IIV Hw 1 R13922A15 陳星佑

1 Timing Analysis of the CAN Protocol: Part I (12pts)

Given a set of periodic messages μ_0, μ_1, μ_2 with their priorities, transmission times, and periods as follows:

Message	Priority (P_i)	Transmission Time (C_i) (msec)	Period (T_i) (msec)
μ_0	0	10	50
μ_1	1	30	200
μ_2	2	20	100

The worst-case response time R_i of μ_i can be computed as

$$R_i = Q_i + C_i, \tag{1}$$

and

$$Q_i = B_i + \sum_{\forall j, P_j < P_i} \left\lceil \frac{Q_i + \tau}{T_j} \right\rceil C_j, \tag{2}$$

where $\tau = 0.1$ in this question. You can consider using the following tables to help you.

1. (4pts) What is the worst-case response time of μ_0?

Iteration	LHS (Q_0)	B_0	RHS	Stop?
1	30	30	30	Y

$\Rightarrow R_0 = Q_0 + C_0 = 30 + 10 = 40$ *.

2. (4pts) What is the worst-case response time of μ_1?

Iteration	LHS (Q_1)	B_1	j	$Q_1 + \tau$	T_j	$\left\lceil \frac{Q_1 + \tau}{T_j} \right\rceil$	C_j	RHS	Stop?
1	30	30	0	30.1	50	1	10	40	N
2	40	30	0	40.1	50	1	10	40	Y

$RHS_1 = 30 + 1 \times 10 = 40 = LHS_1$

$RHS_2 = 30 + 1 \times 10 = 40 = LHS_2$

$\Rightarrow R_1 = Q_1 + C_1 = 40 + 30 = 70$ *

3. (4pts) What is the worst-case response time of μ_2?

Iteration	LHS (Q_2)	B_2	j	$Q_2+\tau$	T_j	$\lceil \frac{Q_2+\tau}{T_j} \rceil$	C_j	RHS	Stop?
1	20	20	0 1	20.1	50 200	1 1	10 30	60	N
2	60	20	0 1	60.1	50 200	2 1	10 30	70	N
3	70	20	0 1	70.1	50 200	2 1	10 30	70	Y

$RHS_1 = 20 + 1\times 10 + 1\times 30 = 60 \neq LHS_1$

$RHS_2 = 20 + 2\times 10 + 1\times 30 = 70 \neq LHS_2$

$RHS_3 = 20 + 2\times 10 + 1\times 30 = 70 = LHS_3$

$\Rightarrow R_2 = Q_2 + C_2 = 70 + 20 = 90$ #.

2 Timing Analysis of the CAN Protocol: Part II (36pts)

Please download the benchmark "input.dat" from NTU COOL. In the benchmark, the first number is n, the number of messages. The second number is τ. Each of the following lines contains the priority (P_i), the transmission time (C_i), and the period (T_i) of each message. You are required to do two things in your submission:

1. You should print out n numbers (one number per line) representing the worst-case response time (R_i) of those messages. Note that you need to follow the message ordering in the benchmark, *e.g.*, the first number in the list is the worst-case response time of the first message in the benchmark.
2. You should also print out your source codes. (For your information, my implementation is less than 100 lines.) We may ask you to provide your source codes which must be the same as those on your printout. If the worst-case response times above are correct but the source codes are clearly wrong implementation, it is regarded as academic dishonesty.

It is highly recommended to write your codes well (*e.g.*, capable of dynamically allocating memory based on n) so that you can reuse them in Homework 2. Ideally, you can test your implementation with the small benchmark in Question 1 and verify its solution by your implementation. Just do not make the same mistake in Questions 1 and 2.

