathrm{safe}}$ when they take rudder angles s_i^* and s_j^* , the rudder angle pair (s_i^*, s_j^*) are infeasible for them.

At the end of this phase, values to variables D_{ij}^{CPA}, T_{ij}^{CPA}, T_{ij}^{m} are determined for any two ships $i, j \in V$, and that domain D_{ij}^{m} of variable T_{ij}^{m} has been constructed.

4. Distributed coordination based on distributed constraint optimization

Based on the ships' trajectories and collision risk parameters, the coordination model of multiple ships is formulated adopting the distributed constraint optimization (DCOP) formalism. Different optimization objectives are proposed and three solution algorithms are presented.

4.1. Distributed constraint optimization

A DCOP is defined as consisting of a set of agents, variables and constraints between variables that reflect the costs/utilities of assignments to variables. Control of values of variables in DCOPs is distributed, with agents only able to assign values to variables that they own. Furthermore, agents are assumed to know only the constraints involving variables that they own. The DCOP formalism has been mainly applied in meeting scheduling, coordination of sensors in networks, resource allocation in disaster evacuation, synchronization of traffic lights, and management of power distribution networks (Hosseini et al., 2013; Zivan et al., 2009; Kumar et al., 2009; Lass et al., 2008). The DCOP method has also been adopted to assist inland ships in finding optimal visiting sequences to different container terminals within a large seaport in our earlier work (Li et al., 2016). Distributed constraint optimization is well suited for formulating those problems since they are distributed by nature.

This paper adopts the DCOP formalism as defined in (Petcu, 2009), in which a DCOP is represented by a triple $\langle \mathcal{A}, \mathcal{COP}, \mathcal{R}^{\text{ia}} \rangle$, where:

- $\mathcal{A} = \{A_1, \ldots, A_M\}$ is a set of M agents;
- $\mathcal{COP} = \{COP_1, \ldots, COP_M\}$ is a set of disjoint, local Constraint Optimization Problems (COPs); COP_m is called the local sub-problem of agent A_m; COP_i is defined by a triple $\langle \mathcal{X}_m, \mathcal{D}_m, \mathcal{R}_i \rangle$, where $\mathcal{X}_m = \{X_{m1}, \ldots, X_{m|\mathcal{X}_m|}\}$ is a set of $|\mathcal{X}_m|$ variables that belong to A_m; $\mathcal{D}_m = \{d_{m1}, \ldots, d_{m|\mathcal{X}_m|}\}$ is a set of finite variable domains of the variables in $\mathcal{X}_m$; $\mathcal{R}_i = \{r_{m1}, \ldots, r_{m|\mathcal{R}_m|}\}$ is a set of $|\mathcal{R}_m|$ utility functions. These utility functions are used to represent objectives, as well as both hard and soft constraints. For hard constraints, the value of the utility function is 0 if the constraint is satisfied; otherwise the value is $-\infty$. For soft constraints, for different combinations of the values for variables, different values will be assigned to the utility functions.
- $\mathcal{R}^{\text{ia}} = \{r_1^{\text{ia}}, \ldots, r_{|\mathcal{R}^{ia}|}^{\text{ia}}\}$ is a set of so-called inter-agent utility functions defined over variables of multiple agents. Each $r_l^{\text{ia}} : scope(r_l^{\text{ia}}) \to \mathbb{R}$ expresses the utility for a joint decision obtained by the agents that have variables involved in r_l^{ia}. The agents that have variables can decide on the values of these variables involved in r_l^{ia} and are called "responsible" for r_l^{ia}. Inter-agent utility functions are considered known to all agents involved, i.e, those agents of which the local variables are part of the inter-agent utility function.

The objective of the agents solving a DCOP is to find the assignment to all variables such that the sum of values of all utility functions (representing the objectives, hard and soft constraints) are maximized. So, the agents determine:

$$X^* = \arg\max \sum_{m=1}^{M} \left(\sum_{v=1}^{|\mathcal{R}_m|} r_{mv}(X_{m1}, \ldots, X_{i|\mathcal{X}_m|}) \right) + \sum_{l=1}^{|\mathcal{R}^{ia}|} r_l^{\text{ia}}$$

.

Since variables from different agents can be constrained via inter-agent utility functions, to make sure these constraints represented by the inter-agent utility functions are satisfied and to find the optimal solution X^*, agents need to communicate and exchange messages. Those messages include information on the assignments of values to variables and the associated utility values.

In our case, the coordination model consists of a number of $|V|$ ship agents, and each ship agent i owns rudder angle variable δ_i, and rudder steering time variables T_{ij}^{m} and T_i^{steering}. The collision risks among ships when they select different rudder angles are represented via utility functions r_{ij}^{inter}, which is constructed based on variables T_{ij}^{m}, T_i^{steering} and T_{ij}^{CPA}. The intra-agent utility functions that represent the relations of T_{ij}^{m} and T_i^{steering} within each ship agent i are represented via utility functions r_i^{intra}.

4.2. *Formulating utility functions and optimization objective*

The inter-agent utility function r_{ij}^{inter} between any two ships $i, j \in V$, and the intra-agent utility function for each ship agent i are expressed as follows. Utility function r_{ij}^{inter} ensures that the rudder steering time should be consistent with the rudder angles that any two ship chooses. Utility function r_{ij}^{intra} is used to represent the time that ships spend in avoiding potential collisions.

$$r_{ij}^{\text{inter}} = \begin{cases} 0 : \text{if } \delta_i = s_i^*, \delta_j = s_j^*, T_{ij}^{\text{m}} = s_{ij}^{m*} & \forall s_i^* \in D_i, \forall s_j^* \in D_j, \forall s_{ij}^{m*} \in D_{ij}^{\text{m}}. \\ +\infty : \text{otherwise} \end{cases} \tag{9}$$

$$r_i^{\text{intra}} = \begin{cases} U_i^{\text{intra-ship}} : \text{if } \delta_i = s_i^*, T_{i1}^{\text{m}} = s_i^*, \cdots, T_{i|V|}^{\text{m}} = s_{i|V|}^{m*} & \forall s_i^* \in D_i, \forall s_{i1}^{m*} \in D_{i1}^{\text{m}}, \cdots, \forall s_{i|V|}^{m*} \in D_{i|V|}^{\text{m}}. \\ +\infty : \text{otherwise} \end{cases} \tag{10}$$

The optimization objective of the formulated problem is to minimize the sum of the utility values of the inter-agent and intra-agent utility functions, defined as

$$\delta^* = \arg\min\left(\sum_{i\in V}\sum_{j\in V, j\neq i} r_{ij}^{\text{inter}} + \sum_{i\in V} r_i^{\text{intra}}\right).$$

Utility value $U_i^{\text{inter-ship}}$ needs to be defined, so as to evaluate the efficiency of the collision avoidance decisions using the proposed utility functions. This paper considers three types of optimization objectives:

1. Obj_1: minimize the sum of rudder steering times of all involved ships. With this optimization objective, the value of $U_i^{\text{inter-ship}}$ equals the longest maneuvering time that ship agent i spends in avoiding collision with the other ships, which means $U_i^{\text{inter-ship}} = \max_{j\in V, j\neq i} T_{ij}^{\text{m}}$.
2. Obj_2: minimize the sum of the travel time to its closest point of approach of all involved ships. With this optimization objective, the value of $U_i^{\text{inter-ship}}$ equals the longest time that ship agent i spends in traveling to its CPAs with the other ships, which means $U_i^{\text{inter-ship}} = \max_{j\in V, j\neq i} T_{ij}^{\text{CPA}}$.
3. Obj_3: minimize the total time all the involved ships spend in avoiding collisions. With this optimization objective, the value of $U_i^{\text{inter-ship}}$ equals the sum of rudder steering time and the traveling time to ship i's CPAs with the other ships, which means $U_i^{\text{inter-ship}} = \max_{j\in V, j\neq i}\left(T_{ij}^{\text{m}} + T_{ij}^{\text{CPA}}\right)$.

