An On-Line Scheduler over Hard Real-Time Communication System

CHEN Hui    XIONG Guangze
(School of Computer Science and Engineering, UESTC  Chengdu  610054  China)

Abstract  By thorough research on the prominent periodic and aperiodic scheduling algorithms, an on-line hard real-time scheduler is presented, which is applicable to the scheduling of packets over a link. This scheduler, based on both Rate Monotonic, pinwheel scheduling algorithm Sr and Polling Server scheduling algorithms, can rapidly judge the schedulability and then automatically generate a bus table for the scheduling algorithm to schedule the packets as the periodic packets. The implementation of the scheduler is simple and easy to use, and it is effective for the utilization of bus link. The orderly execution of the bus table can not only guarantee the performance of the hard real time but also avoid the blockage and interruption of the message transmission. So the scheduler perfectly meets the demand of hard real-time communication system on the field bus domain.

Key words  hard real-time communication; advanced real-time communication scheduler(ARTCS); on-line scheduler;  bus scheduling table;  message

In terms of the requirement of time strictness, real-time systems are classified to soft real-time and hard real-time. Soft real-time systems can tolerate some amount of violation of time constraints. In hard real-time systems, a missed deadline may lead to catastrophic consequences so that hard real-time messages must be properly scheduled to meet their time constraints. For a real-time communication, the link serves as a central resource contested by packets (messages), while the central processor is contended for by tasks in the real-time multitasking system, therefore most real-time scheduling methods are applicable to the scheduling of packets over a link. Based on their arrival patterns over time, real-time messages are commonly divided into two categories: periodic and aperiodic. Periodic messages arrive regularly, and are often used for transmitting sensory data or for intertask communication between periodic tasks. Aperiodic messages arrive randomly or even are executed once only. They are often used to carry alert information or for communication between aperiodic tasks.

In the industrial process control area, field busses are expected to exhibit reliable hard real-time behavior, since they convey the information to and from the terminals of a time-critical system. The information is often composed of a lot of periodic and some aperiodic messages. Scheduling the messages transmission to satisfy the time constraints of every message is extremely important in the field bus systems. But in real-time communication system, scheduling messages for transmission is different from scheduling tasks in real-time multitasking system, because a message transmission can’t be preempted and then resumed without penalty. If a packet transmission is interrupted, it has to be retransmitted all over again[1].

1  Periodic Message Scheduling Algorithms

Periodic messages scheduling algorithms primarily include the rate monotonic (RM) and the earliest deadline first (EDF) algorithms that are used in period task (PT) systems, Pinwheel algorithms that are used in distance constrained task systems (DCTS).
1.1 The Periodic Task Model Scheduling Algorithms

In the periodic task model, system consists of a set of $n$ periodic tasks $T = \{r_1, r_2, \ldots, r_n\}$, where $T_i$ is the period (deadline equal to the task period) of the task $r_i$, and $C_i$ is the worst-case execution time of $r_i$. An active task with a higher priority may preempt an executing task. The utilization of $S$ is defined as

$$U = \sum_{i=1}^{m} (C_i / T_i)$$

Using rate monotonic scheduling algorithm\[2\], tasks with shorter periods are assigned with higher priority. At run time, the scheduler chooses among the active tasks which have been requested but not yet finished, and the one with highest priority is selected to execute. A task set is schedulable under RM if

$$\rho(S) = \sum_{i=1}^{m} C_i / D_i \leq 1$$

Due to its low computational overhead and extremely simple implementation, RM is widely regarded as an appropriate algorithm for scheduling real-time tasks on uniprocessor systems.

EDF is a dynamic priority-driven preemptive scheduling algorithm\[2\]. The task priorities are not fixed but change depending on the closeness of their absolute period (deadline), tasks with earlier deadlines get higher priorities. If the utilization of the task set is no greater than 1, the task set can be scheduled by the EDF. Therefore, the EDF is an optimal uniprocessor scheduling algorithm, since no task set is schedulable if the processor utilization is larger than 1. Though EDF can get higher utilization than static priority scheduling algorithms, it has higher overhead and is more difficult to implement since it requires updating the priority of every task at each scheduling arbitration.

