Article

Scheduling Optimization of Time-Triggered Cyber-Physical Systems Based on Fuzzy-Controlled QPSO and SMT Solver

Jie Jian , Lide Wang *, Huang Chen and Xiaobo Nie

School of Electrical Engineering, Beijing Jiaotong University, Beijing 100044, China; 13117365@bjtu.edu.cn (J.J.); 15117392@bjtu.edu.cn (H.C.); xbnie@bjtu.edu.cn (X.N.)

* Correspondence: ldwang@bjtu.edu.cn

Received: 9 January 2020; Accepted: 30 January 2020; Published: 4 February 2020

Abstract: The time-triggered communication paradigm is a cost-efficient way to meet the real-time requirements of cyber-physical systems. It is a non-deterministic polynomial NP-complete problem for multi-hop networks and non-strictly periodic traffic. A two-level scheduling approach is proposed to simplify the complexity during optimization. In the first level, a fuzzy-controlled quantum-behaved particle swarm optimization (FQPSO) algorithm is proposed to optimize the scheduling performance by assigning time-triggered frame instances to the basic periods of each link. In order to prevent population from high aggregation, a random mutation mechanism is used to disturb particles at the aggregation point and enhance the diversity at later stages. Fuzzy logic is introduced and well designed to realize a dynamic adaptive adjustment of the contraction–expansion coefficient and mutation rate in FQPSO. In the second level, we use an improved Satisfiability Modulo Theories (SMT) scheduling algorithm to solve the collision-free and temporal constraints. A schedulability ranking method is proposed to accelerate the computation of the SMT-based incremental scheduler. Our approach can co-optimize the jitter and load balance of communication for an off-line schedule. The experiments show that the proposed approach can improve the performance of the scheduling table, reduce the optimization time, and reserve space for incremental messages.

Keywords: real-time systems; communication networks; Ethernet networks; particle swarm optimization; scheduling algorithm

1. Introduction

The cyber-physical system (CPS) is the next-generation engineering system to realize a deep integration of computing, communication, and control technology. Its core technology is the close integration of information systems and physical units, which are considered prospects of a networked control system [1]. Physical units act on the control process of automation in the cores of embedded devices, while information systems make full use of communication facilities and resources to acquire, transfer, and apply CPS instructions and information.

The main difficulty of CPS control lies in the co-scheduling of the physical unit and the information system [2,3]. The sampling and control processes of physical units are run on spatially distributed entities, including controllers, actuators, and sensors, and these are generally divided into periodic embedded task sets which have precedence constraints, different rate intervals, strict deadlines, and jitter requirements. Under non-preemptive communication protocol, it is necessary to efficiently co-schedule tasks of physical and information systems to maximize the utilization of network facilities and resources. Instead of a low speed and low bandwidth fieldbus, the communication process of an information system can be undertaken by switched real-time Ethernet (RTE) protocols governed by IEC 61850 [4] and IEC 61158 [5], which make co-scheduling problems more complicated.

Scheduling problems are generally considered as restricted resource allocation problems with specific dependencies and constraints, which can be addressed using a constraint programming approach. Two types of approaches are formally defined: constraint satisfaction problems (CSPs) and constraint optimization problems (COPs) [6]. The purpose of CSPs is to achieve the satisfiability of a set of constraints, while COPs aim at optimizing an objective function on the basis of constraints satisfaction. In time-triggered CPS scheduling, the hard real-time constraints include the period, deadline, precedence, etc., which must be satisfied while some quality of service (QoS) parameters, such as the jitter, latency and load balance of the scheduling table, can be optimized by COP to improve the communication performance.

A multitude of optimal and heuristic techniques have been developed to solve COPs in time-triggered scheduling. The main problem is the complexity of non-deterministic polynomial (NP) complete, which limits the scale of networks and communication tasks. Although advanced methods and algorithms are proposed in past and recent references, it is still a hot topic in the field of optimization theory.

The scheduling optimization of non-strictly periodic traffic further aggravates the NP-complete problem. In recent research and articles, most of the studies are focused on strictly periodic scheduling, which only needs to schedule the first periodic frame instance of traffic flow. However, allowing non-strictly periodic frames can refine the granularity of the schedule table from flow level to frame level and thus heavily improve the solution space and flexibility.

2. Related Works

The survey article [7] proposes a mathematical modeling framework that sets forward fundamental challenges in sensing, decentralized computation, robustness, energy efficiency, and hardware security in CPS. Many references have explained in detail how the design and scheduling of information systems affect the control process and stability of CPS. Derler [8] and Poovendran [9] analyze the real-time and synchronization characteristics of CPSs, which must be guaranteed. Karsai [10] introduces a joint modeling method of an embedded system based on the dynamic integration of software and physics. Xiaofeng Yu [2] focuses on the information transmission path on physical systems and finds an optimal path to meet the requirement of physical system stability. The influence of information system delay on physical process is studied. Bradley [3] establishes control models for an information system and physical system, respectively. Taking the sampling time as the research object, an optimal control of computational–physical state co-regulation is studied in CPS. The results show that CPS needs industrial communication protocol and a real-time scheduling scheme to integrate information and physical systems.

According to the characteristics and services of a control system, the scheduling methods of tasks among multicores or multi-agents in CPSs are also very different. For physical processes that have high variability, non-Gaussianity, and higher order spatiotemporal correlations in personalized medicine, Bogdan defines a cyber-physical task graph as a time-dependent representation and proposes a goal-oriented self-optimization inspired solution for resource allocation on Network-on-Chip-based multicore platforms [11]. In industrial applications, such as aircraft, high-speed trains, power grids, etc., real-time scheduling methods of fieldbus and distributed communication systems (Lonworks [12], WorldFIP [13], CAN [14], FlexRay [15], etc.) have been widely studied in industrial control applications. Moreover, using Ethernet and transmission control protocol/Internet protocol (TCP/IP) technologies to integrate a local area network (LAN) in CPSs allows utilities to be interconnected with automation networks, which has features such as high bandwidth, openness, wide availability, and low cost [16,17]. Although this work focuses on fully switched real-time Ethernet, it can also be used in fieldbus protocol or another switched real-time industrial network by extension. The inherently non-deterministic carrier-sense with collision detection (CSMA/CD) mechanism conditioned the adoption of Ethernet in time-critical applications [18]. To overcome those shortcomings, several real-time protocols have been

introduced to enforce time-critical communication over Ethernet [19]. One cost-efficient way is the time-triggered scheduling, which plans transmission points of frames off-line.

Some articles, such as [20–22] propose QoS-based design methods for fieldbus networks with the bandwidth allocation described as multi-objective optimization problems. Natale [23] solves the problem with an objective to minimize the jitter of activities using simulated annealing. However, the time complexity of NP-complete will become more prominent with the increase of network and traffic scale. The algorithms in these references are only applicable to fieldbus scenarios and are difficult to work at full capacity in a multi-hop switched Ethernet.

Many related studies have done a lot of work to realize real-time Ethernet scheduling. A rate-monotonic algorithm has been used to accelerate scheduling time in [24,25]. The tabu search algorithm has been used to design the scheduling table off-line and improve the overall real-time performance by reducing the packet delay of rate-constrained frames in [26,27]. The pinwheel scheduling solution [28] is regarded as a relaxation of the jittered-bounded scheduling concept, where each activity is required to be scheduled at least once during each predefined number of consecutive time units. Hanzalek in [29] introduces a resource-constrained project scheduling with temporal constraints to minimize the Profinet IO IRT scheduling makespan. However, the jobs are aperiodic and executed only once in a macro period. A formal specification of scheduling constraints for time-triggered multi-hop networks and strictly periodic traffic while solving with the Satisfiability Modulo Theories (SMT) solver is introduced in [30–33]. The strictly periodic model leads to a relatively lower resource utilization and flexibility. Besides, frame constraints are added into the solver without considering the urgency and schedulability of frames. Therefore, the computation time of the schedule table increases rapidly with the number of constraints. In large-scale or non-periodic scheduling, it is hard to get a feasible solution within a time limit. A jitter satisfaction problem is solved by a three-level greedy algorithm in [34]. A strictly periodic multi-cluster scheduling approach is proposed in [35] by dividing flows into intra-cluster and inter-cluster. The off-line and on-line algorithms are presented to fast react to cluster-level changes in networks. A novel search-space pruning technique based on the response times is proposed in [36] to generate a schedule for large distributed time-triggered systems within a reasonable amount of time. Due to the NP-complete of multi-hop scheduling, most of the researches are aimed at increasing the scheduling scale and reducing the scheduling time in strictly periodic scheduling while satisfying the constraints. The optimization of scheduling performance with non-strictly periodic traffic has not been fully studied.

