Section A
Compulsory Questions (60marks)

Question 1

What is a TTPN? Give an example TTPN, and explain its features and annotations.

[6]

SOLUTION

A TTPN is a Transition Timed Petri Net.

[1]

This type of net has times associated with the firing of transitions, and those timings can be specified using functions (eg exponential functions). The result of these functions is to associate probability distributions with the firing of transitions.

[2]

In annotating such a net, labels are associated with the places and the transitions to indicate their interpretation in the model. Timing functions are associated with the transitions to control the delay before they fire.

[3]

Question 2

Construct a TTPN model which represents the flow of traffic on a freeway at one of its entry feeds. Assume that the traffic from the feed lane merges into the continuous flow of traffic on the major road and that traffic flow analysis indicates that openings in the traffic are exponentially distributed. You may assume that traffic flow conditions are stable 24hrs per day.

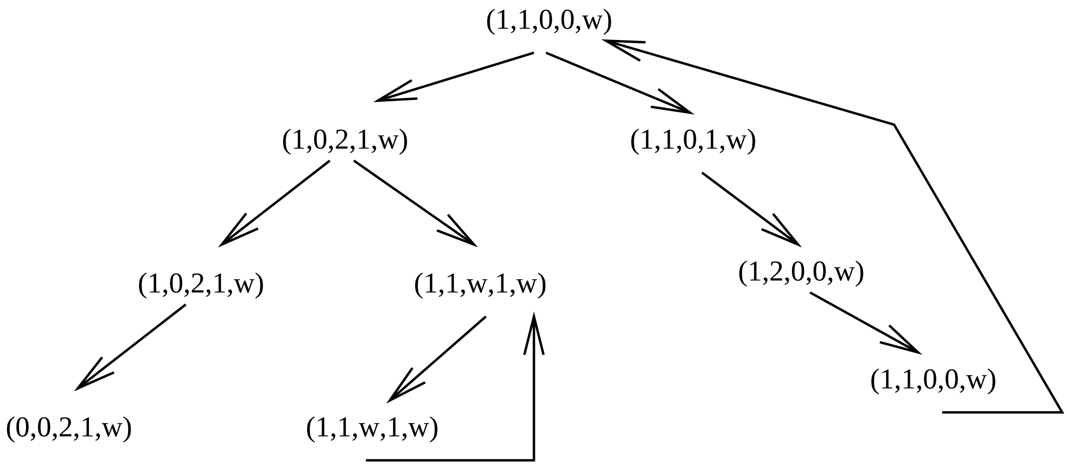
[10]

SOLUTION

A TTPN with two input places and two transitions, one with a weighting of T=1, or 0 (cars can proceed directly), and one with a T=exp(lambda), (indicating that traffic can proceed in an exponentially distributed set of entry slots).

Question 3

Given the following reachability tree; can the system with this tree achieve deadlock?



[4]

SOLUTION

Yes the tree has a leaf node, and thus the system has the potential to enter a deadlock state.

Question 4

List the problems inherent in multi-stage interconnection networks that are partially solved by introducing combining?

SOLUTION

Congestion, hotspot formation, tree saturation

[3]

Question 5

Why is the following code poorly written?

What is the name of the problem demonstrated by this code?

What feature of the scheduling of data operations in parallel programming environments allows this problem to occur?

```
cobegin
    x:= x + 1;
[redacted]
    x:= x + 1;
coend
```

[6]

SOLUTION

The program is poorly written because it modifies a shared variable.

The problem is called a data race, or interference condition.

This is caused by scheduling non-determinism wrt data operations.

Question 6

State two common measures of network desirability mentioned in the the lectures. Provide formulae for these measures for arbitary size torus networks and a MIN with an arbitary number of inputs and outputs. An example MIN is the Omega network shown in the lectures.

[9]

SOLUTION

Common measures are node order, and net diameter.

[2]

Torus, Order= 4, Diameter= floor(X-dim/2) + floor(Y-dim/2)

[3]

MIN is a constant path length network, and so the order of the switch is 4 and the Diameter is the input to output path length = log2(numb_outputs)

[4]

Question 7

Do SIMD machines use control or data parallelism?

