Real-Time Scheduling in Cyber-Physical Systems

Jai-Hoon Kim
Graduate School of Information and Communications
Ajou University, S. Korea
Pervasive Technology Institute, Indiana University, USA
jaikim@ajou.ac.kr

Geoffrey Fox
Pervasive Technology Institute
Indiana University, USA
gcf@indiana.edu

Abstract

Many researches have been performed for real-time scheduling. However, in CPS (cyber-physical system) where computers and physical systems are tightly coupled, we need to consider physical space (location, movement, etc.) as well cyber space (CPU, network, storage systems, etc). In this paper, we propose a new scheduling algorithm for CPS, where servicing node needs to move to serviced node for real-time services. Performance measurement by mathematics analysis shows that our LSTP (Least Slack Time First for CPS) algorithm reduces a deadline miss ration up to 49% and 37% comparing to FIFO (First In First Out) and LST(Least Slack Time First), respectively.

1. Introduction

Timing issues are critical in real-time systems such as robot control, flight control, on-line multimedia systems, and real-time stock trading system, etc. Many real-time scheduling algorithms such as RM(rate monotonic)[6,7], EDF(earliest deadline first)[5,7], and LST(least slack time first)[5,7] deal with resource (CPU and network bandwidth) scheduling to maximize real-time performance (e.g., deadline meet ratio)[5]. As CPS(cyber-physical system)[1,2] such as avionics, transportation, manufacturing processes, energy, healthcare, in which computers and physical systems are tightly coupled and timing is critical, is fast growing, real-time scheduling for CPS become new research issues in the real-time systems[3,4].

Many real-time scheduling algorithms have been proposed and widely used[5,6,7]. However, in CPS (cyber-physical system), we need to consider physical space (location, movement, etc.) as well cyber space (CPU, network, storage systems, etc). Important real-time scheduling issues in CPS systems are as follows:

- Spatial issues: effective release time and deadline of real-time tasks may be different depending on location and physical movement delay of nodes participating in CPS. Real-time scheduling algorithms have to be modified to include spatial factors.

- Conventional cyber real-time system schedules CPU or network bandwidth. However, in real-time scheduling for CPS, location is matter. Location of nodes in CPS might affects on effective release time and deadline.

Table. 1: Real-time Scheduling for CPS

	Conventional Real-Time Scheduling	Real-Time Scheduling for CPS
Resource	CPU, BW, Memory, I/O	CPU, BW, Memory, I/O
Model	● CPU scheduling in cyber environment ● Each task has a period(periodic task), an execution time, a release time, and deadline ● Scheduling algorithm (RM, EDF, LST, etc.)	● CPU scheduling in physical environment ● Each task has a physical movement delay time as well as a period(periodic task), an execution time, a release time, and deadline ● Scheduling algorithm (RM, EDF, LST, etc. including physical movement delay)
Spatial issues	Do not consider spatial issues (sometime consider communication delay)	Consider spatial and movement issues ● physical movement delay (computing node to task, or task to computing node) ● effective release time and deadline considering physical movement ● other physical factors affect on real-time performance

In this paper, we propose new scheduling algorithm for CPS, where computing node needs to move to target node for real-time services. If we assume, for an example, there are many scattered customers randomly requesting real-time services but only one staff exists in the area, real-time scheduling is necessary to maximize performance (e.g., deadline meet ratio). In this case, conventional real-time scheduling algorithm is not proper because the real-time scheduling does not consider spatial issues. In many kinds of CPS, where acting nodes must move to location to perform real-time services, time required for moving has to be included in the real-time scheduling. On the other hand, we can also consider a scenario, where customers move to acting node. But, in this paper, we consider the case in which acting node moves to customers. Many cases of CPS, effective release and deadline are changed. For an example, when a shuttle bus moves to airport through many stops, passengers on different stops have different deadlines to catch the shuttle.

