Front. Eng. Manag. 2025, 12(4): 1196–1219
https://doi.org/10.1007/s42524-025-4173-4

RESEARCH ARTICLE

Zhiwei CHEN, Luogeng ZHANG, Jiayun CHU, Xiaotong FANG, Hongyan DUI

Computational resource configuration analysis and optimization methods for unmanned system considering intended functionality safety

Abstract With the rapid expansion of unmanned system capabilities, integrating and sharing computing resources has become essential. In addition to enhancing resource utilization efficiency, this architecture may also introduce conflicts related to resource competition. Therefore, effective resource-sharing configurations are crucial to ensure the Safety of the Intended Functionality (SOTIF). This paper proposes a computing resource configuration analysis and optimization methods for SOTIF. First, four SOTIF requirements are explored using the computing resource-sharing architecture for unmanned systems, encompassing computing time, computing power, energy consumption restrictions, and mutual exclusion and correlation. Secondly, the computing resource configuration model and its SOTIF constraints are formalized based on the graph and set theories. Subsequently, this study divides the design process of computing resource configuration schemes into resource selection and allocation. It introduces a resource selection optimization method based on Forward Checking and a resource allocation optimization method based on NSGA-II. Finally, a typical unmanned driving scenario is considered as an example, and the optimal resource selection and allocation schemes are sequentially determined using the proposed method on the computing platform.

Keywords safety analysis, unmanned system, safety of the intended functionality, computational resource allocation, optimization.

Received Sep. 15, 2024; revised Feb. 5, 2025; accepted Feb. 11, 2025

Zhiwei CHEN, Luogeng ZHANG
Unmanned systems research institute, Northwestern Polytechnical University, Xi'an 710072, China; National Key Laboratory of Unmanned Aerial Vehicle Technology, Xi'an 710129, China; Integrated Research and Development Platform of Unmanned Aerial Vehicle Technology, Xi'an 710072, China

Jiayun CHU (✉)
China Academy of Launch Vehicle Technology, Beijing 100076, China
E-mail: bhcjy@buaa.edu.cn

Xiaotong FANG
China Institute of Marine Technology & Economy, Beijing 100081, China

Hongyan DUI
School of Management, Zhengzhou University, Zhengzhou 450001, China

This work was supported by the National Natural Science Foundation of China (Grant Nos. 72301296, 72471192, 72101270 and U2341213).

1 Introduction

Automation has rapidly increased across multiple industries worldwide (Guo et al., 2021; Khastgir et al., 2021). Unmanned systems, including aerial and underwater variants (Zhao et al., 2023; Chen et al., 2024), not only enhance performance in various fields but also provide significant economic and environmental benefits. However, with these advancements comes increased complexity, leading to a rise in accidents and safety concerns (Chen et al., 2025; de Koning et al., 2024). Ensuring the safety of the intended functionality (SOTIF) has become crucial as functional inadequacies and performance

limitations of unmanned systems become more apparent. Although the ISO/PAS 21448 standard for "Road Vehicles Safety of the Intended Functionality" was introduced in 2019, it lacks specific technical guidance, underscoring the need for further research (Schnellbach and Griessnig, 2019).

The SOTIF process is a hot topic for unmanned systems which mainly deals with the unreasonable risk due to those hazards caused by performance limitations. Some recent fatalaccidents involving autonomous vehicles have revealed that unmanned systems are highly sensitive to safety hazards due to performance limitations rather

than traditional hardware or software faults or errors (Board, 2019; Pimentel, 2019). In these accidents, the performance boundary is exceeded at a certain point, namely the unmanned system is no longer competent for itscurrent job. However, users are often much overconfident in these smart unmanned systems and thus neglect the signs of danger.With the large-scale use of intelligent technology, similar risks are increasingly troubling designers. Therefore, it is vital to answer the question of whether the intended function of an unmanned system could be declared safe, which is exactly the ultimate goal of SOTIF design. More and more researchers are starting to delve into the SOTIF field. Kinalzyk (2021) detailed the differences between SOTIF and traditional functional safety work and proposed an integrated analysis approach. Skoglund et al. (2021) comprehensively analyzed the work process and focusedon three types of problems: SOTIF, functional safety, and information security, and provided a safety work arrangement for the entire development cycle of the system. Chelouati et al. (2023) explored the case of automatic train safety and security and proposed a GSN-based high-level framework. Grabbe et al. (2020) recommended using FRAM for risk assessment in developing highly automated vehicles, aiming to provide system design suggestions and insights for validation work. Overall, this study aims to extend the existing safety work process to cover the research scope of SOTIF. Similar work generally emphasizes the non-fault features of SOTIF-related hazards, which has reference significance for a deeper understanding of the SOTIF mechanism.

In addition, many researchers are currently committed to analyzing the SOTIF of an unmanned system through case testing. For example, Neurohr et al. (2020) proposed 16 factors to consider in scenario-based unmanned driving testing, including scenario generation, requirement generation, testing bias, testing execution, and testing confirmation. Zhou et al. (2022a) proposed a unified scenario description language and evaluation standard, which can support the expansion of test scenarios and improve data utilization efficiency. In analyzing SOTIF based on case testing, most studies use statistical indicators such as accident rates and takeover request frequencies (corresponding to situations where the system's intended functionality is insufficient) as the basis for the final judgment. Birch et al. (2020) proposed a state machine to structure an argument concerning SOTIF. Cai et al. (2023) predicted the motion behavior of the preceding vehicle based on historical data and improved Markov model. Collin et al. (2020) linked the Rulebooks framework and the SOTIF process. Hu et al. (2022) proposed a novel and implementation-independent SOTIF validation strategy through real-world examples. Jiménez et al. (2023) demonstrated how the integration of the SOTIF process within an existing validation tool suite can be achieved. However, Kalra and Paddock (2016) have proven that at least 100 unmanned systems must be driven continuously for 12.5 years to demonstrate that unmanned systems have the same safety level as existing vehicles. Although many efforts have been made to improve case testingefficiency, the cost of analyzing SOTIF through pure experimental means is prohibitively high. There is a strong demand for considering SOTIF requirements in forward design of the unmanned systems.

Meanwhile, many research scholars have focused on SOTIF and its association with human-machine interactions. For example, Yan et al. (2021) proposed a SOTIF evaluation strategy for safety supervision to support the evaluation of driver errors and lane deviation risks caused by driver errors. Zhang et al. (2021) discussed the effectiveness of system theoretical process analysis and cognitive task analysis methods in addressing SOTIF issues in human factors engineering. Abdulazim et al. (2021) proposed a contextual-based predictive ML model to monitor the intervention between the driver and lane keep assist system. Rau et al. (2019) reviewed the SOTIF process, which described the development of a framework for deriving scenarios. It should be noted that SOTIF issues related to human-machine interaction usually only exist in low-level unmanned systems, which do not apply to high-level unmanned driving (SAE L4 and L5) because the necessary takeover actions by human drivers are no longer required when the unmanned driving function is activated.

SOTIF refers to the state description of whether the unmanned capability meets the actual operating needs under non-fault conditions.Thus, research on operating capabilities in specific operating tasks can also be classified into the scope of consideration for SOTIF. For example, Esterle et al. (2019) used linear logic to judge driving decisions during the operational phase to monitor whether they comply with preset traffic rules.Wang et al. (2022) proposed a robust non-fragile fault-tolerant control strategy as a quantitative risk method for the adaptive cruise control function for ensuring SOTIF. Chu et al. (2023) proposed a method to evaluate SOTIF-oriented perception effectiveness for forward obstacle detection of unmanned systems. Zhang et al. (2019) proposed intended safety systems for intelligent driving, incorporating SOTIF concepts to analyze and evaluate the driving scene and system safety. Luo et al. (2022) proposed a fuzzy reasoningevaluation method based on an analysis of scenarios and elements. Zhou et al. (2022b) investigated the safety of autonomous emergency braking (AEB) perception systems, providing crucial insights for optimizing AEB perception system parameters.

Briefly, the SOTIF concept is a challenging field that is directly related to the type of function being considered in unmanned systems. Although there are many conceptual discussions about SOTIF, there are relatively few forward design methods for specific functions.This paper presents an SOTIF analysis and optimization for

unmanned systembased on computational resource allocation, the main contributions of this study are summarized as follows:

1) Computational resource allocation scheme modeling with multidimensional SOTIF constraints based on graph and set theory: This study constructs a comprehensive model for computational resource allocation in unmanned systems using graph and set theory. It considers task relationships, communication requirements, and SOTIF constraints, enabling a systematic analysis to ensure safety.

2) Optimization method for computational resource selection based on Forward Checking: This study introduces an optimization method for computational resource selection using Forward Checking. It boosts efficiency by narrowing the search space and swiftly identifying optimal allocation schemes that meet SOTIF constraints.

3) Optimization method for computational resource allocation based on NSGA-II: This study uses NSGA-II to optimize computational resource allocation, considering various objectives, such as load distribution and communication cost, while meeting SOTIF constraints. The NSGA-II generates Pareto-optimal solutions, enabling efficient resource use and safety.

This study introduces an innovative framework for analyzing and optimizing computational resource allocation in unmanned systems, enhancing both their efficiency and safety. The paper is organized as follows: Section 2 outlines the objectives and problem definition. Section 3 discusses the SOTIF analysis method based on computational power requirements. Section 4 details the comprehensive optimization methods for resource selection considering SOTIF and computational power costs. Section 5 details the comprehensive optimization method for resource allocation considering SOTIF and rational computational power allocation. Section 6 presents a case study, and Section 7 concludes the study with key findings.

2 Object description and problem description

Unmanned systems are intelligent systems that perceive their environment and make driving decisions autonomously. For specific unmanned driving tasks, the resource entities involved can be categorized into four types: sensors, communication units, processors, and actuators, as shown in Fig. 1. During task execution, sensors capture and preprocess environmental data. This data is transmitted through communication units to processors, where specific applications perform further computations. After multiple processing stages, driving commands are sent via communication units to actuators, which adjust the vehicle's behavior in real-time.

The design process for the computational resource allocation scheme in unmanned systems is illustrated in Fig. 2. It begins with a functional logic model derived from breaking down the driving tasks, which defines the target functions and their interactions. Various applications and messages within this model act as virtual resources deployed on computational platforms, serving as key inputs for the allocation scheme. Additionally, information about available hardware resources is a critical input for this design process. Based on these foundations, the computational resource allocation design can be divided into two key challenges:

1) Hardware Resource Selection Problem: This involves selecting an appropriate combination of hardware entities from the available resource pool to form a computational platform system. This system must support at least one configuration that satisfies SOTIF requirements.

2) Functional Logic Architecture and Computational Platform Mapping Problem: This focuses on balancing multiple design objectives to find the optimal allocation among all configurations that meet SOTIF requirements.

To address these challenges, this study proposes a computational resource allocation method based on graph and set theory. The method leverages functional logic and resource entity models to evaluate resource sharing and constraints related to SOTIF. The optimization process is divided into two stages: resource selection and allocation scheme optimization. Forward Checking and NSGA-II algorithms are employed for optimization. Comprehensive case studies validated the method's effectiveness, demonstrating significant improvements in resource allocation efficiency.

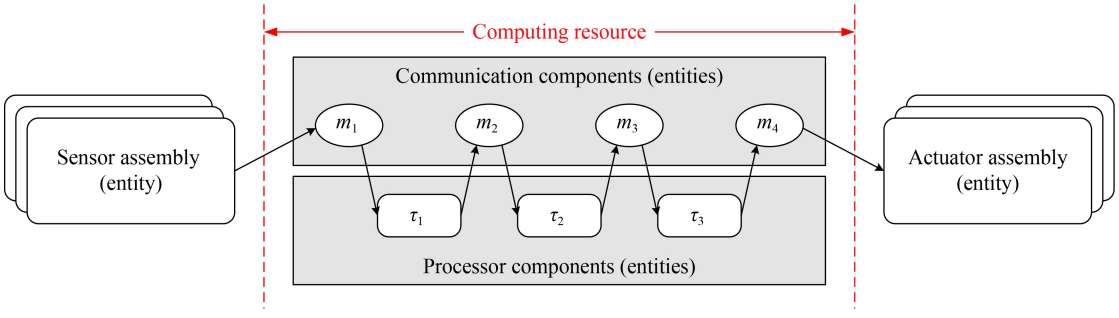

Fig. 1 Examples of resources related to unmanned driving.

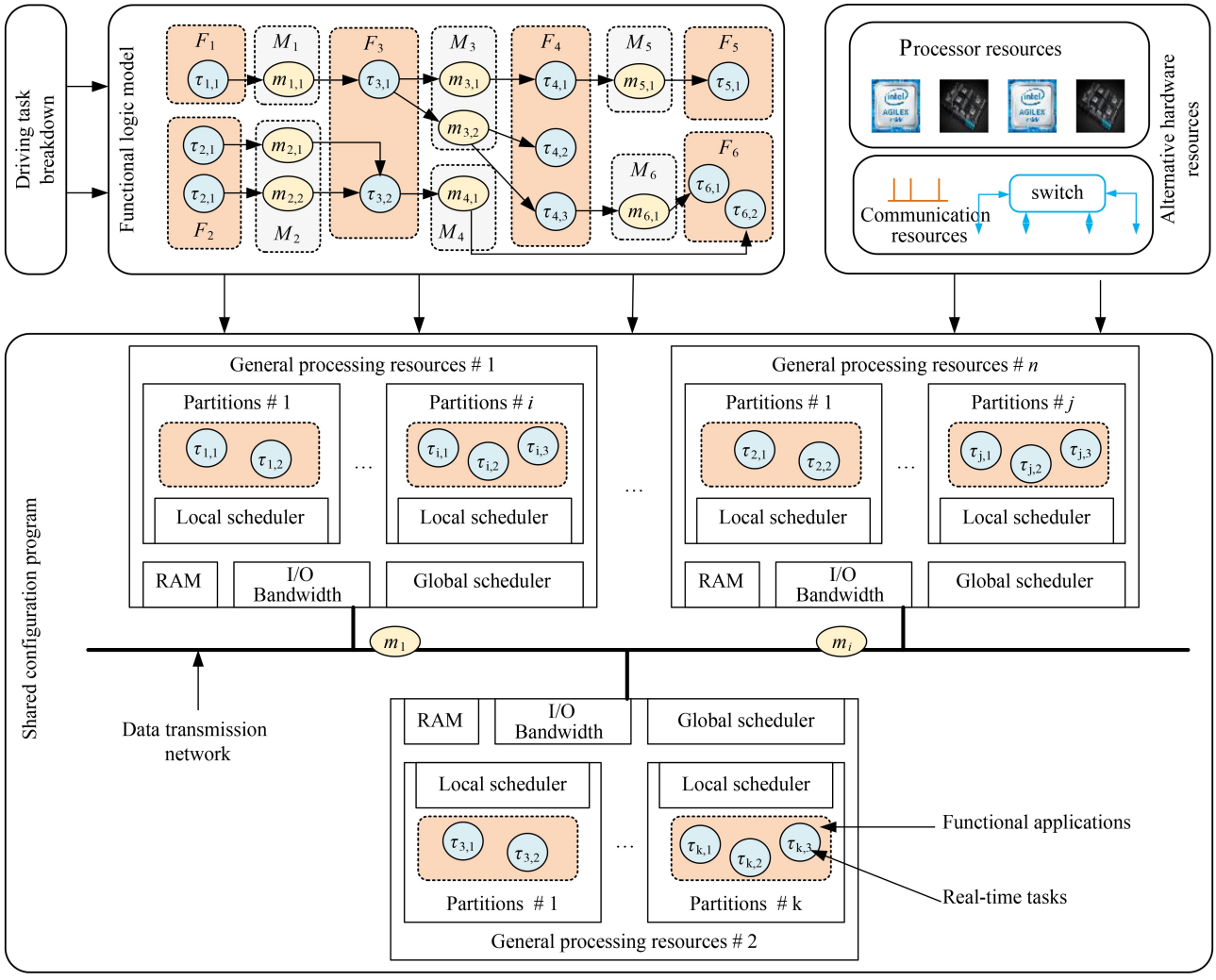

Fig. 2 Computational Resource Allocation Program design process.

