Data Flow – Directed Block Pre-fetching for Distributed Shared Memory System

Current sDSM (Software Distributed Shared Memory) systems are based on demand paging where request to a block from memory (distributed across multiple nodes) is made on every page fault. This approach significantly affects the performance of coarse-grain applications like ADI (Alternating Direction Integration) where there is a high communication requirements due to dependencies among elements distributed across multiple nodes. Since the current system does not provide any latency hiding mechanism or block pre-fetching, so the system cannot do any computation while servicing a page fault.

In this project, we worked on block pre-fetching based on the following principles:

- Compiler analysis of block-level data flow
- Block transfer as soon as the block data is available
- Preliminary runs on IBM Cluster to prove the profitability and potential speedup if appropriate inter-node communication system is provided

We used ADI as the base application for analysis. In order to avoid the effects of current inter-node communication system, we used delay logic (a loop of dummy operations performed by each core) to simulate the block transfer as soon as the block data is generated on each node.

Results:

Figure 1 shows the relationship between delay factor (simulating the communication time for block transfer as soon as the block is available within the L2 and L4 loops of ADI) and block execution time of L2 (row-major) and L4 (column-major) ADI loops. Block transfer time of L4 loop is one of the limiting factors on speedup due to column major execution besides the communication delay of block transfers. In order to achieve a good speedup by block pre-fetching, it is required to have a inter-node communication as fast as it can transfer the data block within the time of its execution. So, it can be made available on time when a thread is going to execute that block.

Following graphs (figure 2,3,4,5) shows the effects of delay factor on speedup. In figure 2 and 3, the speedup is also limited due to the fact of column-major execution in L3 and L4 as also explained above in figure 1. In figure 4 and 5, the effect of column-major has been removed so it shows reasonable speedup with increase in space size.

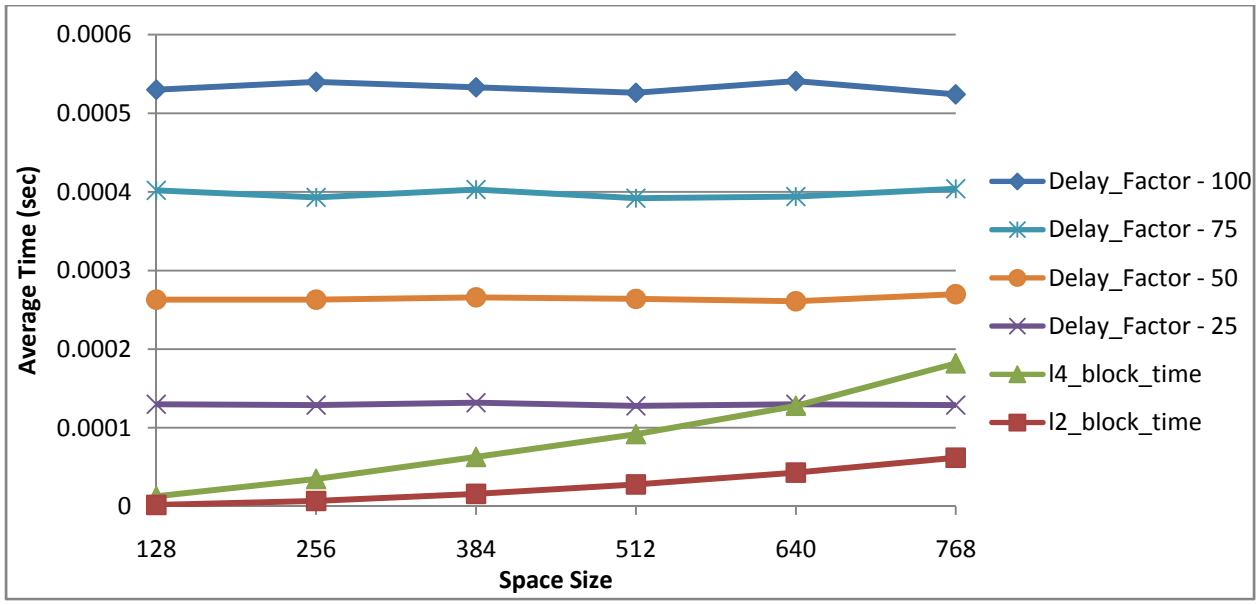

Figure 1: Relation between delay factor and block execution time of L2 and L4 ADI loops