No matter RM or EDF, it is assumed that all tasks can be preempted at any point of their execution. Therefore, the messages may miss their deadlines in the real-time communication systems, because the executing transmission may be preempted by an active task with higher priority so as to be aborted. One way of ensuring exclusive access is to hold the share critical section with semaphores. But Ref.[3] shows that the problem of deciding whether it is possible to schedule a set of periodic tasks is NP-hard when the periodic tasks use semaphores to enforce mutual exclusion. Ref.[4] have proposed the priority ceiling protocol (PCP) to solve the trouble and minimize the effect of priority inversion. The PCP considers the duration in which a higher priority task is blocked by the tasks with lower priority. The RM-schedulability condition under PCP is presented in Ref. [4].

1.2 Distance Constrained Task Model Scheduling Algorithm

In the distance constrained task model, the temporal distance between the finish times of any two consecutive executions of a task should not be longer than a constant. Such real-time systems are called distance constrained task systems (DCTS)\[5,6\]. Formally, a DCTS consists of a set of $n$ tasks $T = \{r_1, r_2, \ldots, r_n\}$, where $D_i$ is the distance constraint of the task $r_i$, and $C_i$ is the worst-case execution time of $r_i$. The total density of task set $S$ is defined as

$$\rho(S) = \sum_{i=1}^{m} C_i / D_i$$

Pinwheel algorithm Sr use a number specialization technique to transform the which satisfies distance constrains into harmonized numbers\[5\]. This transformation method is described as follows: Given any $r$, which satisfies $D_1 / 2 < r \leq D_i$, let $b_1 = r$, $b_i = r(2^j - 1) = 2b_{i-1}$, and $S$ is accordingly transformed to $S'$. If the total density utilization of the DCTS task set $S'$ is no greater than 1, the task set can be scheduled by the Sr. Sr works as follows:

1) Set $l_i = D_i / 2^{\log(D_i / n) + 1}$, get the set $L = \{l_1, l_2, \ldots, l_n\}$, (note that $D_i / 2 < l_i \leq D_i$)
2) Sort $L$’s and remove duplicates, then get the set $K = \{k_1, k_2, \ldots, k_u\}$, where $k_1 < k_2 < \cdots < k_u$ and $u \leq n$.
3) Let $r = k_u$ (1 $\leq u \leq n$), respectively, compute all $\rho(S')_{k_0}$, then select $\min(\rho(S')_{k_0})$.
4) If the min $\rho(S')_{k_0}$ $\leq 1$ (task set $S$ is specialized with $r = k_0$) and transform into $S'$, schedule all tasks using $b_i$’s as the periods; otherwise, it is not schedulable.

Scheduler Sr uses an approach similar to the RM to schedule the transformed tasks set, where it regards distance constrains as periods. So, in the following of
this paper, we won’t strictly distinguish between the term ‘distance constrain’ and ‘period’. Since the periods are harmonized, using the Sr can meet the original distance constrain and eliminate jitter.

**Example 1** Given 5 tasks $S = \{\tau_1, \tau_2, \tau_3, \tau_4, \tau_5\}$ in a real-time system, the distance constraints are $D = \{9.7, 10.5, 10.7, 21.0, 25.6\}$ and the execution times are $C = \{1.5, 2.0, 3.1, 1.4, 3.0\}$ respectively. The Sr works as follows:

1) Let $l = D_i / 2^{\lfloor \log_2 (\rho_i / \tau_i) \rfloor}$, get the set $L = \{l_1, l_2, l_3, l_4, l_5\} = \{9.70, 5.25, 5.35, 5.25, 6.40\}$

2) Remove duplicates from $L$’s, then the set $K = \{k_1, k_2, k_3, k_4\} = \{5.25, 5.35, 6.40, 9.70\}$

3) Let $r = k_i$ $(1 \leq i \leq 4)$ respectively, we have \[\rho(S')_1, \rho(S')_2, \rho(S')_3, \rho(S')_4\} = \{0.98, 1.15, 1.35, 0.91\}

4) According to step 3, it is known that the task set are schedulable using the Sr when $r = 5.25$ or $r = 9.70$. Since the total density of transformed task set $S'$ is minimum, $r = k_4 = 9.7$ is selected.

