Maintaining temporal validity of real-time data on non-continuously executing resources

Tian Bai[1], Hong Lu[2] and Juan Yang[2]

[1]*Hunan Institute of Science and Technology, College of Computer Science, 414000, Yueyang, China*
[2]*Wuhan Maritime Communication Research Institute, 430079, Wuhan, China*

Abstract. This paper studies the problem of temporal validity maintenance on non-continuously-executing resources. Response time bounds for sensor transaction scheduling are derived in the context of the hybrid extended multiprocessor periodic resource model. Then two deadline and period assignment schemes are proposed to maintain the temporal validity of real-time data. The calculation based scheme (DPA-C) uses the response time bounds to compute deadlines and periods. The check based scheme (DPA-A) assigns deadlines and periods directly. It then checks the feasibility of the assignment based on the response time bounds. Experiments are conducted to evaluate the performance of the proposed schemes. The results show that DPA-C performs better than DPA-A in terms of the scheduling success ratio and the imposed update workload.

1 Introduction

Cyber-physical systems (CPS) are widely used in many application areas to monitor the environmental status and make response to critical events. Typical application areas include flight control, health monitoring and industrial process control. In CPS, the real-time data objects are defined to model the current status of external entities. Each data object is associated with a validity interval which specify the lifetime of the object's values. A real-time data object is temporally valid if its current value's validity interval doesn't expire [1]. It's very important to guarantee the real-time data validity, since otherwise CPS would make wrong control decisions based on the invalid data objects.

In recent years, there has been a number of works on temporal validity maintenance. The More-Less (ML) scheme uses sensor transactions to update the values of real-time data objects periodically [1, 2]. The deferrable scheduling algorithm for fixed priority transactions (DS-FP) adopts the sporadic task model [3, 4]. It defers the release times of transaction instances as much as possible by exploiting the semantics of validity constraints. The earliest deadline first (EDF) scheme was adopted in [5, 6] for scheduling sensor transactions. Han et al. proposed two schemes to maintain validity constraints during mode switches [7]. The problem of co-scheduling sensor transactions and application transactions was studied in [8]. Li et al. studied the temporal validity maintenance problem on partitioned multiprocessors [9].

Current studies for maintaining the temporal validity of real-time data assume the processing resource is continuous available for transaction execution. This assumption, however, is not true in many cases. For example, a complex real-time system usually consists of several subsystems. Some subsystems may run on one physical platform. Each of them can only get a fraction of the resource. As another example, in order to achieve thermal resiliency, real-time systems will change between different power modes dynamically. Running in a low power mode can be viewed as running on a platform with discontinuous resource supply.

In this paper, we study how to maintain temporal validity of real-time data on platforms where the resource supply is not continuous. We derives the response time bounds of sensor transactions under global fixed priority scheduling scheme in the context of the hybrid extended multiprocessor periodic resource model. Based on the response time bounds, we then present two schemes to assign relative deadlines and periods to transactions so that the real-time data validity is properly maintained. The calculation based scheme (DPA-C) uses the bounds to compute deadlines and periods. The check based scheme (DPA-A) makes the assignment directly. It then checks whether the transaction set is feasible under the assignment. Experiments are conducted to evaluate the performance of these two schemes. The results show that compared with DPA-A, DPA-C can achieve higher scheduling success ratio while impose lower update workload.

The rest of the paper is organized as follows. Section 2 presents real-time data model and resource model. Section 3 presents the response time bounds of sensor transactions and two schemes for deadline and period assignment. The experimental results are given in section 4 and in section 5 we give the conclusion.

2 System model

Consider a set of real-time data objects $X = \{X_i\}_{i=1}^{n}$ and a set of sensor transaction $\Gamma = \{\tau_i\}_{i=1}^{n}$. The validity interval of data object X_i is V_i. Sensor transaction τ_i update the value of X_i. The worst-case execution time of τ_i is C_i.

Definition 1 X_i is temporally valid at time *t*, if the sampling time of X_i's current value $ts(X_i)$ plus V_i is not less than *t*, that is $ts(X_i)+V_i \geq t$ [1].