The classic particle swarm optimization (PSO) is not a global optimization algorithm. Sun et al. [37] introduces quantum theory into a PSO algorithm and proposes a quantum-behaved particle swarm optimization (QPSO). The contraction–expansion coefficient is the only parameter in QPSO that needs to be set manually besides the population size, particle dimension, and maximum iteration times. Its control strategy has a great influence on the performance of the algorithm. Sun proves that the contraction–expansion (CE) coefficient must be less than 1.781 in order to make the particles converge reliably. The dynamic non-linear decreasing control strategy of the CE coefficient is proposed in [38]. According to the characteristics of different optimization problems, different control curves with concavity and convexity are adopted. However, fixed, linear or non-linear decreasing are all non-feedback control strategies that are obtained by simulation experiments or empirical formulas. This kind of coefficient control strategy has weak self-regulation ability. It cannot adjust the coefficient according to the change of search stage. Liu proposes a novel dynamic CE coefficient for QPSO involving two factors, namely, the evolution speed factor and aggregation degree factor in [39], which can be modified on-line. The fuzzy logic control is one of the most popular control techniques, and it is often used to adjust the parameters of control systems or other algorithms adaptively. The fuzzy logic control is applied in numerous mixes with other algorithms such as the fuzzy logic for inertia weight particle swarm optimization in [40], immune particle swarm optimization in [41], neural networks in [42], model-free adaptive control in [43], and virtual reference feedback tuning in [44]. At present, we have not seen the work of using fuzzy control logic to adaptively control the CE coefficient in QPSO.

This article starts with an introduction of the situation and difficulty of CPS periodic scheduling. Section 2 discusses the related works. A multi-objective optimization problem of non-strictly periodic scheduling in CPS based on time-triggered multi-hop network is proposed in Section 3 for first-level scheduling. A fuzzy-controlled quantum-behaved particle swarm optimization (FQPSO) algorithm for adaptive parameter selection is proposed in Section 4. Section 5 introduces an improved incremental scheduler based on the SMT solver and schedulability ranking of frames to accelerate computation. The experiments and performance discussion are presented in Section 6. This article finishes with conclusion.

This work is different in that it is the first to propose a two-level scheduling approach, which includes optimization and satisfiability parts, to simplify NP-complete complexity. It is also the first to define the load balance of message distribution in the macro cycle in order to improve an incremental schedule of periodic time-triggered messages. Fuzzy logic is firstly used to adaptively adjust CE coefficients in QPSO. An innovation of schedulability ranking in an incremental SMT solver is proposed to solve the problem of excessive callbacks in previous work.

3. Scheduling Optimization

3.1. System Model

This article improves an Ethernet train communication network (TCN) topology in IEC61375-3-4 [45] as an application platform. TCN is a typical real-time CPS, whose topology is hierarchical with one or more train backbone subnets and one or more Ethernet consist network (ECN) subnets. Train backbone nodes (TBN) and links along the train are used to connect active ECN subsets together, while the ECN interconnects end devices (ED) located in one consist and transmits data frames between EDs and between EDs and TBNs through consist network nodes (CNNs) and links. We assume links as full duplex connecting multiprocessor nodes to transmit messages between devices in both directions. Each ED contains a set of homogeneous cores on an electronic control unit (ECU) with output and input ports acting as communication end points that can send or receive only one single message at a time.

The work of this paper is based on time-triggered improved Ethernet and focus on non-strictly periodic process data in TCN. The communication management, routing protocol, communication layer model, and packet format of TCN in our work are regulated by IEC 61375-2-3 [46], which defines a state-of-the-art protocol based on switched Ethernet named train real-time data protocol (TRDP). Our work focuses on the optimization of a static scheduling table that is not confined to specific protocol and is easily extensible to other distributed control system communication networks such as bus, mesh, wireless communication, and so on. Similarly, on the premise of network clock synchronization and determined message routing, the structure of network topology, such as a ring, a tree, or a star, or redundant structures as parallel and ladder structures in IEC 61375-3-4, does not affect the optimization model, analysis, and conclusion of this paper.

The proposed communication topology in this article is shown in Figure 1. A ring topology is formed by three switches and five devices distributed in it. Formally, the physical topology of the information system in CPS can be depicted by a directed graph $G(V, E)$, where $V = ED \cup CNN$ contains all the EDs and CNNs, and E is the set of physical links. One physical link consists of two dataflow links in opposite directions. A dataflow link $l_k \in L$ represents a directed communication link between two adjacent nodes, and L is the set of dataflow links in a network. A dataflow path dp_i is formed by an ordered sequence of l_k connecting one sender to one or several receivers. For example, the dataflow path dp_1 is depicted by a dotted line from ED1 to ED5, which can be also denoted as the ordered l_k sequence, i.e., $l_1 \rightarrow l_5 \rightarrow l_9$ in Figure 1.

All the source and sink ports of EDs in the fieldbus network share the common collision domain so that only one port can occupy the physical link for message transmission. As switched Ethernet is isolated from ports of switches to their respective collision domains, the ports of all EDs can send or

receive messages at the same time. When there is no overlap of dataflow paths, i.e., $dp_1 \cap dp_2 \cap \dots \cap dp_n = \emptyset$, messages can be scheduled as several independent fieldbuses at the same time in the network. Otherwise, $dp_1 \cap dp_2 \cap \dots \cap dp_n \neq \emptyset$, and overlapped links should be co-scheduled through the store and forward mechanism of Ethernet switches, as long as the end-to-end delay can meet the deadline requirements of messages, which enhances the flexibility and utilization of the network.

Figure 1. Topology of the proposed information system in a cyber-physical system (CPS).

In Figure 1, communication between source and sink EDs is distributed globally by dataflow paths and relayed through switches. Messages need to go through several dataflow links during the transmission process. Due to the collision domain isolation characteristics of switch ports, the transmission of messages can be divided into a series of frame instances.

The communication between CPS physical units is undertaken by messages. Assume that a periodic message is transmitted in the payload of a dedicated frame f_i . We denote the set of all frames by F with $f_i \in F$. For each frame, the prior attributes are given as shown in Equation (1):

$$f_i = \{r_i, d_i, e_i, p_i, T_i\}, \quad i = 1, \dots, N \quad (1)$$

where r_i is the release time, which specifies the earliest time that a message can be sent from source ED. d_i is the message deadline, which represents the absolute time from 0 moments to the moment when a message is completely received by its destination. e_i is the end-to-end deadline from the moment when a data frame is sent from source ED to the moment when it is completely received by its destination. p_i is the amount of time that the frame of the message occupies the dataflow link, and T_i is the time period of the message. N is the number of frames.

Under synchronization, the global network communication time is divided into macro periodic units denoted by T_{Mp} , which is equal to the least common multiple (LCM) of all the message periods in the network. In addition, each link has its own basic period denoted by T_{bp} , which is equal to the greatest common divisor (GCD) of the message periods on this link. T_{Mp} is an integral multiple of T_{bp} .

3.2. Objective Function

In order to ensure the real-time performance of time-triggered periodic messages as well as keep more free time slots in the basic periods, the optimization target concentrates on the frame jitter, transmission delay, and load balance of the macro period in the k -th link. In a multi-hop non-strictly periodic scheduling problem, it is usually difficult to satisfy the collision-free constraints through optimization. In this section, we firstly assign frame instances to basic periods, regardless of their overlapping issues.

$s(i, j, l_k)$ and $s(i, j + 1, l_k)$ are the sending time of the j -th and $(j + 1)$ -th instances of frame f_i on dataflow link l_k . M is the number of frames on l_k . The sum of jitters on l_k can be formulated as shown in Equation (2).

$$J(l_k) = \sum_{i=1}^M \sum_{j=1}^{T_{Mp}/T_i} \left| \frac{s(i, j+1, l_k) - s(i, j, l_k) - T_i}{T_i} \right| \quad (2)$$

The periodic frames should be distributed uniformly so as to prevent data overflow and improve the load more balanced for incremental scheduling. The standard deviation of basic period utilization in a macro period can be used to define the load balance and is given by Equation (3):

$$B(l_k) = \sqrt{\frac{1}{N_k} \sum_{j=1}^{N_k} \left(\frac{1}{N_k} \sum_{j=1}^{N_k} \sum_{i=1}^M p_i(r) - \sum_{i=1}^M p_i(r) \right)^2} \quad (3)$$

where N_k is the number of basic periods in one macro period on l_k and equals to $T_{Mp}/T_{bp,k}$, T_{Mp} is the macro period, and $T_{bp,k}$ is the basic period for l_k . $\forall r \in [1, N_k]$, if any frame instance of f_i is located in the r -th basic period, $p_i(r)$ equals the link occupancy time of f_i . Otherwise, $p_i(r) = 0$.