[2]

SOLUTION

Data parallelism through vectorisation.

Question 8

Does MIMD programming provide fine or coarse grain parallelism? Justify your answer.

[6]

SOLUTION

In most cases coarse grain parallelism, since the threads, parallel procedures and parallel loops rely on a large amount of computation to offset the overhead of establishing control flow parallelism.

Question 9

What does a taxonomy of computer architectures aim to achieve?

[2]

SOLUTION

The ability to separate computer systems into useful categories on the basis of their structure and programming characteristics.

Upon what basis does Flynn form the separation of parallel computer systems in his taxonomy.

[4]

SOLUTION

Flynn observes that all processing systems apply instructions to data items. The ability to handle multiples of such streams is what generates parallelism, and thus can be used to classify all parallel computers.

Question 10

Recall that data operations on the DAP can move data on the machine's processor array. Given an 8x8 DAP and the following declarations and initialisations, state the value stored by each of the following statements.

[6]

```
REAL V1(), V2(), V3();

V1= (1, 3, 4, 5, 2, 3, 6, 9)

V2= SHLP(V1,3) + V1()
V3= SHRC(V2, 2) - V1(-)
```

SOLUTION

SHLP(V1, 3) = (5, 2, 3, 6, 9, 0, 0, 0) [1]
V1= (1, 3, 4, 5, 2, 3, 6, 9)

Thus V2 = (6, 5, 7, 11, 11, 3, 6, 9) [2]

SHRC(V2, 2) = (6, 9, 6, 5, 7, 11, 11, 3) [2]
V1(-) = (3, 4, 5, 2, 3, 6, 9, 1)

Thus V3 = (3, 5, 1, 3, 4, 5, 2, 2) [1]

Question 11

Define speedup, and maximum achievable speedup. Use diagrams if necessary.

SOLUTION

One mark each for...

Speedup= Best seq time / k parallel time.

Max Achievable Speedup= Seq time/ inherently seq time.

[2]

Section B
Topic Questions
(Answer TWO questions from this section)

Question 12

A commercial transportation company maintains a fleet of trucks with which to transport customer packages between three major production centres (we will call the production centres A, B and C). Trucks leave A on a regular schedule 7 time units apart, but their destination is exponentially distributed between B and C. Trucks leaving B for A and C leave intermittently, and their frequency of departure is described by a function τ_f. These trucks are allocated to alternating destinations by a dispatcher. Trucks leave C for other destinations every 2 time units and proceed to destinations A and B in a ratio 1:2. That is, two trucks depart for B for every truck that departs for A. Assume the following initial configuration of trucks.

$$A = 40, B = 20, C = 90$$

Explain (in a couple of paragraphs) which of TTPN, PTPN or standard petri nets is best suited to modelling this system. Justify your answer.