4.3. Solution algorithms

Once the DCOP model has been established, a solution algorithm is required to solve the problem. Different DCOP algorithms define in different ways how the variable value assignments and the utility values are passed from one ship agent to another. The DCOP-based coordination strategy does not require a central controller to receive and send information from/to all the ship agents. DCOP solution algorithms can be categorized as complete and incomplete algorithms. Complete algorithms are guaranteed to find optimal solutions, if they exists. Complete algorithms typically do an exhaustive search over the problem space. This paper incorporates traditional complete DCOP algorithms, Synchronous Branch and Bound (SyncBB) (Hirayama and Yokoo, 1997), Dynamic Programming Optimization Protocol (DPOP) (Petcu, 2009), Asynchronous Forward Bounding (AFB) (Gershman et al., 2006) to solve the formulated problem.

4.3.1. SyncBB

The simplest and first complete DCOP algorithm is SyncBB (Hirayama and Yokoo, 1997), which is a straightforward distributed adaptation of the centralized branch-and-bound mechanism. In branch-and-bound, the search for the optimal solution aimed at guiding by a global accumulated cost named bound. SyncBB is a distributed version of branch-and-bound and aims to guide the search through a heuristic applied over the optimization objective.

The solution process of SyncBB is given in Algorithm 1. Firstly, assume that all variables and agents are arranged in a linear order with the priority $\delta_i \succ \delta_m \cdots \succ \delta_n$. The message passing starts with the highest priority variable δ_1, the ship agent that owns this variable sends a so-called single Current Partial Assignment (CPA) message that includes the value assigned to δ_1 and the associated utility value to the next ship agent (line 1-2). Each ship agent that receives the CPA extends it by including a value assignment of its own variable δ_m and the corresponding utility value based on the utility function it shares with other variable assignments appearing in the received CPA (line 7-8). Whenever a CPA reaches a new full assignment at the last ship agent, the accumulated utility value of the CPA is the utility value of the full variable assignment (line 13-15). This utility value will then be broadcast to all other ship agents, and each ship agent can use this utility value as an upper bound (UB). When the utility value of a new CPA exceeds the utility value of the current UB, it will be broadcast again as the new UB (line 10-12). Recursively, each ship agent that receives a CPA checks if its CPA accumulated utility value is larger than the UB. If this is true (line 5-6), implying that the previous variable assignment can be improved, ship agent m will send Backtrack message to the previous agent $m-1$ (line 17), and that the previous agent $m-1$, upon receiving Backtrack message (line 9), will choose the next value in its variable domain, and send it to agent m again. When a ship agent has checked all values in its domain, it broadcasts message TERMINATE to other ship agents (line 16). When the last ship agent has checked all values in its domain, the last discovered full assignment is reported as the optimal solution (line 20-21).

Algorithm 1 Distributed coordination steps of ship agents based on SyncBB.

Require: a linear ordering of variables $\delta_1 \succ \delta_m \cdots \succ \delta_n$

if $m = 1$ **then** $x_i^1 \leftarrow$ choose first $\delta_1^* \in D_1$ such that $u_1(\delta_1^*) < \infty$
  if there exists such a δ_1^* **then** send message (CPA, (δ_1^*), $u_1(\delta_1^*)$) to δ_2
  else broadcast messages INFEASIBLE
for each received message M, ship agent **do**
  if M = (UB,($\delta_1^*, \dots, \delta_n^*$), u) **then**
    $u^* \leftarrow u$ **and** record ($\delta_1^*, \dots, \delta_n^*$), u) as the best solution found so far
  if M = (CPA,($\delta_1^*, \dots, \delta_{m-1}^*$), u) **then**
    ($\delta_1, \dots, \delta_{m-1}$, u) $\leftarrow$ ($\delta_1^*, \dots, \delta_{m-1}^*$) **and** $\bar{u}^m \leftarrow u$
  if M = (Backtrack) **then** $D_m \leftarrow D_m \setminus \{\delta_m^*\}$
// Look for a better value for δ_m
$\delta_m \leftarrow$ find first $\delta_m^* \in D_m$ such that $\bar{u}_m + u_m(\delta_m^*, \cdot) > u^*$
  if there exists δ_m^* **then**
    if $m = n$ **then**
    Record ($\delta_1^*, \dots, \delta_n^*$) as the best solution found so far
    Broadcast message M = (UB,($\delta_1^*, \dots, \delta_n^*$), u^*)
      if D_n = empty **then** broadcast message TERMINATE
      else send message BACK to x_i^{n-1}
    else send message (CPA, ($\delta_1^*, \dots, \delta_m^*$, $\bar{u}_m + u_m(\delta_m^*, \cdot)$) to δ_{m+1}
  else
    if $m = n$ **then** broadcast message TERMINATE
    else send message BACK to δ_{m+1}

Algorithm 2 Distributed coordination steps of ship agents based on DPOP.

Phase 1: DFS structure generation: establish DFS
Run DFS generation algorithm, each variable is considered as a node and each agent controls a set of variables. Label nodes as parent/child nodes, edges are identified as tree/back edges.
Phase 2: UTIL propagation: bottom-up UTIL message propagation
//Join local utility functions:
$u(\delta_i, parent_{\delta_i}, \cdot) \leftarrow \sum_{u \in \{u' \in \mathcal{U} | \delta_i \in scope(u') \wedge scope(u') \cap (children_{\delta_i} \cup pseudo-children_{\delta_i}) = \varnothing\}} u(\delta_i, \cdot)$
//Join with received messages
for each $\delta_j \in children_{\delta_i}$, ship agent i **do**
  Wait for the message (UTIL,$u(\delta_j, \cdot)$)
  $u(\delta_i, parent_{\delta_i}) \leftarrow u_i(\delta_i, parent_{\delta_i}, \cdot) + u_j(\delta_j, \cdot)$
//Project out δ_i:
if δ_i is not the root variable **then**
  $\delta_i^*(parent_{\delta_i}, \cdot) \leftarrow \arg\max_{\delta_i}\{u(\delta_i, parent_{\delta_i}, \cdot)\}$
  $u(paren_{\delta_i}, \cdot) \leftarrow \max_{\delta_i} u(\delta_i, parent_{\delta_i}, \cdot)$
  Send the message (UTIL, $u(parent_{\delta_i}, \cdot)$) to $parent_{\delta_i}$
else $\delta_i^* \leftarrow \arg\max_{\delta_i}\{u(\delta_i)\}$
Phase 3: VALUE propagation: top-down VALUE message propagation
//Root variable sends its optimal value to children nodes
if δ_i is not the root variable **then** receive message (VALUE, $u(\delta_i, \delta_{parent_i}, \cdot)$) from parent of variable node δ_i
  Wait for message (VALUE, $u_{parent}^*, \cdot$) from parent
  $\delta_i \leftarrow \delta_i^*(u_{\delta_i} = u_{\delta_i^*}^*, \cdot)$
for each $\delta \in children_{\delta_i}$, ship agent i **do**
  Send the message (VALUE, δ_i^*, $sep_{\delta_i^*}^*$)

4.3.2. DPOP

DPOP (Petcu, 2009) is based on dynamic programming. The solution process of DPOP is given in Algorithm 2.

Firstly, a depth-first-search (DFS) structure is established using a distributed DFS algorithm. Each variable is considered as a node in this structure. Each ship agent controls a set of variables. The nodes are consistently labeled as parent or child nodes, and the edges that connect variable nodes are identified as tree or back edges. The second phase is called UTIL propagation, which involves the propagation of accumulated utility values from bottom to top through the constructed DFS tree. For each variable δ_i belonging to ship agent i, ship agent i joins all utility functions involving δ_i together, while ignoring all utility functions that involve at least one descendant of x_i^m (line 3-4). Here, a descendant node of a variable node refers to its children nodes or its children's children nodes.

Each ship agent i waits for the reception of a so-called UTIL message from each of the child nodes of $children_i$, and joins them all together with its intra utility function (line 6-8). Finally, agent i eliminates δ_i from the join, and sends the resulting utility value to its parent node of $parent_{\delta_i}$ (line 10-13). The maximum achieved utility for the whole subtree rooted at δ_i, as the function of the value for the parent node of δ_i and also the values for other variables that are higher than δ_i in the DFS tree (line 13). The set of variables in the UTIL message for variables sent by agent i is called the separator $sep(\delta_i)$, which includes δ_i's parent and pseudo-parents nodes, i.e., the ancestors of δ_i but not directly connected, as well as δ_i's descendant's pseudo-parent that are above δ_i in the DFS tree. Therefore, a UTIL message for variable δ_i contains the optimal utility value obtained in the subtree of each instantiation of separator sep_{δ_i}.

