1 - Os computadores de alto desempenho atuais podem ser consideradas como clusters de nós de processamento (PNs) interligados através de uma determinada topologia de rede.

a) Qual é o objetivo das topologias de interligação em sistemas de computação paralela e como afetam o desempenho?

Topologia de interligação refere-se à forma como os nós de processamentos estão interligados entre si numa rede de um sistema paralelo. De modo a garantir um bom desempenho é necessário ter em atenção a largura de banda e a latência e escolher a topologia adequada ao sistema.

b) O que caracteriza a topologia em mesh e de que forma é que a topologia em torus a expande?

A topologia em mesh é uma estrutura em grade/malha em que cada nó tem 4 ligações, o seu tempo de comunicação equivale à raiz quadrada do número de nós e a latência aumenta à medida que a distância entre nós aumenta. A topologia em torus expande a mesh uma vez que os seus nós extremos estão interligados entre si o que garantem uma melhor latência.

c) O que é a decomposição paralela no contexto da computação de alto desempenho?

A decomposição paralela é a distribuição de tarefas em problemas menores, que podem ser executados em paralelo por diferentes processos.

2 - A concorrência é uma abordagem popular para desenvolver os componentes locais de uma aplicação paralela que corre num nó de processamento.

a) O que é uma condição de corrida, ou race condition, e por que é problemática na programação concorrente?

Uma condição de corrida ocorre quando duas ou mais threads ou processos tentam aceder ou modificar o mesmo recurso ou variável partilhada. É problemática porque poderá resultar em resultados inesperados, corrupção de dados ou não terminação do programa. Pode ser evitado usando mecanismos de sincronização como mutexes, semáforos ou monitores.

b) O que é um região crítica e qual a sua relação com a exclusão mútua?

Uma região crítica corresponde a um trecho de código que acede a um recurso partilhado e, por isso, é necessário garantir a exclusão mútua de forma a evitar condições de corrida.

c) Que estratégia pode ser usada para prevenir o deadlock negando a condição de "Hold and Wait"?

Para prevenir o deadlock negando a condição de "hold and wait", a estratégia seria aceder a todos os recursos partilhados de uma vez só assim já não haveria a possibilidade de um processo ou thread ficar agarrado a um recurso enquanto espera por outros.

3 - O MPI (Message Passing Interface) pode ser usado para criar aplicações paralelas a serem executadas em vários nós de uma máquina com memória distribuída.

a) O que é o MPI_Barrier e quando é utilizado?

O MPI_Barrier é um método de sincronização que faz com que as threads esperem umas pelas outras e apenas avancem quando todas chegarem a esse ponto. Basicamente atua mesmo como uma barreira.

b) Qual é a diferença entre MPI_Scatter e MPI_Bcast?

A função MPI_Scatter envia partes diferentes de um vetor ou array, através do processo root, para todos os processos no comunicador, incluindo ele mesmo. Por sua vez, o MPI_Bcast envia mensagens completas, através do processo root, para todos os processos no comunicador, incluindo ele mesmo.

c) Qual é a diferença entre sincronização bloqueante e não bloqueante em MPI?

A sincronização bloqueante faz com que o processo aguarde até que a operação esteja concluída. Na sincronização não bloqueante, o processo continua a sua execução imediatamente após o pedido de envio ou receção.

4 - CUDA é utilizado para programar dispositivos GPU (Graphic Processing Units).

a) O que é um warp em CUDA e quantos threads contém normalmente?

Um warp é a unidade básica de execução paralela dentro de um multiprocessador da GPU contendo um grupo fixo de 32 threads.

b) O que é a geometria de computação no contexto da programação CUDA e por que é necessário defini-la ao lançar um kernel?

A geometria de computação refere-se à forma como as threads são distribuidas em blocos e os blocos em grades. É necessário lançar kernel<<<gridDim, blockDim>>>() para poder definir o número de blocos (gridDim) e o número de threads por bloco (blockDim).

c) Quais são os diferentes tipos de espaços de memória em CUDA? Descreva pelo menos 3 tipos.

Em CUDA, a memória global é visível por todas as threads de todos os blocos, está localizada na memória DRAM da GPU e tem alta latência; a memória partilhada é visível por todas as threads de um mesmo bloco, sendo usada para comunicação rápida e eficiente; por fim, a memória constante é acessível por todas as threads de todos os blocos, é apenas de leitura e tem baixa latência.