3 SOTIF analysis based on computational power requirements

This section performs formal modeling for the computational resource allocation scheme and its related SOTIF requirements to support the subsequent analysis and optimization work for SOTIF requirements.

3.1 The functional logic model

After decomposing and integrating into reusable functions, autonomous driving tasks are represented as a series of executable applications. Communication between these applications ensures task progress. As shown in Fig. 3, a typical functional logic model depicts these relationships. In this study, a quintuple $\langle F, M, E, FM, MF\rangle$ represents the functional logic relationship for autonomous driving tasks, as follows:

$$
\begin{aligned}
&F = \{F_i | 1 \leqslant i \leqslant |F|\},\\
&M = \{M_j | 1 \leqslant j \leqslant |M|\},\\
&E = \begin{pmatrix} 0 & \cdots & e_{1,|F|} \\ \vdots & \ddots & \vdots \\ e_{|F|,1} & \cdots & 0 \end{pmatrix},\\
&FM = \begin{pmatrix} fm_{1,1} & \cdots & fm_{1,|M|} \\ \vdots & \ddots & \vdots \\ fm_{|F|,1} & \cdots & fm_{|F|,|M|} \end{pmatrix},\\
&MF = \begin{pmatrix} mf_{1,1} & \cdots & mf_{1,|F|} \\ \vdots & \ddots & \vdots \\ mf_{|M|,1} & \cdots & mf_{|M|,|F|} \end{pmatrix},
\end{aligned}
\tag{1}
$$

where F represents the set of functions F_i to be executed;

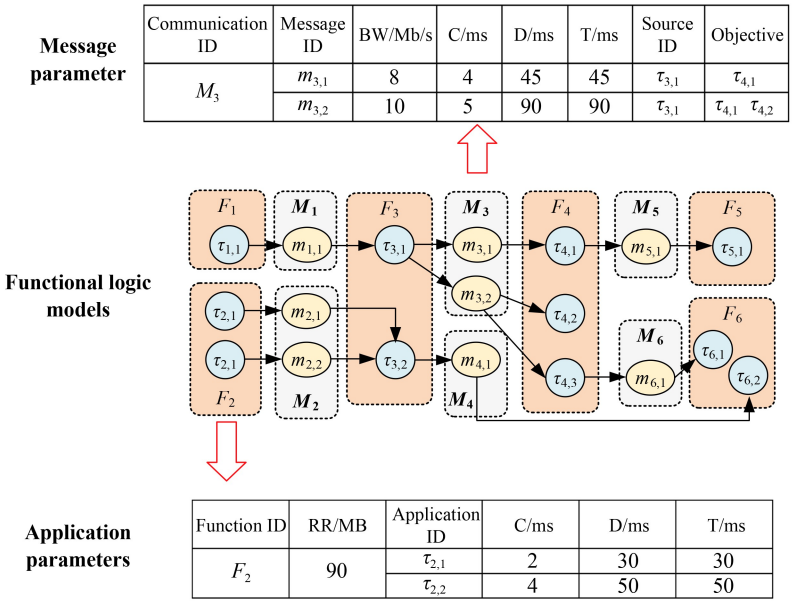

Fig. 3 Example of a functional logic model.

M represents the set of communications M_j to be transmitted; E is the communication cost matrix between functions, where $e_{p,q}$ denotes the bandwidth required to send a message from function F_p to F_q, with main diagonal elements always 0; FM and MF are communication-sending and receiving matrices, respectively. In FM, $fm_{p,q}=1$ if function F_p sends communication requirement M_q, else $fm_{p,q}=0$; in MF, $mf_{p,q}=1$ if function F_q receives communication requirement M_p, else $mf_{p,q}=0$.

Furthermore, each function F_i can be decomposed into a series of real-time application requirements, defined as follows:

$$\begin{aligned} F_i &= \langle RR_i, \Gamma_i \rangle, \\ \Gamma_i &= \{\tau_{i,r} | 1 \leqslant r \leqslant |\Gamma_i|\}, \\ \tau_{i,r} &= \langle C_{i,r}, D_{i,r}, T_{i,r} \rangle, \end{aligned} \tag{2}$$

where RR_i represents the memory requirements for executing function F_i; Γ_i represents the set of application programs within F_i, assumed to be numbered by priority. Each application program $\tau_{i,r}$ is characterized by its worst-case execution time $C_{i,r}$, deadline $D_{i,r}$, and arrival period $T_{i,r}$. The worst-case execution time $C_{i,r}$ depends on the processing resource capability used. Similarly, each communication requirement M_j is decomposed into a series of messages, defined as follows:

$$\begin{aligned} M_j &= \left\{ m_{j,s} | 1 \leqslant s \leqslant \left| M_j \right| \right\}, \\ m_{j,s} &= \left\langle BW_{m_{j,s}}, C_{m_{j,s}}, D_{m_{j,s}}, T_{m_{j,s}}, Pre_{m_{j,s}}, Sub_{m_{j,s}} \right\rangle, \end{aligned} \tag{3}$$

where $m_{j,s}$ represents the information within the communication requirement M_j, numbered by priority. Each information $m_{j,s}$ is described by six parameters: communication cost $BW_{m_{j,s}}$, worst-case transmission time $C_{m_{j,s}}$, transmission deadline $D_{m_{j,s}}$, message period $T_{m_{j,s}}$, source application program $Pre_{m_{j,s}}$, and target application program $Sub_{m_{j,s}}$. The worst-case transmission time $C_{m_{j,s}}$ is dependent on the communication resource capability.

3.2 Resource entity model

As shown in Fig. 4, the types of resource entities included in the computing platform system can be divided into processor and communication resources. The resource entity model is represented by a triple $\langle N, Bus, A \rangle$, whichis defined as follows:

$$\begin{aligned} N &= \{N_i | 1 \leqslant i \leqslant |N|\}, \\ Bus &= \{Bus_j | 1 \leqslant j \leqslant |Bus|\}, \\ A &= \begin{pmatrix} a_{1,1} & \cdots & a_{1,|Bus|} \\ \vdots & \ddots & \vdots \\ a_{|N|,1} & \cdots & a_{|N|,|Bus|} \end{pmatrix}, \end{aligned} \tag{4}$$

where N represents the set of processor resources, and its element N_i is used to refer to the processor entity included in the platform; Bus represents the set of communication resources, and its element Bus_j is used to refer to the communication bus included in the platform; A is the connection matrix, and its element $a_{p,q}$ represents the connection relationship between processor N_p and communication bus Bus_q. When processor N_p is connected to communication bus Bus_q, $a_{p,q}=1$, indicating that data transmission can take place.

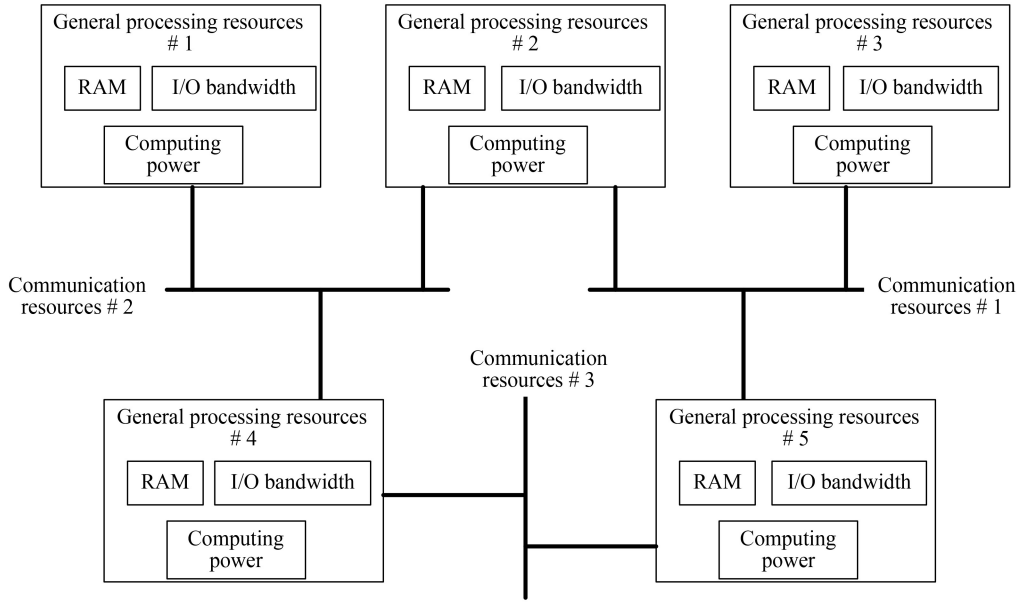

Fig. 4 Example of Resource Entity Model.

The basic performance of any processing resource node is characterized by the memory constraint RAM_i, input/output communication bandwidth constraint BW_i, and computing power Cal_i. Additionally, for a running processor, its energy consumption per unit time includes both static and dynamic energy components (Collin et al., 2019):

$$Power = Power^{\text{idle}} + Power^{\text{runtime}}, \tag{5}$$

where thestatic power consumption, denoted as $Power^{\text{idle}}$, encompasses power consumed by the clock circuit during processor core execution, multi-threaded dynamic scheduling control, on-chip bus communication, and other overheads provided by the processor manufacturer. Dynamic power consumption, denoted as $Power^{\text{runtime}}$, primarily arises from computing and storage units, varying with the executed program. To approximate dynamic power consumption, two parameters, P_i^{memory} for storage space operations and P_i^{compute} for time computing operations, are agreed upon. The calculation method for the dynamic power consumption is as follows:

$$Power^{\text{runtime}} = Memory \cdot P_i^{\text{memory}} + TimeRate \cdot P_i^{\text{compute}}, \tag{6}$$

where *memory* represents the size of memory occupied by the program during execution; $TimeRate$ represents the occupancy rate of the effective time of program execution, determined by the ratio of actual execution time to arrival time interval.

Based on this, any processor resource N_i can be represented by the following six-tuple:

$$N_i = \left\langle RAM_i, BW_i, Cal_i, P_i^{\text{idle}}, P_i^{\text{memory}}, P_i^{\text{compute}} \right\rangle. \tag{7}$$

Similarly, communication resources can be characterized using basic performance parameters such as communication bandwidth BW_{Bus}, and communication capability Com_{Bus}. Unlike processor resources, the energy consumption related to access operations can be ignored when considering communication resource consumption, and instead, static power consumption P_{Bus}^{idle} and power consumption per unit time for transmission operation $P_{Bus}^{\text{transmit}}$ are used to characterize communication resource performance. Therefore, any communication resource Bus_j can be represented using the following four-tuple:

$$Bus_j = \left\langle BW_{Bus_j}, Com_{Bus_j}, P_{Bus_j}^{\text{idle}}, P_{Bus_j}^{\text{transmit}} \right\rangle. \tag{8}$$

Considering the computing power of different processor resources directly impacts the estimation of worst-case execution time for application programs. This paper simplifies this consideration by using relative capacity values. A reference processor is chosen in the computing platform system, with its processing capacity set to 1. The processing capacity parameter of other processors is the ratio of their processing capacity to that of the reference processor. Similarly, relative communication capabilities were used to describe the performance of the communication buses in this study.

3.3 Configuration model and SOTIF constraints

Based on the definitions of the functional logical and resource entity models mentioned above, the following two allocation matrices can be used to characterize the mapping relationship between the application program and messages with their corresponding entity resources:

$$X=\begin{pmatrix} x_{1,1} & \cdots & x_{1,|N|} \\ \vdots & \ddots & \vdots \\ x_{|F|,1} & \cdots & x_{|F|,|N|} \end{pmatrix}, \quad Y=\begin{pmatrix} y_{1,1} & \cdots & y_{1,|Bus|} \\ \vdots & \ddots & \vdots \\ y_{|M|,1} & \cdots & y_{|M|,|Bus|} \end{pmatrix}, \tag{9}$$

where X represents the function allocation matrixand Y represents the communication allocation matrix, where the corresponding elements satisfy:

$$x_{i,j}=\begin{cases} 1 & \text{when } F_i \text{ is deployed on } N_j \\ 0 & \text{else} \end{cases}$$
$$i=1,2,\ldots,|F|, \quad j=1,2,\ldots,|N|,$$
$$y_{p,q}=\begin{cases} 1 & \text{when } M_p \text{ is transmitted via } Bus_q \\ 0 & \text{else} \end{cases}$$
$$p=1,2,\ldots,|M|, \quad q=1,2,\ldots,|Bus|. \tag{10}$$

Note that the functionality allocation matrix X, communication allocation matrix Y and the resource connectivity matrix A described in section 3.2 are inherently correlated. Equations (11) and (12) provide the mathematical relationship between the three matrices from the sending and receiving communication perspective, respectively:

$$\sum_{p=1}^{|M|}\sum_{i=1}^{|F|} x_{i,s}\cdot fm_{i,p}\cdot y_{p,q}\geqslant 1 \Rightarrow a_{s,q}=1, \tag{11}$$

$$\sum_{p=1}^{|M|}\sum_{j=1}^{|F|} x_{j,t}\cdot mf_{p,j}\cdot y_{p,q}\geqslant 1 \Rightarrow a_{t,q}=1, \tag{12}$$

where $s,t=1,2,\ldots,|N|$, $q=1,2,\ldots,|Bus|$. Combining Eqs. (11)and (12), the following identity can be derived.

$$A=\max(\min(X^T\cdot FM\cdot Y,1),\min(X^T\cdot MF^T\cdot Y,1)), \tag{13}$$

where $\max(\cdot)$ and $\min(\cdot)$ representing taking the maximum/minimum value bitwise.