For every message flow, the sending time offset of frames should satisfy not only the constraints on the application level but also on the network level. The temporal constraints of the network are divided into eight types and analyzed in detail [28]. In this article, we use a directed graph to describe them formally. In order to facilitate description, five messages are assumed with their prior attributes defined in Equation (1) shown in Table 1, while the directed graph G is formed as shown in Figure 2.

Table 1. Tasks and temporal constraints of periodic messages. ED: end device.

Message	M1	M2	M3	M4	M5 (Multicast)
Source	ED1	ED3	ED1	ED1	ED2
Destination	ED5	ED5	ED2	ED3	ED1, ED3, ED4, ED5
Links	$l_1 \rightarrow l_5 \rightarrow l_9$	$l_3 \rightarrow l_6 \rightarrow l_9$	$l_1 \rightarrow l_{15}$	$l_1 \rightarrow l_4 \rightarrow l_{14}$	$l_2 \rightarrow \{l_{16}, l_4 \rightarrow l_{14}, l_5 \rightarrow \{l_9, l_{10}\}\}$
$r_i/\mu s$	1.00	13.00	8.00	20.00	5.00
$d_i/\mu s$	40.00	30.00	—	35.00	{—, —, —, 45.00}
$e_i/\mu s$	17.00	19.00	—	—	{—, 24.00, 24.00, 22.00}
$p_i/\mu s$	6.88	6.88	6.88	6.88	11.84
$T_i/\mu s$	32.00	64.00	32.00	64.00	128.00

The link occupancy time $p_i = len \times b_{TD} + T_{GAP}$, where len is the frame length in bits. b_{TD} is the single bit transmission time, which is approximately set as 0.01 μs , and T_{GAP} is the inter-message protection interval with the value of 1.12 μs in this case. For example, if the unicast M1–M4 length is 72 bytes long and multicast M5 is 134 bytes long, we have $p_{1-4} = 6.88 \mu s$, $p_5 = 11.84 \mu s$.

In Figure 2, there are 19 tasks in total marked by N_x with x from 0 to 18. N_0 marks the start time of the system, and the link occupancy time is 0. The positive weight of the directed edges between the task nodes in Figure 2 is depicted as T_{LD}^k representing the link delay T_{LD} on l_k , which consists of two parts: physical links and connected switches. The positive weight between N_0 and the first task node of message transmission represents the release time r_i . The negative weight between the last task node and N_0 is given by $-(d_i - p_i)$ and indicates the absolute deadline of the frame from 0 moments. The negative weight between the last and first task node is given by $-(e_i - p_i)$ and indicates the end-to-end deadline of each message.

Figure 2. Directed graph of task scheduling chart based on temporal constraints.

The weight of the directed edges in Figure 2 forms a matrix W , which represents all the temporal constraints of frame instances during scheduling, as shown in Equation (5). N_x and $N_{x'}$ are tasks in Figure 2 with $x, x' \in [0, 18]$. The element “*” in W indicates that the tasks of N_x and $N_{x'}$ have no constraint relations. The temporal constraints can be formalized as:

$$s(N_x) - s(N_{x'}) \geq w_{x,x'} \quad (4)$$

	N_0	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}	N_{11}	N_{12}	N_{13}	N_{14}	N_{15}	N_{16}	N_{17}	N_{18}
N_0	*	*	*	-33.12	*	*	-23.12	*	*	*	*	-28.12	*	*	*	*	*	*	-33.16
N_1	1.00	*	*	-10.12	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
N_2	*	T_{LD}^1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
N_3	*	*	T_{LD}^5	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
N_4	13.00	*	*	*	*	*	-12.12	*	*	*	*	*	*	*	*	*	*	*	*
N_5	*	*	*	*	T_{LD}^3	*	*	*	*	*	*	*	*	*	*	*	*	*	*
N_6	*	*	*	*	*	T_{LD}^6	*	*	*	*	*	*	*	*	*	*	*	*	*
N_7	8.00	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
N_8	*	*	*	*	*	*	*	T_{LD}^1	*	*	*	*	*	*	*	*	*	*	*
N_9	18.00	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
N_{10}	*	*	*	*	*	*	*	*	*	T_{LD}^1	*	*	*	*	*	*	*	*	*
N_{11}	*	*	*	*	*	*	*	*	*	*	T_{LD}^1	*	*	*	*	*	*	*	*
N_{12}	5.00	*	*	*	*	*	*	*	*	*	*	*	*	*	*	-12.16	*	-12.16	-10.16
N_{13}	*	*	*	*	*	*	*	*	*	*	*	*	T_{LD}^2	*	*	*	*	*	*
N_{14}	*	*	*	*	*	*	*	*	*	*	*	*	T_{LD}^2	*	*	*	*	*	*
N_{15}	*	*	*	*	*	*	*	*	*	*	*	*	*	T_{LD}^4	*	*	*	*	*
N_{16}	*	*	*	*	*	*	*	*	*	*	*	*	T_{LD}^2	*	*	*	*	*	*
N_{17}	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	T_{LD}^5	*	*	*
N_{18}	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	T_{LD}^5	*	*

Therefore, $\forall l_k \in L$, the final objective function can be expressed as shown in Equation (6).

$$\begin{cases} \min F(l_k) = a_1 \frac{J(l_k) - J(l_k)_{\min}}{J(l_k)_{\max} - J(l_k)_{\min}} + a_2 \frac{B(l_k) - B(l_k)_{\min}}{B(l_k)_{\max} - B(l_k)_{\min}} \\ s.t. \sum_{i=1}^M p_i(r) \leq T_{bp,k} \\ s(N_x) - s(N_{x'}) \geq w_{x,x'} \\ a_1 + a_2 = 1, a_1 \geq 0, a_2 \geq 0 \end{cases} \quad (6)$$

where $F(l_k)$ is the objective function of l_k , $J(l_k)$ is the overall jitter value given by Equation (2), and $J(l_k)_{\max}$ and $J(l_k)_{\min}$ are the maximum and minimum value of $J(l_k)$, respectively. $B(l_k)$ is the standard deviation of basic period utilization in a macro period of l_k given by Equation (3), and $B_{\max}$ and $B_{\min}$ are the maximum and minimum value of $B(l_k)$, respectively. a_1 and a_2 are the normalized weight coefficients. The first constraint means that the basic period $T_{bp,k}$ should not overflow with frames. The second constraint is the temporal constraint given by Equation (4).

4. Fuzzy-Controlled Quantum-Behaved Particle Swarm Optimization

4.1. Quantum-Behaved Particle Swarm Optimization

QPSO assumes that the evolutionary system of a PSO algorithm is a quantum system. The aggregation can be described by bounded states existing in the center of particle motion, which is generated by the attraction potential well. Particles in quantum-bounded states can appear at any point in space with a certain probability density.

The exact values of positions and velocity cannot be determined simultaneously with quantum particles. In an N -dimensional search space, the QPSO algorithm consists of M particles representing potential solutions of the problem. At moment t , the relevant variables of the i -th particle are defined as follows with i takes an integer from 1 to M .

- Particles have no velocity vector;
- Position vector is $X_i(t) = [X_{i,1}(t), X_{i,2}(t), \dots, X_{i,N}(t)]$;
- Particle best position is $P_i(t) = [P_{i,1}(t), P_{i,2}(t), \dots, P_{i,N}(t)]$;
- Global best position is $G(t) = [G_1(t), G_2(t), \dots, G_N(t)]$;
- Attractor's vector is $p_i = (p_{i,1}, p_{i,2}, \dots, p_{i,N})$;

Sun et al. [35] present the basic theory and formula of QPSO. Considering that the particle's position $X_{i,j}(t)$, its local attractor $p_{i,j}(t)$, the characteristic length of potential well $L_{i,j}(t)$, and the random variable $u_{i,j}(t)$ develop with the iteration number n , the j -th component of the position of the i -th particle in the $(t+1)$ -th iteration is given by Equation (7).

$$X_{i,j}(t+1) = p_{i,j}(t) \pm \alpha(t) \times \frac{L_{i,j}(t)}{2} \times \ln \left[\frac{1}{u_{i,j}(t)} \right] \quad (7)$$

where $u_{i,j} \sim U(0,1)$, $\alpha(t)$ is the contraction–expansion (CE) coefficient to adjust the convergence rate of particles.