5) Let $r = 9.7$, specialize $D$ with $\{r\}$, get the new distance constraints $D' = \{9.7, 9.7, 9.7, 19.4, 19.4\}$. Fig. 1 shows the result of schedule, $n$ denotes null.

![Fig.1 Sr schedule for example 1](image)

Fig.1 shows that every task of the task set $S$ is scheduled circularly as 19.4, the maximal period of the $S'$. All tasks execution have fixed interval. The execution intervals of $\tau_1$, $\tau_2$, $\tau_3$, $\tau_4$ are themselves transformed distance constraints respectively, whereas the interval of task $\tau_5$ is 7.9 and 11.5 since preemption causes that it can’t be finished during its first execution.

If the distance constraints $D$ are regarded as the periods in the RM, Theorem 1 is proved in Ref.[5].

**Theorem 1** If a task set $S$ is schedulable by Sr, $S$ is also schedulable by RM.

## 2 Aperiodic Message Scheduling Algorithms

The hard real-time system maintains a periodic server for each source of aperiodic messages, and aperiodic messages are treated as periodic messages. Each server has a predefined period and a execution time, therefore it is scheduled together with periodic messages to ensure that it get relevant time during its every period. In terms of the policy of the aperiodic messages usable time, hard real-time aperiodic message scheduling algorithms are mainly classified to polling server (PS), priority exchange (PE) and deferrable server (DS)[17].

Using the PS scheduling algorithm, at the time that the periodic sever is scheduled, if the aperiodic message is ready it execute immediately; otherwise the server execution time is vacancy or is occupied by periodic messages. Even though the aperiodic message arrives but the server has 9 suspended during the execution time, it must wait till the next period of this sever. This shows that the PS doesn’t preserved resource for aperiodic messages execution.

However, the PE and DS scheduling algorithms belong to the resource preserving algorithms. They employ mechanisms to preserve the runtime allocated for the aperiodic message. The PE and DS algorithms differ in the manner in which they preserve their execution time. The PS preserves its execution time by exchanging it with a lower priority periodic message till the aperiodic message is ready or all messages execution has completed. But the DS maintains its aperiodic execution time only for the duration of the server’s period.

## 3 On-Line Hard Real-Time Scheduler ARTCS

For traditional network, maximizing the throughput or minimizing the average message delay is the most important performance criteria. In the hard real-time domain, however, concern focuses on satisfying the time constraints of individual message[8]. The on-line hard real-time scheduler ARTCS presents an approach to absolutely guarantee deadlines of periodic and aperiodic messages in centralized-scheduling hard real-time communication systems on field bus domain. In the centralized-scheduling hard real-time communication system, the access to the bus is controlled by a node on which the bus scheduling
table runs. The bus scheduling table deals with the message as the basic scheduling unit. Each scheduling instruction in the bus table indicates the execution information of every message including the source and destination address, the beginning time and the maximal delay of the message execution, and the other control information. By orderly executing the instructions of this bus table, not only the sending time of a node’s data, but also the channel time used by a node will be controlled. Consequently the hard real-time property of communication is guaranteed.

Using traditional scheduling algorithms directly in a real-time communication system, the transmitting message may be interrupted and has to retransmit all over again, which leads to waste of the bandwidth and more swapping overhead. The PCP avoids messages transmission interruption, while a higher-priority message might be blocked until a lower-priority message completion. Sometimes blocking of the higher-priority message even once may be too long to meet its deadline. On the other hand, when the PCP is applied to give the schedulable analysis for a set of many messages, calculating the blocked times of every messages is complex and not well suitable in the practical industrial computing system. Nevertheless, the Sr can reduce the overhead of judging schedulability since it transforms the distance constraint among the messages of the set into harmonics. Therefore, the ARTCS uses Sr to judge the schedulability of the message set. In the process of ARTCS, it applies the non-blockage and non-interruption approach to adapt the network communication. The ARTCS, based on RM, Sr and PS scheduling algorithms, includes three stages: building message set, judging schedulability and generating bus table. In the following section, the three stages are presented in detail.

