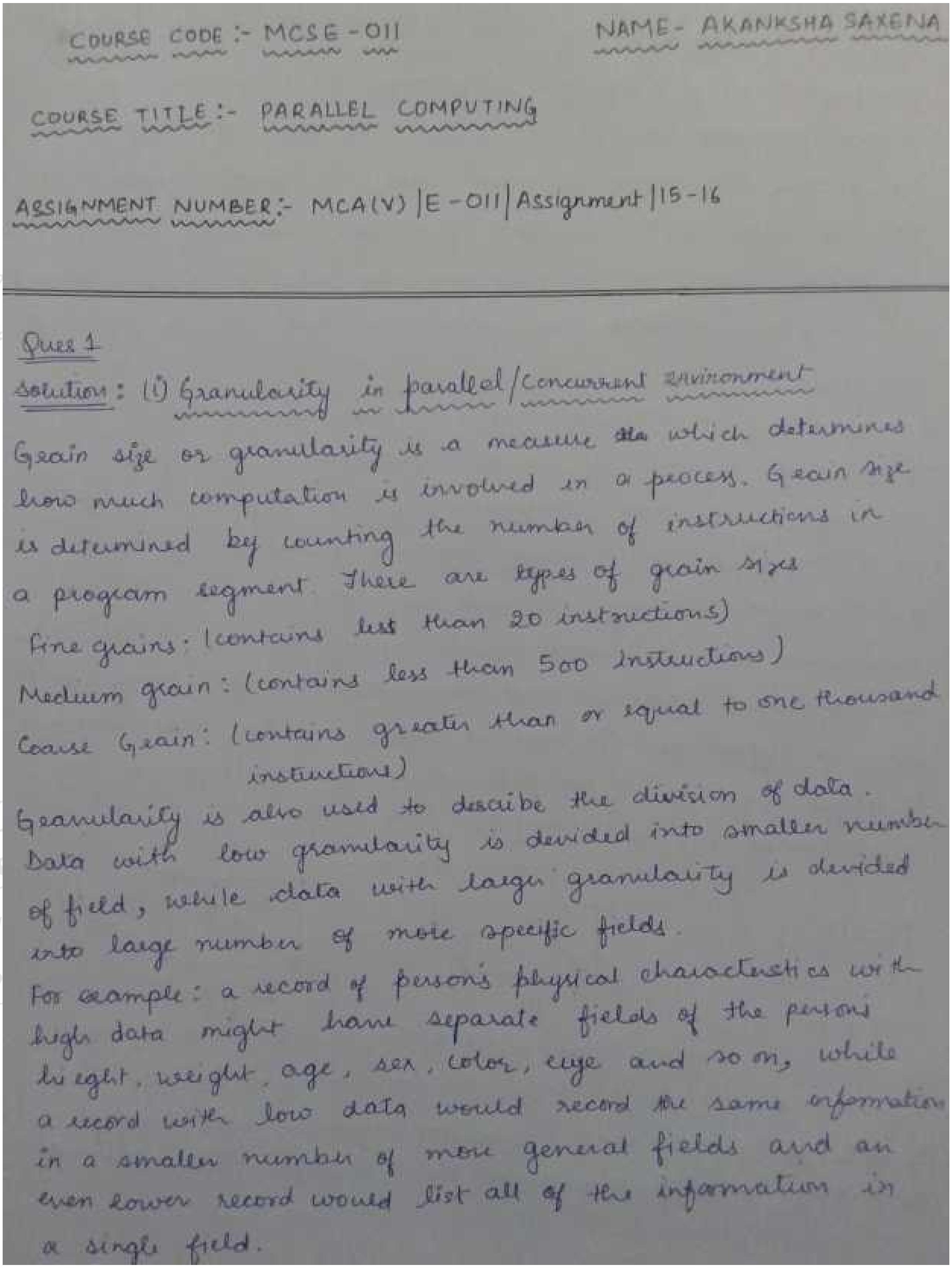
thanx to Akanksha

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i come up with. Please read carefully before writing, I am not responsible for anything th all.



COURSE CODE :- MCSE-011

NAME- AKANKSHA SAXENA

COURSE TITLE :- PARALLEL COMPUTING

ASSIGNMENT NUMBER :- MCA(V)|E-011|Assignment|15-16

Ques 1

Solution : (i) Granularity in parallel/Concurrent environment

Grain size or granularity is a measure ~~the~~ which determines how much computation is involved in a process. Grain size is determined by counting the number of instructions in a program segment There are types of grain sizes

fine grains: (contains less than 20 instructions)

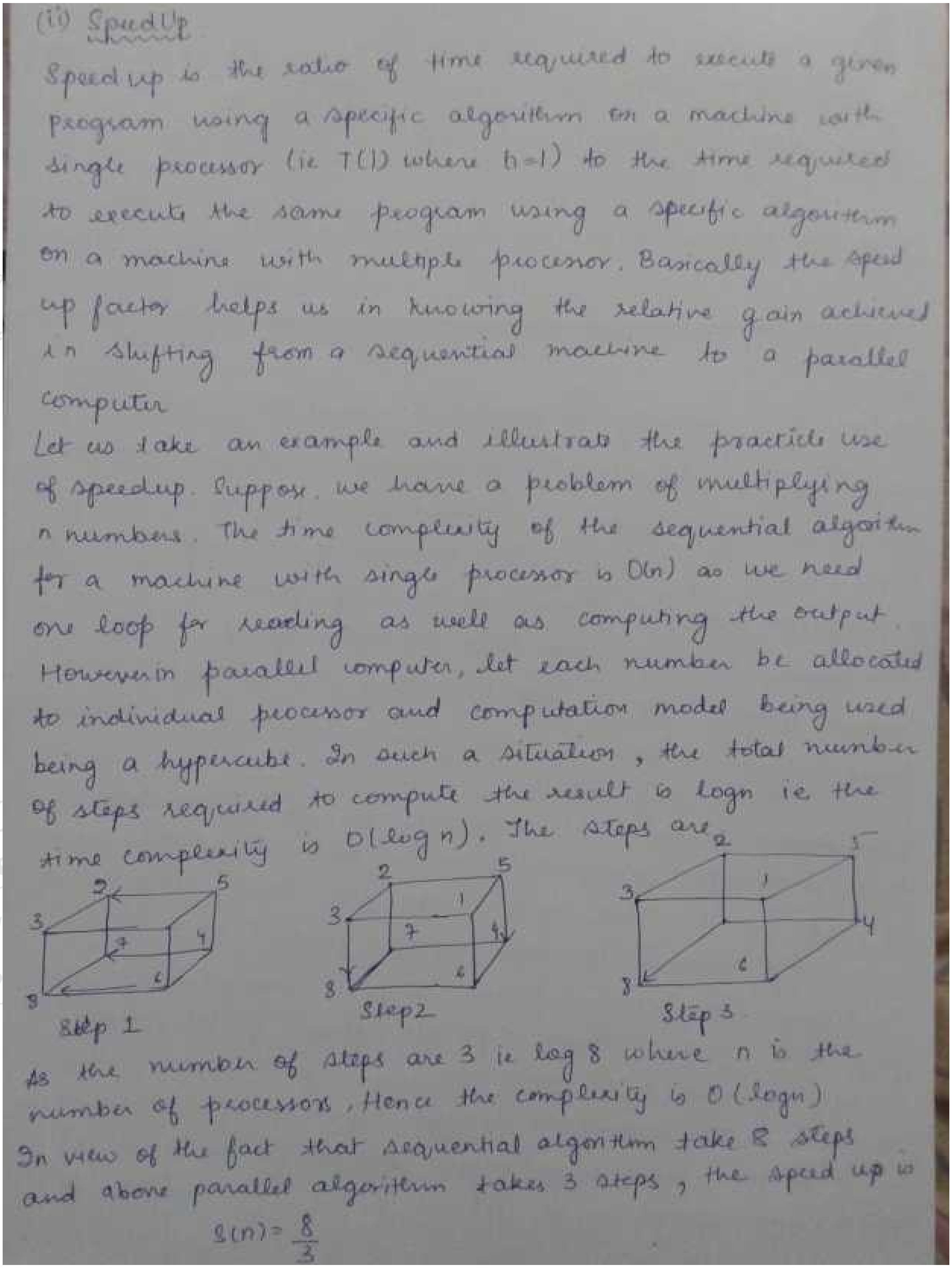
Medium grain: (contains less than 500 Instructions)

Coarse Grain: (contains greater than or equal to one thousand instructions)

Granularity is also used to describe the division of data. Data with low granularity is devided into smaller number of field, while data with large granularity is devided into large number of more specific fields.

For example: a record of person's physical characteristics with high data might have separate fields of the persons height, weight, age, sex, color, eye and so on, while a record with low data would record the same information in a smaller number of more general fields and an even lower record would list all of the information in a single field.

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i come up with. Please read carefully before writing, I am not responsible for anything th all.



(ii) Speed Up

Speed up is the ratio of time required to execute a given program using a specific algorithm on a machine with single processor (ie T(1) where n=1) to the time required to execute the same program using a specific algorithm on a machine with multiple processor. Basically the speed up factor helps us in knowing the relative gain achieved in shifting from a sequential machine to a parallel computer

Let us take an example and illustrate the practicle use of speedup. Suppose we have a problem of multiplying n numbers. The time complexity of the sequential algorithm for a machine with single processor is O(n) as we need one loop for reading as well as computing the output. However in parallel computer, let each number be allocated to individual processor and computation model being used being a hypercube. In such a situation, the total number of steps required to compute the result is logn ie the time complexity is O(log n). The steps are

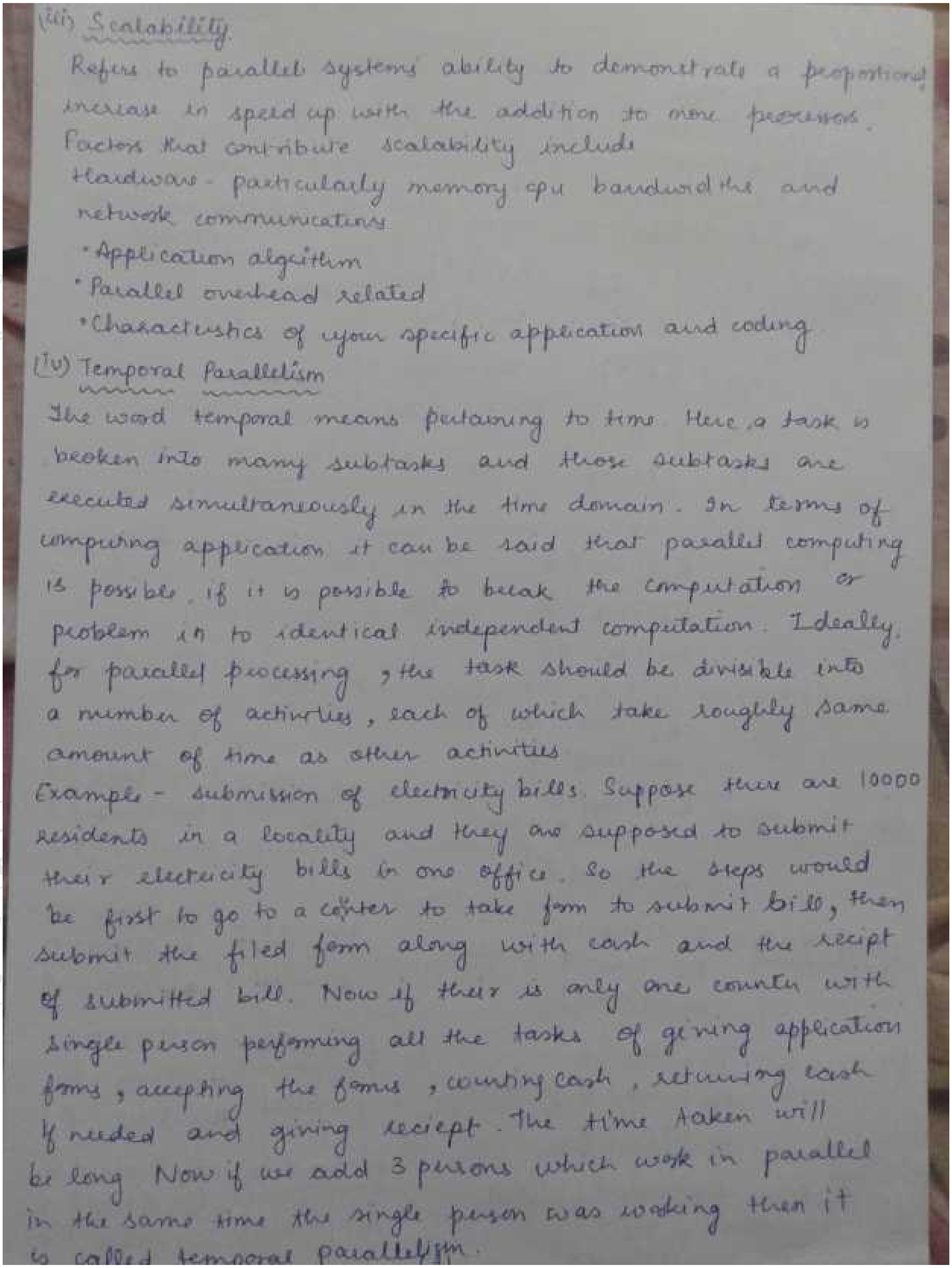
As the number of steps are 3 ie log 8 where n is the number of processors, Hence the complexity is O(logn)

