University of Uppsala

Department of Computer Systems (DoCS)

Final Examination

IT3 (Datorsystem II)

Distributed and Real-Time Systems

INSTRUCTIONS TO CANDIDATES

- This is a SIX (6) hour examination
- Answer all questions on the writing paper provided
- All questions to be answered in English
- Dictionaries are Permitted
- Marks total 85 points
- There are 12 questions
- The exam paper is 8 pages including this cover page

Section A: Distributed Systems

Question 1

Give an English language definition of what a distributed system is.

Discuss how a parallel system is different from a distributed system.

[4]

Question 2

Parallel and distributed programming involves added complications when specifying the implementation code.

List three factors that must be considered when writing a distributed application, and describe how each of these factors makes parallel programming more complex than sequential programming.

[6]

Question 3

What is the foundation of the Flynn Taxonomy of computer archtectures? That is, what characteristics of parallel systems does Flynn use to divide computers into different classes?

Describe the four classes that Flynn defined and present a brief overview of the characteristics of machines that fall into each of these classes.

[4]

Question 4

The following code gives the simple version of the RankSort algorithm that was discussed in the lectures.

```
void RankSort (int Id, int I[], int R[], int F[])
{
    int my_value= I[Id];

    // Locate final location of my_value
    for( int i=0; i < N; i++ )
        if( my_val > I[i] ) R[Id]++;

    // Place my_value into final position in F[]
    F[R[Id]]= my_value;
}
```

This algorithm is limited, and only operates correctly on sets of numbers which consist of unique values.

Eg: 2 9 8 3 5 7 1

The algorithm as given will not deal correctly with sets of numbers in which some values are duplicated.

Given the information above answer the following questions.

(a). Given the following set of values

1 4 2 4 6 3 4 5

show the contents of the arrays R and F after the execution of the RankSort program. Assume that all entries in F were intially zero.

[2]

(b). What is the problem demonstrated by the contents of the array F you computed above?

[1]

(c). Write a revised version of the RankSort program that addresses the problem and results in a correct sorted sequence being stored in F when there are some duplicate values.

[7]

Question 5

Define the difference between shared memory and distributed memory programming models.

Using CSP as an example of a distributed programming language explain how this language allows the distributed parts of the program to communicate data values to each other. What language instructions are used for communication? State the syntax of these instructions and give an example of how they are used, and explain what they do.

[6]

Question 6

Use the Linda language to write a program that finds the maximum of a set of values. Assume that the number of values (N) is known and a power of 2.

[10]

Section B: Real Time Systems

Question 7

(a). Give an English definition of schedulability for a system of tasks with known deadlines. What aspect of the tasks needs to be known in order to know if they can all meet their deadlines?

[3]

(b). Formalise your definition above into an equation. Assume that the tasks are defined by a set $\mathcal{T}$.

[2]

Question 8

In this question you will discuss the formulation and application of the following formula.

$$R_i = C_i + \sum_{\forall j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j$$

(a). Explain the meaning of each of the following terms in the equation given above. First define the meaning of the variable i.

- C_i and C_j
- R_i
- T_j
- hp(i)

[2]

(b). What does the last term in this equation represent? Give an English explanation of what this term means.

[2]

(c). What restrictions must be placed on the values of D_i in this equation in order for the resulting calculations to be meaningful. Explain why this restriction is required.

[2]

Question 9

To compute a schedule we need to know the value of C_i in the equation given in the previous question. Assuming access to instruction timing information explain how to compute a value of C_i for a piece of code which includes an **<u>if</u>** statement. Give a general method.

[3]

Given the following information related to execution costs of Assembler instructions compute a value of C_i for the code segment given below.

Instr	Cycles
MOV	8
CMP BGT JMP ADD	4
MUL	16 + 2*(# of set bits in 1st argument)

The code to be analysed is,

```
    MOV (_x), D1
    MUL D1, #4
    ADD D1, (_A)
L2: CMP D0, #6
    BGT L1

    MOV (D1), D2
    CMP D2, #0
    BGT L3

    MUL D2, #20
    ADD D2, #10
    MOV D2, (D1)

L3: ADD D0, #1
    ADD D1, #4
    JMP L2

L1: ...
```

[8]

Question 10

Given the following task set compute the worst case response time for each task, and determine if the system is schedulable using rate monotonic scheduling. Assign priorities to the tasks and compute the worst case response time using the formula given in the previous question.

Task	T	D	C
T_1	54	12	6
T_2	93	30	11
T_3	100	6	4
T_4	110	65	38

Is there another priority assignment algorithm that is optimal in this situation? If so compute a new set of worst case response times and thus show that the new allocation of priorities allows the system to be scheduled.

[5]

Question 11

Give an English description of the "Jitter" in a task T_1. Describe how "Jitter" can be inherited by,

(a). A message which is launched by T_1.

(b). A task T_2 which is released by the receipt of a message sent by T_1.

(c). What other factors can affect the release of a task by a message? Discuss their importance in terms of deriving a correct schedule.

[5]

Question 12

(a). Blocking needs to be included in the computations of worst case response time for tasks. What is blocking, and how is it different to delay due to pre-emption?

[1]

(b). Describe the two situations where blocking can occur in a distributed system. In each case define, using an English description of the problem, how to compute the worst case blocking time.

[1]

(c). In a system where blocking can occur schedules produced using the standard priority assignment algorithms and worst case response time calculations can fail due to a phenomenon called "priority inversion".

Construct a simple scheduling example that demonstrates how priority inversion can lead to missed deadlines. Draw a diagram of your tasks which shows how the tasks execute, and which task misses its deadline.

[4]

(d). Describe an approach to dealing with priority that resolves this problem.

Using the same example draw a picture that shows how applying the priority handling approach you described in (c). above resolves the problem, and guarantees that all tasks can meet their deadlines.

[2]

(e). Use the following equation to compute the worst case response times for the task set you have proposed using the priority handling approach you described in (c)., and show formally that the system you have described is schedulable.

$$R_i = C_i + B_i + \sum_{\forall j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j$$

where the blocking factor is defined by

$$B_i = \max_{\{s,t | t \in lp(i) \wedge s \in used\_by(t) \wedge ceil(s) \geq pri(i)\}} (cs_{t,s})$$

[5]