3.1 Collecting Information and Building Message Set

ARTCS collects all information relating to message transmission when the network system is initialized, then it creates corresponding original message set. The message set model is consistent with the practical application demand, i.e., either periodic or distance constrained message model. Let $M=\{(C_i,D_i)\}_{1\leq i\leq n}$ be the message set, where $C_i$ is the worse-case execution time and $D_i$ is the deadline (or distance constraint) of $M_i$. Meantime the aperiodic messages are translated into periodic messages by taking the minimal arrival intervals as their periods.

3.2 Judging Schedulability

As we pointed out above, it is difficult to calculate blocked time of every message in practice when the PCP is used. In order to reduce the computational complexity, the ARTCS judges the schedulability of the message set based on the Sr. For the periodic message model, the ARTCS regards the deadline $D_i$ as the distance constraint, and converts the periodic message model into the distance constrained message model. The judging schedulability stage of ARTCS includes two steps: base judgment and emulation judgment.

3.2.1 ARTCS Schedulability Base Judgment

It is supposed that message execution can be preempted during the course of ARTCS schedulability base judgment. According to Sr scheduling algorithm, base judgment works as follows. Let $r = D_1$ or $r = D_n/2^{\log_2(D_n/D_1)}$ respectively ($D_m$ is the maximal distance constraint among the messages of the set $M$), specialize the set $D = \{D_1, D_2, \ldots, D_n\}$ with respect to $\{r\}$, then $D$ is transformed into $D' = \{b_1, b_2, \ldots, b_n\}$. Let $\rho(\tau) = \sum_{i=1}^n C_i/b_i$, if $\rho(\tau) \leq 1$, the set $M$ will suffice the request of base judgment and is processed into next step, emulation judgment.

During this step, in order to decrease calculation overhead, $r$ isn’t presumed all distance constraints of the set $M$ but $D_1$ and $D_n$ typically. $D_m$ is easy to bring smaller total density and provides the bus utilization $1/2 + C_m/D_m$ at least. On the other hand, $D_1$ is the easiest to satisfy the following emulation judgment.

3.2.2 ARTCS Schedulability Emulation Judgment

ARTCS schedulability emulation judgment means to judge the set $M$ schedulability on condition that there is no interruption and blockage in the message execution, which guarantees that a high-priority message transmission never waits for a low-priority message. By using Sr scheduling algorithm, specialize...
D with respect to r. D is transformed into \( D' = \{ b_1, b_2, \cdots, b_n \} \) and have the property that \( b_j \mid b_i \ (j \geq i) \). Therefore, for the transformed set \( M' \), the least common multiple (LCM) of all distance constraints is equal to \( b_n \), so we only consider the schedulability over the duration of period \( b_n \). The method of the emulation judgment is as follows: firstly, divide the \( b_n \) into \( k \) segments, each of them is \( b_1 \); secondly, schedule the highest priority message that is ready among the message set, and the highest priority message will be scheduled only if the message execution can be removed to next segment; thirdly, whenever all messages can be scheduled and satisfy their distance constraints during the period \( b_n \), the set \( M \) can be scheduled successfully on condition that message execution isn’t interrupted and blocked. This process called emulation judgment is showed as follow:

\[
M' = \{ M'_i \mid 1 \leq i \leq n \}
\]

is a set of messages that will be judged, with \( D' = \{ b_1, b_2, \cdots, b_n \} \), \( b_j \mid b_i \ (j \geq i) \), and \( \rho(M') = \sum_{i=1}^{n} C_i / b_i \leq 1. \)

/*k: the current segment for emulation judgment, the length is \( b_1 \)*/
/*getrefer(k): get the reference segment number corresponding to the \( k \)th segment*/
/*R: residual usable time of the current segment*/
/*next: to indicate the first message number that has never been scheduled*/
/*Isint (logk): to determine whether \( \log k \) is an integer, i.e., whether \( k \) is the integer power of 2*/
/*Schedule (P): to determine whether all messages of the \( p \)th kind period have already been scheduled entirely*/
1) next ← 1;
2) for (int k:=1; k:=k+1)
3) do {
4) int j:=getrefer(k);
5) float PT ← 0;
6) for i:=1 to j do {PT:=PT+pti, p:=1;}
7) if (PT=Tnext) then {
8) R:= b1+d;
9) while (R \geq Cnext) {
10) R:= R−Cnext;
11) if (next=n) [return success] /* emulation judgment is success*/
12) next:=next+1;
13) }
14) }
15) Record (k) /*register the time allocation of \( k \)th segment*/
16) if (Isint (logk)) then {
17) P:= log2k+1;
18) if (! Schedule (P)) {return failure} /* emulation judgment is failure*/
19) }
20) }

