Problem 1 (10 pts): Discuss the high-level goals of the partition phase for optimizing parallel performance and their conflicts.

Solution: The high-level goals of partitioning include balancing the workload, reducing wait time at synchronization points, reducing inherent communications, reducing extra work, and reduce artifactual communications.

Problem 2 (15 pts): Discuss the tradeoffs of the two partitioning schemes, i.e., block vs. strip partitioning in the grid solver problem. The grid is of size $n$ by $n$ and $p$ processors are used to parallelize the sequential problem. HINT: compare the amount of inherent communication, communication-to-computation ratio, and the amount of artifactual communications between the two schemes. You can assume that the original grid is stored in a 2-D array and the underlying multiprocessor system has an extended memory hierarchy.

Solution: For the grid solver problem, the block-based partitioning has a lower communication-to-computation ratio compared to the strip-based partitioning (i.e., $\frac{4\sqrt{p}}{n}$ vs. $\frac{2p}{n}$), which also leads to less inherent communications in blocks. However, given a 2-D array, it is difficult to determine which partition scheme will result in more artifactual communications. Block-based partitioning is better at exploiting temporal locality, as process working set size is smaller than that in strip-based partitioning. However, if the grid is stored in a 2-D array, it is difficult for the block-based partitioning to exploit spatial locality because the memory allocation of the grid does not match that in the block-based partitioning.

Problem 3 (10 pts): Discuss why finite replication capacity (e.g., CPU cache and memory) can lead to artifactual communications.

Solution: Due to finite replications, e.g., various levels of caches and physical, cache misses and page faults are inevitable. Since data can be stored on distributed nodes, cache misses and page faults will lead to communications between machines.

Problem 4 (10 pts): Given each of the following memory system traffic, suggest a mitigation technique that is mostly effective to reduce that source of traffic in a shared address machine: 1) compulsory cache misses or cold-start traffic; 2) inherent communication; 3) extra data communication on a miss; 4) capacity-generated communications
Solution:

1) Compulsory cache misses or cold-start traffic – better data placement

2) Inherent communication – Algorithm reorganization

3) Extra data communication on a miss – Data structure reorganization

4) Capacity-generated communications – Large cache sizes

**Problem 5 (15 pts):** Under what conditions would the sum of busy-useful time across processes does not equal the busy-useful time for the sequential program, assuming both the sequential and parallel programs are deterministic? Give examples in which the parallel busy-useful time is larger and smaller than the sequential busy-useful time, respectively.

Solution: parallel busy-useful time can be larger than the sequential busy-useful when there are more instructions executed in the parallel program. Excessive spinning due to load imbalance and additional speculation due to the overlapping of instruction execution and data accesses can increase the busy-useful time in parallel programs. Parallel busy-useful time can also be smaller as parallel machines have a larger aggregate cache capacity, leading to less cache misses.

**Problem 6:** Suppose a parallel application with n data points has a communication-to-computation ratio of $O(1/\sqrt{n})$. If $n = 10^4$ words and the average communication latency in a 1000Mbps (Megabit per second) Ethernet for a word of data is 100 processor cycles. Answer the following questions assuming that a word is 8 bytes, a byte is 8 bit, and a processor cycle is 2ns.

1) (10 pts) With no latency hidden, for what fraction of the execution time is a process stalled due to communication latency?

Solution: Given a communication-to-computation ratio of $1/\sqrt{n}$, For every 1 word of communication, there is $\sqrt{10^4} = 100$ word of computation. Since one word of communication takes 100 cycles and one word of computation takes 1 cycle, 50 percent of time (i.e., 100 out of 200 cycles) a process will be stalled due to communication.

2) (10pts) What would be the impact of execution time of halving the communication latency?

Solution: If the communication latency is halved, the total time for each round becomes 100/2+100=150. The total execution time is improved by 25%.
3) (10 pts) What is the network bandwidth requirement of this parallel application without latency hiding?

Solution: The network should be fast enough to keep the processor busy. Thus, 1 word must be transferred through the network every 200 cycles (for 100 cycles the process will be busy sending 1 word of data, and for other 100 cycles it will perform computation). The bandwidth requirement is 1/200 word every 1 cycle. Therefore, it is 160Mbps.

**Problem 7 (20 pts):** If a parallel application has $m$ phases, each can be accelerated through parallelization. Suppose $f_k$ denotes the fraction of the inherently sequential section of the $k$th phase, and $s_k$ is the speedup of the parallelizable section in phase $k$. Derive an expression for the overall speedup obtained for this application. Show your work.

Solution: We assume each phase takes 1 unit time and all phases take the same amount of time to finish. Thus, the serial time is $m$. The parallel runtime for each phase consists of the serial time $f_k$ and the parallelizable part. So, for each phase the runtime for phase $k$ is $f_k + (1 - f_k) / s_k$. Thus, the speedup is

$$
\frac{m}{\sum_{k=1}^{m} f_k + \sum_{k=1}^{m} \frac{1 - f_k}{s_k}}
$$