Code :

```python
import math

def cal_Bi(i, msg_list):
    lower_equal_priority = [msg for msg in msg_list if msg.p >= msg_list[i].p]
    Bi = max(msg.c for msg in lower_equal_priority)
    return Bi

def cal_rhs(i, msg_list, tau, Qi):
    higher_priority = [msg for msg in msg_list if msg.p < msg_list[i].p]

    Bi = cal_Bi(i, msg_list)

    sum_wtime = sum(math.ceil((Qi + tau) / msg.t) * msg.c for msg in higher_priority)

    rhs = Bi + sum_wtime

    return rhs

def cal_ri(i, msg_list, tau, Qi):
    rhs = cal_rhs(i, msg_list, tau, Qi)

    if rhs + msg_list[i].c > msg_list[i].t:
        return -1
    elif rhs == Qi:
        return Qi + msg_list[i].c
    else:
        return cal_ri(i, msg_list, tau, rhs)

class message:
    def __init__(self, p, c, t):
        self.p = p
        self.c = c
        self.t = t

if __name__ == "__main__":
    try:
        with open("input.dat", "r") as file:
            data = file.readlines()
            msg_len = int(data[0].strip())
            tau = float(data[1].strip())

            msg_list = list()

            for line in data[2:]:
                values = line.strip().split()
                msg_list.append(message(int(values[0]), float(values[1]), int(values[2])))

            for i in range(msg_len):
                Bi = cal_Bi(i, msg_list)
                ri = cal_ri(i, msg_list, tau, Bi)
                print(ri)

    except FileNotFoundError:
        print("input.dat file not found.")
    except ValueError:
        print("Error in converting data.")
```

Result :

```
1.44
2.04
2.56
3.16
3.68
4.28
5.2
8.4
9.0
9.68
10.2
19.360000000000003
19.8
20.32
29.400000000000002
29.76
30.28
```

3 Timing Analysis of Preemptive Fixed-Priority Scheduling (16pts)

The CAN protocol is based on non-preemptive fixed-priority scheduling. For tasks on an Electronic Control Unit (ECU), they are usually scheduled by preemptive fixed-priority scheduling. The worst-case response time R_i of a task τ_i can be computed as

$$R_i = C_i + \sum_{\forall j, P_j < P_i} \left\lceil \frac{R_i}{T_j} \right\rceil C_j, \tag{3}$$

where P_i, C_i, and T_i are the priority, the computation (execution) time, and the period of τ_i, respectively. Given a set of periodic tasks τ_0, τ_1, τ_2 with their priorities, computation times, and periods as follows:

Task	Priority (P_i)	Computation Time (C_i) (msec)	Period (T_i) (msec)
τ_0	0	10	50
τ_1	1	30	200
τ_2	2	20	100

1. (4pts) What is the worst-case response time of τ_0?

Iteration	LHS (R_0)	C_0	RHS	Stop?
1	10	10	10	Y

⇒ $R_0 = 10$ ✱

2. (4pts) What is the worst-case response time of τ_1?

Iteration	LHS (R_1)	C_1	j	R_1	T_j	$\lceil \frac{R_1}{T_j} \rceil$	C_j	RHS	Stop?
1	30	30	0	30	50	1	10	40	N
2	40	30	0	40	50	1	10	40	Y

⇒ $R_1 = 40$ ✱

3. (4pts) What is the worst-case response time of τ_2?

Iteration	LHS (R_2)	C_2	j	R_2	T_j	$\lceil \frac{R_2}{T_j} \rceil$	C_j	RHS	Stop?
1	20	20	0 1	20	50 200	1 1	10 30	60	N
2	60	20	0 1	60	50 200	2 1	10 30	70	N
3	70	20	0 1	70	50 200	2 1	10 30	70	Y

⇒ $R_2 = 70$ ✱.

4. (4pts) Compared with non-preemptive fixed-priority scheduling, preemptive fixed-priority scheduling is expected to be disadvantageous to the lowest-priority message/task. Explain why the the worst-case response time of τ_2 is smaller than the worst-case response time of μ_2 in Question 1.

Answer :

1. Non-Preemptive (CAN) Scheduling
- In non-preemptive scheduling, μ2 must wait until all higher-priority messages have finished their transmissions before it can execute.
- More critically, once μ2 starts transmission, it cannot be interrupted, even if a higher-priority message arrives. The higher-priority message must wait until μ2 completes its transmission.
- This results in a single long blocking time, leading to an extremely high Worst-Case Response Time (WCRT) of 90 ms.

2. Preemptive Scheduling
- In preemptive scheduling, τ2 can be interrupted multiple times by higher-priority messages.
- Although τ2 still experiences interference from higher-priority messages, the interference is distributed rather than concentrated in a single long blocking period.
- As a result, while τ2 still suffers from interruptions, its overall WCRT is shorter, measured at 70 ms in this case.