To ensure the convergence of QPSO, each particle must converge to its own local attractor $p_{i,j}(t)$ given by Equation (8), and $\lim_{t \rightarrow \infty} L_{i,j}(t) = 0$ must be satisfied.

$$\begin{aligned} p_{i,j}(t) &= \varphi_j(t) \times P_{i,j}(t) + [1 - \varphi_j(t)] \times G_j(t) \\ \varphi_j(t) &= \frac{c_1 r_{1,j}(t)}{c_1 r_{1,j}(t) + c_2 r_{2,j}(t)} \end{aligned} \quad (8)$$

where c_1 and c_2 are local and global learning factors, $r_{1,j}(t), r_{2,j}(t) \sim U(0,1)$.

The control strategy of $L_{i,j}(t)$ is the key to affect the convergence and performance of QPSO. The typical $L_{i,j}(t)$ control method is to regulate the distance among particles and the mean best position $C(t)$ is given by Equation (9).

$$\begin{aligned} C(t) &= (C_1(t), C_2(t), \dots, C_N(t)) \\ &= \left(\frac{1}{M} \sum_{i=1}^M P_{i,1}(t), \frac{1}{M} \sum_{i=1}^M P_{i,2}(t), \dots, \frac{1}{M} \sum_{i=1}^M P_{i,N}(t) \right) \end{aligned} \quad (9)$$

The value of $L_{i,j}(t)$ can be given by Equation (10).

$$L_{i,j}(t) = 2\alpha(t) \times |C_j(t) - X_{i,j}(t)| \quad (10)$$

Therefore, the final evolution equation of QPSO is given by Equation (11).

$$\begin{aligned} X_{i,j}(t+1) &= \frac{c_1 r_{1,j}(t) P_{i,j}(t) + c_1 r_{1,j}(t) G_{i,j}(t)}{c_1 r_{1,j}(t) + c_2 r_{2,j}(t)} \\ &\quad \pm \alpha(t) \times |C_j(t) - X_{i,j}(t)| \times \ln \left[\frac{1}{u_{i,j}(t)} \right] \end{aligned} \quad (11)$$

4.2. Fuzzy-Controlled Adaptive CE Coefficient

The relationship between the CE coefficient $\alpha(t)$ and potential well length $L(t)$ is discussed in [35]. The simulation results show that $\alpha(t)$ directly affects the length of $L(t)$ and restricts the searching range of particles. When $\alpha(t)$ and $L(t)$ are large, particle swarm has a strong global searching performance. Otherwise, the local searching performance is enhanced. The fixed, linear, or non-linear strategy of coefficients did not consider the relationship between $\alpha(t)$ and particle positions. In this article, a control strategy of CE coefficient based on fuzzy logic is proposed.

We define a population diversity factor $\beta(t)$ to show the dispersion degree of particles and then to indicate the change of the searching range of particles, as shown in Equation (12).

$$\beta(t) = \frac{1}{M \cdot R} \cdot \sum_{i=1}^M \sqrt{\frac{1}{N} \sum_{j=1}^N (X_{i,j}(t) - \overline{X}_j(t))^2} \quad (12)$$

where R is the longest radius of the searching space, $X_{i,j}(t)$ is the j -th component of the i -th particle, $\overline{X}_j(t)$ is the average value of the j -th component for all the particles, M is the population size, and N is the dimensionality of the problem.

When the optimization begins, the length of potential well $L(t)$ tends to decrease gradually with particles moving to the global optimal position. A larger $\beta(t)$ means a lower aggregation degree and thus leads to a faster moving speed of particles, larger $\alpha(t)$, and $L(t)$. Otherwise, $\alpha(t)$ and $L(t)$ will be smaller.

The fitness increment of Equation (6) is $dF(t)$ and represents the change of fitness value for two successive generations, given by Equation (13):

$$dF(t) = \frac{F(t) - F(t+1)}{F_{\max} - F_{\min}}, dF \in [-1, 1] \quad (13)$$

where $F_{\max}$ and $F_{\min}$ are the maximum and minimum fitness values during iterations, respectively.

Different stages of population evolution are represented as T_{FQPSO} , given by Equation (14):

$$T_{\text{FQPSO}} = \frac{t}{t_{\max}}, T_{\text{FQPSO}} \in (0, 1] \quad (14)$$

where t and $t_{\max}$ are the current iteration number and maximum iteration number, respectively.

With iterations moving on, the diversity of the population inevitably degenerates in later stages of evolution, which makes particle swarm converge prematurely. In order to solve this problem, we propose a stochastic mutation strategy for particle swarm. This algorithm makes particles have the ability to deviate from original position and find a better solution in high-dimensional space.

In a searching space with a dimension of N , the mutation threshold of the algorithm is ρ_0 , while the mutation rate on the components of particle positions is $\rho = [\rho_1, \rho_2, \dots, \rho_N] \in [0, 1]$, $m = 1, 2, \dots, N$. When mutation is executed, the new position of the j -th component in the i -th particle $X'_{i,j}(t)$ is given by Equation (15).

$$X'_{i,j}(t) = \begin{cases} X'_{i,j}(t), & \rho_j \leq \rho_0 \\ X_{i,j}(t), & \rho_j > \rho_0 \end{cases} \quad (15)$$

The fuzzy logic controller is designed as follows. The inputs of the controller include the population diversity $\beta(t)$, fitness increment $dF(t)$, and evolution state T_{FQPSO} . The outputs of the controller are the incremental value of the mutation factor $d\rho$ and CE co-efficient $d\alpha$.

$$\begin{cases} \alpha(t+1) = \alpha(t) + d\alpha \\ \rho(t+1) = \rho(t) + d\rho \end{cases} \quad (16)$$

According to the range of fuzzy variables and corresponding linguistic variables sets in Tables 2 and 3, the membership function of each variable is shown in Figure 3. In a current iteration, the input of fuzzy logic controller $\beta(t)$, T_{FQPSO} , $dF(t)$ are calculated according to Equations (12)–(14). The outputs $d\rho$ and $d\alpha$ are obtained by Figure 3 and updated by Equation (16). They are used to update the particle positions in Equation (11) for the next iteration in FQPSO.

Table 2. Fuzzy rules of $d\rho$ ($d\alpha$) For a list of definitions of abbreviations, see Table 3.

T_{FQPSO}	$dF(t)$	$\beta(t)$				
		EB	B	M	S	ZE
S	For all $dF(t)$	ZE(ZE)	PS (ZE)	PB(PS)	PB(PB)	PB(PB)
M	NB	PS(ZE)	PB(NS)	PB(NS)	PB(NB)	PB(NB)
	NS	ZE(ZE)	PS(NS)	PS(NS)	PB(NS)	PB(NB)
	NES	ZE(NS)	ZE(NS)	ZE(NS)	ZE(NS)	PB(NB)
	PES	ZE(ZE)	ZE(PS)	ZE(PS)	PS(PS)	PB(PB)
	PS	NS(ZE)	NS(ZE)	ZE(PS)	PS(PS)	PB(PS)
	PB	NB(ZE)	NS(ZE)	NS(ZE)	ZE(ZE)	PB(ZE)
B	NB	none	none	NS(NB)	ZE(NB)	PB(NB)
	NS	none	none	NS(NS)	ZE(NS)	PS(NS)
	NES	none	none	NS(NS)	NS(NS)	ZE(NS)
	PES	none	none	NS(PS)	NS(PS)	ZE(PS)
	PS	none	none	NB(ZE)	NS(NS)	ZE(NS)
	PB	none	none	NB(ZE)	NB(NS)	ZE(NS)

Table 3. Fuzzy linguistic variables set.

Symbol	Linguistic Variables	Symbol	Linguistic Variables
S	Small	NS	Negative small
M	Medium	ZE	Zero
B	Big	PS	Positive small
NES	Negative extreme small	PB	Positive big
NB	Negative big	EB	Extreme big
PES	Positive extreme small

Figure 3. Cont.

Figure 3. The membership function for the (a) population diversity, (b) mean fitness, (c) population evolution state, (d) increment of mutation crossover factor, (e) increment of CE co-efficient.

5. Incremental Scheduler Based on SMT Solver

After FQPSO, frame instances are assigned to basic periods and meet the requirement of the objective function and constraints. However, since the collision-free constraints are not considered in the above process, frame instances will overlap each other in the table. The purpose of this section is to adjust the offset of frames to satisfy collision-free and temporal constraints in every basic period based on an improved SMT solver.