The transaction set is executed on a non-continuously-executing resource. In this paper we introduce the hybrid extended multiprocessor periodic resource model (HEMPR). The model is characterized by Ω= (Π, Δ, Θ, m_1, m). It specifies that for a resource that consists of m processors, m_1 processors of it collectively provide Θ unit of resource in every period Π within the deadline Δ (Δ ≤ Π). The rest of the processors each can guarantee full resource supply. The HEMPR model can be viewed as a combination of the MPR model [10] and the dedicated recourse model (DR). If we set $m_1 = m$ and Δ = Π, then we get MPR. If m_1 is set to be zero, then we get DR. Thus our model is more flexible than MPR and DR. Notice that the processors in HEMPR can be either physical or virtual processors. The virtual processors themselves should be allocated to proper physical processors.

There are two approaches to schedule the sensor transactions on multiprocessors: partitioned approach and global approach. In the partitioned approach, a transaction will be allocated to a processor and can be executed only on this processor. In the global approach, transactions are allowed to migrate between different processors during their execution. In this paper the global fixed priority scheduling approach (GFP) is considered. We assume the transactions are sorted according to non-decreasing order of their validity intervals. Higher priorities are assigned to transactions with lower validity intervals. A transaction instance is allowed to be executed on one processor at any time.

3 Temporal validity maintenance on non-continuously executing resources

3.1 Response time bounds

The worst-case resource supply pattern for the HEMPR model can be obtained by making some modifications to the reasoning for MPR model in [10]. In our pattern, the resources provided by m_1 processors in the first period are allocated from time zero. For subsequent periods, the resources are allocated as late as possible without violating the deadline constraint. Then the supply bound function of HEMPR is given by:

$$sbf(t) = \begin{cases} N\Theta + (m-m_1)t + \beta(L) & t \geq a \\ (m-m_1)t & otherwise \end{cases} \quad (1)$$

In (1), $a = \Pi - \lceil \Theta / m_1 \rceil$, $N = \lfloor (t-a)/\Pi \rfloor$, $L = \min\{\Delta,\ t-a-N\Pi\}$. $\beta(L)$ is calculated as follows:

$$\beta(L) = \begin{cases} 0 & L \leq \Delta - \Pi + a \\ \max\{0, \Theta - (\Delta - L + 1)m_1\} + b & otherwise \end{cases} \quad (2)$$

In (2), $b = \Theta - \lfloor \Theta / m_1 \rfloor$.

In order to derive the upper bound of a sensor transaction's response time, we extend Nan's idea of busy period extension that is designed for dedicated resources [11].Consider a transaction instance $\tau_{k,l}$ with release time $r_{k,l}$ and finish time $f_{k,l}$. Let t_o denote the earliest time instant before $r_{k,l}$ such that all resource supply of HEMPR in time interval $[t_o, r_{k,l})$ are consumed by higher priority instances. There are at most $m_1 - 1$ sensor transactions with carry-in instance. Let *t* denote the length of the interval, then the upper bound of total workload of higher priority transactions in the interval is:

$$W_k(t) = \sum_{i=1}^{n} I_k^{NC}(\tau_i, t) + \sum_{m_1-1 \text{ largest}} (I_k^{CI}(\tau_i, t) - I_k^{NC}(\tau_i, t)) \quad (3)$$

The calculation of $I_k^{CI}(\tau_i, t)$ and $I_k^{NC}(\tau_i, t)$ is given by equations (5)-(8) in [11] and is omitted here for brevity.

The next problem is to obtain an upper bound for interference that $\tau_{k,l}$ suffers in $[t_o, f_{k,l})$. Let $I_k(t)$ denote the bound. For a dedicated resource, it is $\lfloor W_k(t)/m \rfloor$. But for HEMPR, it's not the case since the resource supply may not be continuous. In order to obtain $I_k(t)$, we compute the parallel supply functions for HEMPR and then use these functions to calculate the bound. A level-*i* supply function $Y_i(t)$ is the minimum resource supply in every time interval of length *t* with parallelism at most *i* [12]. Based on the worst-case resource supply pattern, we then have:

$$Y_i(t) = \begin{cases} t \cdot i & i \leq m - m_1 \\ sbf(t,i) & otherwise \end{cases} \quad (4)$$

In (4), $sbf(t,i)$ is obtained by replacing *m* in $sbf(t)$ with *i* and Θ with $\Theta - (m-i)(\Pi - a - 1) - \max\{b-i, 0\}$. Let $D(t,j)$ denote the duration over an interval of length *t* during which the time is provided by *j* processors in parallel, then :

$$D(t,j) = \begin{cases} Y_{m-m_1+b+1}(t) - Y_{m-m_1+b}(t) & j = m \\ Y_{m-m_1+1}(t) - Y_{m-m_1}(t) - D(t,m) & j = m - m_1 + b \\ t - D(t,m) - D(t, m-m_1+b) & j = m - m_1 \\ 0 & otherwise \end{cases} \quad (5)$$