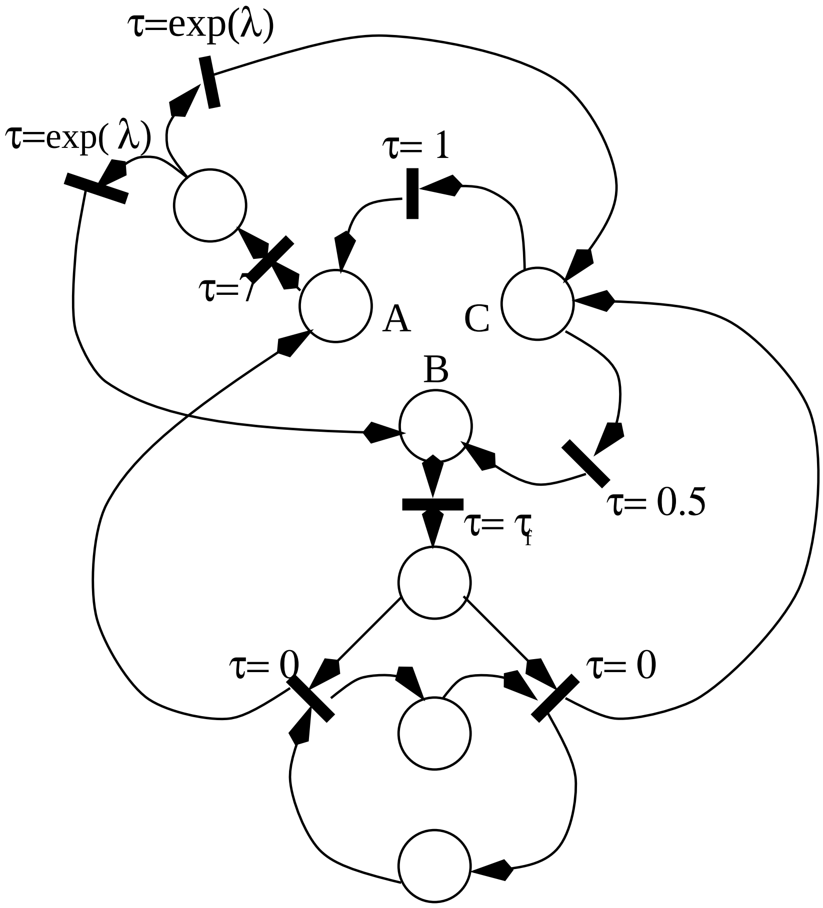
[6]

SOLUTION

I would model the locations as places and the trucks as tokens. This then leaves timed transitions to model the distribution of trucks to destinations. Thus, TTPN, since all the temporal activity lies in the transitions between states.

Draw and annotate an appropriate petri net model of the system outlined above.

SOLUTION



[24]

Question 13

What defines an inherently sequential algorithm?

SOLUTION

Temporal dependency between all steps from 1st to last for correctness of result. Loops which have temporal dependencies between sucessive iterations.

[2]

Give a code example that illustrates the nature of inherently sequential code.

SOLUTION

Selection sort

[4]

Say why your code example is inherently sequential.

SOLUTION

selection sort in the ith iteration assumes that the i previous smallest/largest numbers have been selected and appended/prepended to a list of sorted values. The additions to the sorted list must occur in temporal order congruent with the loop index number for the sort to operate correctly.

[4]

Use the number of data transactions on the bus and the concept of data transformations to define an optimal sequential program.

SOLUTION

An optimal seq prog transforms the source data set into the solution data set in the minimum possible number of bus transactions.

[4]

Use the concept of data partitioning and synchronisation to define the notion of an optimal parallel program.

SOLUTION

An optimal parallel program has perfect data partitioning, does no redundant work, and requires no synchronisation.

[4]

Write a simple k-parallel procedure in Modula-P which sorts an input array and writes the sorted results into an output array. Assume that the procedure is to be executed on a machine with up to 20 processing elements, and that 256K unique integer values are to be sorted. Your procedure must work when called for degrees of parallelism from 1 (sequential) to 20 (most parallel).

SOLUTION

K-parallel rank sort. Divide the numbers into K sets, rank sort each set and deposite the results in the output array. No synchronisation is required since all values have a unique final position in the sorted sequence.

[12]

Question 14

What are the hardware options for implementing data flow programs? List three different alternatives and discuss how each of them deals with recursion. Is recursion possible for all the alternatives that you have described?

[6]

SOLUTION

Logical circuit implementation directly in hardware. There is no support for recursion.

Programmable logic arrays, can implement multiple programs, but are not able to reconfigure themselves at runtime to provide support for recursion.

General purpose dataflow cf. the Manchester machine, provides a general data flow execution environment, which fully supports procedures and recursion.

What is the most important restriction that must be applied to imperative languages to allow them to generate a data flow graph. Think of SISAL when framing your answer.

[2]

SOLUTION

Single assignment

The restriction you have identified above presents difficulties when designing loop commands for data flow languages. What language feature does SISAL introduce to address these difficulties?

[4]

SOLUTION

SISAL introduces the concept of an **old** *version of a variable, which can be used in restricted contexts to access the previous binding to that name in SISAL. The function old can be composed with itself, to render statements like old(old(x)) meaningful.*

Draw a data flow graph that demonstrates the parallelism inherent in the following computations.