On this basis, this study considers a configuration scheme $\langle X,Y\rangle$ to be compliant with SOTIF requirements if and only if it satisfies the following conditions simultaneously:

1) Uniqueness requirement for function and communication allocation

All functions F_i are assigned to one and only one corresponding processor resource entity:

$$\sum_{j=1}^{|N|} x_{i,j}=1 \quad i=1,2,\ldots,|F|. \tag{14}$$

When the sending and receiving functions are located on different processor resources, the communication requirement M_i is sent and received by one and only one corresponding communication resource. When the sending and receiving functions are located on the same processor resource, the communication requirement M_i can be directly fulfilled by accessing the memory of the processor:

$$\sum_{q=1}^{|Bus|} y_{p,q}\leqslant 1, \quad p=1,2,\ldots,|M|, \tag{15}$$

$$\sum_{i=1}^{|F|}\sum_{j=1}^{|F|}\sum_{s=1}^{|N|} fm_{i,p}\cdot x_{i,s}\cdot mf_{p,j}\cdot x_{j,s}=1-\sum_{q}^{|Bus|} y_{p,q},\ p=1,2,\ldots,|M|. \tag{16}$$

2) The configuration scheme satisfies the real-time constraints of all application executions

In an effective configuration scheme, each application $\tau_{i,r}$ needs to complete execution before its corresponding deadline $D_{i,r}$. Considering the computational characteristics of processor resource N_j when application $\tau_{i,r}$ is deployed on processor resource N_j its corresponding equivalent worst-case execution time is $C_{i,r}/Cal_j$. We use the worst-case task response time $R_{i,r}$ to represent the upper limit of the time required to complete application $\tau_{i,r}$. The scheduling mode of programs on processor resources will directly affect their corresponding worst-case task response time. In this article, we consider the most common fixed-priority scheduling mode. Based on this, it can be determined that:

$$R_{i,r}=\frac{C_{i,r}}{Cal_j}+\sum_{\tau_k\in hp(\tau_{i,r})}\left\lceil\frac{R_{i,r}}{T_{\tau_k}}\right\rceil\cdot\frac{C_{\tau_k}}{Cal_j}, \tag{17}$$

where $\lceil\cdot\rceil$ denotes rounding up; $hp(\tau_{i,r})$ represents the set of applications with higher priorities than $\tau_{i,r}$ running on the same processor. Equation (17) reflects the worst-case response time of a task, which consists of the worst-case execution time and the time preempted by higher-priority tasks running on the same processor.

In computing resource sharing, partition scheduling is a widely used mechanism where the rotation period for running partitions on a processing resource module is denoted as RL. The proportion of the total period that an application $\tau_{i,r}$ corresponding to functionality F_i occupies in its partition is denoted as α_i (referred to as the partition coefficient). Considering the inherent mechanism of mutually exclusive partition scheduling, other partitions can be regarded as a cyclic task with higher priority, having a period of RL and execution time of $(1-\alpha_i)RL$ when computing the worst-case response time $R_{i,r}$ of the application $\tau_{i,r}$. Based on this, Eq. (18) can be reformulated:

$$R_{i,r}=\frac{C_{i,r}}{Cal_j}+\sum_{k=1}^{r-1}\left\lceil\frac{R_{i,r}}{T_{i,k}}\right\rceil\cdot\frac{C_{i,k}}{Cal_j}+\left\lceil\frac{R_{i,r}}{RL}\right\rceil(1-\alpha_i)RL. \quad (18)$$

It is known that the partition rotation period RL is a predetermined positive actual number. If $RL\to 0$, it follows that:

$$\left\lceil\frac{R_{i,r}}{RL}\right\rceil=\frac{R_{i,r}}{RL}. \quad (19)$$

According to Eq. (18):

$$R_{i,r}=\frac{C_{i,r}}{Cal_j}+\sum_{k=1}^{r-1}\left\lceil\frac{R_{i,r}}{T_{i,k}}\right\rceil\cdot\frac{C_{i,k}}{Cal_j}+(1-\alpha_i)R_{i,r}. \quad (20)$$

And thus satisfy:

$$\begin{aligned}\alpha_i&\leqslant\frac{1}{Cal_j}\cdot\left(\frac{C_{i,r}}{R_{i,r}}+\sum_{k=1}^{r-1}\left(\frac{R_{i,r}}{T_{i,k}}+1\right)\frac{C_{i,k}}{R_{i,r}}\right)\\&=\frac{1}{Cal_j}\cdot\left(\sum_{k=1}^{r}\frac{C_{i,k}}{R_{i,r}}+\sum_{k=1}^{r-1}\frac{C_{i,k}}{T_{i,k}}\right).\end{aligned} \quad (21)$$

For application $\tau_{i,r}$, It satisfies the real-time requirement if and only if the response time $R_{i,r}$ is not greater than the deadline $D_{i,r}$ (i.e. $R_{i,r}\leqslant D_{i,r}$) which is equivalent to Eq. (22):

$$\frac{1}{Cal_j}\cdot\left(\sum_{k=1}^{r}\frac{C_{i,k}}{D_{i,r}}+\sum_{k=1}^{r-1}\frac{C_{i,k}}{T_{i,k}}\right)\leqslant\alpha_i. \quad (22)$$

Therefore, it can be considered that the real-time requirements of all applications in the partition where function F_i is located can be met as long as they satisfy:

$$\begin{aligned}\frac{1}{Cal_j}\cdot\alpha_i^{lb}&\leqslant\alpha_i,\\\alpha_i^{lb}&=\max_{1\leqslant r\leqslant|\Gamma_i|}\left(\sum_{k=1}^{r}\frac{C_{i,k}}{D_{i,r}}+\sum_{k=1}^{r-1}\frac{C_{i,k}}{T_{i,k}}\right),\end{aligned} \quad (23)$$

where α_i^{lb} is defined to constrain the lower bound of the partition coefficient α_i, which is only related to the function F_i.

Thus, the real-time requirements related to the application can be expressed in the scheduling scheme as follows: "For any processor resource, there exists a feasible combination of partition coefficients, such that each application within each partition is schedulable and the sum of all partition coefficients does not exceed 1." In other words, it satisfies:

$$\sum_{i=1}^{|F|}\frac{1}{Cal_j}\cdot x_{i,j}\cdot\alpha_i^{lb}\leqslant 1,\quad j=1,2,\ldots,|N|. \quad (24)$$

3) The configuration scheme meets the real-time constraints of all message transmissions

When the communication requirement M_j is assigned to the communication resource Bus_q, the worst-case transmission time of message M_j in $m_{j,s}$ on the bus can be represented as follows:

$$R_{m_{j,s}}=\frac{C_{m_{j,s}}}{Com_{Bus_q}}+B_{m_{j,s}}+\sum_{m_k\in hp(m_{j,s})}\left\lceil\frac{R_{m_{j,s}}}{T_{m_k}}\right\rceil\cdot\frac{C_{m_k}}{Com_{Bus_q}}, \quad (25)$$

where $hp(m_{j,s})$ represents the set of messages with higher priority than $m_{j,s}$ on the same data bus; since there is actually no direct priority relationship between different communication requirements M_j, whenever considering the communication requirement M_j in this paper, it is assumed conservatively that other communication requirements M_k $(k\neq j)$ have higher priority than M_j; $B_{m_{j,s}}$ represents the maximum blocking time, which is equivalent to the blocking caused by the longest transmission time of the low-priority message:

$$B_{m_{j,s}}=\max_{s<l\leqslant|M_j|}\frac{C_{m_{j,l}}}{Com_{Bus_q}}. \quad (26)$$

Similarly, according to Eq. (25), it can be inferred that:

$$R_{m_{j,s}}\leqslant\frac{1}{Com_{Bus_q}}\cdot\left(C_{m_{j,s}}+\max_{s<l\leqslant|M_j|}C_{m_{j,l}}+\sum_{m_k\in hp(m_{j,s})}\left(\frac{R_{m_{j,s}}}{T_{m_k}}+1\right)C_{m_k}\right). \quad (27)$$

After simplification:

$$R_{m_{j,s}}\leqslant\frac{C_{m_{j,s}}+\max_{s<l\leqslant|M_j|}C_{m_{j,l}}+\sum_{m_k\in hp(m_{j,s})}C_{m_k}}{\left(Com_{Bus_q}-\sum_{m_k\in hp(m_{j,s})}\frac{C_{m_k}}{T_{m_k}}\right)}. \quad (28)$$

In summary, to strictly ensure that the worst-case transmission time $R_{m_{j,s}}$ is not greater than the deadline $D_{m_{j,s}}$, a feasible approach is to make sure that:

$$\frac{C_{m_{j,s}}+\max_{s<l\leqslant|M_j|}C_{m_{j,l}}+\sum_{m_k\in hp(m_{j,s})}C_{m_k}}{\left(Com_{Bus_q}-\sum_{m_k\in hp(m_{j,s})}\frac{C_{m_k}}{T_{m_k}}\right)}\leqslant D_{m_{j,s}}. \quad (29)$$

For simplicity, concerning any communication requirement $M_k(k=1,2,\ldots,|M|)$, two related variables are defined in this paper, the values of which are solely dependent on the communication requirement M_k:

$$\begin{aligned}C_{M_k}&=\sum_{1\leqslant l\leqslant|M_k|}C_{m_{k,l}},\\\frac{C_{M_k}}{T_{M_k}}&=\sum_{1\leqslant l\leqslant|M_k|}\frac{C_{m_{k,l}}}{T_{m_{k,l}}}.\end{aligned} \quad (30)$$

Therefore, Eq. (29) is equivalent to:

$$\frac{1}{Com_{Bus_q}}\cdot\left(\beta_{m_{j,s}}+\sum_{k=1}^{|M|}y_{k,q}\cdot\left(\frac{C_{M_k}}{D_{m_{j,s}}}+\frac{C_{M_k}}{T_{M_k}}\right)\right)\leqslant 1, \quad (31)$$

where $\beta_{m_{j,s}}$ is defined by Eq. (32), and its value is only related to the message $m_{j,s}$.

$$\beta_{m_{j,s}} = \frac{1}{D_{m_{j,s}}}\left(C_{m_{j,s}} + \max_{s<l\leqslant|M_j|} C_{m_{j,l}} - \sum_{l=s+1}^{|M_j|} C_{m_{j,l}}\right) - \sum_{l=s+1}^{|M_j|} \frac{C_{m_{j,l}}}{T_{m_{j,l}}}. \quad (32)$$

It should be noted that although Eq. (31) can provide the most accurate real-time guarantee for transmission delay, it needs to be calculated separately for each message, which poses a huge challenge for practical analysis. To simplify the calculation, the communication requirement M_j, is defined as:

$$\beta_j^{ub} = \max_{1<s\leqslant|M_j|} \beta_{m_{j,s}}, \quad (33)$$

$$D_j^{lb} = \min_{1<s\leqslant|M_j|} D_{m_{j,s}}. \quad (34)$$

On this basis, the requirement that "the scheduling scheme can meet the real-time constraints of all message transmissions" can be expressed as:

$$\sum_{q=1}^{|Bus|} y_{j,q} \cdot \frac{1}{Com_{Bus_q}} \cdot \left(\beta_j^{ub} + \sum_{k=1}^{|M|} y_{k,q} \cdot \left(\frac{C_{M_k}}{D_j^{lb}} + \frac{C_{M_k}}{T_{M_k}}\right)\right) \leqslant 1, \quad j = 1,2,\ldots,|M|, \quad (35)$$

where Eq. (35) is a sufficient condition for Eq. (31) to hold, i.e., Eq. (35) can provide a stronger feasibility guarantee.

4) The processing capacity requiredunder the maximum processing resources that a processor can provide

For any processor resource module, the application resource requirements deployed on it do not exceed its inherent memory, output bandwidth, and input bandwidth capabilities. It satisfies:

$$\sum_{i=1}^{|F|} x_{i,j} RR_i \leqslant RAM_j, \quad j = 1,2,\ldots,|N|, \quad (36)$$

$$\sum_{i=1}^{|F|} x_{i,j} \sum_{k=1}^{|F|} (1 - x_{i,k}) e_{k,i} \leqslant BW_i, \quad j = 1,2,\ldots,|N|, \quad (37)$$

$$\sum_{i=1}^{|F|} x_{i,j} \sum_{k=1}^{|F|} (1 - x_{i,k}) e_{i,k} \leqslant BW_i, \quad j = 1,2,\ldots,|N|. \quad (38)$$

5) The communication capacity requiredunder the maximum communication resources that can be provided

For any communication resource module, the sum of the required communication bandwidths corresponding to all messages transmitted through the communication resource by the processors connected does not exceed its inherent communication bandwidth capacity limit. This includes both input bandwidth requirements and output bandwidth requirements, where:

$$\sum_{i=1}^{|F|} x_{i,s} \sum_{p=1}^{|M|} mf_{i,p} \cdot y_{p,q} \sum_{j=1}^{|F|} \sum_{k=1}^{|F|} fm_{j,p} \cdot e_{j,i} \cdot mf_{p,i} \leqslant BW_{Bus_q}, \quad s = 1,2,\ldots,|N|, \quad q = 1,2,\ldots,|Bus|, \quad (39)$$

$$\sum_{i=1}^{|F|} x_{i,s} \sum_{p=1}^{|M|} fm_{i,p} \cdot y_{p,q} \sum_{j=1}^{|F|} \sum_{k=1}^{|F|} fm_{j,p} \cdot e_{j,i} \cdot mf_{p,i} \leqslant BW_{Bus_q}, \quad s = 1,2,\ldots,|N|, \quad q = 1,2,\ldots,|Bus|. \quad (40)$$

6) All processor resource power consumption under the threshold

High-performance processor resources enhance the performance of unmanned driving tasks but often result in increased energy consumption and heat dissipation. Processor power consumption comprises static and dynamic components, with the dynamic component further divided into computational and storage operations. A reasonable and safe configuration scheme should ensure that the total power consumption of all processor resources remains within a predefined threshold, as follows:

$$P_j^{\text{idle}} + \frac{P_j^{\text{memory}}}{RAM_j} \sum_{i=1}^{|F|} x_{i,j} RR_i + P_j^{\text{compute}} \sum_{i=1}^{|F|} x_{i,j} \sum_{k=1}^{|\Gamma_i|} \frac{C_{i,k}}{T_{i,k}} \leqslant P_j^{ub}, \quad j = 1,2,\ldots,|N|. \quad (41)$$

7) Power consumption of all communication resources under the thresholds

Similarly, the working power consumption of all communication resources must not exceed a specified threshold. This paper primarily addresses the static power consumption and data transmission power consumption of communication resources, with the following requirements:

$$P_{Bus_q}^{\text{idle}} + P_{Bus_q}^{\text{transmit}} \sum_{p=1}^{|M|} y_{p,q} \sum_{l=1}^{|M_l|} \frac{C_{m_{p,l}}}{T_{m_{p,l}}} \leqslant P_{Bus_q}^{ub}, \quad q = 1,2,\ldots,|Bus|. \quad (42)$$

8) Preset feature allocation mutual exclusion requirements

In designing a shared configuration scheme, it is essential to consider preset allocation constraints and correlations between different functions and between functions and processor resources, as required by unmanned driving tasks.