2. Real-Time Scheduling Model in CPS

In this section, we propose a real-time scheduling for CPS and analyze real-time performance (deadline meet ration) for conventional real-time scheduling and proposed real-time scheduling for CPS. We assume parameters for real-time systems as follows:

- l_i : slack (laxity) time of task *i* (exponential distribution of average $1/\lambda$)
- e_i : execution time of task *i* (evenly distributed[0,E])
- m_i : moving time of computing(servicing) node to task *i* (evenly distributed[0,M])

Deadline meet ratio (DM) of task A without confliction against other tasks is the probability of slack time l_A being greater than moving time m_A (computing node moving to task A within slack time l_A). As distribution of l_A is $\lambda\varepsilon^{-\lambda t}$, deadline meet ratio of a task A (DM_A (λ, m)) is computed as follow:

$$DM_A(\lambda, m) = \int_m^\infty \lambda\varepsilon^{-\lambda t} dt = \varepsilon^{-\lambda m}$$

As *m* is assumed to evenly distributed[0,M]), an average deadline meet ratio is:

$$\text{mean}(DM_A(\lambda, m)) = \frac{1}{M}\int_0^M \varepsilon^{-\lambda m} dm = \frac{1}{\lambda M}(1-\varepsilon^{-\lambda M})$$

For a simple demonstration, we compute a deadline meet ratio when two tasks conflict each other. (As a future work, we will perform simulation in more realistic scenarios.) We compute deadline meet ratios for three different scheduling algorithms: FIFO(first in first serve), LST(least slack time first), LSTP (least slack time first with physical movement delay) scheduling algorithms.

2.1 FIFO

We assume task that task *A* arrived just before the other task *B*. A deadline meet ratio of task *A* is mean(DM_A (λ, m)) as task *A* is performed without confliction. As task *B* can be scheduled after task *A*, the deadline meet ratio of task *B* is the probability of the slack time of task *B* (l_B) being greater than m_A + e_A + m_B. Thus, deadline meet ratio of task *B* following task *A* (mean(DM_B (λ, m_A, e_A, m_B)) is computed as follow:

$$Mean(DM_B(\lambda, m_A, e_A, m_B)) = \int_0^M\int_0^E\int_0^M \varepsilon^{-\lambda(\mathrm{m_A}+\varepsilon_A+\mathrm{m_B})} \mathrm{dm_A}\, \mathrm{d}\varepsilon_A\, \mathrm{dm_B} = \frac{(1-\varepsilon^{-\lambda M})^2(1-\varepsilon^{-\lambda E})}{\lambda^3 \mathrm{M}^2 \mathrm{E}}$$

Now, we obtain the deadline meet ratio of FIFO scheduling algorithm when task *A* and task *B* are conflict.

DM_{fifo}= {mean(DM_A (λ, m)) + mean(DM_B (λ, m_A, e_A, m_B)) }/2= $\{\frac{1}{\lambda M}(1-\varepsilon^{-\lambda M})+\frac{(1-\varepsilon^{-\lambda M})^2(1-\varepsilon^{-\lambda E})}{\lambda^3 M^2 E}\}/2$

2.2 LST

When task *A* and task *B* conflict, a task with least slack time is scheduled first. When we assume that the slack time of task *A* is shorter than that of task *B*, the slack time of task *A* is exponential distribution of average $1/\lambda^2$. On the other hand, the slack of task *B* (shorter one) is exponential distribution of average $1/(2\lambda)$. A deadline meet ratio of task *A* is mean(DM_A (2λ, m)) as task *A* is performed without confliction. As task *B* of longer slack time can be scheduled after task *A* of shorter slack time, the deadline meet ratio of task *B* is the probability of the slack time of task *B* (l_B) being greater than m_A + e_A + m_B. Thus, an average deadline meet ratio of task *B* following task *A* (mean(DM_B (λ^2, m_A, e_A, m_B)) is computed as follow:

$$mean(DM_B\ (\lambda^2,\ m_A,\ e_A,\ m_B)) = \int_0^M\int_0^E\int_0^M \varepsilon^{-\lambda^2(\mathrm{m_A}+\varepsilon_A+\mathrm{m_B})}\mathrm{dm_A}\ \mathrm{d}\varepsilon_A\ \mathrm{dm_B} = \frac{(1-\varepsilon^{-\lambda^2 M})^2(1-\varepsilon^{-\lambda^2 E})}{\lambda^6 M^2 E}$$