The FQPSO part greatly reduces the scale of collision-free constraints, which account for the majority of total constraints for the SMT solver. Delay and precedence can be constrained by Equations (4) and (5). The collision-free constraint is formally defined as shown in Equation (17):

$$\forall l_k \in L, \forall T_{bp,k}^j \in l_k, \forall f_v, f_u \in T_{bp,k}^j$$

$$s(v, T_{bp,k}^j) \geq s(v, T_{bp,k}^j) + p_u \vee s(u, T_{bp,k}^j) \geq s(v, T_{bp,k}^j) + p_v \quad (17)$$

where L is the set of dataflow links, the j -th basic period on dataflow link l_k is denoted by $T_{bp,k}^j$, f_v and f_u are frame instances in $T_{bp,k}^j$, $s(v, T_{bp,k}^j)$ and $s(u, T_{bp,k}^j)$ are their sending offset, and p_v and p_u are their link occupancy times.

We propose an improved SMT solver to solve constraints (4), (5) and (17). This article defines the priority to decide the sequence of incremental frames scheduling. Not only the temporal urgency of frames, i.e., the end-to-end deadline in Figure 2, but also the interference of scheduled messages with unscheduled ones needs to be considered. The period utilization (PU) is defined as the bandwidth needed to transmit a frame f_u on the premise that FQPSO has already allocated more urgent frame instances to basic periods, as given by Equation (18).

$$PU_u = \sum_{k=1}^L \sum_{j=1}^{T_{Mp}/T_{bp,k}} \left(\frac{\sum_{f_v \in T_{bp,k}^j \setminus \{f_u\}} p_v}{T_{bp,k}^j} + \frac{p_u}{T_u} \right) \quad (18)$$

- $\left(\sum_{f_v \in T_{bp,k}^j \setminus \{f_u\}} p_v \right) / T_{bp,k}^j$ means the bandwidth occupied by other frames except f_u in the j -th basic period of dataflow link l_k .
- p_u / T_u means the f_u transmission requires a fraction of time equal to p_u every T_u time units.

Thus, the priority of each frame f_u is set as a vector of two components $\text{priority} = \text{rank}(\min(\text{eu}), \max(\text{PU}_u))$, where eu is the end-to-end deadline of f_u and is compared firstly by the most critical parameter. PU_u is compared secondly for frames that have the same end-to-end deadline. This priority value defines the difficulty of frame instance scheduling. The frame instance with less free scheduling space should be scheduled earlier. Figure 4 shows the incremental SMT scheduler based on the schedulability ranking in our approach.

Prior_Rank is the set of priority queues. Ready_to_Sch is the set of frames that are ready to schedule. Sch_table is the completed schedule table. UnSch is the set of unscheduling frames. Lines 5–9 are schedulability ranking approach, and these are used to compute the priority of frames and sort them to form the Prior_Rank queue. The lower priority is at the tail and the higher is at the head. In each iteration, the frame instance with the highest priority in the queue is added to Ready_to_Sch and scheduled by the SMT solver first.

Line 15 uses the frames in Ready_to_Sch , basic periods scheduling results from FQPSO, and offset of scheduled frames in Sch_table to generate SMT constraints and check whether the logic context is satisfiable (line 16). If so, the SMT solver gives a yices_model that satisfies the logic context (line 17) and returns the feasible solution into local array sch_offset from SMT pointer & smt_offset (line 18). The function $\text{place}()$ in line 19 is used to explicitly add these time offset of frame instances to Sch_table . Then, the Prior_Rank is updated by deleting the successfully scheduled frame f_i from the head of the queue.

If the current frame f_i is not schedulable, the last successfully scheduled frame f_j and its offsets are taken out of Sch_table and pushed back to Ready_to_Sch (lines 22–23). In the next iteration, f_i and f_j are scheduled together by lines 15–20 to find a feasible solution. If there is still no feasible solution when

all the scheduled frames are popped out from Sch_table for backtracking, the frame with minimum priority f_{un} is identified as a problematic frame and put into UnSch to prevent a scheduling crash (lines 25–31).

1 function Schedulability ranking scheduler			
2	Prior_Rank = $\emptyset$, Ready_to_Sch = $\emptyset$, Sch_Table = $\emptyset$, UnSch = $\emptyset$	23	Ready_to_Sch.push(f_j);
3	yices_context ctx = yices_mk_context();	24	endif
4	yices_model model;	25	if Sch_Table = $\emptyset$ do
5	while $F \neq \emptyset$ do	26	$f_{un} = \arg \min(\text{priority})$;
6	$f_i = \arg \min(\text{priority})$;	27	Ready_to_Sch.pop(f_{un});
7	$F.\text{pop}(f_i)$	28	UnSch.push(f_{un})
8	Prior_Rank.push(f_i)	29	endif
9	endwhile	30	endif
10	while Prior_Rank = $\emptyset$ do	31	endwhile
11	if Ready_to_Sch = $\emptyset$ do	32	if UnSch $\neq \emptyset$
12	Prior_Rank.pop(f_i);	33	make_constraints(Sch_Table, Figure 2, UnSch);
13	Ready_to_Sch.push(f_i);	34	if yices_check(ctx) is TRUE do
14	else do	35	model = yices_get_model();
15	make_constraints(Sch_Table, FQPSO.output, Ready_to_Sch);	36	yices_get_real_value(model, sch_offset, &smt_offset);
16	if yices_check(ctx) is TRUE do	37	place(Sch_Table, sch_offset[i]);
17	model = yices_get_model();	38	return Sch_Table;
18	yices_get_real_value(model, sch_offset, &smt_offset);	39	else do
19	place(Sch_Table, sch_offset[i]);	40	return Sch_Table, UnSch;
20	Prior_Rank.pop(f_i);	41	endif
21	else do	42	endif
22	Sch_Table.pop(f_i);		

Figure 4. The algorithm of an incremental Satisfiability Modulo Theories (SMT) scheduler.

After all the frames in Prior_Rank are scheduled, the frames in UnSch are added to the SMT solver together and they check whether there is a feasible solution for them without considering the basic period limits from FQPSO. If so, the final schedule table Sch_table is returned. Otherwise, the Sch_table with currently scheduled frames and UnSch with unscheduled ones are returned.

After the two-level schedule is calculated, if there are a few messages newly adding into the system, the incremental SMT scheduler will be called to determine whether the idle time of the schedule table can meet the constraints of frames. When there is enough appropriate idle time, the scheduler will be updated with new frames online in several to hundreds of milliseconds. Otherwise, the schedule table needs to be recalculated off-line.

6. Experiment

6.1. Experiment Testbench

The experiment testbench is on a local machine with 8 GB memory and Ubuntu 16.10 operating system. We use an Intel(R) core (TM) i5-4590 3.3 GHz processor of Intel Corporation, America. The FQPSO algorithm is implemented by C++ language and the SMT solver is YICES 2.0.

The experimental process on the platform is shown in Figure 5.

1. Define message tasks and their attributes in TCN. In a realistic Ethernet TCN, the load is usually less than 50% of the total bandwidth. Under this, it is difficult to evaluate the performance of our approach. So, we use random traffic to simulate the network load.
2. According to TCN topology, determine the route and dataflow path of messages.
3. According to message attributes, delay parameters, and temporal constraints, complete the directed graph as Figure 2 and objective function.
4. Use FQPSO to optimize the fitness of the objective function and assign frame instances to basic periods. If successful, continue scheduling. Otherwise, return to Step (2) to adjust the route and dataflow path.
5. Call the incremental SMT scheduler to adjust the offset of frame instances in basic periods. If successful, output the time-triggered schedule table. Otherwise, return to Step (2) to adjust the route and dataflow path.
6. When newly added messages are added to TCN, the incremental SMT is called for online scheduling. If successful, the schedule table is updated. Otherwise, return to Step (2) to recalculate the schedule table offline.

Figure 5. Ethernet train communication network topology.

6.1.1. Topology

In the experimental section, we use an improved TRDP-based TCN system in a CR400 high-speed train of China as the realistic scheduling test platform, as shown in Figure 6. Each carriage is equipped with a consist network node (CNN), which adopts an industrial Ethernet switch and loads a time-triggered schedule table in it. Each CNN connects eight EDs and uses a ring topology to provide redundancy within the network. The train backbone node (TBN) uses a linear structure to connect two subnets. The topology contains 10 switches, 64 end devices, 75 physical links, and 150 directed dataflow links. The solid lines represent the Ethernet physical links, and the dotted lines represent the separation between two cars.

Figure 6. Ethernet train communication network topology.