In above processes, according to distance constraints, the messages is sorted into \( P=\text{ceil}(\log(k)/\log(2)) + 1 \) kinds, and each message execution interval is equal to its period. If the \( p \)th kind period length \( b_p \) (except \( b_1 \)) is divided into tow parts averagely, for all messages whose periods are lower than \( b_p \), scheduling them in each segment (segment length \( b_1 \)) of the later part is the same as the previous part entirely. Moreover, when the system allocates time for a certain segment (such as the \( k \)th segment) in the later part, it will firstly schedule the messages whose periods are lower than \( b_p \) consistent with the corresponding reference segment (the \( j \)th segment is the reference segment of the \( k \)th segment) in the previous part. If the messages whose periods are lower than \( b_p \) have been scheduled wholly from the 1st to the \( j \)th segments, given \( d \) is the total time that is allocated to all messages whose periods are lower than \( b_p \) in the reference segment, let residual time \( R \) be equal to \( b_1 \) minus \( d \), then system decides the allocation of residual time according to whether \( R \) is bigger than the execution time \( C \) next of the first message that has never been scheduled. This process protects message execution from being interrupted and preempted. Line 5~7 represent the judgment whether the messages from 1st to \( j \)th segments whose periods are lower than \( b_p \) are scheduled wholly; line 8~14 show how to allocate the residual time of the \( k \)th segment. Whenever each kind period is end, the system will check whether the messages belong to this period have been scheduled wholly. If it is true, the process will turn to the next segment, otherwise the schedulability emulation judgment is failure (line 16~19). If the
system has scheduled all messages within $b_n$, the schedulability emulation judgment is proved successful (line 11).

**Example 2** Let’s consider the example 1 and take emulation judgment for it, the result is presented in Fig.2. It indicates the judgment is success and the execution of task $\tau_i$ is not interrupted.

![Fig.2 The Example of emulation judgment](image)

As we known, when the schedulable sufficient condition of RM is used to judge the schedulability, the utilization of messages must be no greater than $n(2^{1/n}−1)$. Though using the schedulable necessary and sufficient condition of RM can get higher utilization,[2] it isn’t a proper on-line approach since its excessive computational overhead. The Sr scheduling algorithm used by ARTCS can get the utilization at least $n(2^{1/n}−1)^3$. Thus it can be seen that the schedulable analysis approach of ARTCS is able to not only enhance the bandwidth utilization but also reduce the overhead.

On the other hand, after each message of the set $M$ is specialized and transformed into harmonized number, its transformed distance constraint must less than or equal to original distance constraint. According to theorem 1, we can get the theorem 2 as followings. It shows that a message set can be scheduled successfully by the original message model scheduling algorithm on condition of no interruption and blockage, if the message set has passed the ARTCS schedulability judgment.

**Theorem 2** A task set that has passed the ARTCS emulation judgment must be schedulable by the RM on condition of no interruption and blockage.

The proof is omitted here, it is important to say that if a message set $M$ is failure under the basic or emulation judgment, some measures must be employed such as dividing network into multi-segments to reduce the traffic.