Then according to [12], $I_k(t)$ is given by:

$$I_k(t) = D(t, m-m_1) I_{\{m\}}(m_1) + \sum_{i=1}^{|S|} \min\{D(t, s(i)), \left\lfloor \max\{(W_k(t) - \sum_{j=1}^{i-1} D(t, s(j)) s(j)), 0\} / s(i) \right\rfloor\} \quad (6)$$

In (6), $I_{\{m\}}(m_1)$ is an indicator function to take the case $m = m_1$ into consideration. $S = \{m-m_1, m-m_1+b, m\} / \{0\}$. $s(i)$ is *i*-th element of *S*.

Let R_k denote the minimal value that satisfies the equation below:

$$I_k(t) + C_k = t \quad (7)$$

Then $R_k - r_{k,l} + t_o$ must be an upper bound of the response time of τ_k for this particular t_o. The reason is that $I_k(t)$ is an upper bound of interference on $\tau_{k,l}$ in $[t_o, f_{k,l})$, and there are no unused resource units in $[t_o, r_{k,l})$, so it's not possible to obtain a larger response time than $R_k - r_{k,l} + t_o$. Since $r_{k,l}$ is not earlier than t_o, we can see that R_k is a response time bound of τ_k for arbitrary t_o.

3.2 Deadline and period assignment schemes

In order to maintain the temporal validity constraints, the sum of the relative deadline D_k and the period T_k of τ_k must be no larger than the validity interval V_k. In addition, the transaction set must be schedulable by GFP on HEMPR. In this section two schemes are proposed for period and deadline assignment.

Calculation based scheme (DPA-C) uses the response time bounds derived in subsection 3.1 to calculate deadlines and periods directly. This calculation is carried out from τ_1 to τ_n. For τ_k, if R_k is larger than $V_k/2$, Γ is considered as unschedulable. Since $I_k(t)$ and $W_k(t)$ are both non-deceasing functions of t, R_k can be obtained in an iterative way. That is:

$$R_k^{i+1} = I_k(R_k^i) + C_k \tag{8}$$

In (8), R_k^i denotes the value of R_k in i-th iteration. R_k^{-1} is set to zero. DPA-C is presented in Algorithm 1. Notice that when m_1=m, R_i^0 is set to $\Delta - \Pi + 2a + C_i$ which is the length of the interval without resource supply. For τ_k, the time to compute $W_k(t)$ is $O(n)$, then $I_k(t)$ can be calculated in constant time based on $W_k(t)$ since there are at most three different parallel levels to be considered. There are at most $V_k/2 - C_k$ iterations to get R_k. Let $V_{\max} = \max\{V_i \mid 1 \le i \le n\}$, then the time complexity of DPA-C is $O(n^2 V_{\max})$.

Algorithm 1 Calculation based Assignment
Input: Γ and Ω
Output: deadlines and periods of Γ if Γ is schedulable, otherwise report failure

```
for i = 1 to n
  k=0;
  if m_1 < m
    R_i^0 = C_i ;
  else
    R_i^0 = Δ − Π + 2a + C_i ;
  end
  while true
    if R_i^k > V_i / 2
      Γ is unschedulable, return failure;
    else if R_i^k = R_i^{k−1}
      break;
    end
    compute R_i^{k+1} using (8);
    k = k + 1;
  end
  D_i = R_i^k ;
  T_i = V_i − D_i ;
end
```

The iterative computation of DPA-C could be time-consuming for transactions with large validity intervals. Different from DPA-C, the check based scheme (DPA-A) adopts a very simple method to assign deadlines and periods. For τ_k, the scheme sets $D_k = \lfloor V_k/(F+1) \rfloor$ and $T_k = V_k - D_k$. F is the scaling factor of DPA-A. Its value should be not less than 1 and is chosen by system designers.

A schedulability condition for τ_k is obtained from (7):

$$I_k(D_k) + C_k \le D_k \tag{9}$$

Here $W_k(D_k)$ is computed using only $I_k^{CI}(\tau_i, D_k)$ to guarantee a safe bound since the iteration is not used. For each transaction, DPA-A checks whether (9) is satisfied. If not, the transaction set is considered as infeasible. Otherwise DPA-A continues to test subsequent transactions. For each transaction, it's not hard to see that the time to check (9) is $O(n)$. So the time complexity of DPA-A is $O(n^2)$. It is lower than that of DPA-C. However, as one can see from the experimental results, this reduction comes with the cost of lower scheduling success ratio and higher imposed workload.