6.1.2. Message Assignments

Message assignment is done by generating dataflow paths each consisting of one sender and one or a set of receivers. In our testbench, a sender is randomly selected from all the EDs, and every message of the sender belongs to either to a multicast group or a unicast group. A sender of multicast is allowed to transmit frames to a subset of a configurable group size of receivers. The unicast group is allowed only one receiver binding to a sender for the message. In this article, four types of time-triggered traffic are defined: (1) unicast within subnet: the source unicasts messages to one sink belong to the same TBN; (2) multicast within subnet: the source multicasts messages to 8–10 random sinks under the same TBN; (3) unicast between subnet: the source unicasts messages to one sink belonging to different TBNs; (4) multicast between subnet: the source multicasts messages to 8–10 random sinks under different TBNs.

We generate eight datasets for the experiment. Each ED can send any of the four types of time-triggered traffic. Quads of traffic (n_1 , n_2 , n_3 , and n_4) in Table 4 shows the number of messages of unicast within subnet, multicast within subnet, unicast between subnet, and multicast between subnet, relatively. Considering that the real-time data frames in an industry are generally short, the frame length in our testbench is a random integer between 64 to 500 bytes. All dataflow links have the duplex transmission speed of 100 Mbps. The period of the message is $2^x 3^y w$ with $x, y \in \{0, 1, 2\}$ and $w \in \{2, 3\}$, randomly. In Figure 2, the release time is zero, T_{LD} is a random integer between 100 to 400 μs , and the end-to-end delay is set as $e_i = 0.5T_i$.

Table 4. Message distribution of four kinds of time-triggered traffic.

Dataset	Number of Messages for Every ED	Message Composition	Maximum Link Load Utilization Rate/%
1	2	(1,1,0,0)	14.72
2	4	(1,1,1,1)	24.16
3	6	(2,2,1,1)	34.17
4	8	(2,2,2,2)	42.84
5	10	(3,3,2,2)	57.58
6	12	(3,3,3,3)	64.30
7	14	(4,4,3,3)	72.58
8	16	(4,4,4,4)	84.89

6.2. Evaluation of FQPSO

In this article, FQPSO is compared with PSO and QPSO for convergence speed and optimal solution. The PSO algorithm is set as follows: the two acceleration factors are 1.49, the inertia weight $\omega = 0.8$, and the range of $V_{\max}$ is 10% larger than that of the particles. The contraction–expansion coefficients of the QPSO algorithm are linearly decreasing from 1.7 to 0.5 with iterations. The local and global learning factors c_1 and c_2 in Equation (11) are 2 and 2.1.

The parameters of FQPSO are set as follows: the fuzzy membership function is shown in Table 2, and the initial mutation threshold of algorithm is $\rho_0 = 0$. The settings of FQPSO's fuzzy logic function are shown in Section 4.2. For all the problem instances, the particle swarm size is 50. We choose dataset 1, 4, 7 to represent light, medium, and massive load. As for the objective function in Equation (6), we supposed that $a_1 = a_2 = 0.5$.

When the three constraints in Formula (6) are satisfied and there is no better fitness value for 1000 iterations, the computation of FQPSO stops. The constraints in Equation (6) make it so that the incremental SMT scheduler of level 2 can make further frame granularity scheduling based on FQPSO. When there is no better fitness value for 1000 iterations, we get stable and good enough optimization results under an acceptable computation time.

Figure 7 shows the convergence of PSO, QPSO, and FQPSO algorithms for the objective function under different load utilization of dataflow links. Table 5 lists the experiment results of three algorithms for the datasets, and “No solution” indicates that no feasible solution that satisfies Equation (12) can be found within the iterations. Under light load, the PSO algorithm obtains the optimal value in the fewest 38 steps of iteration. However, the minimum fitness value of FQPSO and QPSO is obviously smaller than that of PSO. Due to the low particle dimension, FQPSO has no superiority over QPSO in this case. However, with the increase of load, PSO cannot get the optimal solution under medium and massive load, and the final optimal value of FQPSO is obviously lower than that of QPSO. The results prove that in a high-dimensional space, PSO and QPSO may enter a fully convergent state at the early stage of iterations, while FQPSO outperforms them in overcoming premature convergence and gets a better optimal solution with fewer iterations.

Figure 7. Convergence results of particle swarm optimization (PSO), quantum-behaved particle swarm optimization (QPSO), and fuzzy-controlled quantum-behaved particle swarm optimization (FQPSO) algorithms in (a) light load, (b) medium load and (c) massive load.

Table 5. The calculated results of minimal fitness by algorithms.

Performance	Algorithm	Light	Medium	Massive
Optimal Fitness	PSO	0.5634	No solution	No solution
	QPSO, α = linear	0.1659	0.236	0.323
	FQPSO	0.1403	0.066	0.225
Iteration times	PSO	38	No solution	No solution
	QPSO, α = linear	1194	2288	405
	FQPSO	1540	2147	1530

6.3. Evaluation of Incremental Scheduler

In our experiment, 100 problem instances are randomly generated for each dataset in Table 4. Each problem instance is optimized by FQPSO first. Then, an incremental scheduler for the frame instances is implemented based on the SMT in each basic period, and the average computation time is calculated. In order to evaluate the performance of each incremental scheduler under different priority ranking strategies, random ranking, period ascending, and proposed deadline-PU based priority are selected in this article. Table 6 shows the comparison of computation time. The upper limit of computation time for the SMT set in this article is 3600 s. If the result is not obtained within time limit, it will return “No solution” as unsolved.

Table 6. Computation time with three ranking methods (in seconds).

Dataset	1	2	3	4	5	6	7	8
Random	2	10	32	584	2256	No solution	No solution	No solution
Period ascending	1	4	21	29	67	328	1241	2874
Proposed deadline-PU	1	5	11	24	46	75	202	413

The constraints given by Steiner in [28] are too many and complicated for the case of non-strictly periodic scheduling. It is very easy for the computation time to exceed the upper time limit and fail to obtain a feasible schedule. In this article, each frame instance is fixed in its own basic period by FQPSO, which can greatly reduce the searching space and number of constraints.

The experiment results in Table 6 show that the computation time of random frame ranking is significantly higher than that of the other two ranking methods due to no consideration of the schedulability of frames. For period ascending and deadline-PU priority ranking, when the problem scale is not large, the incremental scheduler performance is close. With the increase of time-triggered traffic in the network, the incremental scheduler based on the deadline-PU priority ranking has an obviously lower computation time compared with period ascending ranking, because the schedulability of a time-triggered frame depends not only on the period of the message, but also on the frame length, basic period length, and parameters of the other messages scheduled in the same basic period. These factors are comprehensively considered in the priority method proposed in this article and result in a more reasonable scheduling queue of frames.

6.4. Performance Results

After completing the FQPSO optimization and incremental scheduler, we perform scheduling performance analysis on all successful instances in the datasets. The incremental scheduler in [28] is used as a contrastive algorithm. Since the strictly periodic scheduling is used in these references, the collision-free constraint in [28] is changed as shown in Equation (19) to meet the requirements of

non-strict periodic scheduling in this article. Delay and precedence can be constrained by Equations (4) and (5). The other assignments and constraints are the same as those in Section 6.1.

$$\forall l_k \in L, \forall f_i, f_u \in l_k, \begin{cases} s(i, j, l_k) \geq s(u, v, l_k) + p_u \vee s(u, v, l_k) \geq s(i, j, l_k) + p_i \\ (j-1)T_i \leq s(i, j, l_k) - s(i, 1, l_k) \leq jT_i \\ (v-1)T_u \leq s(u, v, l_k) - s(u, 1, l_k) \leq vT_u \end{cases} \quad (19)$$

where L is the set of dataflow links, f_i and f_u are frames on l_k , $s(i, j, l_k)$ and $s(u, v, l_k)$ are the sending offsets of their j -th and v -th instances, $s(i, 1, l_k)$ and $s(u, 1, l_k)$ are the sending offsets of the first instance of frame f_i and f_u on l_k , and p_i and p_u are the link occupancy time of frames.

In two cases, the proposed approach may fail to provide solutions. Firstly, in level 1 of the proposed approach, FQPSO cannot get solutions satisfying the constraints in Equation (6). Secondly, in level 2, although the result of level 1 is satisfied, the collision-free problem cannot be solved, or the effective solution cannot be provided within the time limit by the incremental SMT scheduler due to the heavy network load.