Mutual exclusion in allocation means that "two target functions cannot be deployed on the same processor resource." This requirement is satisfied by:

$$\sum_{j=1}^{|N|} x_{i,j} \cdot x_{k,j} = 0, \tag{43}$$

when F_i and F_k are preset as mutually exclusive functions.

Allocated correlation means that "two target functions must be deployed on the same processing resource." This requirement is satisfied by:

$$\sum_{j=1}^{|N|} x_{i,j} \cdot x_{k,j} = 1, \tag{44}$$

when F_i and F_k are preset as correlated functions.

In summary, Eqs. (14)–(16), (24), (35)–(44) collectively constitute the constraint criteria for the feasibility of the configuration $\langle X, Y \rangle$. To facilitate batch operation analysis, this paper introduces auxiliary matrices listed in Table 1, which enable a vectorized representation of each constraint.

On this basis, SOTIF constraints of the configuration scheme can be reorganized into the following matrix expression:

$$\begin{aligned}
&Con01: \quad X \cdot SN = SF,\\
&Con02: \quad Y \cdot SBus \leqslant SM,\\
&Con03: \quad rdiag(MF \cdot X \cdot X^T \cdot FM) = 1 - Y \cdot SBus,\\
&Con04: \quad diag(SCal) \cdot X^T \cdot SAlpha \leqslant SN,\\
&Con05: \quad diag(SBeta) \cdot Y \cdot SCOMB + diag((SDLb)^{-1})\\
&\qquad \cdot Y \cdot diag(SCOMB) \cdot Y^T \cdot SCM + Y\\
&\qquad \cdot diag(SCOMB) \cdot Y^T \cdot SCTM \leqslant SM,\\
&Con06: \quad X^T \cdot SRR \leqslant SRAM,\\
&Con07: \quad rdiag((1 - X^T) \cdot E \cdot X) \leqslant SBW,\\
&Con08: \quad rdiag((1 - X^T) \cdot E^T \cdot X) \leqslant SBW,\\
&Con09: \quad X^T \cdot MF^T \cdot diag(rdiag(MF \cdot E^T \cdot FM)) \cdot\\
&\qquad Y \leqslant BWNB,\\
&Con10: \quad X^T \cdot FM \cdot diag(rdiag(MF \cdot E^T \cdot FM)) \cdot\\
&\qquad Y \leqslant BWNB,\\
&Con11: \quad SPI + diag(SPM) \cdot X^T \cdot SRR + diag(SPC) \cdot\\
&\qquad X^T \cdot SCT \leqslant SPUb,\\
&Con12: \quad SPIB + diag(SPTB) \cdot Y^T \cdot SCTM \leqslant SPUbB,\\
&Con13: \quad \max(X \cdot X^T, ME) = ME,\\
&Con14: \quad \min(X \cdot X^T, MC) = MC,
\end{aligned} \tag{45}$$

where $diag(\cdot)$ denotes the operation of transforming a column vector into a diagonal matrix, while $rdiag(\cdot)$ denotes the operation of extracting the diagonal elements of a square matrix to form a column vector. Equation (46) provides examples of the operations of these two operators; Constraints $Con01$, $Con02$ and $Con03$ correspond to uniqueness requirements; constraints $Con04$ and $Con05$ correspond to temporal requirements; constraints $Con06$, $Con07$, $Con08$, $Con09$, $Con10$ correspond to capability requirements; constraints $Con11$ and $Con12$ correspond to energy consumption limit requirements; and constraints $Con13$ and $Con14$ correspond to mutual exclusion and correlation requirements.

$$diag\left(\begin{pmatrix} 1 \\ 2 \end{pmatrix}\right) = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}, \quad rdiag\left(\begin{pmatrix} 1 & 3 \\ 4 & 2 \end{pmatrix}\right) = \begin{pmatrix} 1 \\ 2 \end{pmatrix}. \tag{46}$$

4 Comprehensive optimization for resource selection considering SOTIF and computational power costs

This section presents a method for optimizing SOTIF and computational power costs for unmanned systems under varying computational power conditions. It introduces Constraint Satisfaction Problems (CSP) and a forward-checking algorithm, illustrating how hardware resource selection can be addressed through CSP construction.

4.1 Constraint satisfaction problems and Forward Checking algorithm

CSP has been applied to many practical problems since Mackworth (1977) first developed it. Classic examples include map coloring, job scheduling, and the n-queens problem. A formally defined CSP consists of a set of variables, each assigned values from a specific domain, and a set of constraints that must be simultaneously satisfied. This can be represented as the following triple:

$$\langle X, D, C \rangle, \tag{47}$$

where $X = \{x_1, \ldots, x_n\}$ represents a set of variables; $D = \{D_1, \ldots, D_n\}$ represents the corresponding domain of variables, and each element D_i $(i = 1, \ldots, n)$ reflects all possible values for each variable $x_1 \in X$; $C = \{c_1, \ldots, c_n\}$ represents a finite set of constraints that limit feasible values for each group of variables.

A feasible solution for a CSP is a set of specific assignments $\{x_1 = a_1, \ldots, x_n = a_n\}$, from the variable domains, which guarantees that all constraints are not violated. Solving a CSP involves assigning values to variables to find feasible solutions. If at least one solution exists, the CSP is considered satisfiable; otherwise, it is unsatisfiable. There are two main approaches to solving a CSP: backtracking search and constraint propagation. Backtracking systematically explores all possible value combi-

Table 1 Definition of auxiliary matrices related to SOTIF constraints on computing resources

Index	Symbol Label	Size	Element Value Assignment
1	SN	$\|N\|\times 1$	$SN_{i,1}=1$
2	SF	$\|F\|\times 1$	$SF_{i,1}=1$
3	SM	$\|M\|\times 1$	$SM_{i,1}=1$
4	$SBus$	$\|Bus\|\times 1$	$SBus_{i,1}=1$
5	$SAlpha$	$\|F\|\times 1$	$SAlpha_{i,1}=\alpha_i^{lb}$
6	$SCal$	$\|N\|\times 1$	$SCal_{i,1}=\dfrac{1}{Cal_i}$
7	$SBeta$	$\|M\|\times 1$	$SBeta_{i,1}=\beta_i^{ub}$
8	$SCOMB$	$\|Bus\|\times 1$	$SCOMB_{i,1}=\dfrac{1}{Com_{Bus_i}}$
9	SCM	$\|M\|\times 1$	$SCM_{i,1}=\sum_{1\leqslant j\leqslant\|M_i\|}C_{m_{i,j}}$
10	$SDLb$	$\|M\|\times 1$	$SDLb_{i,1}=D_j^{lb}$
11	$SCTM$	$\|M\|\times 1$	$SCT_{i,1}=\sum_{1\leqslant j\leqslant\|M_i\|}\dfrac{C_{m_{i,j}}}{T_{m_{i,j}}}$
12	SRR	$\|F\|\times 1$	$SRR_{i,1}=RR_i$
13	$SRAM$	$\|N\|\times 1$	$SRAM_{i,1}=RAM_i$
14	SBW	$\|N\|\times 1$	$SBW_{i,1}=BW_i$
15	$BWNB$	$\|N\|\times\|Bus\|$	$SBWB_{i,j}=BW_{Bus_j}$
16	SPI	$\|N\|\times 1$	$SPI_{i,1}=P_i^{\text{idle}}$
17	SPM	$\|N\|\times 1$	$SPM_{i,1}=P_i^{\text{memory}}$
18	SPC	$\|N\|\times 1$	$SPC_{i,1}=P_i^{\text{compute}}$
19	SCT	$\|F\|\times 1$	$SCT_{i,1}=\sum_{1\leqslant j\leqslant\|F_i\|}\dfrac{C_{i,j}}{T_{i,j}}$
20	$SPUb$	$\|N\|\times 1$	$SPUb_{i,1}=P_i^{ub}$
21	$SPIB$	$\|Bus\|\times 1$	$SPIB_{i,1}=P_{Bus_i}^{\text{idle}}$
22	$SPTB$	$\|Bus\|\times 1$	$PTB_{i,1}=P_{Bus_i}^{\text{transmit}}$
23	$SPUbB$	$\|Bus\|\times 1$	$SPUbB_{i,1}=P_{Bus_i}^{ub}$
24	ME	$\|F\|\times\|F\|$	$CAL_{i,j}=\begin{cases}0 & F_i \text{ and } F_j \text{ are mutually exclusive}\\ 1 & \text{others}\end{cases}$
25	MC	$\|F\|\times\|F\|$	$CAL_{i,j}=\begin{cases}1 & F_i \text{ and } F_j \text{ are correlated}\\ 0 & \text{others}\end{cases}$

nations, often using depth-first search, to identify feasible solutions. In contrast, constraint propagation narrows the range of valid variable values by enforcing "local compatibility" through continuous reasoning. Haralick and Elliott combined these methods into the Forward Checking algorithm Kondrak and Van Beek (1997), which prunes the search tree early to reduce the number of nodes visited, minimizing the search effort.

Table 2 presents the pseudocode for the main program using Forward Checking. It integrates two heuristic techniques: 'Select-Unassigned-Variable' and 'Order-Domain-Values.' These heuristics allow for efficient variable and value ordering based on the problem's requirements, improving search performance. Common heuristic methods include minimum remaining values, minimum degree, and minimum constraint values, among others.

Table 2 The pseudocode of the main program of Forward Checking

```
function Backtracking-Search (csp) returns a solution, or failure
Return Backtrack ({},csp)
function Backtrack (assignment, csp) returns a solution, or failure
if assignment is complete then return assignment
var ← Select-Unassigned-Variable (csp)
foreach value in Order-Domain-Values (var, assignment, csp) do
if value is consistent with assignment then
add {var = value} to assignment
inferences ← Inference (csp, var, value)
if inferences ≠ failure then
add inferences to assignment
result ← Backtrack (assignment, csp)
if result ≠ failures then
return result
remove {var = value; add inferences from assignment
return failure
```

4.2 Computational resource selection optimization method

Figure 5 illustrates the process of optimizing compute resource selection. This involves constructing a CSP in a limited search space and ensuring it meets SOTIF requirements using the forward-checking algorithm. The 'compute resource selection' challenge is divided into two stages: 'processor resource selection' and 'communication resource selection'. The goal is to create a cost-effective plan that fulfills all SOTIF requirements.

4.2.1 Processor resource selection

In addressing the "processor resource selection" problem, communication resource constraints are temporarily ignored, assuming sufficient communication resources for message transmission. In Eq. (45), the constraints directly related to processor resources include eight items: $Con01$, $Con04$, $Con06$, $Con07$, $Con08$, $Con11$, $Con13$, $Con14$. These are used as criteria to assess the feasibility of processor resource combinations. For all processing resource sets $AR=\{AR_i|1\leqslant i\leqslant|AR|\}$ in the resource library, a processing resource combination can be defined by the following variables:

$$NumAR=(numAR_1 \quad numAR_2 \quad \cdots \quad numAR_{|AR|}), \tag{48}$$

where $numAR_i$ represents the number of resources selected for the ith candidate processing resource type AR_i.

This study defines the relationship between the processor resource combination scheme, $NumAR$, and the processor resource set $N=\{N_i|1\leqslant i\leqslant|N|\}$ in the resource entity

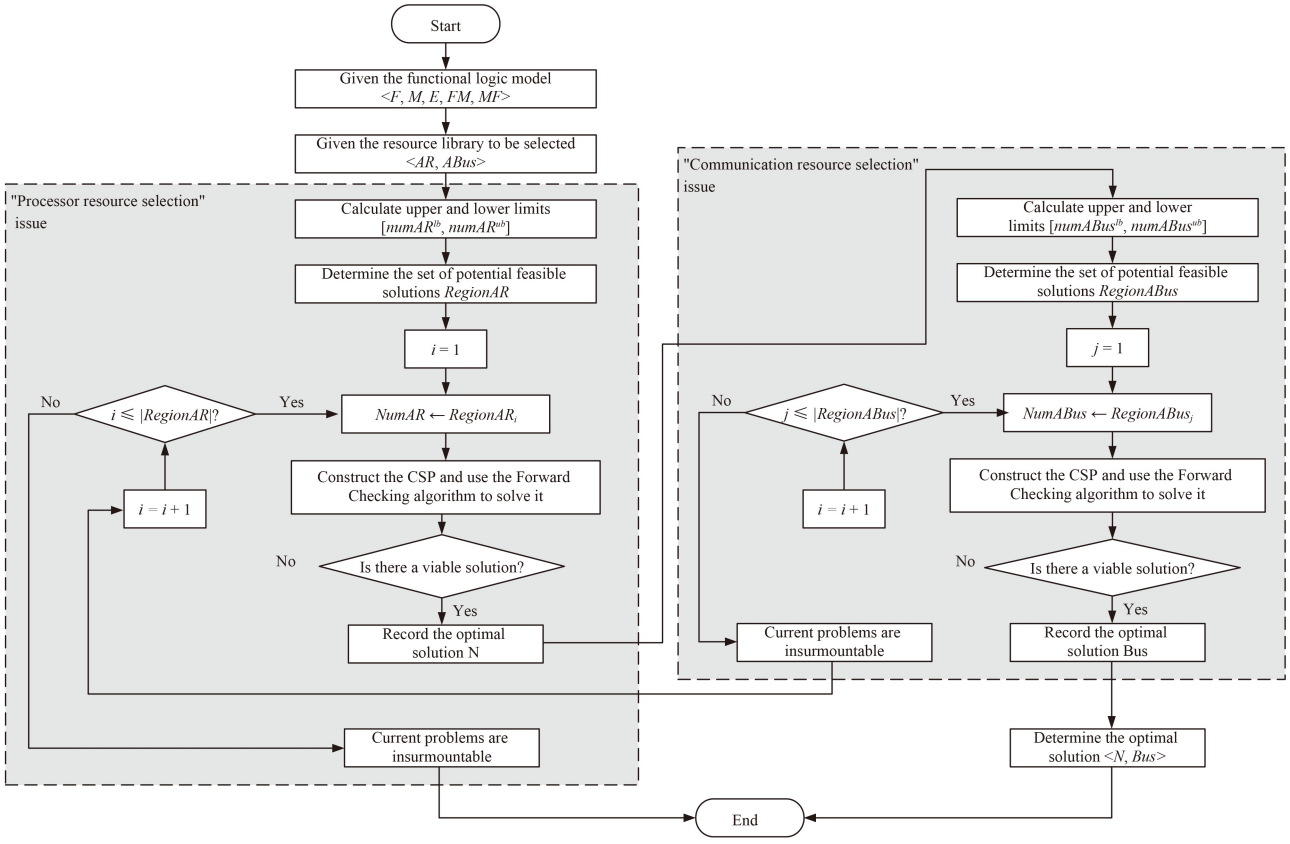

Fig. 5 The basic process of the compute resource selection optimization method.