4 Performance evaluation

In this section the results of the experimental evaluation on DPA-C and DPA-A are presented. There are two performance metrics. The first metric is the scheduling success ratio. It's defined as the ratio of the schedulable transaction sets under the particular scheme to the total number of transaction sets. The second metric is the update workload imposed by sensor transactions defined as $\sum_{1 \le i \le n} u_i / ((\Theta + (m - m_1)\Pi)/(m\Pi))$. This definition takes the discontinuity of the resource supply into account.

The parameters and their default settings are given in table 1. The resource deadline of HEMPR is set to be the same as the resource period. The ratio between Θ and $m_1\Pi$ is chosen randomly in [0.7, 1]. The ratio between m_1 and m is chosen randomly in [0.5, 1]. The value of Θ and m1 can then be obtained based on the ratios.

Table 1. Parameters and settings.

Parameters	Meaning	Value
V_i	Validity interval	[100,300]
C_i	Computation time	[5,20]
N_p	Number of processors	[2,8]
Π	Resource period	200

Fig.1 shows the scheduling success ratio of DPA-C and DPA-A when the density of the transaction set is varied from 1.5 to 2.5. The density is defined as the sum of the ratio of a transaction's computation time and its validity interval. In the experiment, N_p is set to 4 and the number of dedicated processors is set to be half of N_p. The ratio between Θ and $m_1\Pi$ is fixed at 0.8. The scaling factor for DPA-A is set to 2. It can be seen that the scheduling success ratio of DPA-C is constantly no lower than that of DPA-A. For example, when the density is about 1.9, almost all transaction sets can't be accepted by DPA-A while DPA-C still can accept any transaction set. The reason is DPA-A uses looser bounds for feasibility test which will lead to higher reject rate of transactions.

Fig.2 shows the update workload generated from two

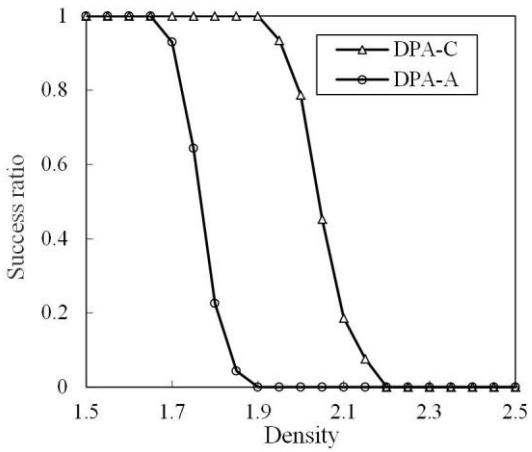

Figure 1. Scheduling success ratio comparison.

schemes. The parameter settings are the same as those described for fig.1. It can be seen that the update workload produced by DPA-C is lower than that of DPA-A. Both schemes' update workload becomes higher when the density increases. For temporal validity maintenance schemes, the update workload is an import metric. Reducing the workload of sensor transactions will leave more resources to other types of transactions, thus the overall system performance is improved.

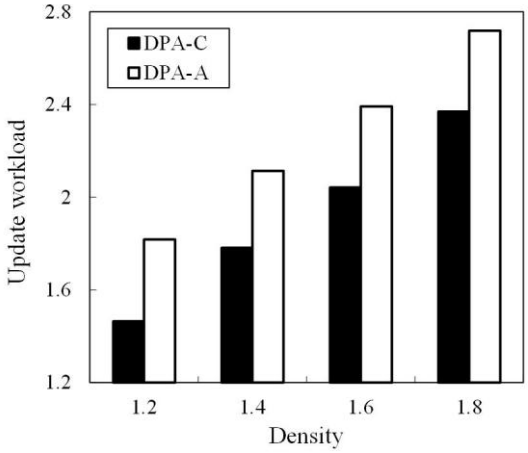

Figure 2. Update workload comparison.

Fig.3 shows the scheduling success ratio of two schemes under different ratios between Θ and $m_1\Pi$, while the density is fixed at 1.75. Other parameter settings remain unchanged. It is observed that the success ratio of DPA-C is constantly higher than that of DPA-A. In addition, both schemes' success ratio will increase when the recourse capacity increases. More experiments are carried out by varying parameter settings. The results are similar to what we have presented here and are omitted.