Figure 8a shows the schedulability in the percentage of each dataset within the time limit of the standard incremental SMT [28] and FQPSO-SMT in this article. With non-strictly periodic frames, rapidly increasing the quantity of constraints makes it more difficult for the standard incremental SMT approach to find a feasible solution. The schedulability of our approach is always higher than that of standard incremental SMT and is always 100% when the dataflow link utilization is lower than 25%. No solution is obtained in the standard incremental SMT of datasets 7 and 8, while our approach achieves the schedulability of 39% and 26% within the time limit.

Figure 8. (a) Schedulability of SMT and FQPSO-SMT algorithms. (b) Computation time of SMT and FQPSO-SMT algorithms.

Figure 8b shows the average computation time of standard incremental SMT and FQPSO-SMT in this article, and our approach always obtains a significantly lower result in all the datasets. On one hand, our FQPSO approach can optimize the QoS parameters of scheduling, such as macro period load balance, jitter, etc., at the cost of increasing computation time. On the other hand, our approach decreases the number of constraints by solving the problem in certain basic periods instead of the whole macro period, thus reducing the complexity and computation time. As a result, the standard incremental SMT computation time is slightly lower than that of FQPSO-SMT in the light load scenario of dataset 1 and is around 1.7 to 4.7 times that of datasets 2 to 6 in our testbench. The average computation time of our approach for dataset 7 and 8 is 1264 and 2060 s, while the results of incremental SMT are outside of the time limits.

For the link with the largest traffic load in every problem instance, we define the load balance as the standard deviation for basic period utilization as shown in Equation (3). The max frame jitter is divided by its message period as $Jitter_i/T_i$. The experimental results are the average value of all the successful instances in each dataset. The statistics are shown in Table 7. If the result is not obtained within time limit of 3600 s, it will return “No solution” as unsolved.

Table 7. Load balance and jitter results.

Dataset	Load Balance (μ s)		Max Frame Jitter	
	SMT	FQPSO-SMT	SMT	FQPSO-SMT
1	83.64	28.52	0.126	0
2	88.55	34.63	0.314	0.067
3	102.72	37.26	0.540	0.114
4	106.46	44.19	0.652	0.175
5	110.45	50.23	0.963	0.248
6	123.07	56.66	0.998	0.327
7	No solution	73.11	No solution	0.402
8	No solution	81.86	No solution	0.467

There are few research studies on jitter and load balance optimization of a basic period model for non-strictly periodic scheduling in switched Ethernet. We can only use the results of standard incremental SMT as a contrast and compare them with the results of proposed FQPSO-SMT approach in this article.

In Table 7, the load balance of FQPSO-SMT in all the test datasets is less than that of the contrast. The value obtained by FQPSO-SMT in dataset 8, which has the heaviest traffic, is 61.86 and less than the 63.64 obtained by incremental SMT in dataset 1, which has the smallest load. It proves that the FQPSO-SMT can make the schedule of time-triggered frames more balanced in the communication model based on a basic period structure. This balance optimization can improve the overall bandwidth utilization of the network. Furthermore, it provides more flexible time slots for the incremental scheduling of newly added network traffic.

The jitter in Table 7 deteriorates rapidly with the increase of traffic load without optimization. The optimized jitter of FQPSO-SMT is better than that of the contrast in all the datasets. When using the FQPSO-SMT algorithm, the increasing speed of jitter is significantly slower than that of the contrast algorithm, and the maximum frame jitter is less than half of its period. Therefore, our approach is more suitable for non-strictly periodic scheduling while the standard incremental SMT scheduler is only suitable for strictly periodic frames. In a train communication scenario, the time-triggered service generally does not exceed 50% of the total bandwidth, and the remaining bandwidth can be used by event-triggered and best effort traffic. The frame jitter at this point is less than 25% of its period as shown in dataset 5, which can meet the jitter requirements of train real-time services.

7. Conclusions

This article introduces a scheduling approach to design and optimize a time-triggered schedule table for non-periodic frames and tasks with hard real-time requirements that are executed on Ethernet in CPS. Our approach includes two levels: the optimization part in the first level and the incremental scheduler part in the second level. We present a co-scheduling model of load balance and frame jitter for the first level. Furthermore, an improved fuzzy-controlled QPSO method is proposed to realize the dynamic adaptive adjustment of CE coefficient and mutation rate in FQPSO and improve the algorithm performance when searching in a high-dimensional space. Based on these properties, an improved SMT incremental scheduler with a schedulability ranking method is proposed, which can fast solve the collision-free and temporal constraints of frames in basic periods. We evaluate the efficiency and performance of the proposed approach on the testbench of a train Ethernet network system with different traffic loads. The experiment demonstrates the effectiveness and scalability of our approach.

Our approach is an enhancement and improvement of a previous optimization algorithm and SMT solver scheduler. It is an exploration of NP-complete complexity mitigation for multitask scheduling optimization. This approach still has a few disadvantages. When the online incremental SMT scheduling of newly added traffic cannot succeed, the entire schedule table needs to be recalculated offline. A hierarchical submodel may be a solution and will be our future work. Another future work is the extension of the proposed approach to systems with mixed hardware properties, such as for example the Internet of Things, networks with different wire speeds, or wireless communication.

Author Contributions: Conceptualization, methodology, and writing—original draft preparation, J.J.; formal analysis, software, and programming, J.J. and H.C.; supervision, project administration, and writing—review and editing, L.W. and X.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Fundamental Research Funds for the Central Universities of China under Grant 2018YJS149, and the Natural Science Foundation of Beijing Municipality under Grant L171009.

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

References

1. Rajkumar, R.; Lee, I.; Sha, L.; Stankovic, J. Cyber-physical systems: The next computing revolution. In Proceedings of the Design Automation Conference, Anaheim, CA, USA, 13–18 June 2010; pp. 731–736.
2. Yu, X.F.; Tomovic, K. Application of linear matrix inequalities for load frequency control with communication delays. *IEEE Trans. Power Syst.* **2004**, *19*, 1508–1515. [\[CrossRef\]](#)
3. Bradley, J.M.; Atkins, E.M. Computational-Physical State Co-regulation in Cyber-Physical Systems. In Proceedings of the 2011 IEEE/ACM Second International Conference on Cyber-Physical Systems, Chicago, IL, USA, 12–14 April 2011; pp. 119–128.
4. International Electrotechnical Commission. *Industrial Communication Networks—Profiles—Part 1: Fieldbus Profiles*. International Electrotechnical Commission 61784-1; International Electrotechnical Commission: Geneva, Switzerland, 2014.
5. International Electrotechnical Commission. *Industrial Communication Networks—Fieldbus Specifications—Part 2: Physical Layer specification and Service Definition*, International Electrotechnical Commission 61158-2; International Electrotechnical Commission: Geneva, Switzerland, 2014.
6. Beji, S.; Hamadou, S.; Gherbi, A.; Mullins, J. SMT-Based Cost Optimization Approach for the Integration of Avionic Functions in IMA and TTEthernet Architectures. In Proceedings of the 2014 IEEE/ACM 18th International Symposium on Distributed Simulation and Real Time Applications, Toulouse, France, 1–3 October 2014; pp. 165–174.
7. Xue, Y.K.; Li, J.; Nazarian, S. Fundamental Challenges toward Making the IoT a Reachable Reality: A Model-Centric Investigation. *ACM Trans. Des. Autom. Electron. Syst. (Todaes)* **2017**, *22*, 1–25. [\[CrossRef\]](#)
8. Derler, P.; Lee, E.A.; Vincentelli, A.S. Modeling Cyber-Physical Systems. *Proc. IEEE* **2012**, *100*, 13–28. [\[CrossRef\]](#)
9. Poovendran, R. Cyber-Physical Systems: Close Encounters between Two Parallel Worlds [Point of View]. *Proc. IEEE* **2010**, *98*, 1363–1366. [\[CrossRef\]](#)
10. Karsai, G.; Sztipanovits, K.; Ledecz, A.; Bapty, T. Model-integrated development of embedded software. *Proc. IEEE* **2003**, *91*, 145–164. [\[CrossRef\]](#)
11. Bogdan, P. A Cyber-Physical Systems Approach to Personalized Medicine: Challenges and Opportunities for NoC-based Multicore Platforms. In Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 9–13 March 2015; pp. 253–258.
12. Alonso, J.M.; Ribas, J.; del Coz, J.J. Intelligent control system for fluorescent lighting based on LonWorks technology. In Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society (Cat. No.98CH36200), Aachen, Germany, 31 August–4 September 1998; Volume 1, pp. 92–97.
13. Liang, G.; Wang, H. Communication performance analysis and comparison of two patterns for data exchange between nodes in World FIP fieldbus network. *ISA Trans.* **2010**, *49*, 567–576. [\[CrossRef\]](#)
14. Ferreira, J.; Almeida, L.; Fonseca, A.; Pedreiras, P.; Martins, E.; Rodriguez-Navas, G.; Rigo, J.; Proenza, J. Combining operational flexibility and dependability in FTT-CAN. *IEEE Trans. Ind. Inform.* **2006**, *2*, 95–102. [\[CrossRef\]](#)