The third phase of DPOP is called VALUE propagation. At the end of the UTIL propagation phase, the root variable (the variable node that initiates the DFS tree, with itself as the root) has obtained its own local optimal value based on the messages it received (line 14). The ship agent that controls the value of the root variable sends this optimal value to each of the child nodes of the root variable via VALUE messages (line 15). Recursively, for each variable δ_i, when the corresponding agent receives a VALUE message from the parent node, it searches for its own optimal value assignments during the UTIL propagation phase (line 16-18). To each of the child nodes $children_i$ of i, ship agent i sends to them not only the optimal value of δ_i, but also the optimal values for all the variables in δ_i's children node's separator, altogether with the VALUE message agent i receives (line 19-20). Optimal decisions are hereby propagated down the DFS tree, until all variables have been assigned optimal values.

4.3.3. AFB

AFB (Asynchronous Forward Bounding) (Gershman et al., 2006) to solve the formulated problem. Algorithm 3 shows the distributed coordination steps of multiple ship agents based on AFB. Starting from the ship with largest number is the root node, and the ship with the smallest number is the top node. This is because the most risky ship will be more willing to change its rudder angle to avoid collision risks with other ships. Firstly, if ship agent m is the on the root node, it creates an empty CPA (Current Partial Assignment) message regarding the value assigned to its variable δ_m and starts process *assign_CPA* (line 22-28), and send CPA and the current utility value to higher agents. Procedure *assign_CPA* is used to find a value assignment for the current agent within the current bounds of the CPA.

Firstly, clear all estimates related to previous assignments (line 23). Then the agent try to assign every value in its domain, and the assigned value must make sure that the sum of the utility values in the current CPA and the estimate are smaller than the upper-bound (line 24). If so, the agent adds the selected value on the CPA (line 25-26). Otherwise, the assignments of some higher priority agents must be changed, and thus *backtrack* is performed (line 22). If the agent is the last agent, then a complete assignment will be reached, and it is broadcast to all agents through SOLUTION and CPA messages. If it is not the last agent, a CPA message will be send to the next agent (line 28).

When a ship agent receives the CPA from the lower ship agents (line 13), it adds the variable assignments in its local view only if the current utility value is smaller than the global lower-bound. Otherwise, procedure *backtrack* is carried out and the this agent sends the CPA to the previous agent to revise its variable assignment. A timestamp mechanism is adopted to determine the most up-to-date CPA and discard old CPAs. When the message is not discarded, the agent updates the timestamp of the CPA (line 14-15) and saves the received variable assignment (line 15). Only one agent performs one assignment on the CPA at a time. If the utility of received variable assignment exceeds the upper-bound, it performs backtrack (line 16), otherwise, *assign_CPA* is carried out to reassign another value to its variable (line 17).

Each agent adds its variable assignment in CPA and replies through FB_CPA messages for agent who do not have assignments in the current CPA. When the agent receives a FB_CPA message, the upper-bound is computed

Algorithm 3 Distributed coordination steps of ship agents based on AFB.

Require: a linear ordering of variables $\delta_1 \succ \delta_m \succ \cdots \succ \delta_n$
if $m = 1$, starting from ship m, **then** find the first value $\delta_m^* \in D_m$ such that $u_m(\delta_m^*) < \infty$, $\delta_m \leftarrow \delta_m^*$
    if there exists such a δ_m^* **then** send message (CPA, δ_m^*, $u_m(\delta_m^*)$) to higher ship agent n
    else broadcast messages INFEASIBLE
for each received message M **do**
    if M = (FB_CPA,$(\delta_1^*, \ldots, \delta_{m-1}^*)$, u) **then**
        if FB_CPA.$timestamp > (m-1).timestamp$ **then**
        $timestamp \leftarrow$ FB_CPA.$timestamp$; $estimate_m \leftarrow$ local utility + future utility for δ_m
        send message (FB_ESTIMATE, $(\delta_1^*, \ldots, \delta_{m-1}^*)$, m, $estimate_m, u$) to δ_m
    if M = (FB_ESTIMATE,$(\delta_1^*, \ldots, \delta_{m+1}^*)$,$m+1$, $estimate_{m+1}$) **then**
        if FB_ESTIMATE.$timestamp > (m-1).timestamp$, $m-1$) **then**
        $estimate_m \leftarrow estimate_{m+1}$,
            if u + all saved estimates $\leq$ UB **then** calls *assign_CPA* to change variable assignment
    if M = (CPA,$(\delta_1^*, \ldots, \delta_{m-1}^*)$, u) **then**
        if CPA.$timestamp > m.timestamp$ **then**
        $timestamp \leftarrow$ CPA.$timestamp$; $D'_m \leftarrow D_m$, $(\delta_1, \ldots, \delta_{m-1}, u) \leftarrow (\delta_1^*, \ldots, \delta_{m-1}^*)$, $\bar{u}_m \leftarrow u$,**and** update CPA
            if $\bar{u}_{m-1} \leq$ UB **then** backtrack
            else calls *assign_CPA*
    if M = (SOLUTION,$(\delta_1^*, \ldots, \delta_n^*)$, u) **then**
        if SOLUTION has not already been recorded **then**
        record assignments and optimal utility value
        broadcast TERMINATE
// procedure *assign_CPA*
    clear estimates
    iterate (from last assigned value) over D_m until found $v \in D_m$ such that δ_m + $estimate$(CPA,m,v) $\leq$ UB
        if no such value exists **then** backtrack
        else assign $\delta_m = v$
            if CPA is full assignment **then** $u^* \leftarrow u$, and broadcast (SOLUTION,CPA,u^*)
            else send CPA to δ_{m+1} **and** send (FB_CPA, δ_m, CPA) to ship agent $\forall n > m$
// procedure *backtrack*
    clear estimates
    if $m = n$ **then** broadcast TERMINATE
    else send CPA to δ_{m-1}

considering the utility of variable assignments on the CPA and the current agent (line 5-8). This estimate utility value is returned to the sender by a FB_ESTIMATE message (line 9-12). Upon receiving SOLUTION message, each agent checks to see if the SOLUTION has already been recorded, and record the variable assignments if necessary and optimal utility value, and terminates all agents (line 27).

5. Experimental results

Simulation experiments are carried out to assess and analyze the effectiveness of the proposed method. Firstly, an example of the proposed coordination structure, and an example of a 7-ships encounter situation and simulated trajectories of ships, as well as the distances between ships when they are operating to avoid collisions are given. Secondly, the performance of the adopted DCOP algorithms with different optimization objectives are evaluated with respect to computation time and communication costs. Finally, an analysis of the experimental results is given.

5.1. Experimental settings

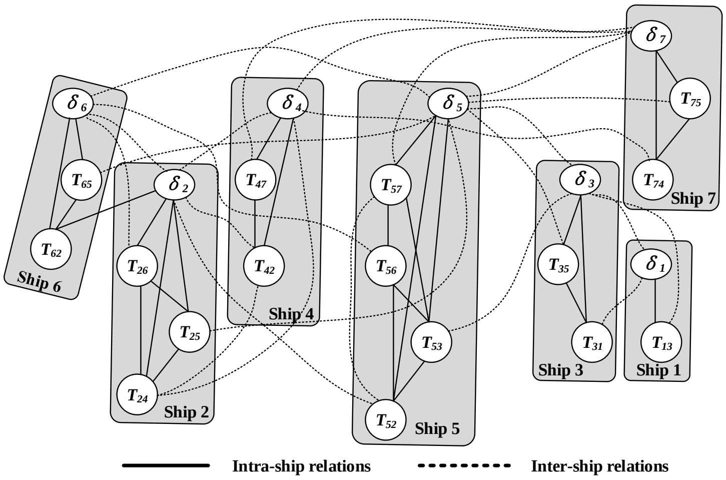
Our experiments are performed on an Intel Core i7-7500 CPU with 8GB RAM running Windows 10. The proposed method is implemented in MATLAB R2017a. The AFB algorithm is implemented with a latest version of the FRODO2 toolbox (version 2.16) (Léauté et al., 2009). This paper selects the KVLCC2 tanker as a sample ship and adopts the ship parameters from (SIMMAN 2008 committee, 2008). We set up 10 scenarios in which 7 homogeneous ships are encountering, with different courses and coordinates. Computation and communication costs for coordinating the ships' anti-collision decisions are calculated and evaluated based on these scenarios. The minimum safety distance that each two ships should keep is set as $D^{\mathrm{DCPA}} = 150m$. It is noted that due to the lack of real-world AIS data, this paper assumes that the ships that are approaching with one another within a circular area of 2.5km constitute a multiple ships coordination problem.