5 Conclusion

In this paper, we study the problem of maintaining temporal validity on platforms where the resource supply is not continuous. The response time bounds of sensor transactions scheduled by GFP for HEMPR model are derived. Two deadline and period assignment schemes,

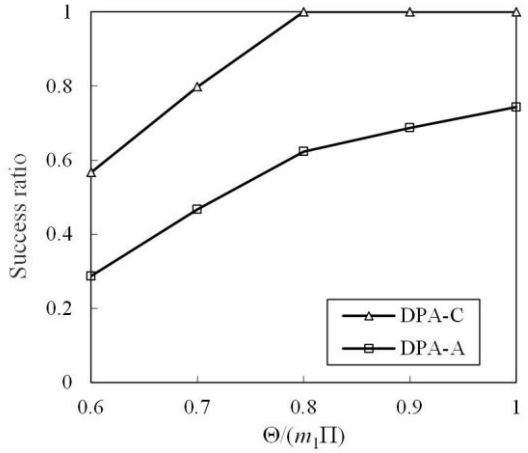

Figure 3. Scheduling success ratio comparison (density=1.75).

named calculation based scheme (DPA-C) and check based scheme (DPA-A), are proposed to maintain the validity constraints. DPA-C calculates deadlines and periods by using the response time bounds. DPA-A makes the assignment directly and then checks whether the assignment is feasible based on the derived bounds. The experimental results show that DPA-C achieves higher scheduling success ratio and imposes lower update workload than DPA-A does. This paper only considers the sensor transaction scheduling problem. In the furfure we will study how to simultaneously schedule sensor transactions and user transactions on non-continuously-executing resources.

References

1. M. Xiong, K. Ramamritham, Deriving deadlines and periods for real-time update transactions, IEEE Trans Comput,**53**(5), 567–583(2004)
2. J. Wang, S. Han, K. Y. Lam, et al., Maintaining data temporal consistency in distributed real-time systems, Real-Time Syst, **48**(4), 387–429(2012)
3. S. Han, D. Chen, M. Xiong, et al., Schedulability Analysis of Deferrable Scheduling Algorithm for Maintaining Real-Time Data Freshness, IEEE Trans Comput, **63**(4), 979–994(2014)
4. M. Xiong, S. Han, D. Chen, et al., Desh: overhead reduction algorithms for deferrable scheduling, Real-Time Syst, **44**(1), 1–25(2010)
5. M. Xiong, Q. Wang, K. Ramamritham, On earliest deadline first scheduling for temporal consistency maintenance, Real-Time Syst, **40**(2), 208–237(2008)
6. J. Li, M. Xiong, V. C. S. Lee, et al., Workload-Efficient Deadline and Period Assignment for Maintaining Temporal Consistency under EDF, IEEE Trans Comput,**62**(6), 1255–1268(2013)
7. S. Han, D. Chen, M. Xiong, et al., Online scheduling switch for maintaining data freshness in flexible real-time systems, *Proceeding of Real-Time Systems Symposium,* IEEE, Washington D.C., 115–124(2009)
8. S. Han, K. Y. Lam, J. Wang, et al., On Co-Scheduling of Update and Control Transactions in Real-Time Sensing and Control Systems: Algorithms,

Analysis, and Performance, IEEE Trans on Knowl Data En, **25**(10), 2325–2342(2013)
9. J. Li, D. Chen, M. Xiong, et al., Workload-Aware Partitioning for Maintaining Temporal Consistency upon Multiprocessor Platforms, *Proceedings of the IEEE 32nd Real-Time Systems Symposium*, IEEE, Vienna, 126–135 (2011)
10. I. Shin, A. Easwaran, I. Lee, Hierarchical Scheduling Framework for Virtual Clustering of Multiprocessors, *Euromicro Conference on Real-time Systems*, IEEE, Washington D.C., 181–190(2008)
11. N. Guan, M. Stigge, W. Yi, et al., New Response Time Bounds for Fixed Priority Multiprocessor Scheduling, IEEE Trans Commun, **51**(2), 387–397 (2010)
12. E. Bini, M. Bertogna, S. Baruah, Virtual Multiprocessor Platforms: Specification and Use, *IEEE Real-time Systems Symposium* , IEEE, Washington D.C., 437–446 (2009)