4 Timing Analysis of TDMA-Based Protocols (12pts)

Following the assumptions (each time slot has the same length, each time slot serves exactly one frame, and a frame is transmitted only if the whole time slot is available) in the lecture, please compute the worst-case response time of the "asynchronous" message with the frame arrival pattern $(4, 10, 0, 3, 5, 6)$ and the schedule pattern $(2, 5, 1, 2)$ by completing the following steps.

1. (2pts) Please duplicate the schedule pattern (hint: $(4, 10, 1, 2, \ldots)$). No intermediate work is needed here.

schedule : (4, 10, 1, 2, 6, 7)

2. (2pts) Please duplicate the arriving times of frames in the frame arrival pattern but fix $m = 4$ and $p = 10$. No intermediate work is needed here.

frame : (4, 10, 0, 3, 5, 6, 10, 13, 15, 16)

3. (2pts) Please duplicate the starting times of time slots in the schedule pattern but fix $n = 4$ and $q = 10$. No intermediate work is needed here.

schedule : (4, 10, 1, 2, 6, 7, 11, 12, 16, 17)

4. (4pts) Please complete the following table:

k	$\max_{1\le j\le n}(s_{j+k} - s_j)$	=	$\min_{1\le i\le m}(a_{i+k-1} - a_i)$	=	(Column-3) − (Column-5)
1	$\max_{1\le j\le 4}(s_{j+1} - s_j)$	4	$\min_{1\le i\le 4}(a_i - a_i)$	0	4
2	$\max_{1\le j\le 4}(s_{j+2} - s_j)$	5	$\min_{1\le i\le 4}(a_{i+1} - a_i)$	1	4
3	$\max_{1\le j\le 4}(s_{j+3} - s_j)$	9	$\min_{1\le i\le 4}(a_{i+2} - a_i)$	3	6
4	$\max_{1\le j\le 4}(s_{j+4} - s_j)$	10	$\min_{1\le i\le 4}(a_{i+3} - a_i)$	6	4

5. (2pts) Please compute the worst-case response time (which is waiting time plus transmission time) of the message.

1 + max(4, 4, 6, 4) = 7 ✱

5 MILP Linearization (12pts)

We will prove or make the following propositions are equivalent so that we can transform constraints to linear forms and thus apply the Mixed Integer Linear Programming (MILP). Note that "$\Longleftrightarrow$" denotes "equivalence" and "$\wedge$" denotes "logical conjunction" (AND).

1. (4pts) Given α, β, γ which are binary variables, prove

$$\alpha+\beta+\gamma \neq 2 \Longleftrightarrow (\alpha+\beta-\gamma \leq 1) \wedge (\alpha-\beta+\gamma \leq 1) \wedge (-\alpha+\beta+\gamma \leq 1)$$

by filling "T" (True) or "F" (False) in the following table (if LHS=RHS in all cases, then LHS and RHS are equivalent):

α	β	γ	LHS	$\alpha+\beta-\gamma \leq 1$	$\alpha-\beta+\gamma \leq 1$	$-\alpha+\beta+\gamma \leq 1$	RHS	LHS=RHS?
0	0	0	T	T	T	T	T	T
0	0	1	T	T	T	T	T	T
0	1	0	T	T	T	T	T	T
0	1	1	F	T	T	F	F	T
1	0	0	T	T	T	T	T	T
1	0	1	F	T	F	T	F	T
1	1	0	F	F	T	T	F	T
1	1	1	T	T	T	T	T	T

2. (4pts) Given α, β, γ which are binary variables, prove

$$\alpha\beta=\gamma \Longleftrightarrow (\alpha+\beta-1 \leq \gamma) \wedge (\gamma \leq \alpha) \wedge (\gamma \leq \beta)$$

by filling "T" (True) or "F" (False) in the following table (if LHS=RHS in all cases, then LHS and RHS are equivalent):