model, as illustrated in Fig. 6. The elements in both sets satisfy the following criteria:

$$N_i \leftarrow AR_j \quad \text{when} \quad \sum_{k=1}^{j} numAR_k \geqslant i > \sum_{k=1}^{j-1} numAR_k. \tag{49}$$

Designers can set an acceptable upper limit on the number of processor resources, $numAR^{ub}$, in advance, due to the constraints of onboard space. Additionally, a conservative constraint is that the number of processor resources will not exceed the number of target functions, since the worst-case scenario would involve assigning each task to a different processing resource. Thus, $numAR^{ub} \leqslant |F|$. Inspired by the memory resource capacity constraint in Eq. (36), it can be determined for the final resource entity scheme $\langle N, Bus, A\rangle$ that:

$$Con15: \quad \sum_{j=1}^{|N|}\sum_{i=1}^{|F|} x_{i,j} RR_i = \sum_{i=1}^{|F|} RR_i \leqslant \sum_{j=1}^{|N|} RAM_j. \tag{50}$$

The constraint $Con15$ is a necessary condition for constraint $Con06$, which is independent of the allocation of functions and can support rapid prunin search space pruning. According to Eq. (50), it can be determined that:

$$\sum_{i=1}^{|NumAR|} numAR_i \geqslant \frac{\sum_{i=1}^{|F|} RR_i}{\max\limits_{1\leqslant i\leqslant |NumAR|} RAM_{AN_i}} = numAR^{lb}. \tag{51}$$

Based on this, the "processor resource selection" problem can be formulated as the following optimization problem:

$$\begin{aligned} &\min_{NumAR} \sum_{i=1}^{|NumAR|} numAR_i \cdot costAR_i, \\ &\text{s.t.} \quad numAR^{ub} \geqslant \sum_{i=1}^{|NumAR|} numAR_i \geqslant numAR^{lb}, \\ &\qquad \text{is a positive integer.} \\ &\exists X, \quad Con01 \wedge Con04 \wedge Con06 \wedge Con07 \wedge Con08 \\ &\qquad \wedge Con11 \wedge Con13 \wedge Con14, \end{aligned} \tag{52}$$

where X represents the mapping relationship between functions and processing resources, defined by Eq. (9).

In fact, due to the constraint that $numAR^{ub} \geqslant \sum_{i=1}^{|NumAR|} numAR_i \geqslant numAR^{lb}$ ($numAR_i$ is a positive integer), Eq. (52) can be understood as a finite-state search problem. By considering only the feasible domain and constraint $Con15$, we can easily obtain all potential solutions and calculate the corresponding design costs. We

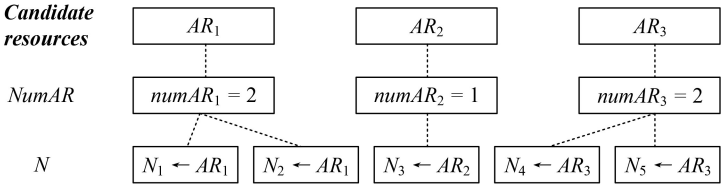

Fig. 6 The mapping relationship between the combination of processor resources and the set of processor resources.

use $RegionAR = \{RegionAR_i | 1 \leqslant i \leqslant |RegionAR|\}$ to represent the set of all potential solutions, where the solutions are numbered in ascending order of cost, i.e., $Cost(RegionAR_i) \leqslant Cost(RegionAR_j)$ (when $i \leqslant j$). After determining $RegionAR$, Eq. (52) can be solved by sequentially checking whether a feasible allocation scheme X.

The symbol $NumAR = (numAR_1\ numAR_2\ \cdots\ numAR_{|AR|})$ is used to represent any processor resource combination corresponding to a traversal process. To reduce the search space when judging the feasibility of the function allocation scheme X, this paper defines the following auxiliary column vector for the function allocation scheme by borrowing the requirement of allocation uniqueness:

$$XZ = \begin{pmatrix} XZ_1 \\ \vdots \\ XZ_{|F|} \end{pmatrix}, \tag{53}$$

where XZ_i represents the processor resource number on which function F_i is deployed, the element in the allocation scheme X satisfies:

$$x_{i,j} = \begin{cases} 1 & j = XZ_i \\ 0 & j \neq XZ_i \end{cases} \quad i = 1, 2, \ldots, |F|. \tag{54}$$

On this basis, the problem of determining whether SOTIF requirements can be satisfied for a given processor resource combination $NumAR$ can be formulated as a CSP problem with the following structure.

$$\begin{aligned} &\text{variable}Z \quad \leftarrow XZ \\ &\text{domain}D \quad \leftarrow \{1, 2, \ldots, |N|\} \\ &\text{constraint}C \quad \leftarrow \{Con04, Con06, Con07, Con08, \\ &Con11, Con13, Con14\}, \end{aligned} \tag{55}$$

where $|N| = \sum_{i=1}^{|NumAR|} numAR_i$.

This paper introduces two heuristic methods for selecting unassigned variables:

1) Minimum Remaining Values (MRV): This method selects the variable with the fewest remaining values in its domain among the unassigned variables.

2) Maximum Memory Requirement (MMR): This method chooses the variable with the highest memory requirement among the unassigned variables.

Although the auxiliary column vector XZ offers a better representation for reducing the search space, the original matrix X is more practical for checking constraint violations. Therefore, during consistency checks in the Forward Checking algorithm, it is essential to convert between the auxiliary column vector XZ and the original matrix X. The conversion rules refer to Eqs. (53) and (54).

4.2.2 Communication resource selection

For all communication resource sets $ABus = \{ABus_i | 1 \leqslant i \leqslant |ABus|\}$ in the resource library, a communication resource combination can be defined using the following variables:

$$NumABus = (numABus_1\ numABus_2\ \cdots\ numABus_{|ABus|}). \tag{56}$$

where $numABus_i$ represents the number of resources selected for the ith candidate communication resource type $ABus_i$.

The communication resource combination $NumABus$ is defined similarly as in Fig. 6, and its correspondence with the communication resource set $Bus = \{Bus_j | 1 \leqslant j \leqslant |Bus|\}$ in the resource entity model satisfies the following condition:

$$Bus_i \leftarrow ABus_j \ \text{ when } \ \sum_{k=1}^{j} numABus_k \geqslant i > \sum_{k=1}^{j-1} numABus_k. \tag{57}$$

Similarly, designers can preset an acceptable upper limit of communication resource quantity $numABus^{ub}$, while the lower limit of communication resource quantity $numABus^{lb}$ can be determined according to Eq. (58):

$$numABus^{lb} = \begin{cases} 1 & \sum_{i=1}^{|NumAR|} numAR_i \geqslant 2 \\ 0 & \sum_{i=1}^{|NumAR|} numAR_i = 1 \end{cases}. \tag{58}$$

Note that when evaluating the communication resource combination $NumABus$, the satisfiability of the resource connection matrix A must be checked simultaneously. According to Eq. (13), the resource connection matrix A

can be directly determined from matrix X and matrix Y. Therefore, it is unnecessary to assign values separately to the resource connection matrix A.

In summary, this paper represents the "communication resource selection" problem as the following optimization problem:

$$\begin{aligned}&\min_{NumABus}\sum_{i=1}^{|NumABus|} numABus_1\cdot costABus_1\\ &\text{s.t.}\quad numABus^{ub}\geqslant\sum_{i=1}^{|numABus|} numABus_i\geqslant numABus^{lb}\\ &numABus_i \text{ is a positive integer.}\\ &\exists X,Y\quad Con01\sim Con14\end{aligned}\tag{59}$$

Similarly, the "communication resource selection" problem is also regarded as a search problem in a finite state space. It is assumed that $RegionABus=\{RegionABus_i|1\leqslant i\leqslant|RegionABus|\}$ represents the set of all potential solutions for the "communication resource selection" optimization problem, and the solutions are numbered in ascending or concerning $Cost(RegionABus_i)\leqslant Cost(RegionABus_j)$ $(i\leqslant j)$. Therefore, Eq. (59) can be solved by sequentially checking the feasibility of the communication resource combination corresponding to each potential solution $RegionABus_i$ concerning the functionality allocation matrix X and the communication allocation matrix Y.

The symbol $NumABus=(numABus_1\ numABus_2\ \cdots\ numABus_{|ABus|})$ is used to represent the combination of communication resources during any traversal process. Similar to constructing the auxiliary column vector XZ for the functionality allocation matrix X in Eq. (53), this paper defines the auxiliary column vector YZ corresponding to the communication allocation matrix Y as follows:

$$YZ=\begin{pmatrix}YZ_1\\ \vdots\\ YZ_{|M|}\end{pmatrix},\tag{60}$$

where YZ_i represents the communication resource number connected to the communication requirement M_i. In addition, it is agreed that when the communication demand M_i does not need to be transmitted through any communication resource, its corresponding element $YZ_i=-1$, that is $YZ_i=-1$ when $\sum_{j=1}^{|Bus|}y_{i,j}=0$. Correspondingly, the elements in the communication allocation matrix Y satisfy:

$$y_{i,j}=\begin{cases}1 & j=YZ_i\\ 0 & j\neq YZ_i\end{cases}\qquad i=1,2,\ldots,|M|.\tag{61}$$

Subsequently, the problem of determining the feasibility of SOTIF requirements for a given communication resource combination scheme $NumABus$ is constructed in the followingCSP form:

$$\begin{aligned}&\text{variable } Z_1\leftarrow XZ\\ &\text{variable } Z_2\leftarrow YZ\\ &\text{corresponding domain } D_1\leftarrow\{1,2,\ldots,|N|\}\\ &\text{corresponding domain } D_2\leftarrow\{-1,1,2,\ldots,|Bus|\}\\ &\text{constraint } C\leftarrow Con01\sim Con14\end{aligned},\tag{62}$$

where $|N|=\sum_{i=1}^{|NumAR|}numAR_i$, $|Bus|=\sum_{i=1}^{|NumABus|}numABus_i$.

When a feasible solution is found for the CSP in Eq. (62), the current communication resource combination scheme is optimal and meets all requirements. If no feasible solution exists, the scheme must be revised, and the next option in the set $RegionABus$ should be evaluated. The final optimal computing resource combination scheme, denoted as $\langle NumAR,NumABus\rangle$, is determined by combining the best solutions from both the "processor resource selection" and "communication resource selection" problems. To maintain consistency with the symbols used in Section 3.2, we define $N\leftarrow NumAR$ and $Bus\leftarrow NumABus$. Therefore, the optimal computing resource combination scheme is represented as $\langle N,Bus\rangle$.

5 Comprehensive optimization for resource allocation considering SOTIF and rational computational power allocation

This subsection discusses trade-offs among design objectives and optimal solution selection in computational resource allocation schemes for functional safety. It introduces constrained optimization problems and NSGA-II, detailing optimization based on safety constraints.

5.1 Constraint optimization problems and the NSGA-II algorithm

In this study, the Non-Dominated Sorting Genetic Algorithm (NSGA-II) (Deb et al. 2002) is used to design the computing resource allocation scheme. Table 3 provides the pseudocode for NSGA-II. This algorithm enhances traditional genetic algorithms by integrating Pareto sorting to obtain Pareto-optimal solutions for multiple objectives. NSGA-II generally comprises the following six steps: 1) population initialization; 2) non-dominated sorting; 3) crowding distance assignment; 4) optimization selection; 5) genetic population generation (crossover and mutation); 6) population recombination and generation.

Table 3 Pseudocode of NSGA-II main program

1: population initialization P_0
2: $t \leftarrow 0$
3: while condition
4: $Q_t \leftarrow$Make-New-Population (P_t)
5: $R_t \leftarrow P_t \cup Q_t$
6: $F \leftarrow$ Fast-Nondominated-Sort (R_t)
7: $P_{t+1} \leftarrow \varnothing$ and $i \leftarrow 1$
8: **while** $|P_{t+1}| + |F_i| \leqslant N$
9: Crowding-Distance-Assignment (F_i)
10: $P_{t+1} \leftarrow P_{t+1} \cup F_i$
11: $i \leftarrow i+1$
12: **end while**
13: Sort ($F_i, \prec_n$)
14: $P_{t+1} \leftarrow P_{t+1} \cup F_i[1:(N-|P_{t+1}|)]$
15: $t \leftarrow t+1$
16: **end while**

5.2 Optimization methods for computing resource allocation schemes

For a given combination of computing resources $\langle N, Bus\rangle$, there may be multiple feasible allocation schemes for computing resources, with differences in load distribution and communication costs. According to Eq. (13), the resource connection matrix A can be derived from the functional allocation matrix X and the communication allocation matrix Y. Therefore, the independent variables that need to be determined in the process of designing the computing resource allocation scheme include the functional allocation matrix X and the communication allocation matrix Y We represent the computing resource allocation scheme as $\langle X, Y\rangle$ (or $\langle X, Y, A\rangle$), and consider the following three optimization objectives in this paper:

1) Achieving as uniform a load distribution as possible, balancing the task utilization of processor resources in the system

Task utilization is a key metric for evaluating processor resource performance. It represents the proportion of time that processor resources are actively executing tasks during system operation. The utilization of processor resources depends on the functions assigned and running on them. For a function F_i, its task utilization is defined as follows:

$$U(F_i) = \sum_{r=1}^{|F_i|} U(\tau_{i,r}) = \sum_{r=1}^{|F_i|} \frac{C_{i,r}}{T_{i,r}}. \tag{63}$$

Therefore, for a given configuration scheme $\langle X, Y\rangle$, considering the task utilization factor, this article uses the second-order central moment to represent the optimization objective as follows:

$$U(X,Y) = X^T \cdot U(F)$$
$$\min UXY = \frac{\left(U(X,Y) - \overline{U(X,Y)}\right)^T \cdot \left(U(X,Y) - \overline{U(X,Y)}\right)}{|N|} \tag{64}$$

where $U(F)$ represents a $|F|\times 1$ column vector composed of $U(F_i)$; $U(X,Y)$ represents a $|N|\times 1$ column vector composed of the task utilization factor of each processor resource under the configuration scheme $\langle X, Y\rangle$; and $\overline{U(X,Y)}$ represents the average task utilization factor of each processor resource.