### 3.3 According to the Scheduling Algorithm, Generating Bus Scheduling Table

If the message set $M$ has passed emulation judgment, ARTCS will schedule $M$ and generate bus scheduling table according to original message model scheduling algorithm. That is to say, for a periodic model message set, it will be scheduled by the RM; for a distance constrained model message set, it will be scheduled by the Sr. In the following, we will introduce the process of generating bus table, and assume periodic model message set to demonstrate. ARTCS uses the RM other than EDF to schedule the message set in order to reduce overhead. At the same time, the RM still employs the non-interruption and non-blockage approach to achieve the demand of real-time communication. ARTCS on-line schedule process to periodic message set is presented as follow:

```c
/* $d_i$: the slack time of the message $M_i$ */
/* $c_i$: the needed execution time of message $M_i$ in the duration $d_i$ */
/* $M_1$={($C_i,D_i$)} $1\leq i\leq n$. $D_1\leq D_2\leq \cdots \leq D_{n-1}\leq D$ is the set of messages */
1) for $i=1$ to $n$ do { $c_i:=C_i$; $d_i:=D_i$; } 2) do { 3) $i--$; 4) while ($i\leq n$ and $c_i=0$) { $i:=i+1$ } 5) if $i=n$ then { 6) $r:=\min \{d_1,d_2,\cdots,d_{i-1}\}$ 7) if ($c_i\leq r$) { $t:=c_i$; $c_i:=0$; } /* allocate time $t$ to $M_i$ */ 8) else { $r:=r$; } /* allocate time $t$ to vacancy */ 9) else { $r:=\min \{d_1,d_2,\cdots,d_n\}$; $t:=r$; } /* allocate time $t$ to vacancy */ 10) for $j=1$ to $n$ do { 11) $d_j:=d_j-t$; 12) if ($d_j=0$) { $c_j:=C_j$; $d_j:=D_j$; } 13)} 14) } forever
```

In above processes, the ARTCS sets two variable $c_i$ and $d_i$ to each message $M_i$, the highest priority message $m_i$ in $M$ that waits for scheduled can be got by determining the value of $c_i$. In order to ensure there is no interruption and blockage in message execution, set variable $r$ denotes the shortest residual time before the next period arriving among the messages whose priorities are higher than $m_i$. If $r$ is equal to or bigger than $c_i$, time $t=c_i$ will be allocated to the message $M_i$ and $c_i$ is set 0 (line 7); if $r$ is smaller than $c_i$, this implies it will bring on preemption and interruption if
the message $M_i$ is scheduled, time $t = r$ will be allocated to vacancy (line 8). If all messages execution time are 0, this implies that all messages of set $M$ have been scheduled in their current periods and their next periods don’t arrive, the shortest time before the next period arriving among these message of the set $M$ will be allocated to vacancy (line 9). Whenever time $t$ is allocated to a certain message or vacancy, all slack times $d(s)$ should reduce $t$. If the slack time of a certain message becomes 0, this implies that the next period of the message arrives, the system will set $c_j = C_j, d_j = D_j$ (from line 10~12).

With running the above process forever, the bus scheduling base table is generated. Since the resource preserving algorithms may bring on preemption and interruption, for the aperiodic messages, they are solved by the PS algorithm and converted into periodic messages. Owing to the burst characteristic of the aperiodic messages, they can occupy the time including their execution time and vacancy time. The node on which the bus table runs will judge whether the usable time can meet the demand of a certain message to determine schedulability of the message. On the other hand, if the practical communication system has non-real-time traffic, it can use the idle time that aperiodic messages remain for supporting the traffic. The node on which the bus table runs will adjust the bus scheduling base table according to the above strategy and obtain the formal bus scheduling table. By executing command lines of the bus table orderly, it will provide the hard real-time, non-interruption and non-blockage communication performance.

4 Conclusions

An on-line hard scheduler ARTCS is proposed for the hard real-time communication system on the field bus domain. The scheduler includes three stages: building the message set according to the practical application, judging the schedulability of the message set, and generating the bus scheduling table. By using the improved pinwheel scheduling algorithm to judge the schedulability, the ARTCS is able to not only enhance the bandwidth utilization but also reduce the overhead, and guarantee the hard real-time transmission of messages without being interrupted and blocked. The ARTCS can be used for both the periodic message model and the distance constrained message model to schedule the periodic and aperiodic message transmissions. By executing command lines of the bus table orderly, the hard real-time transmission is guaranteed in a centralized-scheduling network.

According to our experiment, it is proved that the scheduler is easy to be implemented and can gracefully suffices the practical requirement of hard real-time communication system. Based on this result, in further work to be done is how to on-line optimize the bus table for improving its performance.

References


Brief Introduction to Author(s)

CHEN Hui 陈慧 was born in 1973. She is now pursuing the Ph.D. degree in UESTC. Her research interests include: real-time communication technology and application. E-mail: chenhui@uestc.edu.cn

XIONG Guangze 熊光泽 was born in 1938. He is now a Professor in UESTC. His research interests include: real-time computing system and its application.