In view of the fact that sequential algorithm take 8 steps and above parallel algorithm takes 3 steps, the speed up is

$$S(n) = \frac{8}{3}$$

Thanx to all of you, i m now in malaysia..
btw i have a question "would u like to be a friendship with me?"
yaar bahut lonely ho gya hu guyz :):)

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i
come up with. Please read carefully before writing, I am not responsible for anything th
all.



(iii) Scalability

Refers to parallel systems' ability to demonstrate a proportional increase in speed up with the addition to more processors. Factors that contribute scalability include

Hardware - particularly memory cpu bandwidth and network communications

- Application algorithm
- Parallel overhead related
- Characteristics of your specific application and coding

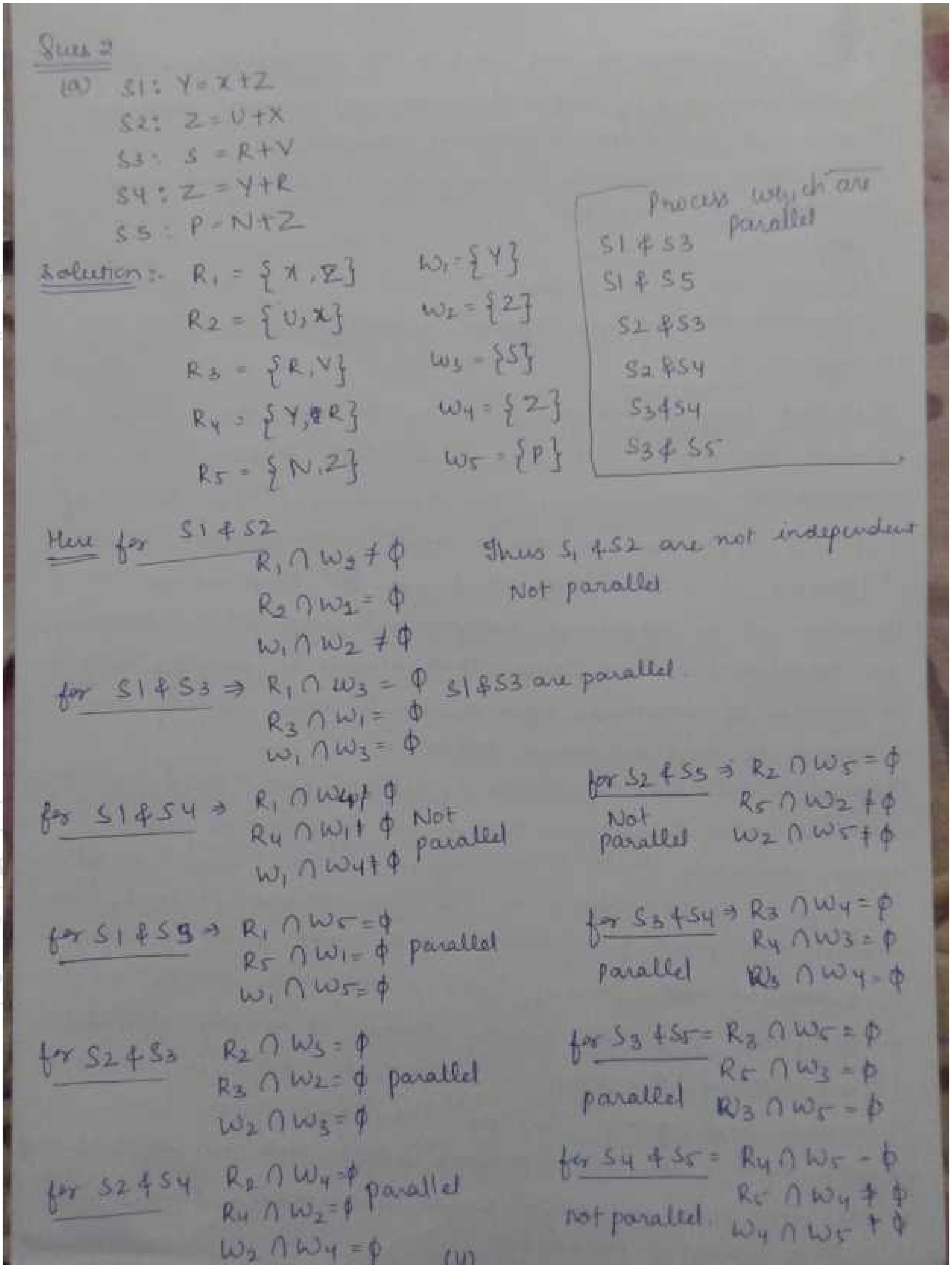
(iv) Temporal Parallelism

The word temporal means pertaining to time. Here, a task is broken into many subtasks and those subtasks are executed simultaneously in the time domain. In terms of computing application it can be said that parallel computing is possible. if it is possible to break the computation or problem in to identical independent computation. Ideally, for parallel processing, the task should be divisible into a number of activities, each of which take roughly same amount of time as other activities

Example - submission of electricity bills. Suppose there are 10000 residents in a locality and they are supposed to submit their electricity bills in one office. So the steps would be first to go to a center to take form to submit bill, then submit the filed form along with cash and the reicpt of submitted bill. Now if their is only one counter with single person performing all the tasks of giving application forms, accepting the forms, counting cash, returning cash if needed and giving reciept. The time taken will be long. Now if we add 3 persons which work in parallel in the same time the single person was working then it is called temporal parallelism.

thanx to Akanksha

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i come up with. Please read carefully before writing, I am not responsible for anything th all.