α	β	γ	LHS	$\alpha+\beta-1 \leq \gamma$	$\gamma \leq \alpha$	$\gamma \leq \beta$	RHS	LHS=RHS?
0	0	0	T	T	T	T	T	T
0	0	1	F	T	F	F	F	T
0	1	0	T	T	T	T	T	T
0	1	1	F	T	F	T	F	T
1	0	0	T	T	T	T	T	T
1	0	1	F	T	T	F	F	T
1	1	0	F	F	T	T	F	T
1	1	1	T	T	T	T	T	T

3. (4pts) Given β which is a binary variable, x, y which are non-negative real variables, and a constraint $x \leq 2025$, select a value of M to guarantee

$$\beta x = y \iff 0 \leq y \leq x \wedge x - M(1-\beta) \leq y \wedge y \leq M\beta,$$

where you can refer to the following table:

β	LHS	$0 \leq y \leq x$	$x - M(1-\beta) \leq y$	$y \leq M\beta$	RHS
0	$0 = y$	$0 \leq y \leq x$	$x - M \leq y$	$y \leq 0$	$x - M \leq y = 0 \leq x$
1	$x = y$	$0 \leq y \leq x$	$x \leq y$	$y \leq M$	$0 \leq y = x \leq M$

$x \leq 2025$, $x \leq M$, $y \leq M$, $y = x$ 都需要成立

$\Rightarrow M = 2025$ #.

6 Signal Packing (12pts)

Bit stuffing does not need to be considered in this problem, *i.e.*, you can assume that the length of a message is the length of its data field plus 44 plus 3. Note that the length of a data field must be 8, 16, 24, ..., or 64 bits, even if the message itself is shorter. Assume that there are 4 Electronic Control Units (ECUs), $\varepsilon_0, \varepsilon_1, \varepsilon_2, \varepsilon_3$, and 4 messages, $\mu_0, \mu_1, \mu_2, \mu_3$, as follows:

Message	Sender	Receiver(s)	Number of Bits (Data Field)	Period (msec)
μ_0	ε_0	ε_1	6	50
μ_1	ε_0	ε_1	10	50
μ_2	ε_1	$\varepsilon_2, \varepsilon_3$	10	50
μ_3	ε_0	ε_3	16	100

A system designer redesigns the messages as follows:

Message	Sender	Receiver(s)	Number of Bits (Data Field)	Period (msec)
μ_0'	ε_0	ε_1	16	50
μ_2	ε_1	$\varepsilon_2, \varepsilon_3$	10	50
μ_3	ε_0	ε_3	16	100

where the first 6 bits of μ_0' are the bits from μ_0 and the following 10 bits of μ_0' are the bits from μ_1.

1. (4pts) Regarding the number of bits which need to be transmitted, do you think that the new design is better? Please explain.

Original (w/ 8n bits)

訊息	實際 bits	overhead 47 bits	總bits	頻率 (每秒)	總bits/秒
μ_0	8	47	55	20	1100 bits
μ_1	16	47	63	20	1260 bits
μ_2	16	47	63	20	1260 bits
μ_3	16	47	63	10	630 bits
合計					4250 bits

Redesigned (w/ 8n bits)

訊息	實際 bits	overhead 47 bits	總bits	頻率 (每秒)	總bits/秒
μ_0'	16	47	63	20	1260 bits
μ_2	16	47	63	20	1260 bits
μ_3	16	47	63	10	630 bits
合計					3150 bits

3150 < 4250 ⇒ New design is better

(每秒傳的 bits 較少)

2. (4pts) Can you further merge μ_2 into μ_0'?

μ_0' 的 sender 為 ε_0

μ_2 的 sender 為 ε_1

⇒ sender 不同 無法 merge ✗.

3. (4pts) In most cases, it does not hurt to have more frequent messages, but it is not allowed to have less frequent messages. Following this policy, can you further improve the number of bits which need to be transmitted? Please explain.

若將 μ_3 頻率提高至 50 ms，可與 μ_0' merge：

After merge：

訊息	實際 bits	overhead 47 bits	總bits	頻率 (每秒)	總bits/秒
$\mu_0'+\mu_3$	32	47	79	20	1580 bits
μ_2	16	47	63	20	1260 bits
合計					2840 bits

⇒ 2840 < 3150 (redesigned)

⇒ 將 μ_0' 和 μ_3 merge 後可再 improve ※