Parallel Computing

Lab 4 Report

Name	Sec	BN
Sara Gamal	1	20
Eman Ibrahim	1	14

1- Compare work efficient and work inefficient implementations:

(Pageable memory without using streams)

	Efficient Prefix Sum	Inefficient Prefix Sum
1 milion	7.011ms	12.008ms
3 milion	21.326ms	36.114ms

(Pageable memory with using streams)

	Efficient Prefix Sum	Inefficient Prefix Sum
1 milion	12.300ms	18.347ms
3 milion	38.113ms	49.493ms

2- Compare work efficient algorithm but using different memories:

	Pageable	Pinned	Memory mapped	Unified
1 milion	7.0118ms	6.9936ms	10.253ms	9.2009ms
3 milion	21.326ms	21.293ms	30.348ms	26.048ms

Comments:

1. Work-Efficient vs Work-Inefficient:
   - The work-efficient prefix sum is significantly faster than the work-inefficient version.
   - The speedup becomes more noticeable as the input size increases (e.g., at 3 million elements (without using streams), efficient is ~41% faster).

2. Streams vs No Streams:

   Streams were slower because the kernel execution is very fast, so the overhead of managing multiple streams and synchronizing them dominates.

3. Memory Comparison:
    - Pinned memory performs slightly better than pageable memory because pinned memory allows faster data transfers between host and device by avoiding internal memory copies.
    - Memory-mapped (zero-copy) access is slower because even though it avoids explicit copies, it accesses host memory during kernel execution, which introduces latency.
    - Unified memory performance is reasonable but slower than pinned memory due to on-demand page migration between host and device memory.
    - Overall, pinned memory gives the best performance among memory types tested for this application.