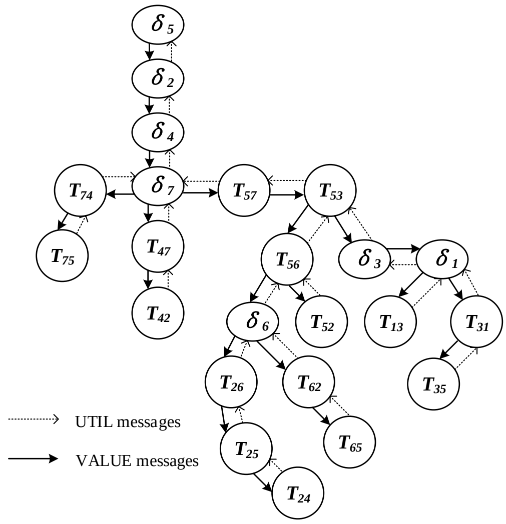
5.2. Coordination structures

Figure 5 presents an example that illustrates the coordination structure of the proposed approach. Each variable constitutes a node. These variable nodes are connected based on different structures: nodes in SyncBB and AFB are connected with a linear ordering; while the nodes in DPOP are connected with a depth-first-search structure. Based on the way the variable nodes are structured, the variable assignments and the corresponding utility values are passed from one node that is owned by an agent to the nodes owned by other agents. In Figure 5(a), Ship agent 1 owns variables δ_1 and T_{13}, this implies that Ship 1 only has collision risk with Ship 3. Similarly, Ship agent 5 owns variables $\delta_5, T_{57}, T_{56}, T_{53}$ and T_{52}, which means Ship 5 has collision risks with Ships 2, 3, 7 and 6. In addition, Ship Variables that are owned by the same ship agent are connected with utility function r_i^{intra}. Inter-agent utility function r_{ij}^{inter} connects variables that are owned by ship i and ship j, if collision risk exists between them. Figure 5(b) shows the DFS structure that connects different variables owned by different agents based on DPOP algorithm, and Figures 5(c) and 5(d) describes the linear structure that connects variables based on SyncBB and AFB algorithm. Information regarding each ship agent's currently chosen values for its variables and the corresponding utility values are propagated along with these structures.

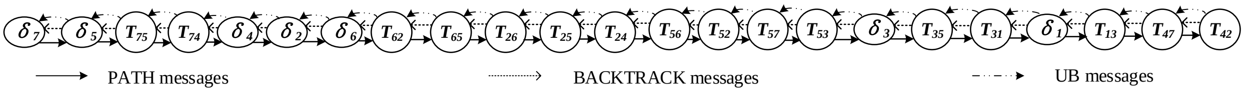
5.3. An example of optimized collision avoidance decisions

Figure 6 presents an extreme case in which 7 ships are encountering one another as an example of a multi-ships encountering situation. It shows the ships' predicted trajectories when they keep their original courses unchanged. The initial coordinates of ships are marked with $\circ$. After solving this problem with the proposed approach using DPOP algorithm with objective Obj_1, rudder angles and the rudder steering time for each ship has been obtained. Ships 1, 2, 3, 4, 5, 6 and 7 take rudder angle $\delta_1 = -20°$, $\delta_2 = -10°$, $\delta_3 = -10°$, $\delta_4 = -20°$, $\delta_5 = -20°$, $\delta_6 = -20°$ and $\delta_7 = -20°$, respectively. The rudder steering time for these ships is 145 seconds. The rudder steering time of all ships are equal because all 7 ships have collision risks with one another, and that the rudder steering time for them should be consistent. Figure 7 shows the simulated trajectories of ships with the optimized rudder angle alterations and operation time of rudder steering. The initial coordinate of each ship is marked with $\circ$, and the time at which rudder maneuvering ends is marked with $*$. The point at which each ship can pass the CPA with the other ships is

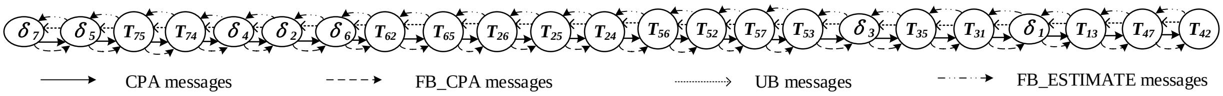




(a) Constraint graph of ship agents. (b) DPOP structure.



(c) SyncBB structure.



(d) AFB structure.

Figure 5: Coordination structure of the proposed approach

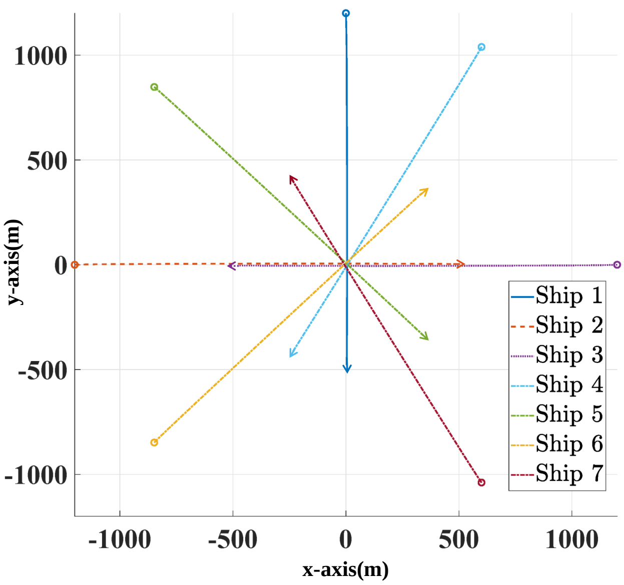


Figure 6: Simulated trajectories of 7 ships with original courses.

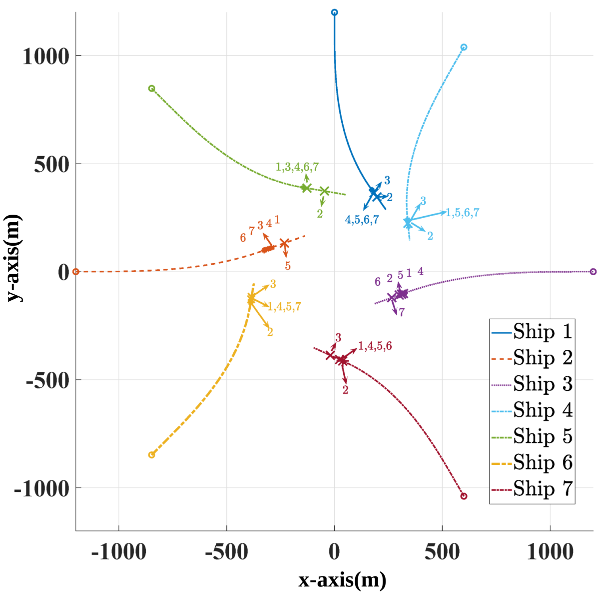


Figure 7: Optimized ship trajectories to avoid collisions.