Ques 2

(a) S1: $Y = X + Z$

S2: $Z = U + X$

S3: $S = R + V$

S4: $Z = Y + R$

S5: $P = N + Z$

Solution:- $R_1 = \{X, Z\}$ $W_1 = \{Y\}$

$R_2 = \{U, X\}$ $W_2 = \{Z\}$

$R_3 = \{R, V\}$ $W_3 = \{S\}$

$R_4 = \{Y, R\}$ $W_4 = \{Z\}$

$R_5 = \{N, Z\}$ $W_5 = \{P\}$

Process which are parallel

S1 & S3

S1 & S5

S2 & S3

S2 & S4

S3 & S4

S3 & S5

Here for S1 & S2

$R_1 \cap W_2 \neq \phi$

$R_2 \cap W_1 = \phi$

$W_1 \cap W_2 \neq \phi$

Thus S_1 & S2 are not independent

Not parallel

for S1 & S3 ⇒ $R_1 \cap W_3 = \phi$ S1 & S3 are parallel

$R_3 \cap W_1 = \phi$

$W_1 \cap W_3 = \phi$

for S1 & S4 ⇒ $R_1 \cap W_4 \neq \phi$

$R_4 \cap W_1 \neq \phi$ Not parallel

$W_1 \cap W_4 \neq \phi$

for S2 & S5 ⇒ $R_2 \cap W_5 = \phi$

$R_5 \cap W_2 \neq \phi$

$W_2 \cap W_5 \neq \phi$

Not parallel

for S1 & S5 ⇒ $R_1 \cap W_5 = \phi$

$R_5 \cap W_1 = \phi$ parallel

$W_1 \cap W_5 = \phi$

for S3 & S4 ⇒ $R_3 \cap W_4 = \phi$

$R_4 \cap W_3 = \phi$

$R_3 \cap W_4 = \phi$

parallel

for S2 & S3 $R_2 \cap W_3 = \phi$

$R_3 \cap W_2 = \phi$ parallel

$W_2 \cap W_3 = \phi$

for S3 & S5 = $R_3 \cap W_5 = \phi$

$R_5 \cap W_3 = \phi$

$R_3 \cap W_5 = \phi$

parallel

for S2 & S4 $R_2 \cap W_4 = \phi$

$R_4 \cap W_2 = \phi$ parallel

$W_2 \cap W_4 = \phi$

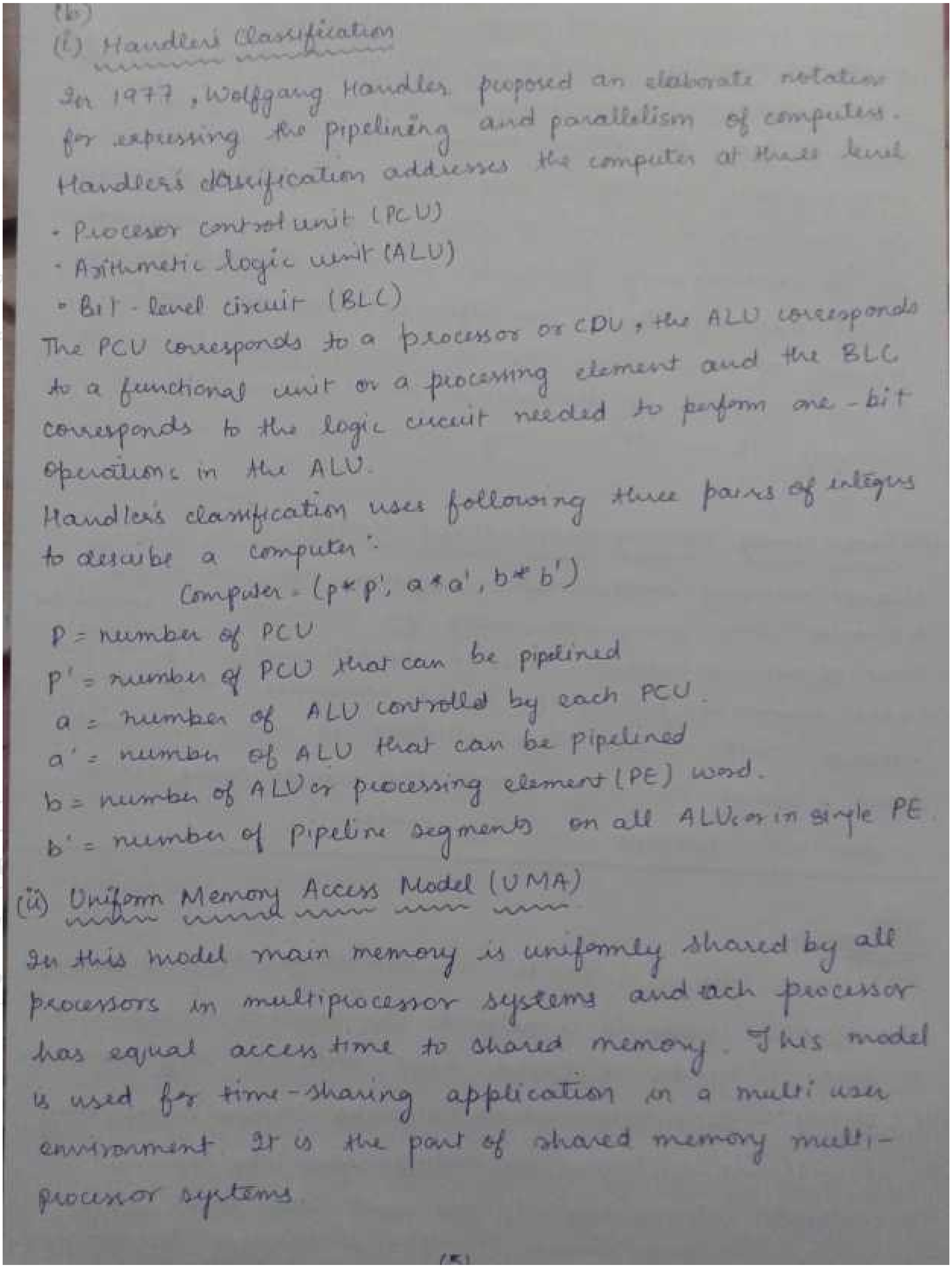
for S4 & S5 = $R_4 \cap W_5 = \phi$

$R_5 \cap W_4 \neq \phi$

$W_4 \cap W_5 \neq \phi$

not parallel

(ii)



(b)

(i) Handler's Classification

In 1977, Wolfgang Handler proposed an elaborate notation for expressing the pipelining and parallelism of computers. Handler's classification addresses the computer at three level

- Processor control unit (PCU)
- Arithmetic logic unit (ALU)
- Bit-level circuit (BLC)

The PCU corresponds to a processor or CPU, the ALU corresponds to a functional unit or a processing element and the BLC corresponds to the logic circuit needed to perform one-bit operations in the ALU.

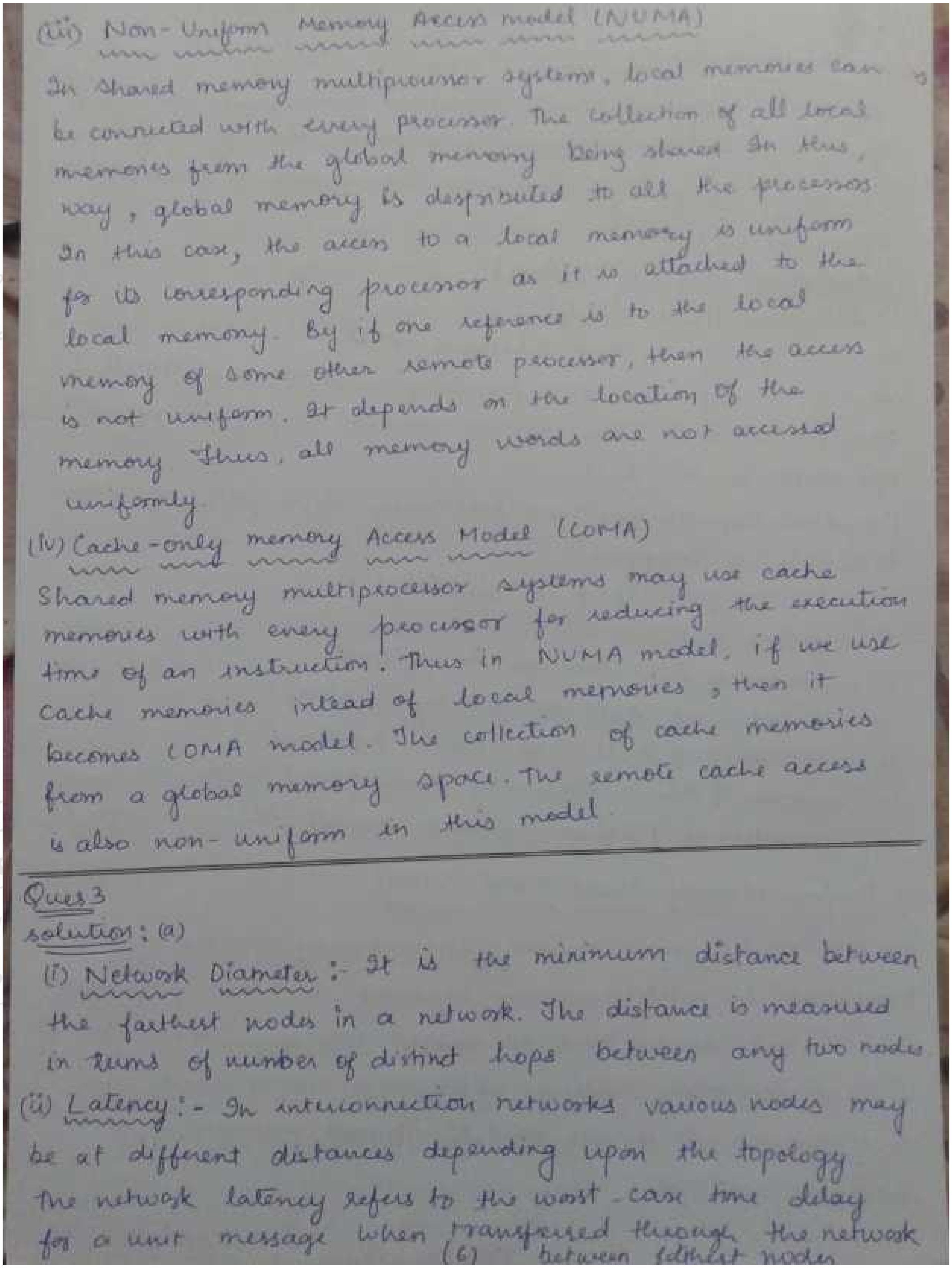
Handler's classification uses following three pairs of integers to describe a computer:

Computer = (p*p', a*a', b*b')

p = number of PCU
p' = number of PCU that can be pipelined
a = number of ALU controlled by each PCU.
a' = number of ALU that can be pipelined
b = number of ALU or processing element (PE) word.
b' = number of pipeline segments on all ALUs or in single PE.

(ii) Uniform Memory Access Model (UMA)

In this model main memory is uniformly shared by all processors in multiprocessor systems and each processor has equal access time to shared memory. This model is used for time-sharing application in a multi user environment. It is the part of shared memory multi-processor systems.



(iii) Non-Uniform Memory Access model (NUMA)

In shared memory multiprocessor systems, local memories can be connected with every processor. The collection of all local memories from the global memory being shared. In this way, global memory is desptributed to all the processors. In this case, the access to a local memory is uniform for its corresponding processor as it is attached to the local memory. By if one reference is to the local memory of some other remote processor, then the access is not uniform. It depends on the location of the memory. Thus, all memory words are not accessed uniformly

(IV) Cache-only memory Access Model (COMA)

Shared memory multiprocessor systems may use cache memories with every processor for reducing the execution time of an instruction. Thus in NUMA model, if we use cache memories intead of local memories, then it becomes COMA model. The collection of cache memories from a global memory space. The remote cache access is also non-uniform in this model

Ques 3

solution: (a)

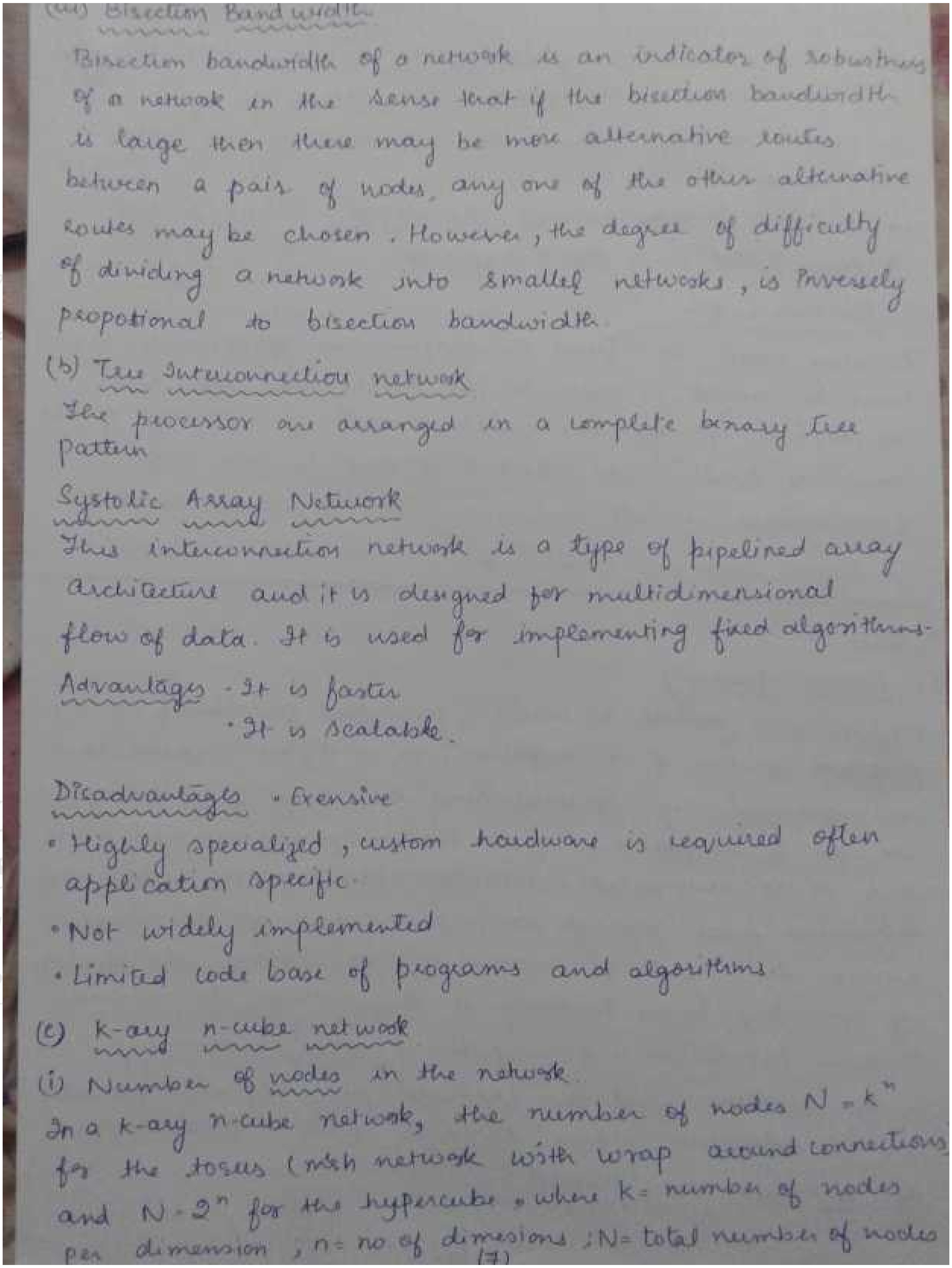
(i) Network Diameter :- It is the minimum distance between the farthest nodes in a network. The distance is measured in terms of number of distinct hops between any two nodes

(ii) Latency :- In interconnection networks various nodes may be at different distances depending upon the topology. The network latency refers to the worst-case time delay for a unit message when transferred through the network between fathest nodes

btw i have a question "would u like to be a friendship with me?"

thanx to Akanksha

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i
come up with. Please read carefully before writing, I am not responsible for anything th
all.