2) Allocate the load as evenly as possible and balance the memory utilization of processor resources in the system

Similarly, considering the memory utilization of processor resources, this paper uses the second-order central moment to represent the optimization objective as follows:

$$R(X,Y) = (diag(SRAM))^{-1} \cdot X^{\mathrm{T}} \cdot SRR$$
$$\min RXY = \frac{\left(R(X,Y) - \overline{R(X,Y)}\right)^{\mathrm{T}} \cdot \left(R(X,Y) - \overline{R(X,Y)}\right)}{|N|}, \tag{65}$$

where $R(X,Y)$ represents an $|N|\times 1$ column vector composed of the memory utilization of each processor resource under the configuration $\langle X, Y\rangle$, and $\overline{U(X,Y)}$ represents the average memory utilization of each processor resource.

3) Minimize the communication cost required by the system as much as possible

Considering the total communication cost required by the system, this paper represents the optimization objective using the first-order moment as follows:

$$\min TXY = SN^{\mathrm{T}} \cdot rdiag((1 - X^{\mathrm{T}}) \cdot E \cdot X). \tag{66}$$

Considering the three factors mentioned above, the optimization problem for allocating computing resources $\langle X, Y\rangle$ can be defined as the following multi-objective constrained optimization problem:

$$\min_{\langle X,Y\rangle} [UXY, RXY, TXY]$$
$$\text{s.t.} \quad Con01 \sim Con14. \tag{67}$$

This paper uses NSGA-II to solve the multi-objective constrained optimization problem described in Eq. (67). When using NSGA-II, a chromosome encoding format, as shown in Fig. 7 is adopted. The chromosome consists of a total of $|F| + |M|$ bits, where the first $|F|$ bits correspond to the allocation of one objective function each, and the last $|M|$ bits correspond to a specific connection of processor resources. It is easy to convert the chromosome

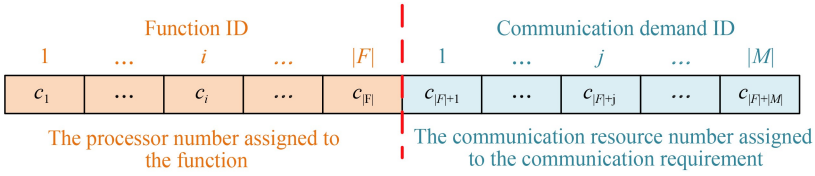

Fig. 7 The chromosome encoding method for optimization design of computing resource allocation scheme.

encoding into a computing resource allocation scheme $\langle X, Y\rangle$ using Eqs. (53) and (60).

Considering the nonlinear nature of SOTIF constraints, this paper uses the penalty function method to transform the multi-objective optimization problem into an unconstrained optimization problem. The encoding method illustrated in Fig. 1 automatically satisfies the constraints $Con01$ and $Con02$ for the resource allocation scheme $\langle X, Y\rangle$. Therefore, penalties are designed for constraints $Con03 \sim Con14$, as follows:

$$
\begin{aligned}
PE_1 =& \max(\|rdiag(MF\cdot X\cdot X^{\mathrm{T}}\cdot FM)\\
&-(1-Y)\cdot SBus\|_1, 0)\\
PE_2 =& SN^{\mathrm{T}}\cdot \max(diag(SCal)\cdot X^{\mathrm{T}}\cdot SAlpha - SN, 0)\\
PE_3 =& SM^{\mathrm{T}}\cdot \max(diag(SBeta)\cdot Y\cdot SCOMB\\
&+(diag(SDLb))^{-1}\cdot Y\cdot diag(SCOMB)\\
&\cdot Y^{\mathrm{T}}\cdot SCM + Y\cdot diag(SCOMB)\cdot Y^{\mathrm{T}}\\
&\cdot SCTM - SM, 0)\\
PE_4 =& SN^{\mathrm{T}}\cdot \max(X^{\mathrm{T}}\cdot SRR - SRAM, 0)\\
PE_5 =& SN^{\mathrm{T}}\cdot \max(rdiag((1-X^{\mathrm{T}})\cdot E\cdot X) - SBW, 0)\\
PE_6 =& SN^{\mathrm{T}}\cdot \max(rdiag((1-X^{\mathrm{T}})\cdot E^{\mathrm{T}}\cdot X) - SBW, 0)\\
PE_7 =& SN^{\mathrm{T}}\cdot \max(X^{\mathrm{T}}\cdot MF^{\mathrm{T}}\cdot diag(rdiag(MF\cdot E^{\mathrm{T}}\cdot FM))\\
&\cdot Y - BWNB, 0)\cdot SBus\\
PE_8 =& SN^{\mathrm{T}}\cdot \max(X^{\mathrm{T}}\cdot FM\cdot diag(rdiag(MF\cdot E^{\mathrm{T}}\cdot FM))\\
&\cdot Y - BWNB, 0)\cdot SBus\\
PE_9 =& SN^{\mathrm{T}}\cdot \max(SPI + diag(SPM)\cdot X^{\mathrm{T}}\cdot SRR\\
&+diag(SPC)\cdot X^{\mathrm{T}}\cdot SCT - SPUb, 0)\\
PE_{10} =& SBus^{\mathrm{T}}\cdot \max(SPIB + diag(SPTB)\cdot Y^{\mathrm{T}}\\
&\cdot SCTM - SPUbB, 0)\\
PE_{11} =& \max(\|\max(X\cdot X^{\mathrm{T}}, ME) - ME\|_1, 0)\\
PE_{12} =& \max(\|\min(X\cdot X^{\mathrm{T}}, MC) - MC\|_1, 0),
\end{aligned}
\tag{68}
$$

where $\|\cdot\|_1$ represents the column sum norm of a matrix, which is the maximum value of the absolute sum of all column vectors of the matrix; $\max(\cdot)$ represents the maximum value taken element-wise. In this article, we further integrate all penalty functions and express them in the form of Eq. (69):

$$
PEXY = \sum_{i=1}^{12}\theta_i\cdot PE_i = \Theta^{\mathrm{T}}\cdot PE, \tag{69}
$$

where θ_i represents the weight value of each normalized penalty function indicator; Θ and PE represent the weight values and row vector composed of each penalty function indicator, respectively.

By combining the objective function and penalty function, the fitness function used in NSGA-II for problem-solving can be expressed as:

$$
\begin{aligned}
Fitness_1 &= \omega_1\cdot UXY + PEXY,\\
Fitness_2 &= \omega_2\cdot RXY + PEXY,\\
Fitness_3 &= \omega_3\cdot TXY + PEXY,
\end{aligned}
\tag{70}
$$

where ω_i is also a weight value used for normalization.

At this point, based on the chromosome encoding defined in Fig. 7. and the fitness function defined in Eq. (70), the NSGA-II algorithm provided in Table 3 can be smoothly executed.In practice, the following tricks are recommended in order to obtain the optimal computed resource allocation $\langle X, Y\rangle$:

1) All the weight values should be trailed before the NSGA-II algorithm is executed.On the one hand, throughestimating the magnitudes of UXY, RXY and TXY, we canpreset the value of ω_i to ensure that $\omega_1\cdot UXY$, $\omega_2\cdot RXY$, and $\omega_3\cdot TXY$ are of the same magnitude. On the other hand, because lower fitness values indicate better solutions, it is believedthat an unsatisfied-scheme $\langle X, Y\rangle$ will reach a rather high fitness value by setting the condition $\theta_i \gg \omega_i$. Therefore, throughestimating the magnitudes of thepenalty functions, we can preset the value of Θ to ensure the $PEXY$ will be significantly higher than $\omega_1\cdot UXY$, $\omega_2\cdot RXY$, and $\omega_3\cdot TXY$.

2) The control parameters of the NSGA-II should be scientifically designed.Control parameters, such as population size Num_{pop}, crossover rate p_c and mutation rate p_m, have a direct impact on the performance and stability of the algorithm. According to practical experience,the preferredranges of initialvalues for Num_{pop}, p_c and p_m are [20, 200], [0.6, 1], and [0.01, 0.1], respectively. There are several options to better determine these control parameters. For example, we can carry out the control parameter tuning experiments and observe the specific impact on the performance of the algorithm.Besides, adaptive probabilities of crossover and mutation have also been a favored choice (Srinivas and Patnaik, 1994).

3) The correctness of the final solutions should be verified after the optimal process. Because ofdefining the penalty functions, it is impossible to completely rule out

the possibility that the final solution does not satisfy the SOTIF constraints. Therefore, it is necessary to verify the correctness of the Pareto optimal solution through automated solution tools. In practice, the feasibility verification code of the solution is similar to the penalty function code, and thus partial code can be reused.

Finally, by combining the computed resource combination $\langle N, Bus\rangle$ obtained in Section 3 and the computed resource allocation $\langle X, Y\rangle$, determined in this section, a computed resource configuration scheme that satisfies all SOTIF requirements is generated.

6 Case study

In this section, we analyze and optimize computing resource allocation schemes for unmanned driving functions in relation to SOTIF requirements. Tables 4 and 5 provide detailed parameters for the expected functions/applications and communication/messages. Based on Table 5, the communication cost matrix E, communication sending matrix FM and communication receiving matrix MF defined in Eq. (71) can be determined as follows:

$$E=\begin{pmatrix}
0 & 0 & 21 & 0 & 10 & 0 & 5 & 12 & 5 & 12\\
12 & 0 & 0 & 11 & 0 & 34 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 24 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
15 & 0 & 0 & 0 & 0 & 17 & 22 & 0 & 22 & 0\\
0 & 0 & 0 & 0 & 21 & 0 & 0 & 22 & 0 & 22\\
0 & 0 & 0 & 16 & 0 & 0 & 0 & 15 & 0 & 0\\
0 & 0 & 12 & 0 & 16 & 0 & 30 & 0 & 0 & 0\\
0 & 0 & 0 & 16 & 0 & 0 & 0 & 0 & 0 & 15\\
0 & 0 & 12 & 0 & 16 & 0 & 0 & 0 & 30 & 0
\end{pmatrix}, \tag{71}$$

$$FM=\begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1
\end{pmatrix}, \tag{72}$$

$$MF=\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{pmatrix}^{\mathrm{T}}. \tag{73}$$

Table 4 Expected parameters related to the functions/applications to be executed

Function ID	Memory requirement RR (MB)	Application ID	Worst execution time C (ms)	Relative time frame D (ms)	Periodicity T (ms)
F_1	70	$\tau_{1,1}$	1	20	20
		$\tau_{1,2}$	2	35	35
		$\tau_{1,3}$	3	40	40
F_2	90	$\tau_{2,1}$	2	30	30
		$\tau_{2,2}$	4	50	50
F_3	50	$\tau_{3,1}$	4	55	55
		$\tau_{3,2}$	5	80	80
F_4	100	$\tau_{4,1}$	7	90	90
F_5	80	$\tau_{5,1}$	2	35	35
		$\tau_{5,2}$	2	40	40
		$\tau_{5,3}$	3	40	40
F_6	80	$\tau_{6,1}$	1	25	25
		$\tau_{6,2}$	2	50	50
		$\tau_{6,3}$	4	60	60
F_7	70	$\tau_{7,1}$	3	35	35
		$\tau_{7,2}$	4	60	60
F_8	60	$\tau_{8,1}$	2	30	30
		$\tau_{8,2}$	2	50	50
F_9	70	$\tau_{9,1}$	3	35	35
		$\tau_{9,2}$	4	60	60
F_{10}	60	$\tau_{10,1}$	2	30	30
		$\tau_{10,2}$	2	50	50

6.1 Computing resource selection

The selection and optimization of computing resources according to SOTIF requirements is divided into two phases: "Processor resource selection" and "Communication resource selection." In general terms, the power consumption threshold is set to 150 W for processor resources and 70 W for communication resources.