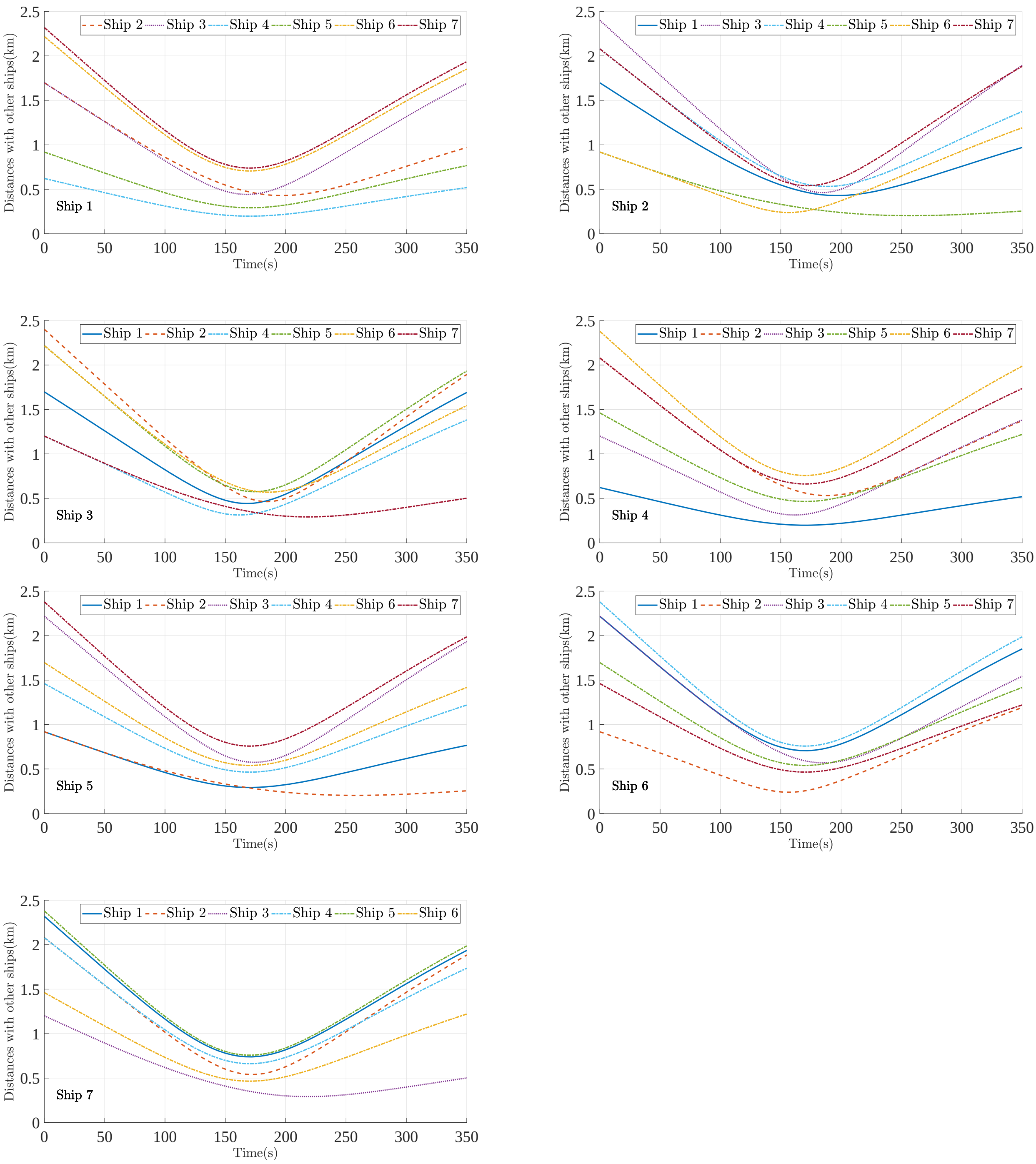


Figure 8: The distances between the involved ships.

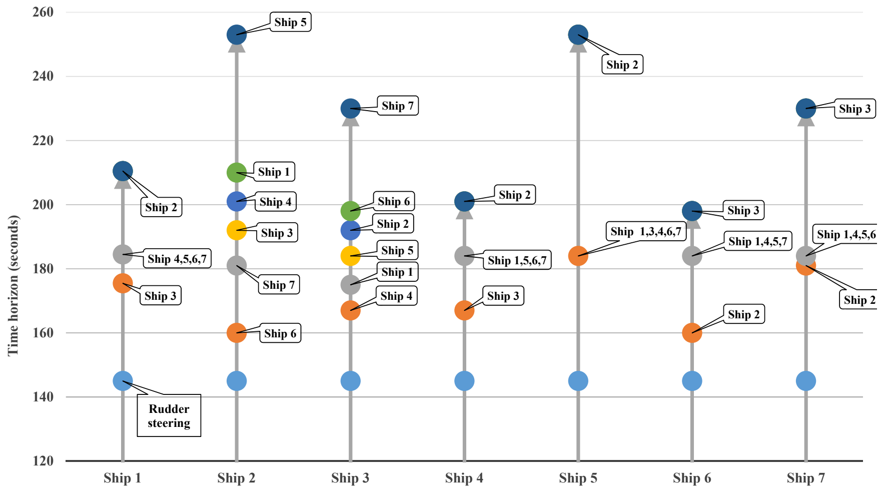


Figure 9: Ship's encountering time with each other.

marked with $\times$ along with the Ship ID. Figure 8 shows how the distances between the ships change over time. It can be seen that the minimum safety distance is kept by any two ships.

Figure 9 shows the rudder steering time for each ship and its encountering time with the other ships. As an example, it can be seen that Ship 1 passes its CPA with Ship 3 at time $T_{13}^{\text{CPA}} = 175s$, and then passes its CPA with Ships 4,5,6,7 simultaneously at time $T_{14}^{\text{CPA}} = T_{15}^{\text{CPA}} = T_{16}^{\text{CPA}} = T_{17}^{\text{CPA}} = 184s$, and finally passes the last ship, which is Ship 2 at time $T_{12}^{\text{CPA}} = 210s$. Similarly, Ship 2 passes its CPAs with the rest of the ships with a sequence of Ship 6 $\rightarrow$ Ship 7 $\rightarrow$ Ship 3 $\rightarrow$ Ship 4 $\rightarrow$ Ship 1, and finally passes the last ship, which is Ship 5 at time $T_{25}^{\text{CPA}} = 253s$. After each ship passes the CPA with the last ship, collision avoidance operation terminates and they can switch back to their original courses. Therefore, Ships 1, 2, 3, 4, 5, 6 and 7 can go back to their original courses at $210s, 253s, 230s, 201s, 253s, 198s, 230s$, respectively.

5.4. *Comparison of computation time*

Figure 10 shows the computation time for solving the DCOP. The values reported in the figure are the maximum, minimum and average of the computation times for solving the multiple ships collision avoidance problem with different solution algorithms and optimization objectives in the 10 cases. It can be seen that among the three solution algorithms, DPOP can solve the problem with the shortest time, while AFB solves the problem in relatively longer time, and that SyncBB required the longest computation time. In addition, among the three optimization objectives, objective Obj_2 requires the shortest computation time, while objective Obj_3 requires the longest computation time. This may be caused by the fact that Obj_3 takes both the TCPA and rudder steering time into account via utility functions, which means that the size of messages that include the information of utility values is larger that the size of such messages in objectives Obj_1 and Obj_2.

5.5. *Comparison of communication costs*

To evaluate the communication costs in the coordination of multiple ships, this section analyzes the number of size and messages that ships exchange, as well as the distribution of different types of messages during the information exchange.

5.5.1. *Number and size of messages exchanged during coordination*

In order to compare the performance of different solution algorithms and optimization objectives in terms of number and size of messages exchanged among ship agents during coordination, Figures 11, 12 and 13 summarize the total number of messages, the total amount of information (in bytes) and the size of largest message (in bytes) sent. It is noted that the values in these figures are ratios, which are calculated by dividing the total number of messages/total amount of information/largest message size that each algorithm with each optimization objective requires in each

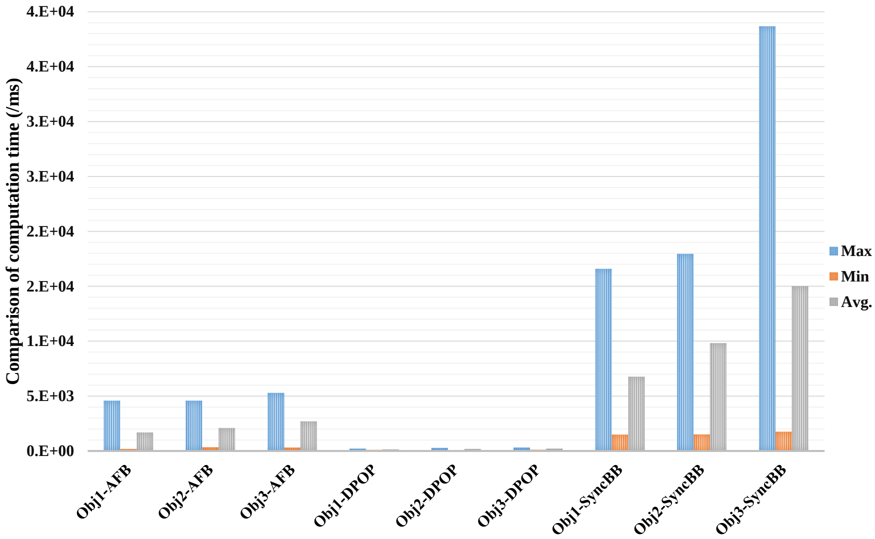


Figure 10: Comparison of computation time.

cases, by the minimum total number of messages/total amount of information/largest message size that are required by the algorithms and optimization objectives in that case.