(iii) Bisection Bandwidth

Bisection bandwidth of a network is an indicator of sobustness of a network in the sense that if the bisection bandwidth is large then there may be more alternative routes between a pair of nodes, any one of the other alternative routes may be chosen. However, the degree of difficulty of dividing a network into smaller networks, is inversely propotional to bisection bandwidth.

(b) Tree Interconnection network

The processor are arranged in a complete binary tree pattern

Systolic Array Network

This interconnection network is a type of pipelined array architecture and it is designed for multidimensional flow of data. It is used for implementing fixed algorithms.

Advantages · It is faster
· It is scalable.

Disadvantages · Exensive

- Highly specialized, custom hardware is required often application specific.
- Not widely implemented
- Limited code base of programs and algorithms

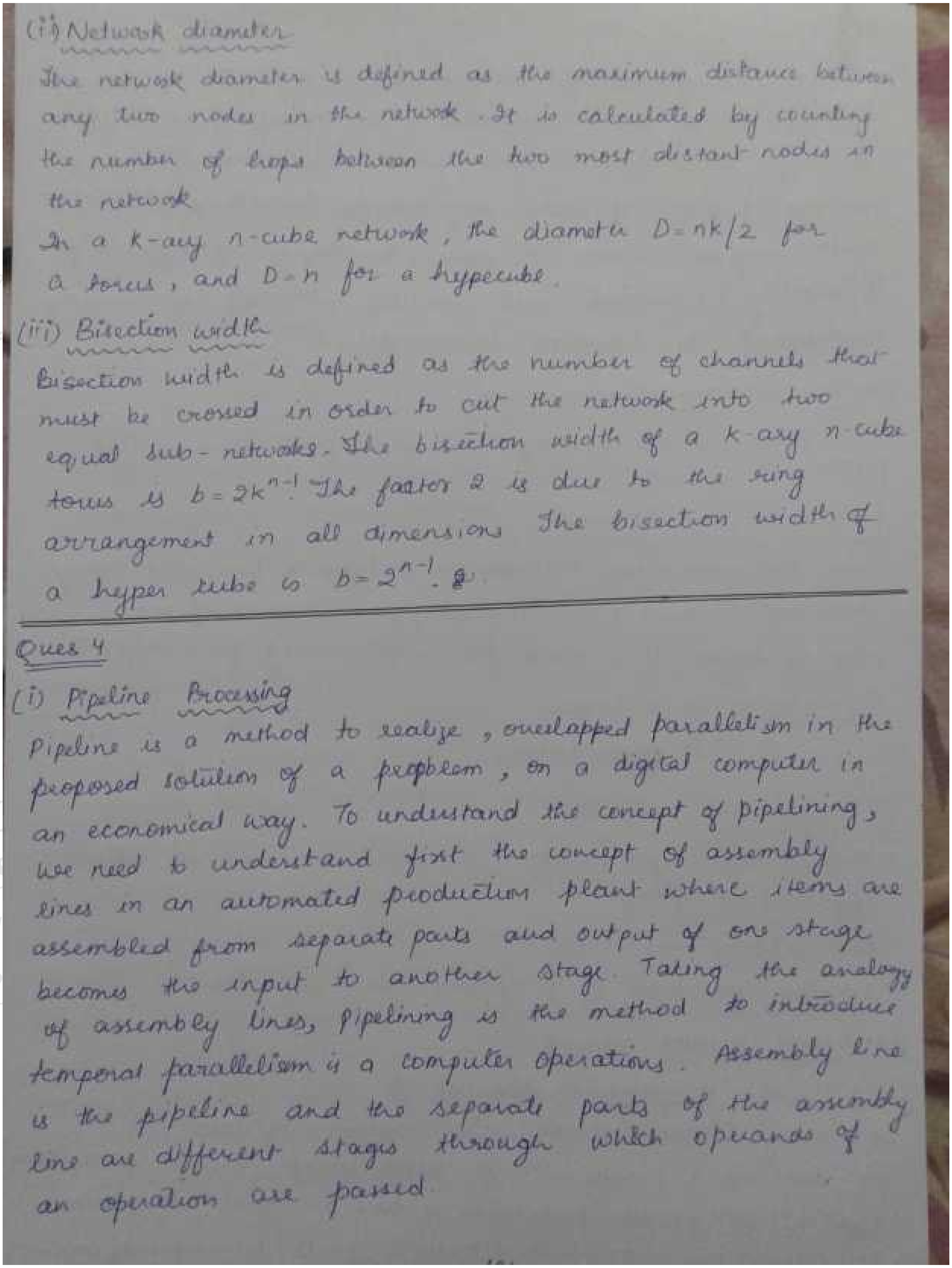
(c) k-ary n-cube network

(i) Number of nodes in the network

In a k-ary n-cube network, the number of nodes $N = k^n$ for the torus (mesh network with wrap around connections) and $N = 2^n$ for the hypercube, where k = number of nodes per dimension, n = no of dimensions; N = total number of nodes

Thanx to all of you, i m now in malaysia..
btw i have a question "would u like to be a friendship with me?"
yaar bahut lonely ho gya hu guyz :):)

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i come up with. Please read carefully before writing, I am not responsible for anything th all.



(ii) Network diameter

The network diameter is defined as the maximum distance between any two nodes in the network. It is calculated by counting the number of hops between the two most distant nodes in the network

In a k-ary n-cube network, the diameter $D = nk/2$ for a torus, and $D = n$ for a hypercube.

(iii) Bisection width

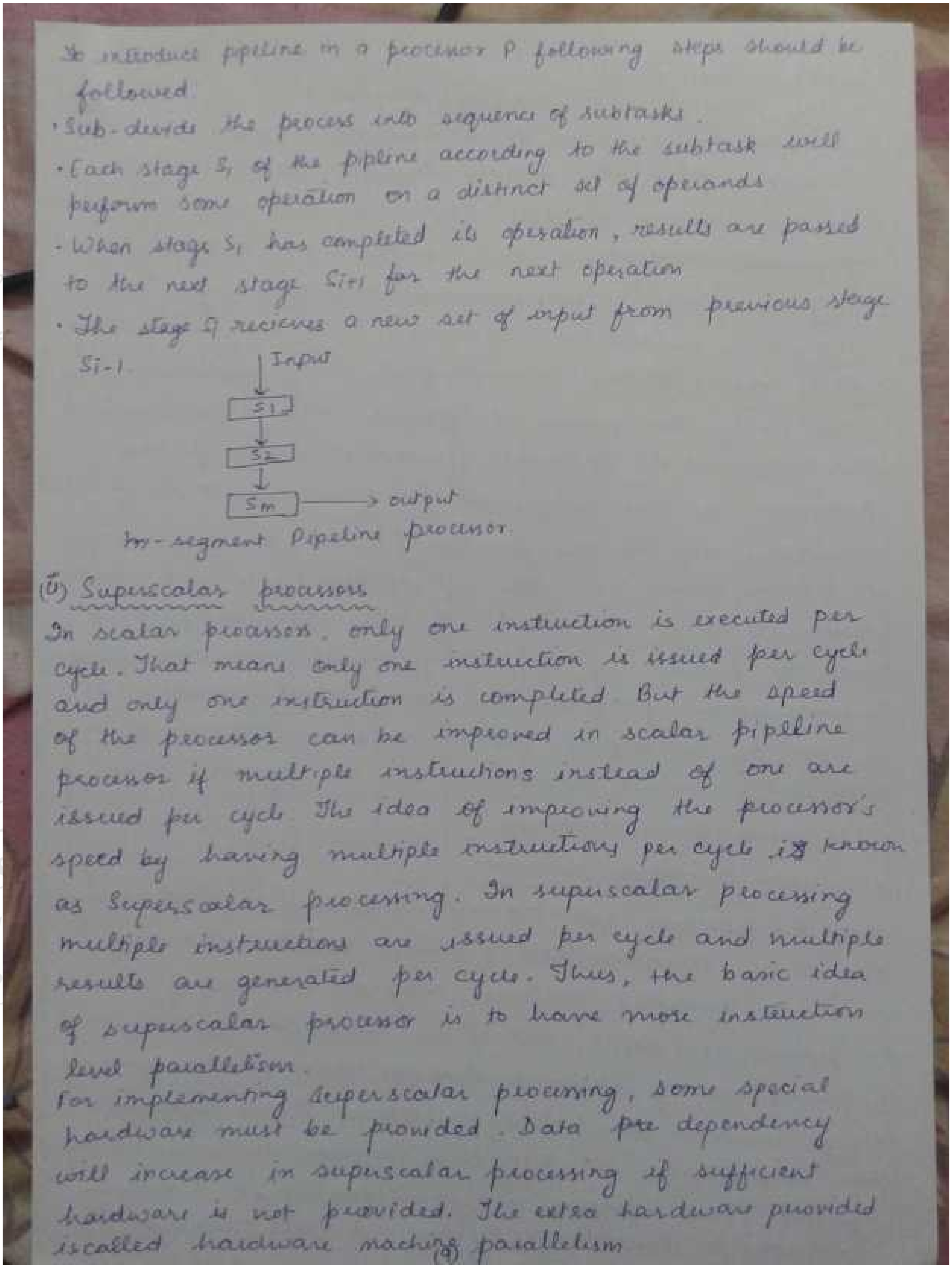
Bisection width is defined as the number of channels that must be crossed in order to cut the network into two equal sub-networks. The bisection width of a k-ary n-cube torus is $b = 2k^{n-1}$. The factor 2 is due to the ring arrangement in all dimensions The bisection width of a hyper cube is $b = 2^{n-1} \cdot 2$

Ques 4

(i) Pipeline Processing

Pipeline is a method to realize, overlapped parallelism in the proposed solution of a propblem, on a digital computer in an economical way. To understand the concept of pipelining, we need to understand first the concept of assembly lines in an automated production plant where items are assembled from separate parts and output of one stage becomes the input to another stage. Taking the analogy of assembly lines, Pipelining is the method to introduce temporal parallelism is a computer operations. Assembly line is the pipeline and the separate parts of the assembly line are different stages through which operands of an operation are passed

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i come up with. Please read carefully before writing, I am not responsible for anything th all.



To introduce pipline in a processor P following steps should be followed:

- Sub-divide the process into sequence of subtasks.
- Each stage S_i of the pipline according to the subtask will perform some operation on a distinct set of operands
- When stage S_i has completed its operation, results are passed to the next stage S_{i+1} for the next operation
- The stage S_i recieves a new set of input from previous stage S_{i-1}.