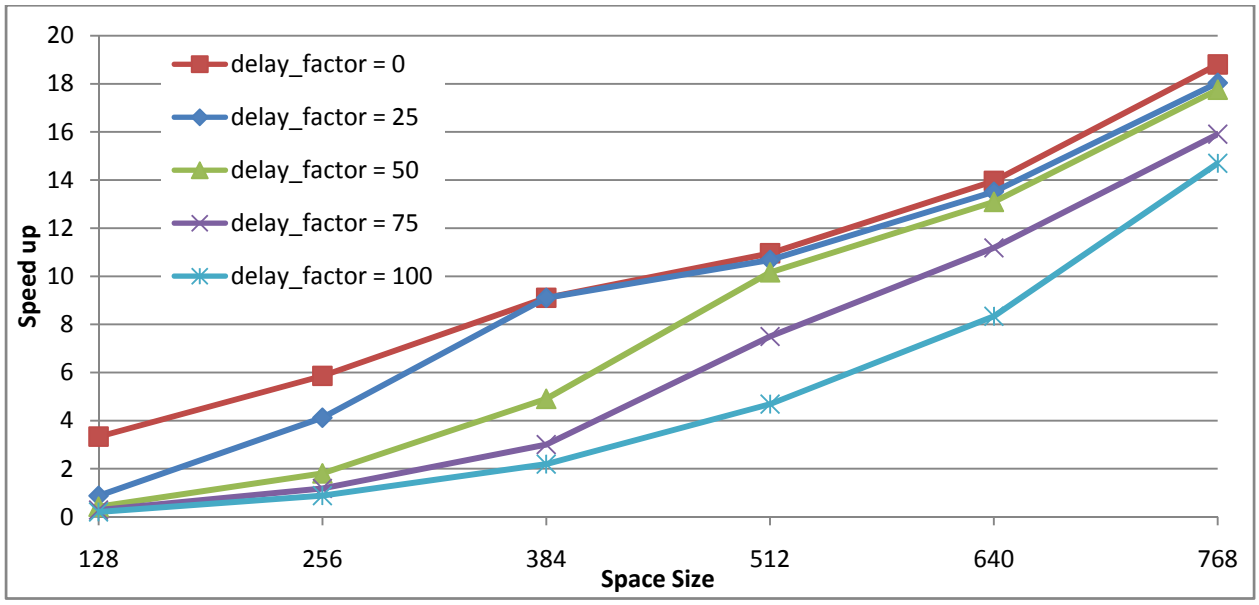

Figure 2: Effects on speedup with different Space Size and Delay Factor

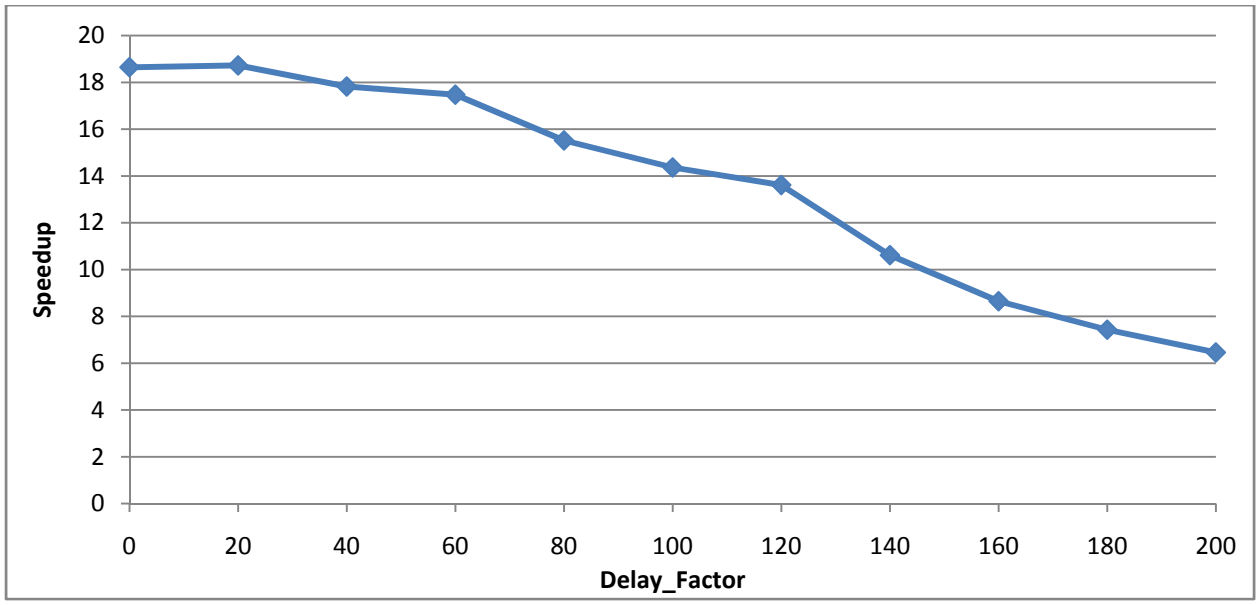