Now, we obtain the deadline meet ratio of LST scheduling algorithm when two task *A* and task *B* conflict.

$$DM_{lst} = (\mathrm{mean}(DM_A\ (2\lambda,\ m)) + \mathrm{mean}(DM_B\ (\lambda^2,\ m_A,\ e_A,\ m_B)))/2$$

$$= \{\frac{1}{2\lambda M}(1-\varepsilon^{-2\lambda M})+\frac{(1-\varepsilon^{-\lambda^2 M})^2(1-\varepsilon^{-\lambda^2 E})}{\lambda^6 M^2 E}\}/2$$

2.3 LSTP

Preemptive LST is an optimal algorithm in real-time scheduling algorithm. However, in CPS, we need to consider physical environments. As an example, we have to consider moving time of computing (acting) node to the location of task serviced. When task *A* and task *B* conflict, a task with *least slack* time including moving time is scheduled *first*. Let's denote $l_{eff,i}$, be an effective slack time of task *i* (slack time including moving time), then $l_{eff,i}$, is computed as following:

$$l_{eff,i} = l_i - m_i$$

Now, we compute the $l_{eff,i}$. As distribution of l_i is $\lambda\varepsilon^{-\lambda t}$, $l_{eff,i}$ (when $l_{eff,i}$ > 0) distribution is computed as follow:

$$\frac{1}{M}\int_0^M \lambda\varepsilon^{-\lambda(t+m)}dm = \frac{\varepsilon^{-\lambda t}}{M}(1-\varepsilon^{-\lambda M})$$

$l_{eff,i}$ (when –M < $l_{eff,i}$ <0) distribution is computed as follow:

$$\frac{1}{M}\int_{-t}^M \lambda\varepsilon^{-\lambda(t+m)}dm = \frac{1}{M}(1-\varepsilon^{-\lambda(t+M)})$$

An average deadline meet ration of task *A* (without conflict) is the probability of $l_{eff,i} > 0$

$$\text{mean}(DM_A\ (\lambda,\ m)) = \int_0^\infty \frac{\varepsilon^{-\lambda t}}{M}(1-\varepsilon^{-\lambda M})dt = \frac{1}{\lambda M}(1-\varepsilon^{-\lambda M})$$

Deadline meet ratio of task *B* following task *A* is:

$$DM_B\ (\lambda,\ m_A,\ e_A,\ m_B) = \int_{m_A+e_A}^\infty \frac{\varepsilon^{-\lambda t}}{M}(1-\varepsilon^{-\lambda M})\varepsilon^{-\lambda t}dt = \frac{(1-\varepsilon^{-\lambda M})}{\lambda M}\varepsilon^{-\lambda(m_A+e_A)}$$

As we assume m_A and e_A are evenly distributed on [0,M] and [0,E], respectively, mean(DM_B (λ, m_A, e_A, m_B)) is computed as:

$$\text{mean}(DM_B\ (\lambda,\ m_A,\ e_A,\ m_B)) = \frac{1}{ME}\int_0^E\int_0^M \frac{(1-\varepsilon^{-\lambda M})}{\lambda M}\varepsilon^{-\lambda(m_A+e_A)}dmde = \frac{(1-\varepsilon^{-\lambda M})^2}{\lambda^3 M^2 E}(1-\varepsilon^{-\lambda E})$$

We can find that mean(DM_A (λ, m)) and mean(DM_B (λ, m_A, e_A, m_B) are same as those obtained in subsection 2.1. As parameters using in two analyses are the same but $l_{eff,i,} = l_{i,} - m_i$, two deadline meet ratios computed in 2.1 and 2.3 must be the same. (One uses $l_i > m_i$ while the other $l_{eff,i}$ $(= l_{i,} - m_i\ ) > 0$, which is basically same, to compute deadline meet ratio.) Let *p* be probability of meeting deadline of firstly scheduled task.

$$p = \text{mean}(DM_A\ (\lambda,\ m)) = \int_0^\infty \frac{\varepsilon^{-\lambda t}}{M}(1-\varepsilon^{-\lambda M})dt = \frac{1}{\lambda M}(1-\varepsilon^{-\lambda M})$$