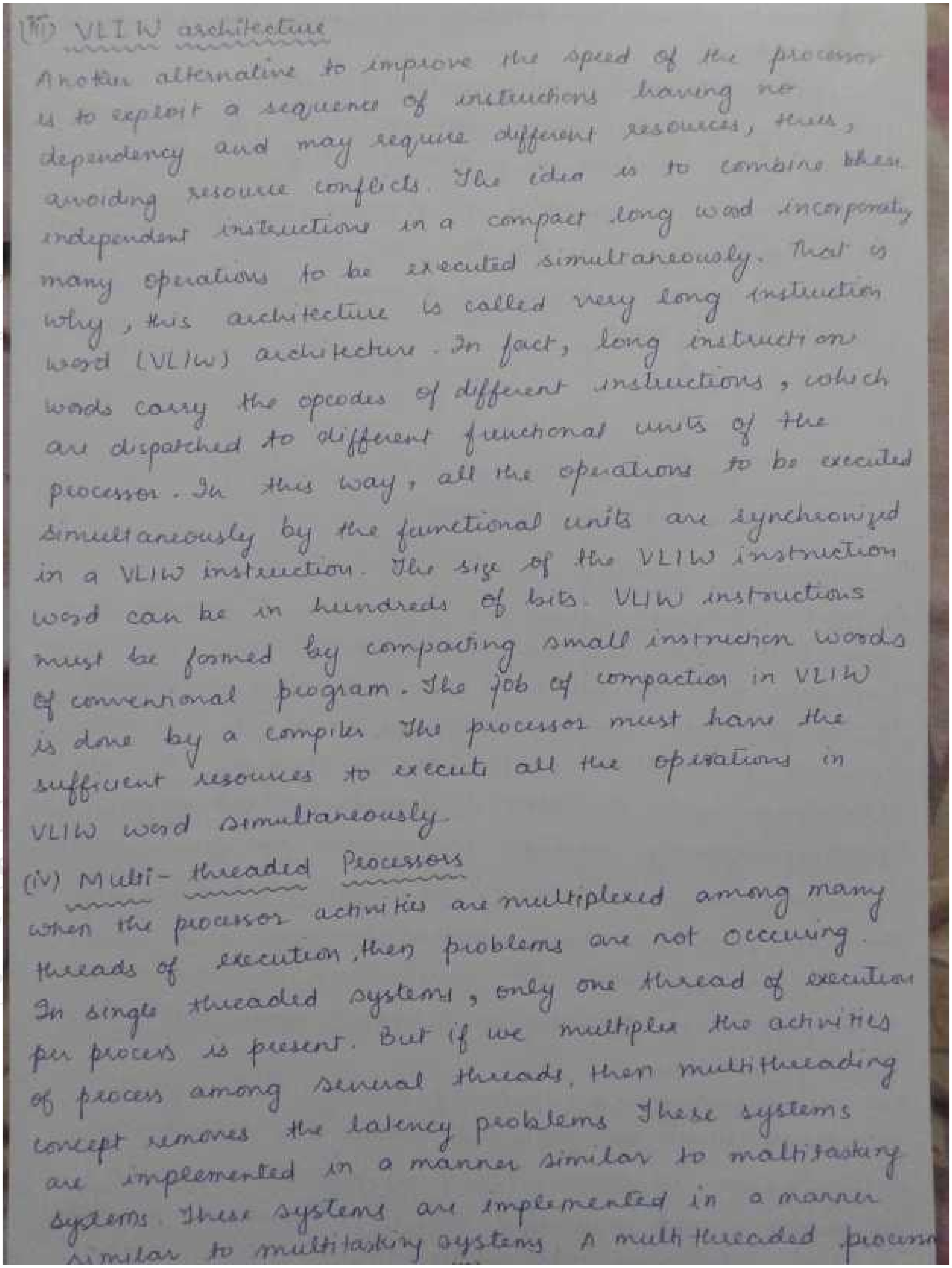
m-segment Pipeline processor

(ii) Superscalar processors

In scalar processor, only one instruction is executed per cycle. That means only one instruction is issued per cycle and only one instruction is completed. But the speed of the processor can be improved in scalar pipeline processor if multiple instructions instead of one are issued per cycle. The idea of improving the processor's speed by having multiple instructions per cycle is known as Superscalar processing. In superscalar processing multiple instructions are issued per cycle and multiple results are generated per cycle. Thus, the basic idea of superscalar processor is to have more instruction level parallelism.

For implementing superscalar processing, some special hardware must be provided. Data pre dependency will increase in superscalar processing if sufficient hardware is not provided. The extra hardware provided iscalled hardware maching parallelism

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i
come up with. Please read carefully before writing, I am not responsible for anything th
all.



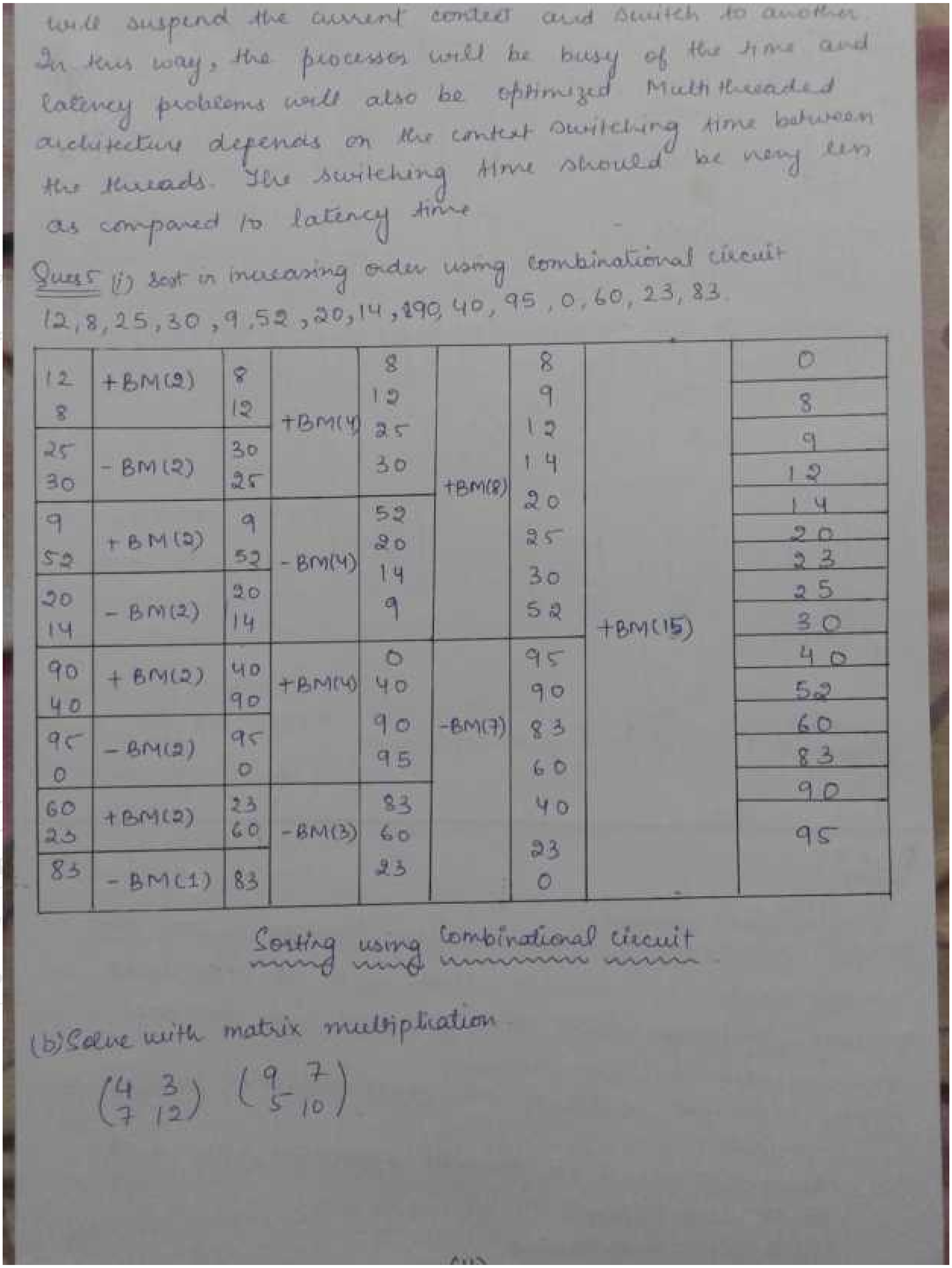
(iii) VLIW architecture

Another alternative to improve the speed of the processor
is to exploit a sequence of instructions having no
dependency and may require different resources, thus,
avoiding resource conflicts. The idea is to combine these
independent instructions in a compact long word incorporating
many operations to be executed simultaneously. That is
why, this architecture is called very long instruction
word (VLIW) architecture. In fact, long instruction
words carry the opcodes of different instructions, which
are dispatched to different functional units of the
processor. In this way, all the operations to be executed
simultaneously by the functional units are synchronized
in a VLIW instruction. The size of the VLIW instruction
word can be in hundreds of bits. VLIW instructions
must be formed by compacting small instruction words
of conventional program. The job of compaction in VLIW
is done by a compiler. The processor must have the
sufficient resources to execute all the operations in
VLIW word simultaneously.

(iv) Multi- threaded Processors

when the processor activities are multiplexed among many
threads of execution, then problems are not occuring.
In single threaded systems, only one thread of execution
per process is present. But if we multiplex the activities
of process among several threads, then multithreading
concept removes the latency problems. These systems
are implemented in a manner similar to multitasking
systems. These systems are implemented in a manner
similar to multitasking systems. A multi threaded processor

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i come up with. Please read carefully before writing, I am not responsible for anything th all.



will suspend the current context and switch to another In this way, the processor will be busy of the time and latency problems will also be optimized Multi threaded architecture depends on the context switching time between the threads. The switching time should be very less as compared to latency time

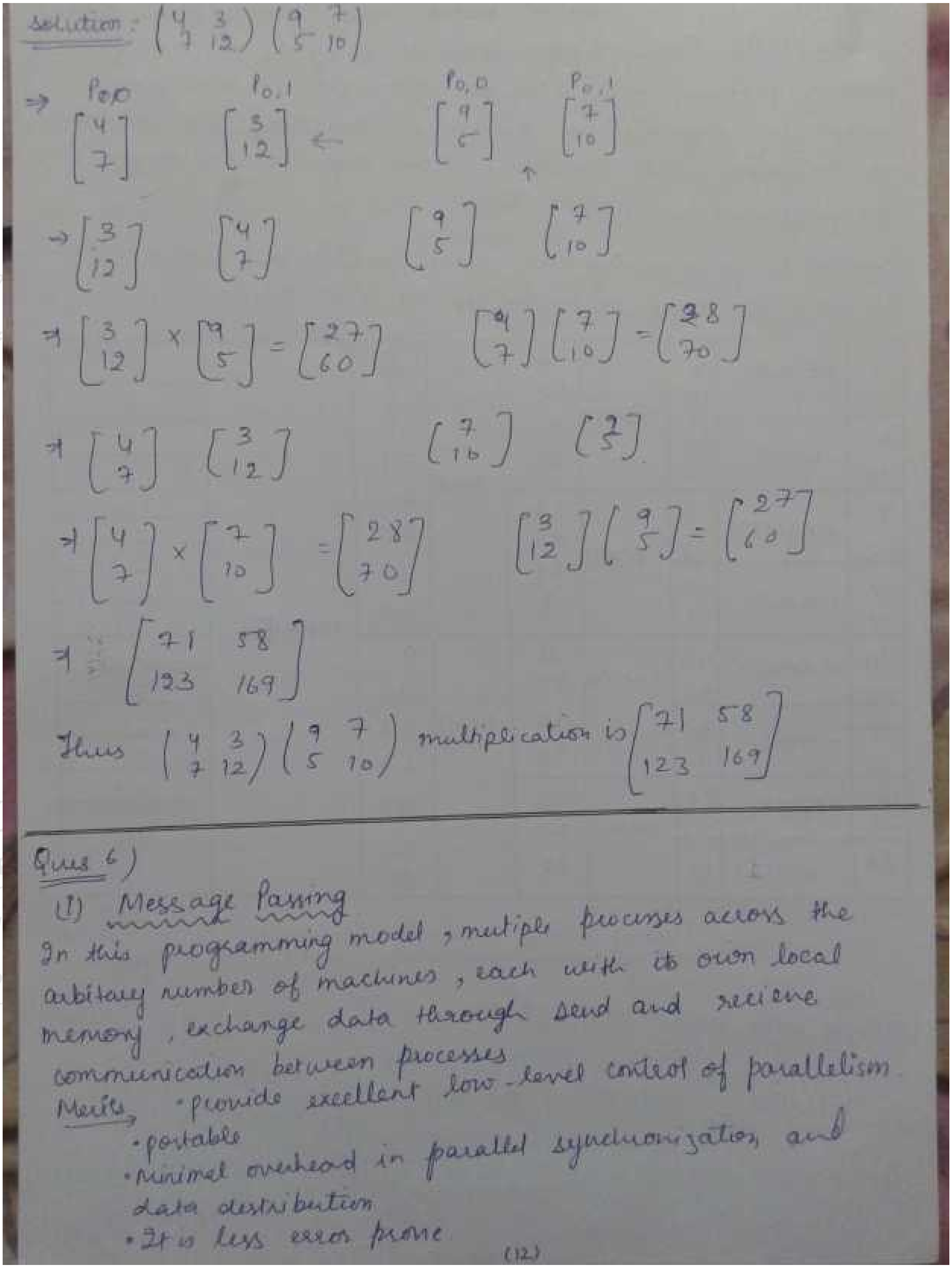
Ques 5 (i) Sort in increasing order using combinational circuit
12, 8, 25, 30, 9, 52, 20, 14, 890, 40, 95, 0, 60, 23, 83