By analyzing the Processor Resource Selection problem, the lower and upper limits for the number of processor resources selected in the repository AR shown in Table 6 are first defined as follows:

$$numAR^{lb} = \frac{730}{512} = 1.43 numAR^{ub} = 4. \tag{74}$$

Furthermore, a total of $2793(7^2+7^3+7^4)$ potential solution sets were constructed, and a preliminary screening was performed based on constraint $Con15$ defined in Eq. (50), resulting in the set $RegionAR$, composed of 2587 potential solutions, sorted in ascending order of cost. Based on the Forward Checking algorithm, the corresponding constraint satisfaction problem was solved, and the 431st processor resource combination $NumAR = (0\ 0\ 1\ 1\ 2\ 0\ 0)$ was ultimately determined as the optimal solution that satisfies SOTIF constraints ($Con01$, $Con04$, $Con06$ $Con08$, $Con11$, $Con13$, $Con14$) which selects one AR_3, one AR_4, and two AR_2. For this processor resource solution, the assignment order and feasible solutions obtained using the "least remaining value" and "maximum memory requirement" heuristics with the Forward Checking algorithm are as follows:

$$\begin{aligned} &F4:1\ \ F5:2\ \ F6:2\ \ F1:1\ \ F2:2\ \ F7:3 \\ &F9:4\ \ F8:3\ \ F10:4\ \ F3:1, \end{aligned} \tag{75}$$

$$XZ = (1\ 2\ 1\ 1\ 2\ 2\ 3\ 3\ 4\ 4)^{\mathrm{T}}. \tag{76}$$

This paper compares the actual number of assignment operations with the expected upper limit of operations when applying the Forward Checking algorithm to solve the CSP for each candidate solution $RegionAR_i$, as shown in Fig. 8. The results indicate a significant reduction in the number of assignment operations due to the use of the "minimum remaining value" and "maximum memory requirement" heuristics, which effectively minimize unnecessary operations.For example, for $RegionAR_{200} = (0\ 0\ 2\ 0\ 0\ 2\ 0)$, the total number of hardware resources used is 4. If a full search traversal is performed, the overall expected upper limit of assignment operations is

Table 5 Expected communication/message-related parameters to be transmitted

Communication ID	Message ID	Bandwidth requirements BW (Mb/s)	Transfer time C (ms)	Relative time rame D (ms)	Periodicity T (ms)	Source ID	Objective ID
M_1	$m_{1,1}$	7	0.04	20	20	$\tau_{1,1}$	$\tau_{3,1}$
	$m_{1,2}$	14	0.06	40	40	$\tau_{1,3}$	$\tau_{3,2}$
M_2	$m_{2,1}$	10	0.05	35	35	$\tau_{1,2}$	$\tau_{5,1}$
M_3	$m_{3,1}$	5	0.03	20	20	$\tau_{1,1}$	$\tau_{7,1}$
M_4	$m_{4,1}$	12	0.06	35	35	$\tau_{1,2}$	$\tau_{8,2}$
M_5	$m_{5,1}$	5	0.03	20	20	$\tau_{1,1}$	$\tau_{9,1}$
M_6	$m_{6,1}$	12	0.06	35	35	$\tau_{1,2}$	$\tau_{10,2}$
M_7	$m_{7,1}$	12	0.06	30	30	$\tau_{2,1}$	$\tau_{1,2}$
M_8	$m_{8,1}$	11	0.06	50	50	$\tau_{2,2}$	$\tau_{4,1}$
M_9	$m_{9,1}$	14	0.06	30	30	$\tau_{2,1}$	$\tau_{6,2}$
	$m_{9,2}$	20	0.1	50	50	$\tau_{2,2}$	$\tau_{6,3}$
M_{10}	$m_{10,1}$	24	0.12	80	80	$\tau_{3,2}$	$\tau_{4,1}$
M_{11}	$m_{11,1}$	15	0.07	40	40	$\tau_{5,3}$	$\tau_{1,3}$
M_{12}	$m_{12,1}$	17	0.08	40	40	$\tau_{5,2}$	$\tau_{6,2}$
M_{13}	$m_{13,1}$	10	0.05	35	35	$\tau_{5,1}$	$\tau_{7,1}$
	$m_{13,2}$	12	0.06	40	40	$\tau_{5,2}$	$\tau_{7,2}$
M_{14}	$m_{14,1}$	10	0.05	35	35	$\tau_{5,1}$	$\tau_{9,1}$
	$m_{14,2}$	12	0.06	40	40	$\tau_{5,2}$	$\tau_{9,2}$
M_{15}	$m_{15,1}$	7	0.04	25	25	$\tau_{6,1}$	$\tau_{5,1}$
	$m_{15,2}$	14	0.06	25	25	$\tau_{6,1}$	$\tau_{5,3}$
M_{16}	$m_{16,1}$	5	0.03	25	25	$\tau_{6,1}$	$\tau_{8,1}$
	$m_{16,2}$	17	0.08	50	50	$\tau_{6,2}$	$\tau_{8,2}$
M_{17}	$m_{17,1}$	5	0.03	25	25	$\tau_{6,1}$	$\tau_{10,1}$
	$m_{17,2}$	17	0.08	50	50	$\tau_{6,2}$	$\tau_{10,2}$
M_{18}	$m_{18,1}$	16	0.08	60	60	$\tau_{7,2}$	$\tau_{4,1}$
M_{19}	$m_{19,1}$	15	0.06	35	35	$\tau_{7,1}$	$\tau_{8,2}$
M_{20}	$m_{20,1}$	12	0.06	50	50	$\tau_{8,2}$	$\tau_{3,2}$
M_{21}	$m_{21,1}$	16	0.08	30	30	$\tau_{8,1}$	$\tau_{5,3}$
M_{22}	$m_{22,1}$	14	0.06	30	30	$\tau_{8,1}$	$\tau_{7,1}$
	$m_{22,2}$	16	0.07	50	50	$\tau_{8,2}$	$\tau_{7,2}$
M_{23}	$m_{23,1}$	16	0.08	60	60	$\tau_{9,2}$	$\tau_{4,1}$
M_{24}	$m_{24,1}$	15	0.06	35	35	$\tau_{9,1}$	$\tau_{10,2}$
M_{25}	$m_{25,1}$	12	0.06	50	50	$\tau_{10,2}$	$\tau_{3,2}$
M_{26}	$m_{26,1}$	16	0.08	30	30	$\tau_{10,1}$	$\tau_{5,3}$
M_{27}	$m_{27,1}$	14	0.06	30	30	$\tau_{10,1}$	$\tau_{9,1}$
	$m_{27,2}$	16	0.07	50	50	$\tau_{10,2}$	$\tau_{9,2}$

$4^7 = 16384$, However, after only 40 assignment operations using the Forward Checking algorithm, it is determined that the current processor resource combination cannot satisfy all SOTIF requirements.

On the basis of the optimal processor resource combination $NumAR = (0\ 0\ 1\ 1\ 2\ 0\ 0)$ obtained in this paper, further optimization design is carried out for the communication resource selection problem. Similarly, the lower and upper limits of the number of communication resources selected in the resource library $ABus$ described in Table 7 are defined as:

$$numABus^{lb} = 1 numABus^{ub} = 4. \tag{77}$$

Furthermore, a set of $120(\cdot)$ potential solutionsare constructed in $RegionABus$, where the solution numbers

Table 6 The candidate processor resource parameters

ID	*RAM* (MB)	*BW* (Mb/s)	Computing power *Cal*	Static power consumption P^{idle} (W)	Power consumption for storage operation P^{memory} (W/MB)	Calculating operational power consumption P^{compute} (W)	Price *Cost*
AR_1	256	100	1	100	0.0016	80	200
AR_2	256	100	1.1	110	0.0016	90	300
AR_3	256	100	0.9	90	0.0016	70	170
AR_4	256	150	1	100	0.0016	80	250
AR_5	256	80	0.9	90	0.0016	70	140
AR_6	128	100	0.8	80	0.0016	65	150
AR_7	512	100	1.2	120	0.0016	95	400

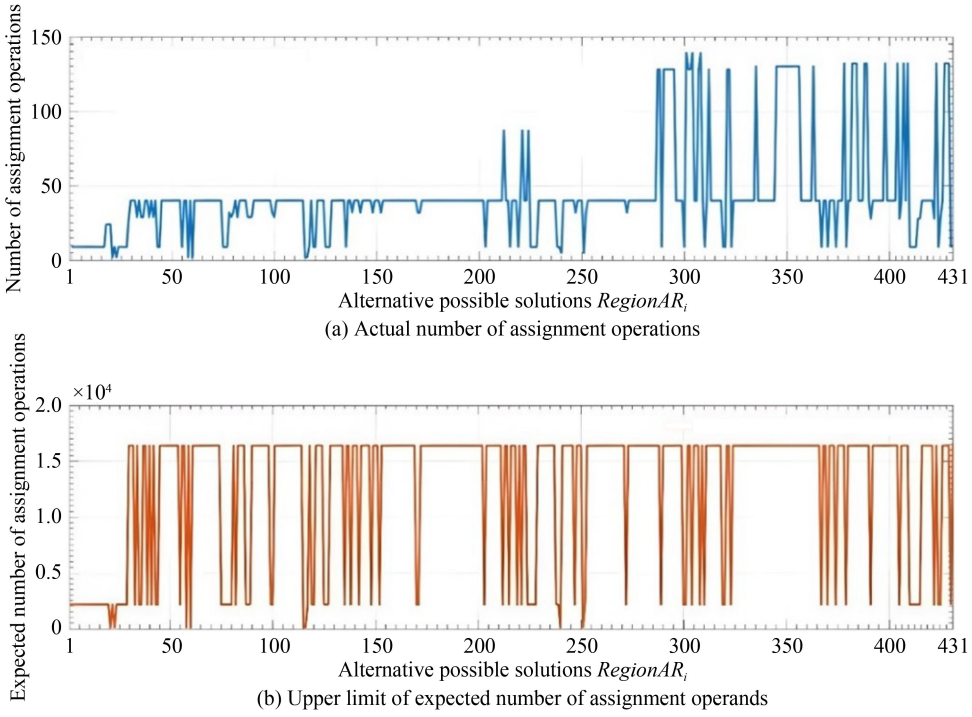

Fig. 8 Curve showing the change in assignment operations with respect to the candidate solution $RegionAR_i$.

Table 7 The candidate communication resource parameters

ID	BW_{Bus} (Mb/s)	Communication capability Com_{Bus}	Static power consumption P_{Bus}^{idle} (W)	Power consumption for communication operation $P_{Bus}^{\text{transmit}}$ (W)	Price *Cost*
$ABus_1$	100	1	45	35	270
$ABus_2$	80	0.8	40	33	240
$ABus_3$	50	0.5	36	29	200

are arranged in ascending order of cost. Based on the Forward Checking algorithm, the corresponding constraint satisfaction problem is solved, and the 9th communication resource combination solution $NumABus = (0\ \ 2\ \ 0)$ is determined as the optimal solution that can satisfy SOTIF constraints ($Con01 - Con14$), i.e., two $ABus_2$. are selected. The assignment sequence and feasible solutions for this communication resource solution are shown below:

$$\begin{gathered} F4:1\ \ F5:2\ \ F6:2\ \ M12:-1\ \ M15:-1\ \ F7:3 \\ M13:1\ \ M18:2\ \ F9:4\ \ M14:1\ \ M23:1\ \ F2:1 \\ F1:2\ \ F8:3\ \ F10:4\ \ F3:1\ \ M1:1\ \ M2:-1 \\ M3:1\ \ M4:2\ \ M5:1\ \ M6:2\ \ M7:1\ \ M8:-1 \\ M9:1\ \ M10:-1\ \ M11:-1\ \ M16:2\ \ M17:2 \\ M19:-1\ \ M20:1\ \ M21:1\ \ M22:-1\ \ M24:-1 \\ M25:1\ \ M26:1\ \ M27:-1 \end{gathered} \tag{78}$$

$$XZ = (2\ 1\ 1\ 1\ 2\ 2\ 3\ 3\ 4\ 4)^{\mathrm{T}}$$
$$YZ = (1\ -1\ 1\ 2\ 1\ 2\ 1\ -1\ 1\ -1\ -1\ -1\ 1\ 1\ -1\ 2\ 2\ 2\ -1\ 1\ 1\ -1\ 1\ -1\ 1\ 1\ -1)^{\mathrm{T}}. \quad (79)$$

In summary, the optimal result for the selection of computing resources that meets SOTIF requirements is:

$$NumAR = (0\ 0\ 1\ 1\ 2\ 0\ 0) \quad NumABus = (0\ 2\ 0). \quad (80)$$

6.2 Computingresource allocation

Based on the optimal compute resource selection scheme determined by Eq. (80), this section will further use the NSGA-II algorithm to determine the optimal compute resource allocation scheme that satisfies SOTIF requirements. To set the penalty coefficient more reasonably, this paper first makes a preliminary estimate of the order of magnitude of the objective functions defined in Eqs. (64)–(66) and the penalty function defined in Eq. (69), where the magnitudes of UXY, RXY and TXY are approximately 0.01, 0.01, and 100, respectively, while the penalty function is generally not less than 10. Therefore, based on experience, the weighting coefficients related to the objective functions in Eqs. (69) and (70) are specified as follows:

$$\omega_1 = 100\ \omega_2 = 100\ \omega_3 = 0.001\ \theta_i = 1000\ (i = 1, 2, \ldots, 12). \quad (81)$$

In this case, in order to ensure the search strength for the solution space,the population size and the maximum genetic generation are set as 200 and 500, respectively. Meanwhile, the adaptive crossover rate p_c and mutation rate p_m is used to enhancethe performance and stability of the algorithm. Drawing on Srinivas' widely recognized research (Srinivas and Patnaik, 1994), detailed definitions of crossover rate p_c and mutation rate p_m for each generation are specified as follows:

$$p_c = \begin{cases} \prod_{i=1}^{3} \min\left(\dfrac{Fitness_i' - Fitness_{\min}}{\overline{Fitness_i} - Fitness_{\min}},\ 1\right) & \exists Fitness_i' \leqslant \overline{Fitness_i} \\ 1 & \forall Fitness_i' > \overline{Fitness_i} \end{cases}, \quad (82)$$

$$p_m = \begin{cases} 0.5 \cdot \prod_{i=1}^{3} \min\left(\dfrac{Fitness_i - Fitness_{\min}}{\overline{Fitness_i} - Fitness_{\min}},\ 1\right) & \exists Fitness_i \leqslant \overline{Fitness_i} \\ 0.5 & Fitness_i > \overline{Fitness_i} \end{cases}, \quad (83)$$

where $Fitness_i'$ is the larger of thefitness values $Fitness_i$ of the solutions to be crossed in each generation; $\overline{Fitness}_{\min}$ is the minimum of fitness values $Fitness_i$; $\overline{Fitness_i}$ is the average of fitness values $Fitness_i$. Both Eqs. (82) and (83) reflect that the closerto the $Fitness_{\min}$, the smaller rate should be set.

Figure 9 illustrates the NSGA-II optimization process. Figure 9(a) Shows the average fitness values of 200 individuals over generations, with initial sharp drops indicating elimination of infeasible solutions. As shown in Fig. 9(b), after about 8 generations, the curve of the Pareto optimal total number of individuals has remained at 200.

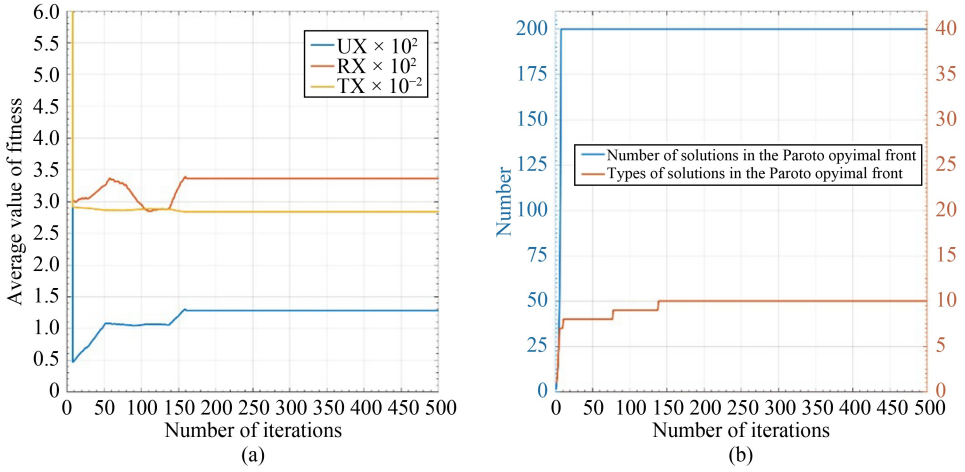

Fig. 9 Optimization process of computing resource allocation scheme.