Let *q* be probability of meeting deadline of the secondly scheduled task.

$$q = \text{mean}(DM_B\ (\lambda,\ m_A,\ e_A,\ m_B)) = \frac{1}{ME}\int_0^E\int_0^M \frac{(1-\varepsilon^{-\lambda M})}{\lambda M}\varepsilon^{-\lambda(m_A+e_A)}dmde = \frac{(1-\varepsilon^{-\lambda M})^2}{\lambda^3 M^2 E}(1-\varepsilon^{-\lambda E})$$

We use somewhat different approach from LST scheduling to compute deadline meet ratio for LSTP scheduling. LSTP scheduling considers moving time as well as slack time to improve deadline meet ratio. When task *A* and task *B* conflict, LSTP schedules tasks (*A* followed by *B* or *B* followed by *A*), which maximize a deadline meet ratio. On a schedule of *A* followed by *B*, there are four cases:

- Both *A* and *B* meet the deadline (probability of pq): In this case, LSTP does not change schedule (choose schedule of *A* followed by *B*)
- *A* only meets the deadline(probability of p(1-q)): In this case, LSTP changes schedule (*B* followed by *A*) if both *A* and *B* meet the deadline. Probability of meeting deadline for both *A* and *B* by changing schedule is (p-q)/(1-q)*q/p. ((probability of *B* meeting the deadline at scheduling of *B* followed by *A* on the condition of missing the deadline at scheduling of *A* followed by *B*)*(probability of *A* meeting the deadline also even at scheduling of B followed by *A* on the condition of meeting the deadline at scheduling of *A* followed by *B*))
- *B* only meets the deadline (probability of (1-p)q). In this case, *A* cannot meet deadline at any scheduling.

- Neither *A* nor *B* meets deadline (probability of (1-p)(1-q)): In this case, LSTP changes schedule (*B* followed by *A*) if *B* meets the deadline. Probability of meeting deadline for *B* by changing schedule is (p-q)/(1-q) (probability of B meeting the deadline at scheduling of *B* followed by *A* on the condition of missing the deadline at scheduling of *A* followed by *B*). In this case, *A* cannot meet deadline at any scheduling.

The other schedule, *B* followed by *A*, has also four cases. LSTP choose the schedule which maximize deadline meet ratio by considering moving time as well as slack time.

Deadline meet/miss on schedule A➔B (A followed by B)	probability	LSTP schedule	probability of LSTP choosing this schedule	no. of task meeting deadline
meet A, meet B	pq	A➔B	pq	2
meet A, miss B	p(1-q)	change schedule B➔A if meet both A and B	p(1-q)*(p-q)/(1-q)*q/p	2
		A➔B If B➔A is not better	p(1-q)*{1-(p-q)/(1-q)*q/p}	1
miss A, meet B	(1-p)q	A➔B (B➔A)	(1-p)q	1
miss A, miss B	(1-p)(1-q)	schedule B➔A if meet B	(1-p)(1-q)*(p-q)/(1-q)	1
		A➔B (if If B➔A is not better)	(1-p)(1-q)*{1-(p-q))/(1-q)}	0

From the above table, we can obtain the deadline meet ratio of LSTP scheduling algorithm when task *A* and task *B* conflict. We can compute expected number of tasks meeting the deadline by summation of products of columns "*probability of LSTP choosing this schedule*" and "*no. of task meeting deadline*". After that, deadline meet ratio is the half of expected number of tasks meeting the deadline as there are two tasks.

DM_{lstp}= 2pq + 2p(1-q)*(p-q)/(1-q)*q/p + p(1-q)*{1-(p-q)/(1-q)*q/p} + (1-p)q + (1-p)(1-q)*(p-q)/(1-q)

= $(2p - p^2 + 2pq - q^2)/2$,

where p = (mean(DM_A (λ, m)) = $\frac{1}{\lambda M}(1-\varepsilon^{-\lambda M})$

and q = (mean(DM_B (λ, m_A, e_A, m_B))) = $\frac{(1-\varepsilon^{-\lambda M})^2(1-\varepsilon^{-\lambda E})}{\lambda^3 M^2 E}$.