12 8	+BM(2)	8 12	+BM(4)	8 12 25 30	+BM(8)	8 9 12 14 20 25 30 52	+BM(15)	0 8 9 12 14 20 23 25 30 40 52 60 83 90 95
25 30	- BM(2)	30 25						
9 52	+ BM(2)	9 52	- BM(4)	52 20 14 9				
20 14	- BM(2)	20 14						
90 40	+ BM(2)	40 90	+BM(4)	0 40 90 95	-BM(7)	95 90 83 60 40 23 0		
95 0	- BM(2)	95 0						
60 23	+BM(2)	23 60	- BM(3)	83 60 23				
83	- BM(1)	83						

Sorting using Combinational circuit

(b) Solve with matrix multiplication

$$\begin{pmatrix} 4 & 3 \\ 7 & 12 \end{pmatrix} \begin{pmatrix} 9 & 7 \\ 5 & 10 \end{pmatrix}$$



solution : $\begin{pmatrix} 4 & 3 \\ 7 & 12 \end{pmatrix} \begin{pmatrix} 9 & 7 \\ 5 & 10 \end{pmatrix}$

$\Rightarrow$ $P_{0,0}$ $\begin{bmatrix} 4 \\ 7 \end{bmatrix}$ $P_{0,1}$ $\begin{bmatrix} 3 \\ 12 \end{bmatrix} \leftarrow$ $P_{0,0}$ $\begin{bmatrix} 9 \\ 5 \end{bmatrix}$ $\uparrow$ $P_{0,1}$ $\begin{bmatrix} 7 \\ 10 \end{bmatrix}$

$\rightarrow \begin{bmatrix} 3 \\ 12 \end{bmatrix} \quad \begin{bmatrix} 4 \\ 7 \end{bmatrix} \qquad \begin{bmatrix} 9 \\ 5 \end{bmatrix} \quad \begin{bmatrix} 7 \\ 10 \end{bmatrix}$

$\rightarrow \begin{bmatrix} 3 \\ 12 \end{bmatrix} \times \begin{bmatrix} 9 \\ 5 \end{bmatrix} = \begin{bmatrix} 27 \\ 60 \end{bmatrix} \qquad \begin{bmatrix} 4 \\ 7 \end{bmatrix} \begin{bmatrix} 7 \\ 10 \end{bmatrix} = \begin{bmatrix} 28 \\ 70 \end{bmatrix}$

$\rightarrow \begin{bmatrix} 4 \\ 7 \end{bmatrix} \quad \begin{bmatrix} 3 \\ 12 \end{bmatrix} \qquad \begin{bmatrix} 7 \\ 10 \end{bmatrix} \quad \begin{bmatrix} 9 \\ 5 \end{bmatrix}$

$\rightarrow \begin{bmatrix} 4 \\ 7 \end{bmatrix} \times \begin{bmatrix} 7 \\ 10 \end{bmatrix} = \begin{bmatrix} 28 \\ 70 \end{bmatrix} \qquad \begin{bmatrix} 3 \\ 12 \end{bmatrix} \begin{bmatrix} 9 \\ 5 \end{bmatrix} = \begin{bmatrix} 27 \\ 60 \end{bmatrix}$

$\rightarrow \begin{bmatrix} 71 & 58 \\ 123 & 169 \end{bmatrix}$

Thus $\begin{pmatrix} 4 & 3 \\ 7 & 12 \end{pmatrix} \begin{pmatrix} 9 & 7 \\ 5 & 10 \end{pmatrix}$ multiplication is $\begin{bmatrix} 71 & 58 \\ 123 & 169 \end{bmatrix}$

Ques 6)

(I) Message Passing

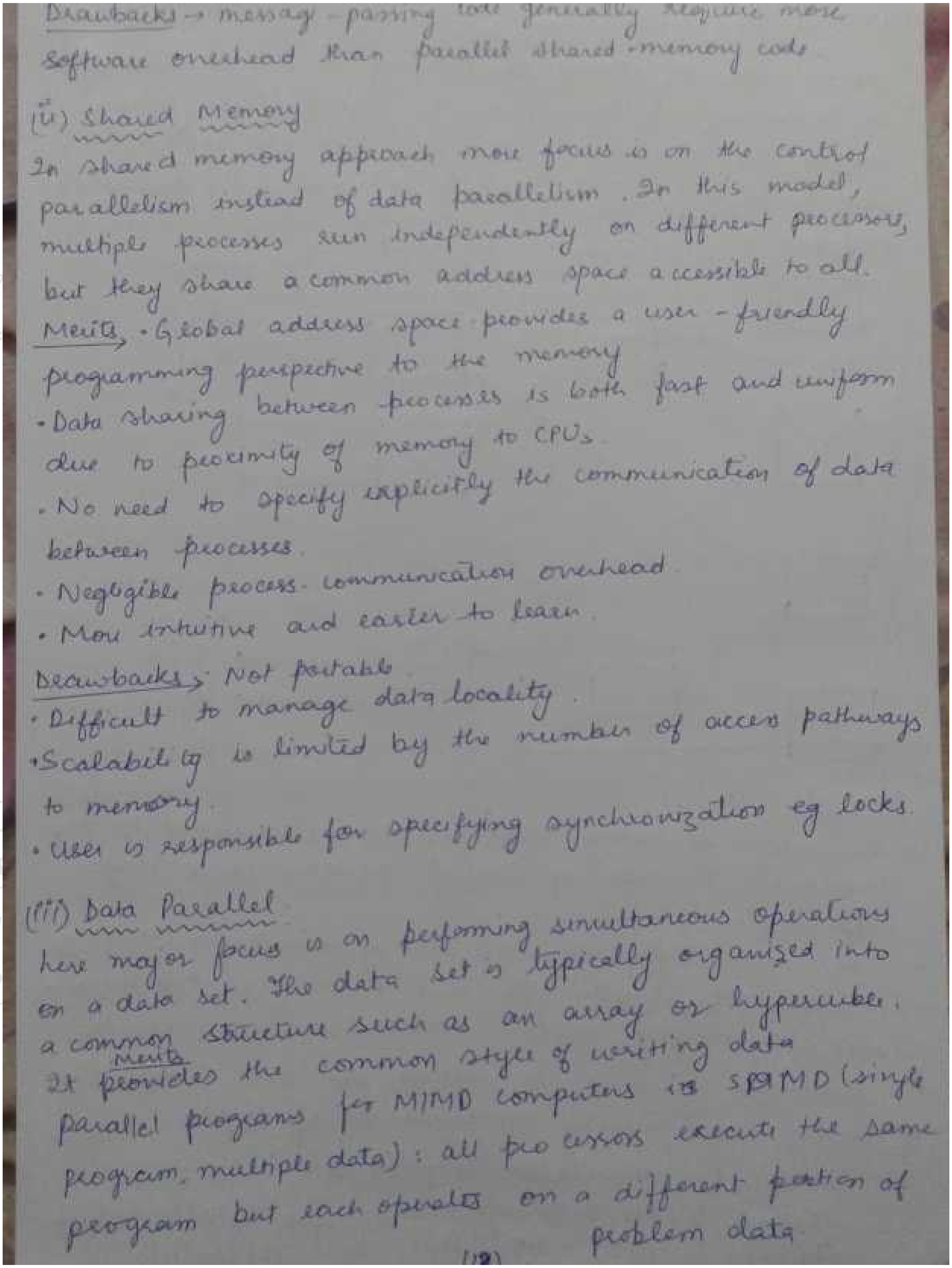
In this programming model, multiple processes across the arbitrary number of machines, each with its own local memory, exchange data through send and recieve communication between processes

Merits → • provide excellent low-level control of parallelism

• portable

• Minimal overhead in parallel synchronization and data distribution

• It is less error prone



Drawbacks → message-passing code generally require more software overhead than parallel shared-memory code

(ii) Shared Memory

In shared memory approach more focus is on the control parallelism instead of data parallelism. In this model, multiple processes run independently on different processors, but they share a common address space accessible to all.

Merits → • Global address space provides a user-friendly programming perspective to the memory

• Data sharing between processes is both fast and uniform due to proximity of memory to CPUs

• No need to specify explicitly the communication of data between processes.

• Negligible process communication overhead

• More intuitive and easier to learn.

Drawbacks → Not portable

• Difficult to manage data locality.

• Scalability is limited by the number of access pathways to memory.

• User is responsible for specifying synchronization eg locks.