However, the curve for the Pareto optimal total types of individual types only began to converge after about 139 generations, and finally stabilized at 10. This convergence indicates solution stability despite maintaining solution diversity.

For the Pareto optimal solution generated after 500 iterations, 70 duplicate individuals were further removed in the last generation. And the remaining 10 different solutions are presented in Table 8. Obviously, the fitness values of all Pareto optimal solutions will already have their corresponding penalty functions at 0 after 500 iterations. Therefore, the original objective functions UXY, RXY and TXY can be directly obtained based on the fitness values, as shown in Fig. 10. Since these resource allocation schemes are Pareto optimal and thus often require further analysis in practical engineering applications. In this case, there is a cluster of points near the origin of the UXY–RXY view, there is a cluster of points near the origin. Solutions within this cluster are recommended if the focus is on average resource usage, with total communication cost as a secondary consideration. Consequently, the 3 rdsolution is identified as the most suitable allocation scheme in this case.

Based on the analysis in Sections 6.1 and 6.2 the optimal computing resource allocation scheme can be determined as follows:

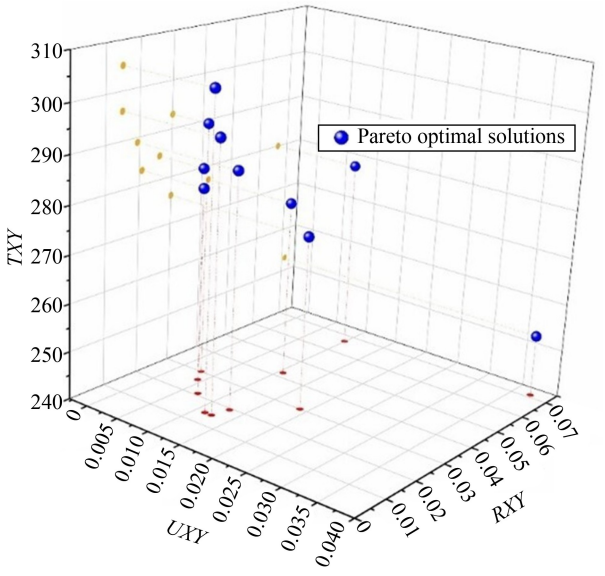

Fig. 10 Visualization results of Pareto optimal solutions for computational resource allocation.

$$\begin{aligned} &NumAR = (0\ \ 0\ \ 1\ \ 1\ \ 2\ \ 0\ \ 0), \quad NumABus = (0\ \ 2\ \ 0)\\ &XZ = (4\ \ 3\ \ 4\ \ 3\ \ 2\ \ 1\ \ 2\ \ 2\ \ 1\ \ 1)^{\mathrm{T}},\\ &YZ = (-1\ \ 2\ \ 1\ \ 1\ \ 2\ \ 1\ \ 1\ \ -1\ \ 1\ \ 2\ \ 1\ \ 1\ \ -1\\ &\quad 2\ \ 2\ \ 1\ \ -1\ \ 2\ \ -1\ \ -1\ \ 2\ \ -1\ \ 1\ \ 1\ \ -1)^{\mathrm{T}},\\ &A = \begin{pmatrix} 1 & 1\\ 1 & 1\\ 1 & 1\\ 1 & 1 \end{pmatrix}, \end{aligned} \tag{84}$$

where the resource connection matrix A is determined according to Eq. (13).

Table 8 The Pareto optimal solutions of the optimization of the computing resource allocation scheme and their corresponding fitness function values

Index	Pareto optimal solutions of computing resource allocation scheme		Fitness function		
	XZ	YZ	$UXY\times100$	$RXY\times100$	$TXY\times0.01$
1	(2,3,3,3,2,1,4,4,1,1)	(1,−1,1,1,2,2,2,−1,2,−1,−1,1,2,2,1,1,−1,2,−1,2,1,−1,2,−1,2,1,−1)	0.291	3.00	2.93
2	(1,3,3,3,2,2,4,4,1,1)	(1,2,1,1,−1,−1,2,−1,1,−1,2,−1,2,2,−1,1,1,2,−1,2,2,−1,2,−1,2,2,−1)	0.427	2.62	2.85
3	(4,3,4,3,2,1,2,2,1,1)	(−1,2,1,1,2,1,1,−1,1,2,1,1,−1,2,2,1,−1,2,−1,2,−1,−1,2,−1,1,1,−1)	0.705	2.09	2.83
4	(4,3,3,3,2,1,2,2,1,1)	(1,2,1,1,2,2,2,−1,1,−1,1,1,−1,2,1,1,−1,2,−1,2,−1,−1,2,−1,2,1,−1)	0.991	6.67	2.8
5	(2,3,3,3,2,2,4,4,1,1)	(1,−1,1,1,1,2,2,−1,1,−1,−1,−1,2,2,−1,1,2,2,−1,2,2,−1,2,2,−1,2,2,−1)	1.063	4.22	2.77
6	(2,3,2,3,1,2,4,4,1,1)	(−1,2,1,1,1,1,2,−1,2,1,1,1,2,−1,2,1,2,2,−1,2,2,−1,2,−1,1,−1,−1)	1.133	1.48	3.05
7	(2,3,2,3,2,1,4,4,1,1)	(−1,−1,1,1,2,1,1,−1,1,2,−1,1,2,2,1,1,−1,2,−1,2,1,−1,2,−1,1,1,−1)	1.237	1.48	2.96
8	(2,3,1,3,2,2,4,4,1,1)	(2,−1,1,1,2,1,2,−1,1,1,−1,−1,2,2,−1,1,1,2,−1,2,2,−1,2,−1,−1,2,−1)	1.297	1.94	2.89
9	(1,3,1,3,2,2,4,4,1,1)	(−1,2,1,1,−1,−1,1,−1,1,1,2,−1,2,2,−1,1,1,2,−1,2,1,−1,2,−1,−1,1,−1)	1.893	3.00	2.76
10	(1,2,1,3,2,2,4,4,1,1)	(−1,1,1,1,−1,−1,2,1,−1,1,1,−1,2,2,−1,1,2,2,−1,2,2,−1,2,−1,−1,1,−1)	3.734	7.12	2.53

7 Conclusions

This paper proposes an innovative computational resource allocation method for unmanned systems considering SOTIF, aiming at enhancing system performance and safety. First, a comprehensive computational resource allocation model is established by considering multidimensional SOTIF constraints, which can ensure SOTIF across various driving scenarios. Secondly, a forward-checking-based optimization method is proposed for computational resource selection, which can reduce search space and improve efficiency. This method can rapidly identify and exclude infeasible resource combinations, and expedite the discovery of optimal allocation schemes meeting all SOTIF constraints. Furthermore, NSGA-II is employed for multi-objective optimization of computational resource allocation schemes by considering SOTIF constraints and balancing various design objectives, such as task load distribution, memory utilization, and communication cost. NSGA-II can effectively deal with multi-objective optimization problems under complex constraints, offering an efficient and practical optimization strategy for computational resource allocation in unmanned systems. Based on the analytical optimization methods proposed in this paper, the optimal result of computing resource selection and the optimal allocation scheme is supplied, which can give some reference for unmanned system SOTIF design.

Compared with the state-of-the-art techniques associated with the SOTIF issues of unmanned systems, this paper proposes a computational resource allocation scheme modeling with multidimensional SOTIF constraints based on graph and set theory as well as a two-step optimization method for computational resource selection and computational resource allocation. Not only is it an important effort to guide the design of computational resource of unmanned system, but also it is asignificant attempt to combine SOTIF analysis with product forward design. Besides, it is noted that our work also has some disadvantages. This paper sets two general search heuristics, namely minimum residual value and maximum memory requirement, for optimizing SOTIF computing resource selection based on forward checking. Future heuristics can supplement these to enhance the optimization efficiency further.

Competing Interests The authors declare that they have no competing interests.

References

Abdulazim A, Elbahaey M, Mohamed A (2021). Putting safety of intended functionality SOTIF into practice. SAE Technical Paper, 2021-01-0196

Birch J, Blackburn D, Botham J, Habli I, Higham D, Monkhouse H, Price G, Ratiu N, Rivett R (2020). A structured argument for assuring Safety of the Intended Functionality (SOTIF). In: Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops. Lecture Notes in Computer Science, 12235: 408–414

Board T (2019). Highway accident report: Collision between vehicle controlled by developmental automated driving system and pedestrian. Available at trb.org/view/1751168

Cai B, Zhao L, Wang Q, Yan M, Fang T (2023). A strategy of vehicle following on slope road at night considering the safety of the intended functionality. Physica A, 624: 128951

Chelouati M, Boussif A, Beugin J, El Koursi E M (2023). Graphical safety assurance case using Goal Structuring Notation (GSN) — challenges, opportunities and a framework for autonomous trains. Reliability Engineering & System Safety, 230: 108933

Chen Z W, Yin S Y, Li L F, Cui W W, Hong D P (2024). Resilience metric and dynamic assessment of unmanned system-of-systems considering cooperative reconfiguration strategies. IEEE Transactions on Reliability, early access: 1–13

Chen, Z W, Zhang, S R, Dui, H Y (2025. Importance-based risk evaluation methodology in transportation cyber-physical systems. Frontiers of Engineering Management,early access: 1–14

Chu J Y, Zhao T D, Jiao J, Yuan Y, Jing Y F (2023). SOTIF-oriented perception evaluation method for forward obstacle detection of autonomous vehicles. IEEE Systems Journal, 17(2): 2319–2330

Collin A, Bilka A, Pendleton S, Tebbens R D (2020). Safety of the intended driving behavior using rulebooks. In: IEEE Intelligent Vehicles Symposium (IV). IEEE, 2020: 136–143

Collin A A C (2019). A systems architecture framework towards hardware selection for autonomous navigation. Dissertation for the Doctoral Degree. Massachusetts Institute of Technology

de Koning M, Machado T, Ahonen A, Strokina N, Dianatfar M, De Rosa F, Minav T, Ghabcheloo R (2024). A comprehensive approach to safety for highly automated off-road machinery under Regulation 2023/1230. Safety Science, 175: 106517

Deb K, Pratap A, Agarwal S, Meyarivan T (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 6(2): 182–197

Esterle K, Aravantinos V, Knoll A (2019). From specifications to behavior: maneuver verification in a semantic state space. In: IEEE Intelligent Vehicles Symposium (IV). 2140–2147

Grabbe N, Kellnberger A, Aydin B, Bengler K (2020). Safety of automated driving: The need for a systems approach and application of the Functional Resonance Analysis Method. Safety Science, 126: 104665

Guo K, Ye Z S, Liu D T, Peng X Y (2021). UAV flight control sensing enhancement with a data-driven adaptive fusion model. Reliability Engineering & System Safety, 213: 107654

Hu J, Xu T, Yan X R, Zhang R C (2022). Validation on Safety of the Intended Functionality of automated vehicles: concept development. SAE International Journal of Connected and Automated Vehicles, 6(12-06-01-0006): 83–97

Expósito Jiménez V J, Winkler B, Castella Triginer J M, Scharke H, Schneider H, Brenner E, Macher G (2024). Safety of the Intended Functionality concept integration into a validation tool suite. ACM SIGAda Ada Letters, 43(2): 69–72

Kalra N, Paddock S M (2016). Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? Transportation Research Part A, Policy and Practice, 94: 182–193

Khastgir S, Brewerton S, Thomas J, Jennings P (2021). Systems approach to creating test scenarios for automated driving systems. Reliability Engineering & System Safety, 215: 107610

Kinalzyk D (2021). SOTIF process and methods in combination with functional safety. In: European Conference on Software Process Improvement. Cham: Springer International Publishing: 612–623

Kondrak G, Van Beek P (1997). A theoretical evaluation of selected backtracking algorithms. Artificial Intelligence, 89(1–2): 365–387

Luo Q, Zhang D, Zhou H, Pang S, Li X, Wang C (2022). Evaluation on driving scenarios for safety of intended functionality of intelligent vehicles. China Safety Science Journal, 32: 140–145

Mackworth A K (1977). Consistency in networks of relations. Artificial Intelligence, 8(1): 99–118

Neurohr C, Westhofen L, Henning T, De Graaff T, Mohlmann E, Bode E (2020). Fundamental considerations around scenario-based testing for automated driving. In: 2020 IEEE intelligent vehicles symposium (IV): 121–127

Pimentel J (2019). Safety of the Intended Functionality. SAE International. Warrendale, PA, USA

Rau P, Becker C, Brewer J (2019). Approach for deriving scenarios for Safety of the Intended Functionality. In: 26th International Technical Conference on the Enhanced Safety of Vehicles (ESV): 1–15

Schnellbach A, Griessnig G (2019). Development of the ISO 21448. In:European Conference on Software Process Improvement. Cham: Springer International Publishing: 585–593

Skoglund M, Warg F, Hansson H, Punnekkat S (2021). Synchronisation of an automotive multi-concern development process. International Conference on Computer Safety, Reliability, and Security. Cham: Springer International Publishing: 63–75

Srinivas M, Patnaik L M (1994). Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Transactions on Systems, Man, and Cybernetics, 24(4): 656–667

Wang B, Luo Y, Zhong Z, Li K (2022). Robust non-fragile fault tolerant control for ensuring the Safety of the Intended Functionality of cooperative adaptive cruise control. IEEE Transactions on Intelligent Transportation Systems, 23(10): 18746–18760

Yan M Y, Chen W W, Wang Q D, Zhao L F, Liang X T, Cai B X (2021). Human–machine cooperative control of intelligent vehicles for lane keeping—Considering Safety of the Intended Functionality. In: Actuators. MDPI, 10(9): 210

Zhang X Y, Zhou M, Shao W B, Luo T, Li J (2019). The architecture of the intended safety system for intelligent driving. In: 2019 IEEE International Symposium on Circuits and Systems (ISCAS): 1–4

Zhang Y, Lintern G, Gao L, Zhang Z (2021). A study on functional safety, SOTIF and RSS from the perspective of human-automation interaction. SAE Technical Paper, 2021–01–0858

Zhao X, Lv Z H, Qiu Q A, Wu Y G (2023). Designing two-level rescue depot location and dynamic rescue policies for unmanned vehicles. Reliability Engineering & System Safety, 233: 109119

Zhou B, Chen C, Zhai Y, Zhao S (2022a). A study on scenario generalization and optimization for ADS. SAE Technical Paper, 2022-01-7007

Zhou H, Li X Y, He X, Li P F, Xiao L Y, Zhang D W (2022b). Research on safety of the intended functionality of automobile AEB perception system in typical dangerous scenarios of two-wheelers. Accident Analysis and Prevention, 173: 106709