2.4 Performance Comparisons for Real-Time CPS

We measure performance by varying parameters, λ and *M*. (we assume that *M*=*E*.) We compare performance among FIFO, LST, and LSTP. Fig. 1 shows deadline meet ratios for FIFO, LST, and LSTP scheduling algorithms. Fig. 2 shows relative views of Figure 1 (relative deadline miss ratios of LSTP to FIFO and LSTP to LST) LSTP algorithm can reduce deadline meet ratios up to 49% and 37% comparing to FIFO and LST algorithms, respectively.

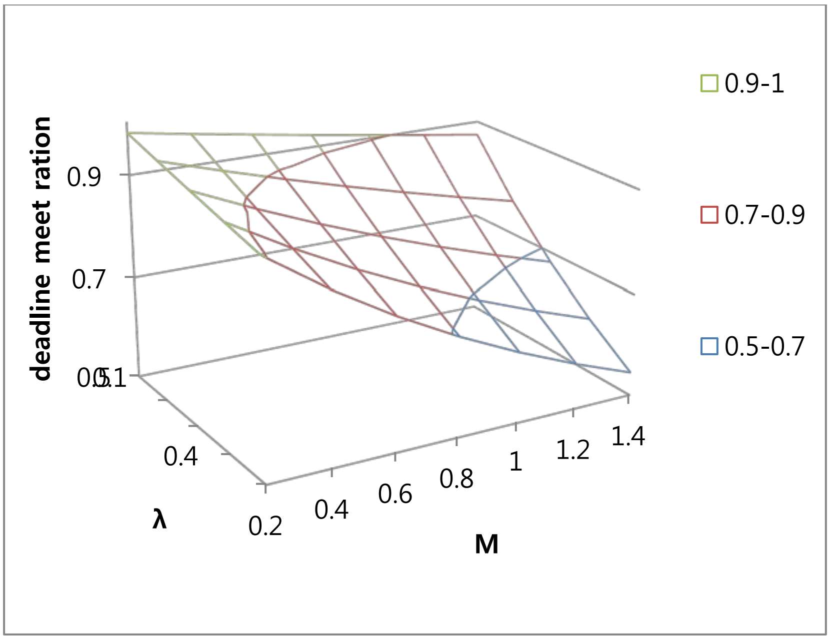


Fig.1 (a) Deadline meet ratio (FIFO)

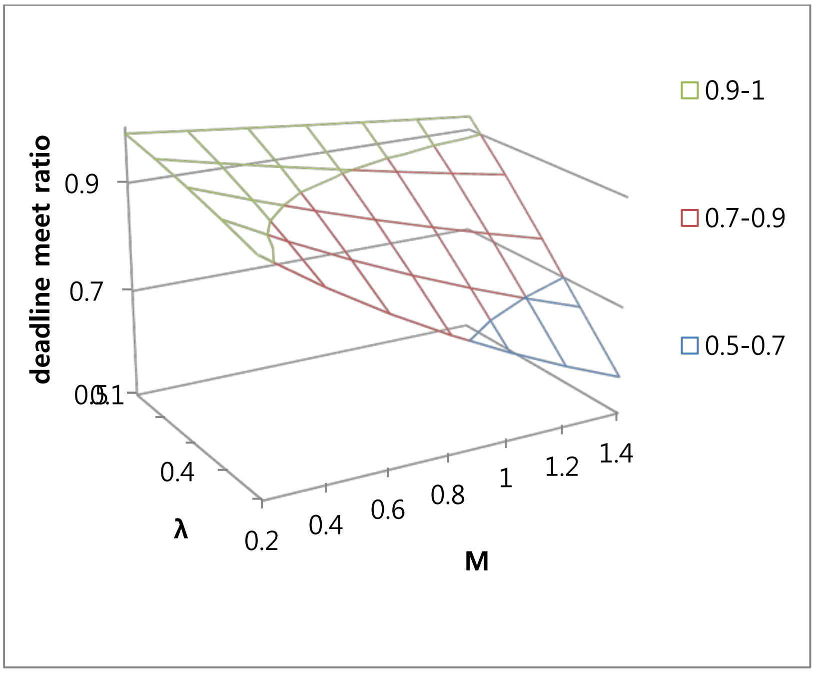


Fig.1 (b) Deadline meet ratio (LST)