15. Pop, T.; Pop, P.; Eles, P.; Peng, Z.; Andrei, A. Timing analysis of the FlexRay communication protocol. In Proceedings of the 18th Euromicro Conference on Real-Time Systems (ECRTS'06), Dresden, Germany, 5–7 July 2006.
16. Skeie, T.; Johannessen, S.; Brunner, C. Ethernet in substation automation. *IEEE Control Syst. Mag.* **2002**, *22*, 43–51.
17. Skeie, T.; Johannessen, S.; Holmeide, O. Timeliness of real-time IP communication in switched industrial Ethernet networks. *IEEE Trans. Ind. Inform.* **2006**, *2*, 25–39. [[CrossRef](#)]
18. *IEEE Standard for Ethernet, IEEE 802.3*; IEEE Standards Association: Piscataway, NJ, USA, 2015.
19. Decotignie, J.D. Ethernet-Based Real-Time and Industrial Communications. *Proc. IEEE* **2005**, *93*, 1102–1117. [[CrossRef](#)]
20. Bai, T.; Zhi-Ming, W.; Gen-Ke, Y. Optimal bandwidth scheduling of networked control systems (NCSs) in accordance with jitter. *J. Zhejiang Univ.—Sci. A: Appl. Phys. Eng.* **2005**, *6*, 535–542.
21. Yan, X.; Li, H.-B.; Wang, L.; Shen, P. Optimal scheduling algorithm for reducing jitter in networked control systems based on estimation of distribution algorithms. *J. Southeast Univ. Natrual Sci. Ed.* **2013**, *43*, 38–43.
22. Barreiros, J.; Costa, E.; Fonseca, J.; Coutinho, F. Jitter reduction in a real-time message transmission system using genetic algorithms. In Proceedings of the 2000 Congress on Evolutionary Computation, La Jolla, CA, USA, 16–19 July 2000; Volume 2, pp. 1095–1101.
23. Natale, M.D.; Stankovic, J.A. Scheduling distributed real-time tasks with minimum jitter. *IEEE Trans. Comput.* **2000**, *49*, 303–316. [[CrossRef](#)]
24. Xu, X.F.; Cao, C.; Guo, J.; Liu, Z. TT-RMS: Communication table generation algorithm of time-triggered network. *J. Beijing Univ. Aeronaut. Astronaut.* **2015**, *41*, 1403–1408.
25. Zhang, C.; Nan, J.G.; Chu, W.K. An Improved Communication Table Generation Algorithm based on Time-Triggered Rate Monotonic Scheduling. *J. Air Force Eng. Univ.: Nat. Sci. Ed.* **2016**, *17*, 82–87.
26. Tamas-Selicean, D.; Pop, P.; Steiner, W. Synthesis of communication schedules for TTEthernet-based mixed-criticality systems. In Proceedings of the 8th IEEE/ACM/IFIP International Conference on Hardware/software Codesign and System Synthesis, Tampere, Finland, 7–12 October 2012; pp. 473–482.
27. Tamas-Selicean, D.; Pop, P.; Steiner, W. Design optimization of TTEthernet-based distributed real-time systems. *Real-Time Syst.* **2015**, *51*, 1–35. [[CrossRef](#)]
28. Mok, A.; Rosier, L.; Tulchinsky, I.; Varvel, D. The pinwheel: A real-time scheduling problem. In Proceedings of the Twenty-Second Annual Hawaii International Conference on System Sciences, Kailua-Kona, HI, USA, 3–6 January 1989; Volume II: Software Track, pp. 693–702.
29. Hanzalek, Z.; Burget, P.; Sucha, P. Profinet IO IRT Message Scheduling with Temporal Constraints. *IEEE Trans. Ind. Inform.* **2010**, *6*, 369–380. [[CrossRef](#)]
30. Steiner, W. An Evaluation of SMT-Based Schedule Synthesis for Time-Triggered Multi-hop Networks. In Proceedings of the 31st IEEE Real-Time Systems Symposium, San Diego, CA, USA, 30 November–3 December 2010; pp. 375–384.
31. Pozo, F.; Rodriguez-Navas, G.; Hansson, H.; Steiner, W. SMT-based synthesis of TTEthernet schedules: A performance study. In Proceedings of the 10th IEEE International Symposium on Industrial Embedded Systems (SIES), Siegen, Germany, 8–10 June 2015; pp. 1–4.
32. Huang, J.; Blech, J.O.; Raabe, A.; Buckl, C.; Knoll, A. Static scheduling of a Time-Triggered Network-on-Chip based on SMT solving. In Proceedings of the 2012 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 12–16 March 2012; pp. 509–514.
33. Craciunas, S.S.; Oliver, R.S. Combined task and network-level scheduling for distributed time-triggered systems. *Real-Time Syst.* **2015**, *52*, 161–200. [[CrossRef](#)]
34. Minaeva, A.; Akesson, B.; Hanzálek, Z.; Dasari, D. Time-Triggered Co-Scheduling of Computation and Communication with Jitter Requirements. *IEEE Trans. Comput.* **2018**, *67*, 115–129. [[CrossRef](#)]
35. Wang, N.; Yu, Q.; Wan, H.; Song, X.; Zhao, X. Adaptive Scheduling for Multicenter Time-Triggered Train Communication Networks. *IEEE Trans. Ind. Inform.* **2019**, *15*, 1120–1130. [[CrossRef](#)]
36. Syed, A.; Fohler, G. Efficient offline scheduling of task-sets with complex constraints on large distributed time-triggered systems. *Real-Time Syst.* **2019**, *55*, 209–247. [[CrossRef](#)]
37. Sun, J.; Fang, W.; Wu, X.; Palade, V.; Xu, W. Quantum-Behaved Particle Swarm Optimization: Analysis of Individual Particle Behavior and Parameter Selection. *Evol. Comput.* **2012**, *20*, 349–393. [[CrossRef](#)] [[PubMed](#)]

38. Huang, W.; Xu, X.; Pan, X. Control strategy of contraction-expansion coefficient in quantum-behaved particle swarm optimization. *Appl. Res. Comput.* **2013**, *30*, 1967–1970.
39. Liu, P.; Leng, W.; Fang, W. Training ANFIS Model with an Improved Quantum-Behaved Particle Swarm Optimization Algorithm. *Math. Probl. Eng.* **2013**, *2013*, 595639. [[CrossRef](#)]
40. Yadmellat, P.; Salehizadeh, S.M.A.; Menhaj, M.B. A New Fuzzy Inertia Weight Particle Swarm Optimization. In Proceedings of the 2009 International Conference on Computational Intelligence and Natural Computing, Wuhan, China, 6–7 June 2009; pp. 507–510.
41. Wang, Y.; Wang, L.; Yan, X.; Shen, P. Fuzzy immune particle swarm optimization algorithm and its application in scheduling of MVB periodic information. *J. Intell. Fuzzy Syst.* **2017**, *32*, 3797–3807. [[CrossRef](#)]
42. Xu, B.; Zhang, X.; Liu, L. The Failure Detection Method of WSN Based on PCA-BDA and Fuzzy Neural Network. *Wirel. Pers. Commun.* **2018**, *102*, 1657–1667. [[CrossRef](#)]
43. Roman, R.C.; Precup, R.E.; Petriu, E.M. Model-free adaptive control with fuzzy component for tower crane systems. In Proceedings of the 2019 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Bari, Italy, 6–9 October 2019; pp. 1–6.
44. Roman, R.C.; Precup, R.E.; Bojan-Dragos, C.A. Combined Model-Free Adaptive Control with Fuzzy Component by Virtual Reference Feedback Tuning for Tower Crane Systems. *Procedia Comput. Sci.* **2019**, *162*, 267–274. [[CrossRef](#)]
45. International Electrotechnical Commission. *Electronic Railway Equipment—Train Communication Network (TCN)—Part 3-4: Ethernet Consist Network (ECN)*, International Electrotechnical Commission 61375-3-4; International Electrotechnical Commission: Geneva, Switzerland, 2014.
46. International Electrotechnical Commission. *Electronic Railway Equipment—Train Communication Network (TCN)—Part 2-3: TCN Communication Profile*, International Electrotechnical Commission 61375-2-3; International Electrotechnical Commission: Geneva, Switzerland, 2015.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).