The values reported in Figure 11 are the maximum, minimum and average values, with respect to the ratios of the total number of messages exchanged in each algorithm and optimization objective. The values reported in Figure 12 are the maximum, minimum and average values, with respect to the ratios of the total amount of information exchanged in each algorithm and optimization objective. The values reported in Figure 13 are the maximum, minimum and average values, with respect to the ratios of the largest message size sent/received in each algorithm and optimization objective. Figures 11 and 12 show that algorithm DPOP requires fewest number of messages and the least amount of information exchanged among ship agents, compared with algorithm AFB and SyncBB. Algorithm AFB requires the largest information exchange with respect to message numbers and sizes. This implies that the frequency of information exchange in ships' coordination with algorithm AFB are substantially higher than that of algorithm DPOP and SyncBB. Meanwhile, Figure 13 also shows that the largest message sizes that are required during coordination with DPOP are substantially larger than that of AFB and SyncBB. In addition, among different optimization objectives, it can be seen that objective Obj_1 requires less information exchange comparing with objectives Obj_2 and Obj_3.

5.5.2. Distribution of types of messages involved during coordination

Table 2 and Figure 14 conclude the message types in SyncBB and their distribution in the total number and sizes of messages in the information exchange between ship agents. It can be seen from Figure 14 that the main messages exchanged in SyncBB are PATH, ELECT and Backtrack, as they make up a large proportion in both the total number of messages (>99%) and amount of information (>95%) exchanged during coordination. This is because PATH messages include the current partial assignments to the variables, which are the basis of the information that ship agents exchange to find the variable assignments with the smallest utility values. ELECT messages are also important for each ship agent to determine next variable node to sent messages. A Backtrack message is generated whenever a ship agent cannot find any value assignments from its domain to make the current utility value smaller than the global upper bound, and that this agent needs to send information to the previous agent to make adjustments on its value assignments. The number of Backtrack messages depends on the domain sizes of each variable, if the variable domain is large, ship agents need to spend more efforts in searching for possible value assignments to make the utility values smaller than the global upper bound.

Table 3 and Figure 15 show the message types in DPOP and their distribution in the total number and sizes of messages in the information exchange between ship agents. It can be seen from Figure 15(a) that ParallelDFSwrapper

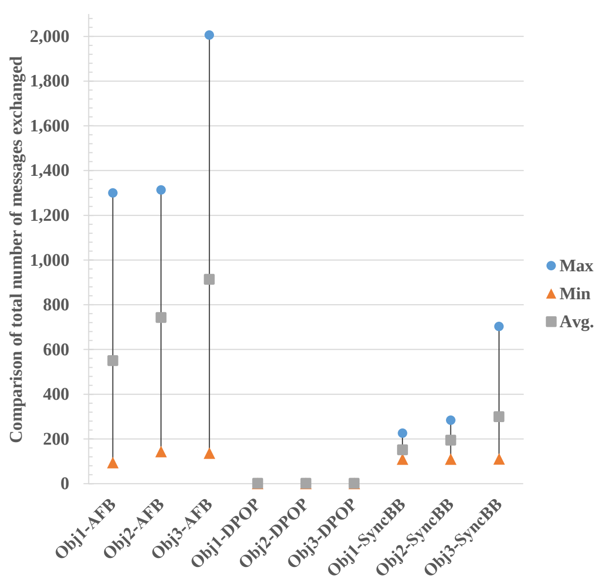


Figure 11: Comparison of number of messages exchanged during coordination.

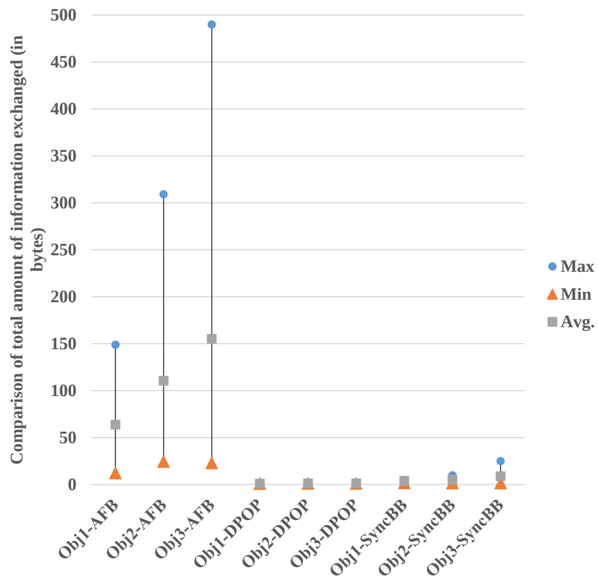


Figure 12: Comparison of amount of information exchanged during coordination.

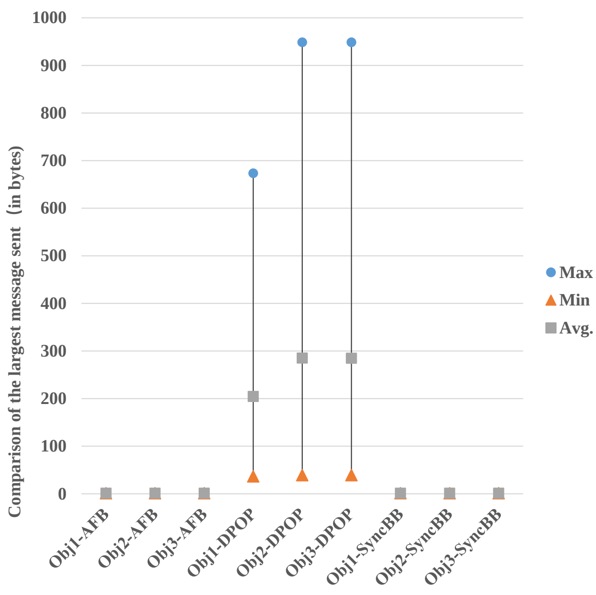
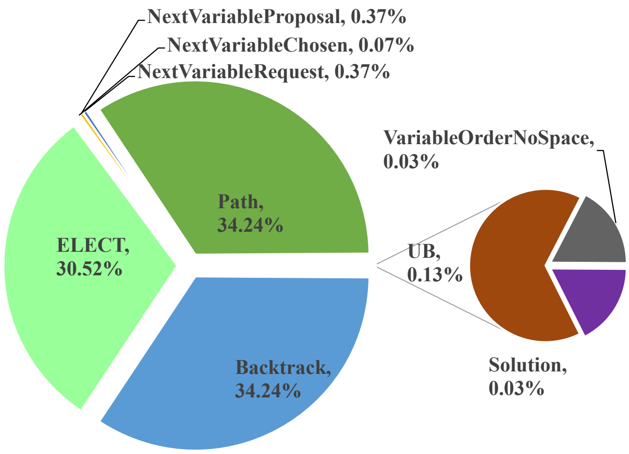


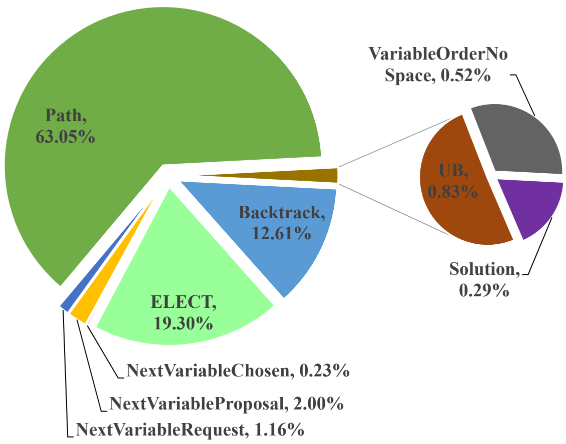
Figure 13: Comparison of size of largest messages required during coordination.