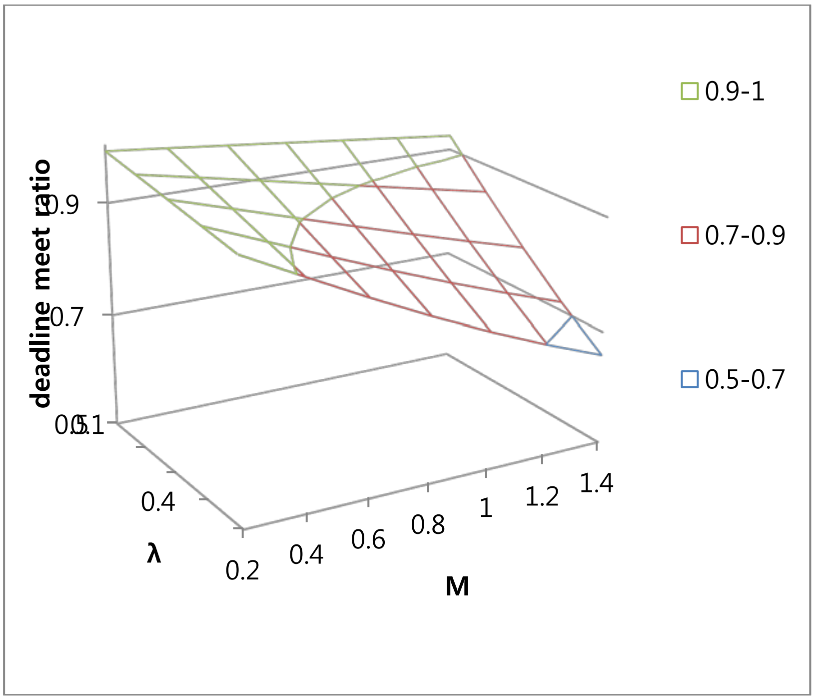
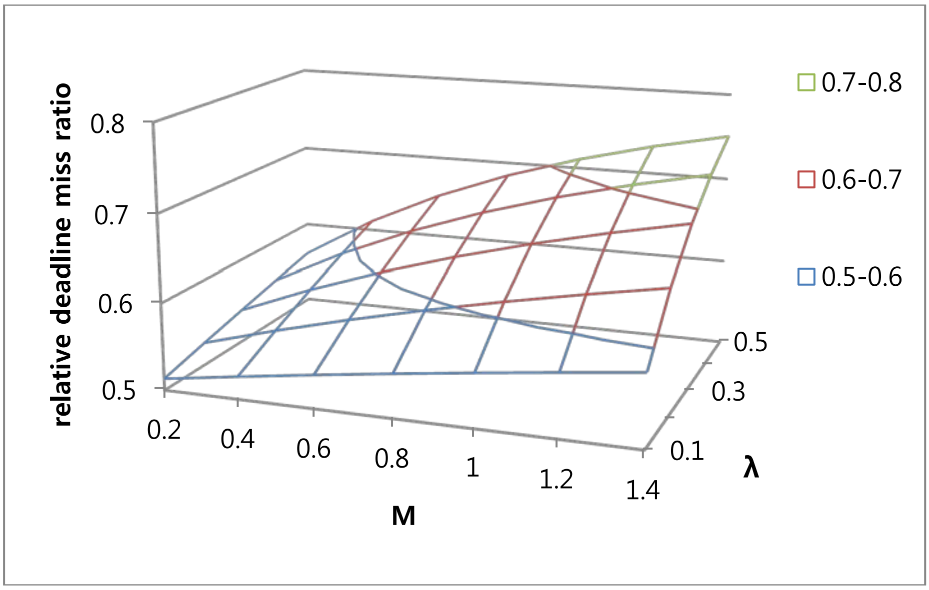
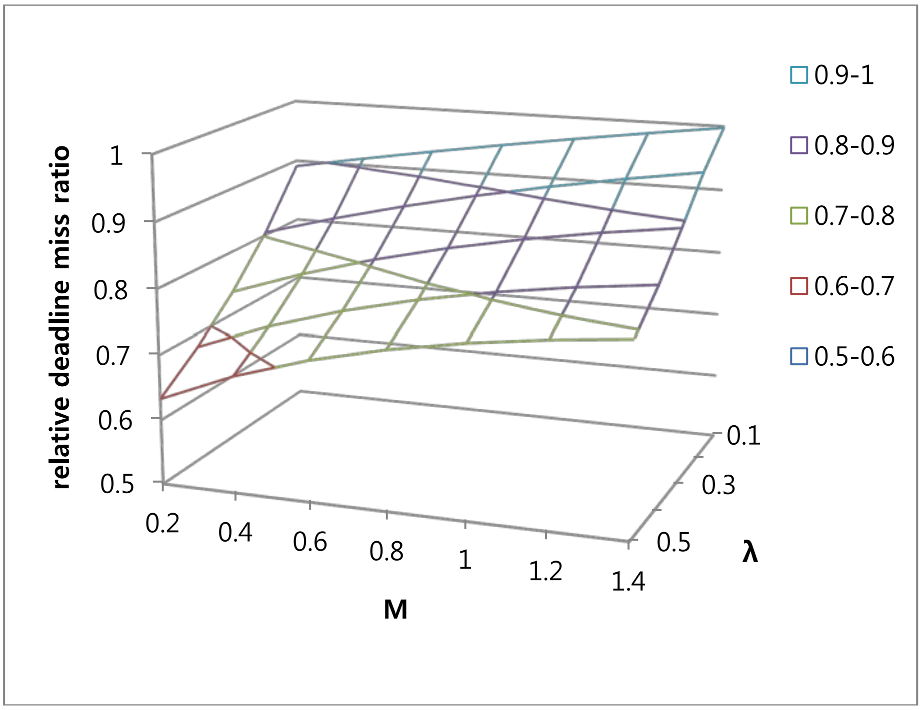