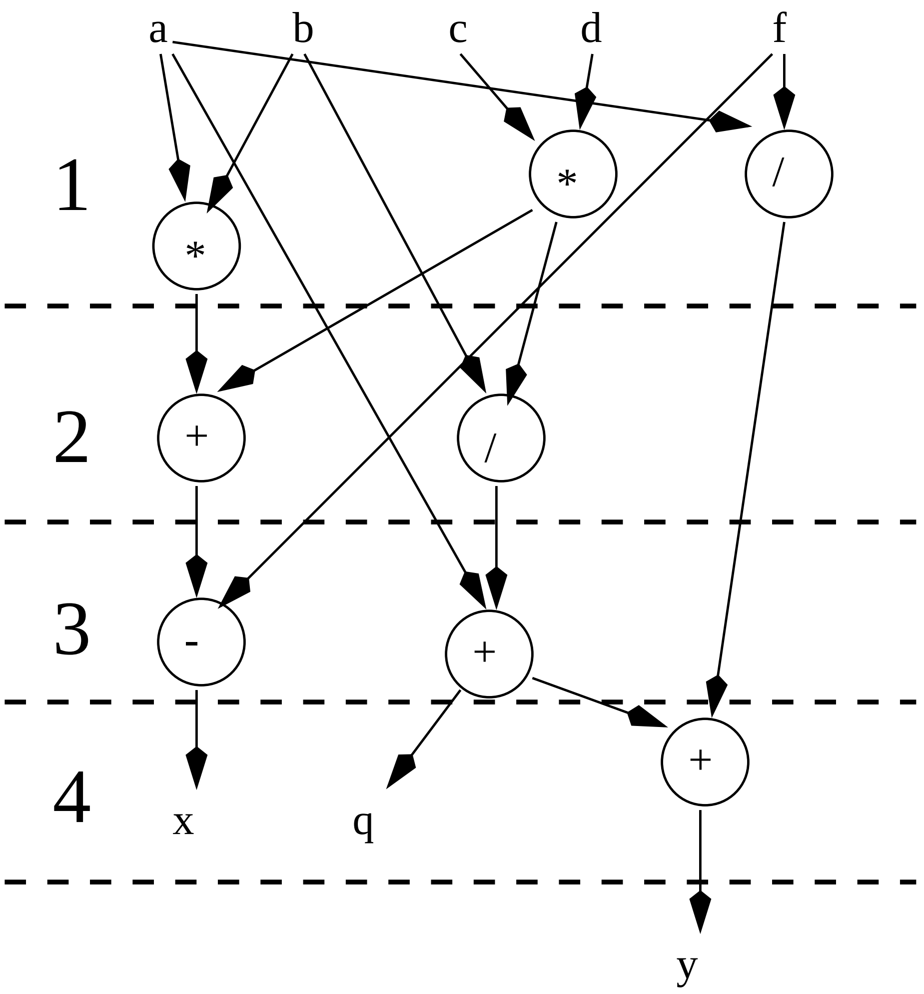
$$x = a * b + c * d - f$$

$$q = a + b/(c * d)$$

$$y = q + a/f$$

[9]

SOLUTION



Assume that all these operations take 1 clock cycle to execute, and that data transmission is instantaneous. How many clock cycles are required to complete all these computations?

SOLUTION

4 cycles.

[2]

What is the principal bottleneck in the Manchester data flow computer architecture?

[2]

SOLUTION

The matching unit. In particular the interaction between the matching unit and the overflow store when programs become large.

Name the three types of dataflow nodes which must be introduced to handle loops. Draw a picture of each node type, labelling its inputs and outputs and indicating their data types

OVER/

(Eg. boolean or normal data)

[5]

SOLUTION

Boolean test node, data in, boolean out.

[1]

Switch node, boolean and data in, data switched between outputs depending on the boolean.

[2]

Join node, two data inputs, and one interleaved data output stream.

[2]