(iii) Data Parallel

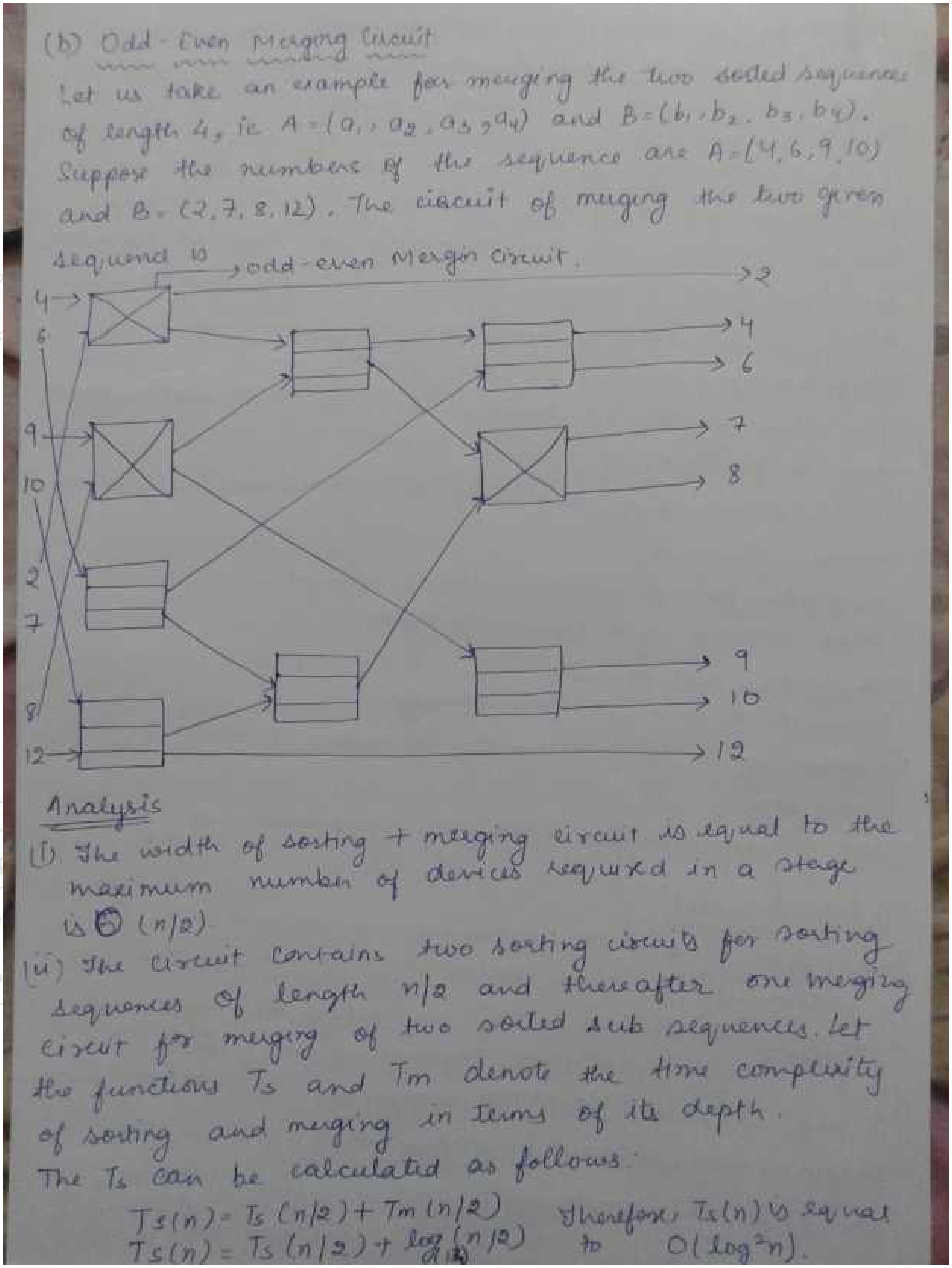
here major focus is on performing simultaneous operations on a data set. The data set is typically organised into a common structure such as an array or hypercube.

Merits

It provides the common style of writing data parallel programs for MIMD computers is SPMD (single program, multiple data) : all processors execute the same program but each operates on a different portion of problem data

thanx to Akanksha

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i
come up with. Please read carefully before writing, I am not responsible for anything th
all.



(b) Odd - Even Merging Circuit

Let us take an example for merging the two sorted sequences of length 4, ie $A = (a_1, a_2, a_3, a_4)$ and $B = (b_1, b_2, b_3, b_4)$. Suppose the numbers of the sequence are A = (4, 6, 9, 10) and B = (2, 7, 8, 12). The circuit of merging the two given sequence is odd-even Mergin circuit.

4
6
9
10
2
7
8
12
2
4
6
7
8
9
10
12

Analysis

(i) The width of sorting + merging circuit is equal to the maximum number of devices requixed in a stage is $\Theta(n/2)$.

(ii) The circuit contains two sorting circuits for sorting sequences of length n/2 and thereafter one merging circuit for merging of two sorted sub sequences. let the functions Ts and Tm denote the time complexity of sorting and merging in terms of its depth.

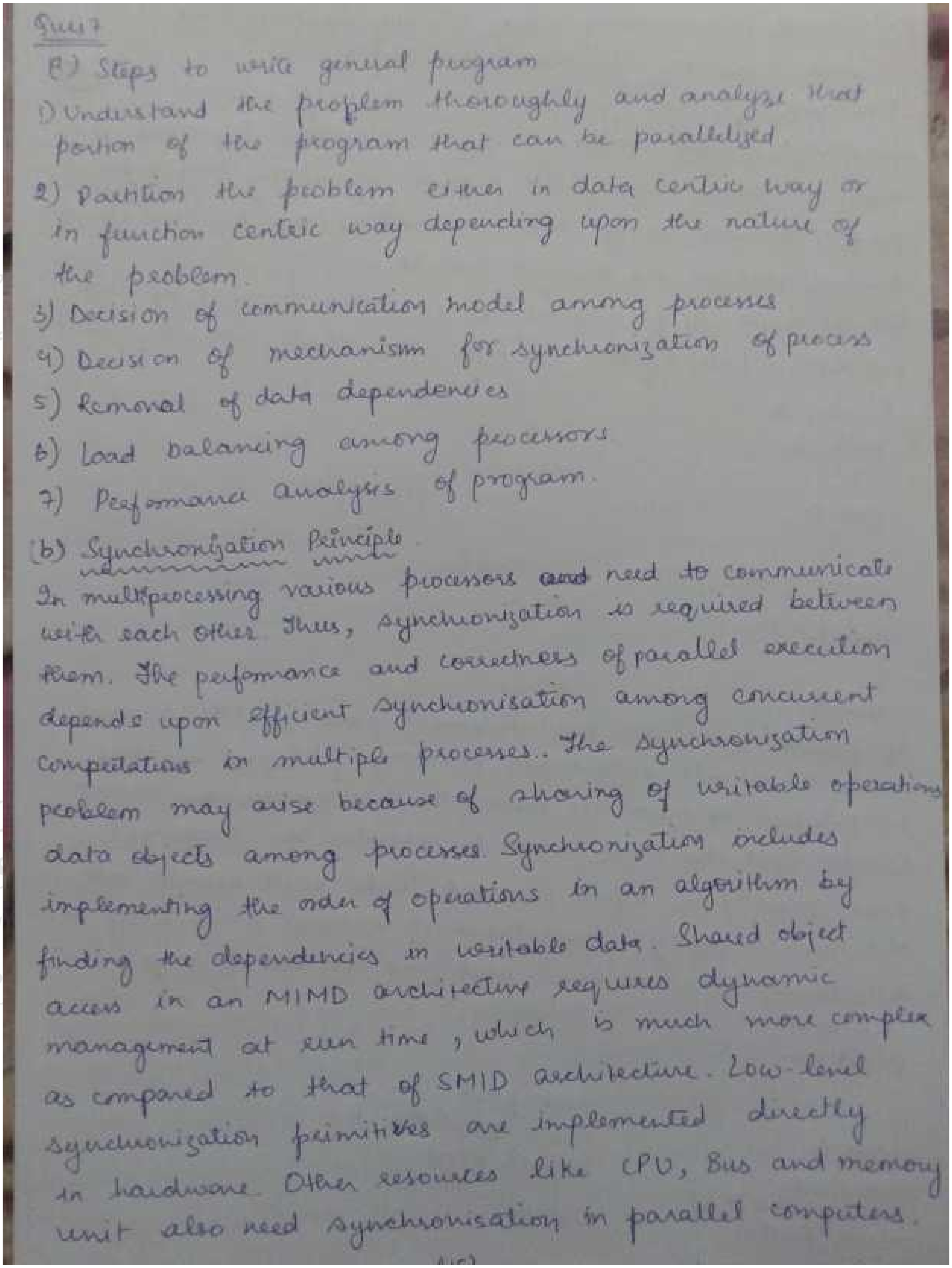
The Ts can be calculated as follows:

$$Ts(n) = Ts(n/2) + Tm(n/2)$$

$$Ts(n) = Ts(n/2) + \log(n/2)$$

Therefore, Ts(n) is equal to $O(\log^2 n)$.

Thanx to all of you, i m now in malaysia..
btw i have a question "would u like to be a friendship with me?"
yaar bahut lonely ho gya hu guyz :):)



Ques 7

(a) Steps to write general program

1) Understand the problem thoroughly and analyze that portion of the program that can be parallelized

2) Partition the problem either in data centric way or in function centric way depending upon the nature of the problem.

3) Decision of communication model among processes

4) Decision of mechanism for synchronization of process

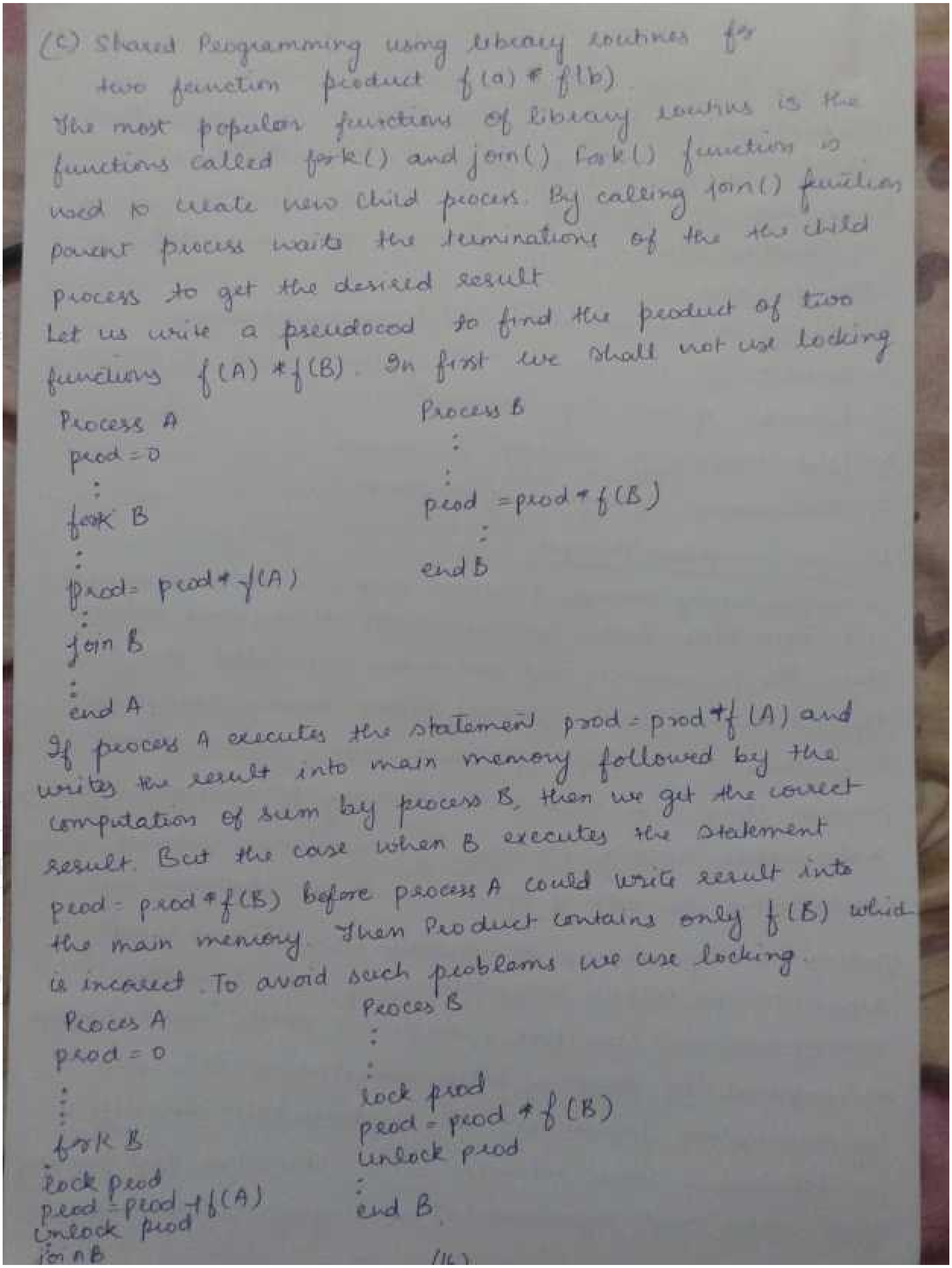
5) Removal of data dependencies

6) Load balancing among processors

7) Performance analysis of program.