Table 2: Type of messages involved in SyncBB.

Message name	Definition
Backtrack	The backtrack messages
UB	The messages that broadcast by the last ship agent containing the current upper bound of utility values
SOLUTION	The chosen assignments to variables
PATH	The messages containing the current partial assignment
ELECT	The message containing a protocol to elect a variable/agent
NextVarChosen	The message containing the next variable chosen
NextVarProposal	The message containing a proposal for the next variable to put in the variable order
NextVarRequest	The message requesting a proposal for the next variable to put in the variable order
VarOrderNoSpace	The messages containing the chosen linear order of variables



(a) Distribution of different message types in total number of messages.



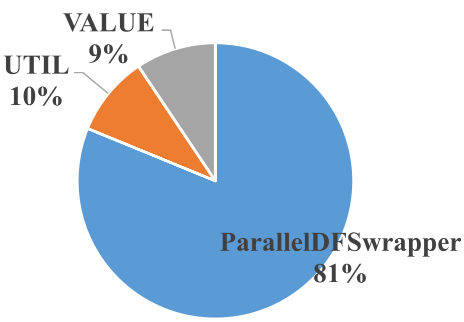
(b) Distribution of different message types in total amount of information.

Figure 14: Distribution of types pf messages involved during coordination with SyncBB in percentages.

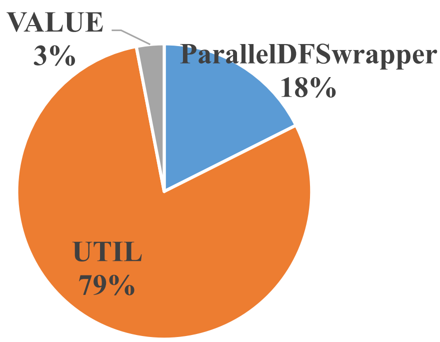
message constitute a large proportion of the total number of messages, due to the fact that ship agents need to constantly exchange this type of message to construct the DFS-tree structure. During the coordination of multiple ships based on DPOP, UTIL message include all possible utility values, which is ship rudder steering time (Obj_1), or time to closest point of approach (Obj_2), or the total time a ship spends in avoiding collisions (Obj_3); VALUE message include all possible rudder angles a ship selects. As the utility values are always larger than the values assigned to rudder angle variables, a UTIL message has a larger size than that of a VALUE message. Therefore, while UTIL message makes up a small proportion (10%) with respect to message numbers in Figure 15(a), it constitutes a large proportion regarding total amount of information in Figure 15(b).

Table 3: Type of messages involved in DPOP.

Message name	**Definition**
UTIL	The messages containing utility values
VALUE	The messages containing values assigned to variables
ParallelDFSwrapper	The messages containing DFS messages for a particular candidate variable root



(a) Distribution of different message types in total number of messages.



(b) Distribution of different message types in total amount of information.

Figure 15: Distribution of types pf messages involved during coordination with DPOP in percentages.

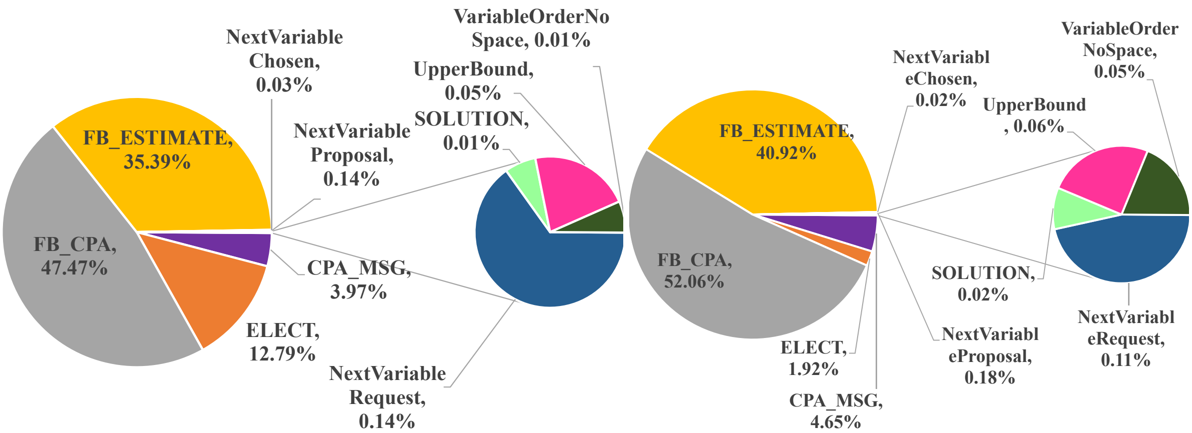
Table 4 and Figure 16 show the message types in AFB and their distribution in the total number and sizes of messages in the information exchange between ship agents. It can be seen that CPA message only occupies a small proportion in the information exchange and that FB-CPA and FB-ESTIMATE messages constitute the major proportion with respect to both number and size of information exchanged. This is because FB-CPA messages are the extension of CPAs, which are constantly sent from one ship agent to the other agents in order to dynamically calculate the upper bound of the utility values, and the other agents return the estimated utility values via FB-ESTIMATE messages.

5.6. Result analysis

Firstly, the proposed approach is able to provide ships with optimal decisions on rudder angle alteration and the rudder steering time. Secondly, DPOP-based coordination is able to find optimal solutions in a shorter time than AFB- or SyncBB-based coordination. Meanwhile, DPOP-based coordination also requires a lower communication frequency and less amount of information exchange than the other two algorithms. Nevertheless, DPOP-based coordination requires messages in larger sizes. For situations in which the communication bandwidth is limited, DPOP may not be applicable. Under this circumstance, AFB-based coordination would be more suitable, as its message sizes are smaller and the computation time is also acceptable. In addition, among optimization objectives Obj_1, Obj_2 and Obj_3, objective Obj_1 is relatively better, as it has lower requirements on information exchange and needs shorter computation time to find optimal solutions.

Table 4: Type of messages involved in AFB.

Message name	Definition
CPA	The message containing the current partial assignment of variables
FB-CPA	The messages containing CPA that will be sent from a ship agent to a destination receiver
FB-ESTIMATE	The estimated lower bound of the utility value returned from the receiver to the sender ship agent
SOLUTION	The chosen assignments to variables
UB	The messages that broadcast by the last ship agent containing the current upper bound of utility values
ELECT	The message containing a protocol to elect a variable/agent
NextVarChosen	The message containing the next variable chosen
NextVarProposal	The message containing a proposal for the next variable to put in the variable order
NextVarRequest	The message requesting a proposal for the next variable to put in the variable order
VarOrderNoSpace	The messages containing the chosen linear order of variables



(a) Distribution of different message types in total number of messages.

(b) Distribution of different message types in total amount of information.

Figure 16: Distribution of types pf messages involved during coordination with AFB in percentages.

6. Conclusions and future work

This paper proposes a distributed coordination strategy for assisting ships in making decisions on the most efficient anti-collision operations when multiple ships encounter with one another and that collision risks exist among them. The proposed method considers both the ship dynamics and the inter-related characteristic of the anti-collision decision making. Based on the ships' current states, the proposed method provides them with the rudder angles they should choose and the corresponding rudder steering time. In addition, the type, number, and size of messages that are calculated from the experiments can provide insight for practitioners regarding the communication costs to implement such coordination strategies. This could increase the safety and reliability of a ships automated navigation, reduce the psychological and physical burden of ship operators, and reduce the occurrence of ship collisions.

To enhance the applicability of the proposed method, further research is required. Firstly, its compatibility with COLREGs should be extensively investigated, considering the fact that it is the foundation of ship collision avoidance today. The integration of COLREGs in intelligent collision avoidance method is critical to ensure a smooth transitioning from manned ships to future unmanned ships. Secondly, this paper deals with the multiple ships collision avoidance problem from a distributed decision making perspective and does not consider the heading control or tracking control regarding courses and trajectories of ships.