Fig.1 (c) Deadline meet ratio (LSTP)



Figiure 2 (a) Relative deadline miss ratio (LSTP to FIFO)



Figiure 2 (b) Relative deadline miss ratio (LSTP to LST)

3. Conclusion and Future Works

As conventional real-time scheduling algorithm considers system resources in cyber space such CPU, network bandwidth, and memory, it does not proper in physical space. We propose real-time scheduling algorithm for CPS, where physical factors (e.g., location, movement delay, etc.) affect on real-time performance. To demonstrate real-time scheduling algorithm for CPS, we assume a simple CPS environment in which computing node moves around physically distributed tasks to perform real-time services. Performance measurement by mathematics analysis shows that our LSTP (Least Slack Time First for CPS) algorithm reduces a deadline miss ration up to 49% and 37% comparing to FIFO (First In First Out) and LST(Least Slack Time First), respectively. We plan to perform extensive simulations to verify performance of LSTP in more realistic environment.

References

[1] Ragunathan Rajkumar, Insup Lee, Lui Sha, and John Stankovic, "Cyber-Physical Systems: The Next Computing Revolution," *Design Automation Conference 2010*, pp. 731-736, 2010.

[2] Edward A. Lee, "CPS Fouldations," Design Automation Conference 2010, pp. 737-742, 2010.

[3] John Eidson, Edward A. Lee, Slobodan Matic, Sanjit A. Seshia, Jia Zou. "Distributed Real-Time Software for Cyber-Physical Systems". IEEE Proceedings, December 2011; To appear.

[4] Edward A. Lee. "Computing Needs Time". Communications of the ACM, 52(5):70-79, May 2009.
[5] Jane W. S. Liu, Real-Time Systems, Prentice Hall, 2000.
[6] C. L. Liu and J. Layland (1973), "Scheduling algorithms for multiprogramming in a hard real-time environment", *Journal of the ACM* **20** (1): 46–61, 1973.
[7] A. Burns, "Scheduling Hard Real-Time Systems: a review,", Software Engineering Journal, pp. 116-128, May 1991.