(b) Synchronization Principle

In multiprocessing various processors ~~and~~ need to communicate with each other. Thus, synchronization is required between them. The performance and correctness of parallel execution depends upon efficient synchronisation among concurrent computations in multiple processes. The synchronization problem may arise because of sharing of writable operations data objects among processes. Synchronization includes implementing the order of operations in an algorithm by finding the dependencies in writable data. Shared object access in an MIMD architecture requires dynamic management at run time, which is much more complex as compared to that of SMID architecture. Low-level synchronization primitives are implemented directly in hardware. Other resources like CPU, Bus and memory unit also need synchronisation in parallel computers.



(c) Shared Programming using library routines for two function product f(a) * f(b).

The most popular functions of library routines is the functions called fork() and join(). fork() function is used to create new child process. By calling join() function parent process waits the terminations of the the child process to get the desired result.

Let us write a pseudocod to find the product of two functions f(A) * f(B). In first we shall not use locking

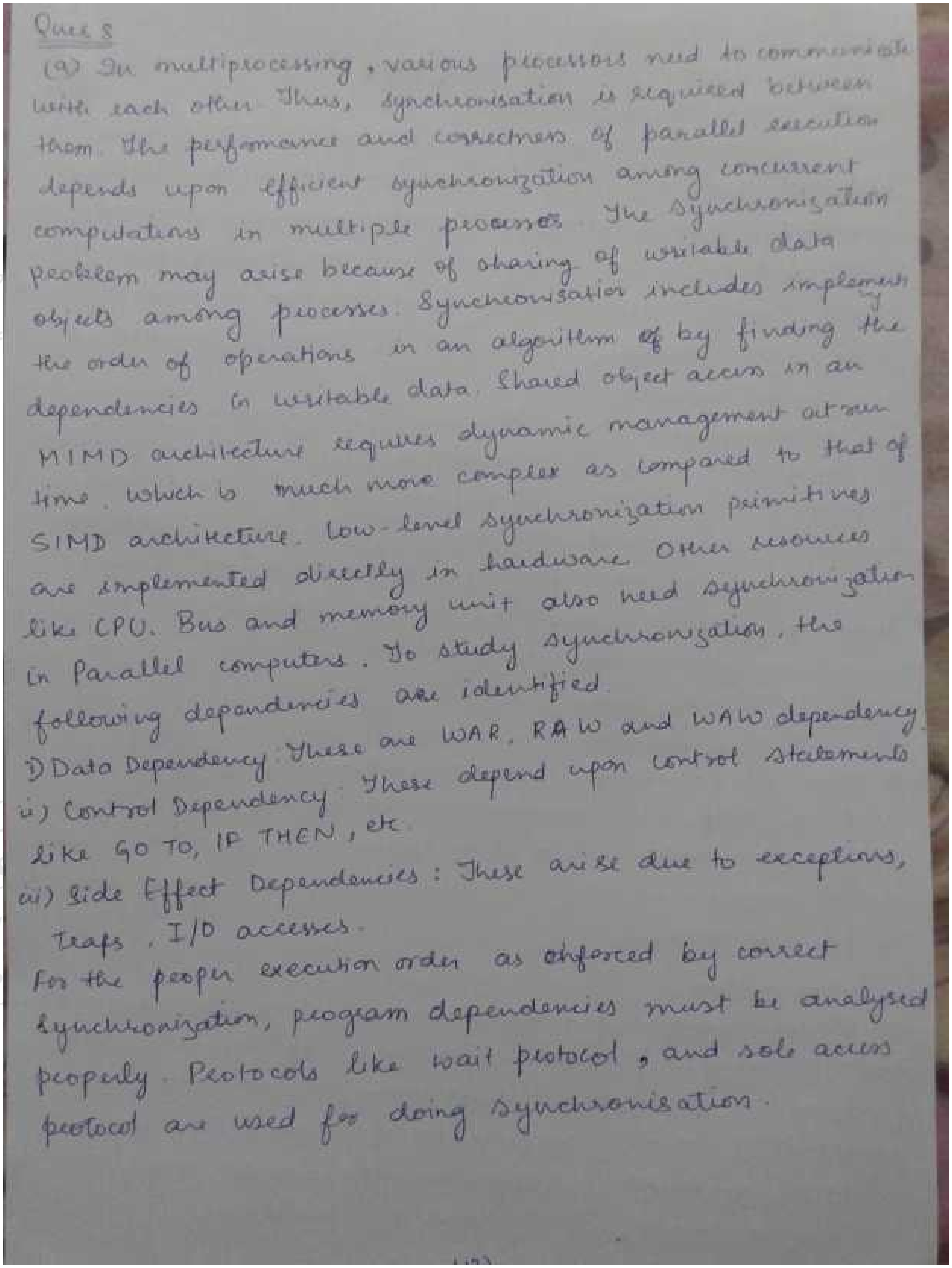
```
Process A                      Process B
 prod = 0                          :
   :                               :
 fork B                        prod = prod * f(B)
   :                               :
 prod = prod * f(A)            end B
   :
 join B
   :
 end A
```

If process A executes the statement prod = prod * f(A) and writes the result into main memory followed by the computation of sum by process B, then we get the correct result. But the case when B executes the statement prod = prod * f(B) before process A could write result into the main memory. Then Product contains only f(B) which is incorrect. To avoid such problems we use locking.

```
Process A                 Process B
prod = 0                      :
  :                           :
  :                       lock prod
fork B                    prod = prod * f(B)
lock prod                 unlock prod
prod = prod * f(A)            :
unlock prod               end B.
join B
```

(16)

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i come up with. Please read carefully before writing, I am not responsible for anything th all.



Ques 5

(a) In multiprocessing, various processors need to communicate with each other. Thus, synchronisation is required between them. The performance and correctness of parallel execution depends upon efficient synchronization among concurrent computations in multiple processes. The synchronisation problem may arise because of sharing of writable data objects among processes. Synchronisation includes implementing the order of operations in an algorithm ~~of~~ by finding the dependencies in writable data. Shared object access in an MIMD architecture requires dynamic management at run time, which is much more complex as compared to that of SIMD architecture. Low-level synchronization primitives are implemented directly in hardware. Other resources like CPU. Bus and memory unit also need synchronization in Parallel computers. To study synchronization, the following dependencies are identified.

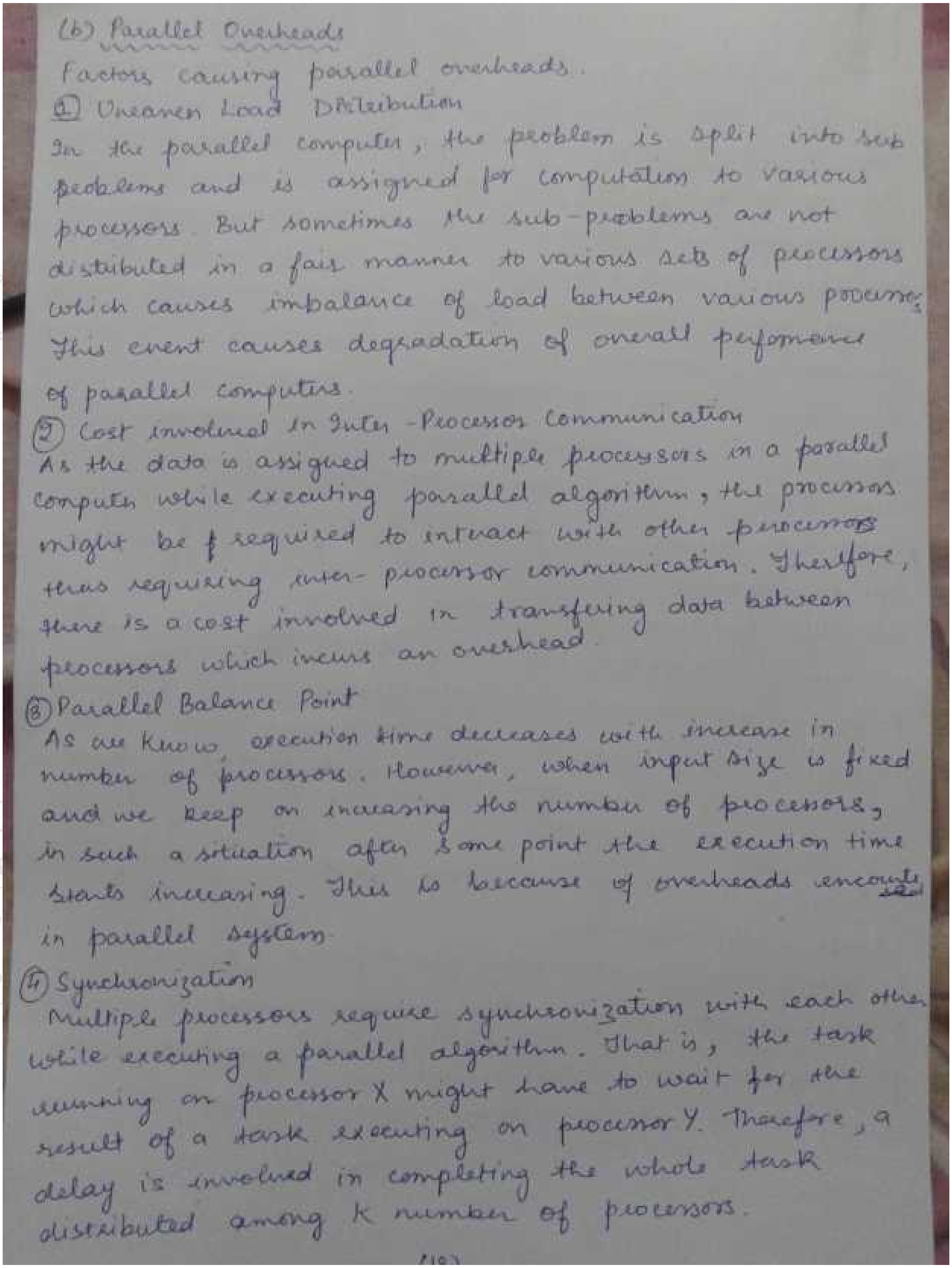
i) Data Dependency: These are WAR, RAW and WAW dependency.

ii) Control Dependency: These depend upon control statements like GO TO, IF THEN, etc.

iii) Side Effect Dependencies: These arise due to exceptions, Traps, I/O accesses.

For the proper execution order as enforced by correct synchronization, program dependencies must be analysed properly. Protocols like wait protocol, and sole access protocol are used for doing synchronisation.

Hi everyone, I am ***Akanksha***. I am not sure if all the answers are correct or not but this i come up with. Please read carefully before writing, I am not responsible for anything th all.



(b) Parallel Overheads

Factors causing parallel overheads.

① Uneaven Load Distribution

In the parallel computer, the problem is split into sub problems and is assigned for computation to various processors. But sometimes the sub-problems are not distributed in a fair manner to various sets of processors which causes imbalance of load between various processors. This event causes degradation of overall pefomance of parallel computers.

② Cost involved in Inter-Processor Communication

As the data is assigned to multiple processors in a parallel computer while executing parallel algorithm, the processors might be ~~f~~ required to interact with other processors thus requiring inter-processor communication. Therefore, there is a cost involved in transfering data between processors which incurs an overhead.

③ Parallel Balance Point

As we know, execution time decreases with increase in number of processors. However, when input size is fixed and we keep on increasing the number of processors, in such a situation after some point the execution time starts increasing. This is because of overheads encountered in parallel system.

④ Synchronization

Multiple processors require synchronization with each other while executing a parallel algorithm. That is, the task running on processor X might have to wait for the result of a task executing on processor Y. Therefore, a delay is involved in completing the whole task distributed among k number of processors.