To further improve the efficiency of the anti-collision operations, advanced heading control algorithms should be integrated with the coordination strategy, so that the ships can finish rudder steering operations more efficiently and accurately. In addition, this paper adopts TCPA/DCPA as the main collision risk indicators, it would be interesting to also consider other risk indicators and compare their performances in the future. Moreover, extensive experiments based on real-world are required to validate the practical effectiveness and applicability of the proposed method. Last but not least, it is also interesting to consider applying the proposed method to solve similar collision avoidance problems in restricted waters.

Acknowledgments

Supported by National Natural Science Foundation of China (51709217) , Hubei Provincial Natural Science Foundation of China (2018CFB640) and State Key Laboratory of Ocean Engineering (Shanghai Jiao Tong University) (Grant No. 1707).

References

Escario, J.B., Jimenez, J.F., Giron-Sierra, J.M., 2012. Optimisation of autonomous ship manoeuvres applying ant colony optimisation metaheuristic. Expert Systems with Applications 39, 10120–10139.

Gershman, A., Meisels, A., Zivan, R., 2006. Asynchronous forward-bounding for distributed constraints optimization. Frontiers in Artificial Intelligence and Applications 141, 103–107.

Hirayama, K., Yokoo, M., 1997. Distributed partial constraint satisfaction problem, in: Proceedings of the 3rd International Conference on Principles and Practice of Constraint Programming. Springer, Linz, Austria, pp. 222–236.

Hornauer, S., Hahn, A., Blaich, M., Reuter, J., 2015. Trajectory planning with negotiation for maritime collision avoidance. TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation 9, 335–341.

Hosseini, S., Samaneh, Basir, O.A., 2013. Target to sensor allocation: A hierarchical dynamic distributed constraint optimization approach. Computer Communications 36, 1024–1038.

Johansen, T.A., Perez, T., Cristofaro, A., 2016. Ship collision avoidance and colregs compliance using simulation-based control behavior selection with predictive hazard assessment. IEEE Transactions on Intelligent Transportation Systems PP, 1–16.

Kim, D., Hirayama, K., Okimoto, T., 2017. Distributed stochastic search algorithm for multi-ship encounter situations. Journal of Navigation 70, 1–20.

Kim, D.G., Hirayama, K., Okimoto, T., 2015. Ship collision avoidance by distributed tabu search. Transnav,the International Journal on Marine Navigation and Safety of Sea Transportation 9, 23–29.

Kumar, A., Faltings, B., Petcu, A., 2009. Distributed constraint optimization with structured resource constraints, in: Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems, Budapest, Hungary. pp. 923–930.

Lass, R.N., Kopena, J.B., Sultanik, E.A., Nguyen, D.N., Dugan, C.P., Modi, P.J., Regli, W.C., 2008. Coordination of first responders under communication and resource constraints, in: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems, Estoril, Portugal. pp. 1409–1412.

Lazarowska, A., 2014. Ant colony optimization based navigational decision support system. Procedia Computer Science 35, 1013–1022.

Lazarowska, A., 2017a. Multi-criteria aco-based algorithm for ships trajectory planning. Transnav, the International Journal on Marine Navigation and Safety of Sea Transportation 11, 31–36.

Lazarowska, A., 2017b. A new deterministic approach in a decision support system for ships trajectory planning. Expert Systems with Applications 71, 469 – 478.

Léauté, T., Ottens, B., Szymanek, R., 2009. FRODO 2.0: An open-source framework for distributed constraint optimization, in: Proceedings of the IJCAI'09 Distributed Constraint Reasoning Workshop (DCR'09), Pasadena, California, USA. pp. 160–164. https://frodo-ai.tech.

Li, S., Negenborn, R.R., Lodewijks, G., 2016. Distributed constraint optimization for addressing vessel rotation planning problems. Engineering Applications of Artificial Intelligence 48, 159–172.

Liu, J., Hekkenberg, R., Quadvlieg, F., Hopman, H., Zhao, B., 2017. An integrated empirical manoeuvring model for inland vessels. Ocean Engineering 137, 287 – 308.

Liu, Y., Bucknall, R., 2015. Path planning algorithm for unmanned surface vehicle formations in a practical maritime environment. Ocean Engineering 97, 126–144.

MarineTraffic, 2018. A scrrenshot of the maritime traffic near Yangshan Port in Shanghai, P. R. China on the website of Marine Traffic. Accessed online through https://www.marinetraffic.com/ on September 10th 14:00pm, 2018.

Mohamed-Seghir, M., 2012. The branch-and-bound method and genetic algorithm in avoidance of ships collisions in fuzzy environment. Polish Maritime Research 19, 45–49.

Naeem, W., Irwin, G.W., Yang, A., 2012. Colregs-based collision avoidance strategies for unmanned surface vehicles. Mechatronics 22, 669–678.

Perera, L.P., Carvalho, J.P., Soares, C.G., 2011. Fuzzy logic based decision making system for collision avoidance of ocean navigation under critical collision conditions. Journal of Marine Science and Technology 16, 84–99.

Perera, L.P., Ferrari, V., Santos, F.P., Hinostroza, M.A., Soares, C.G., 2015. Experimental evaluations on ship autonomous navigation and collision avoidance by intelligent guidance. IEEE Journal of Oceanic Engineering 40, 374–387.

Petcu, A., 2009. A class of algorithms for distributed constraint optimization. Ph.D. thesis. École Polytechnique Fédérale de Lausanne. Lausanne, Switherland.

SIMMAN 2008 committee, 2008. MOERI Tanker KVLCC2: Geometry and test conditions. Accessed online from http://www.simman2008.dk/KVLCC/KVLCC2/tanker2.html, June 1, 2018.

Simsir, U., Amasyal, M.F., Bal, M., elebi, U.B., Ertugrul, S., 2014. Decision support system for collision avoidance of vessels. Applied Soft Computing 25, 369–378.

Szlapczynska, J., 2015. Data acquisition in a manoeuver auto-negotiation system. TransNav Journal 9, 343–348. doi:10.12716/1001.09.03.06.

Szlapczynski, R., 2011. Evolutionary sets of safe ship trajectories: A new approach to collision avoidance. Journal of Navigation 64, 169181.

Szlapczynski, R., 2013a. Evolutionary sets of safe ship trajectories within traffic separation schemes. Journal of Navigation 66, 65–81.

Szlapczynski, R., 2013b. Evolutionary ship track planning within traffic separation schemes evaluation of individuals. Transnav, the International Journal on Marine Navigation and Safety of Sea Transportation 7, 301–308.

Szlapczynski, R., 2015. Evolutionary planning of safe ship tracks in restricted visibility. Journal of Navigation 68, 39–51.

Szlapczynski, R., Szlapczynska, J., 2016. An analysis of domain-based ship collision risk parameters. Ocean Engineering 126, 47–56.

Tam, C.K., Bucknall, R., 2010. Path-planning algorithm for ships in close-range encounters. Journal of Marine Science and Technology 15, 395–407.

Tam, C.K., Bucknall, R., 2013. Cooperative path planning algorithm for marine surface vessels. Ocean Engineering 57, 25–33.

Tam, C.K., Bucknall, R., Greig, A., 2009. Review of collision avoidance and path planning methods for ships in close range encounters. Journal of Navigation 62, 455–476.

Wang, X., Liu, Z., Cai, Y., 2017. The ship maneuverability based collision avoidance dynamic support system in close-quarters situation. Ocean Engineering 146, 486–497.

Xu, Q., Wang, N., 2014. A survey on ship collision risk evaluation. Promet - Traffic - Traffico 26, 475–486.

Zhang, J., Zhang, D., Yan, X., Haugen, S., Soares, C.G., 2015. A distributed anti-collision decision support formulation in multi-ship encounter situations under colregs. Ocean Engineering 105, 336–348.

Zhen, R., Riveiro, M., Jin, Y., 2017. A novel analytic framework of real-time multi-vessel collision risk assessment for maritime traffic surveillance. Ocean Engineering 145, 492–501.

Zivan, R., Glinton, R., Sycara, K., 2009. Distributed constraint optimization for large teams of mobile sensing agents, in: Proceedings of the 2009 International Conference on Web Intelligence and Intelligent Agent Technology, Milano, Italy. pp. 347–354.