Figure 3: Effects of delay factor on speedup

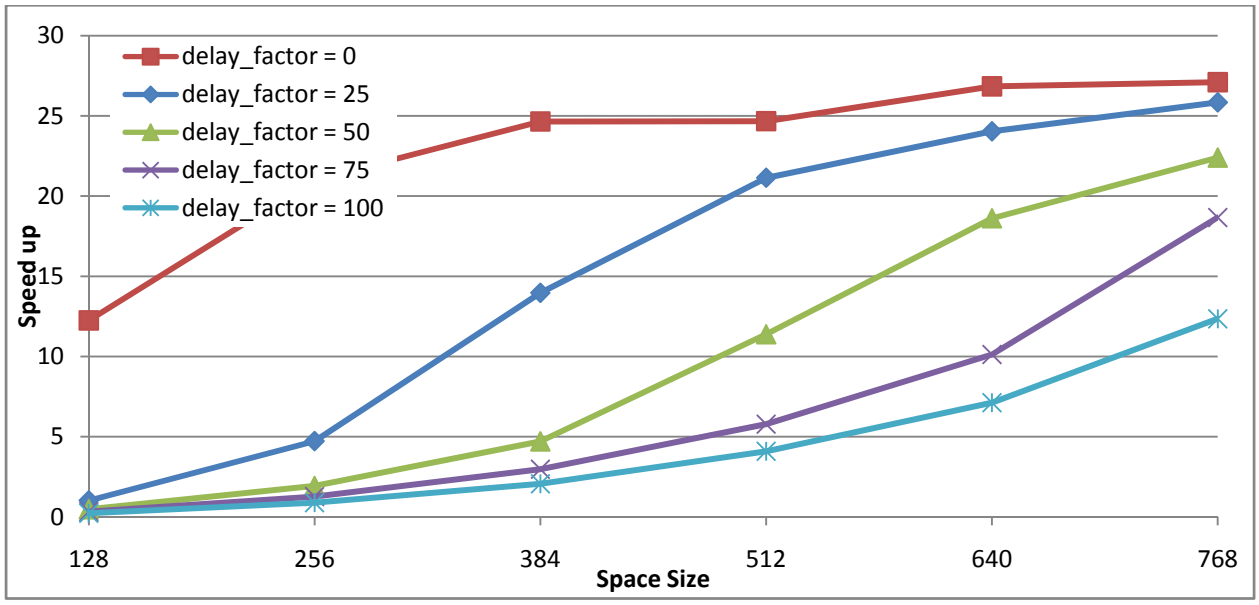

Figure 4: Effects on speedup with different space size and delay factor (without L3 and L4)

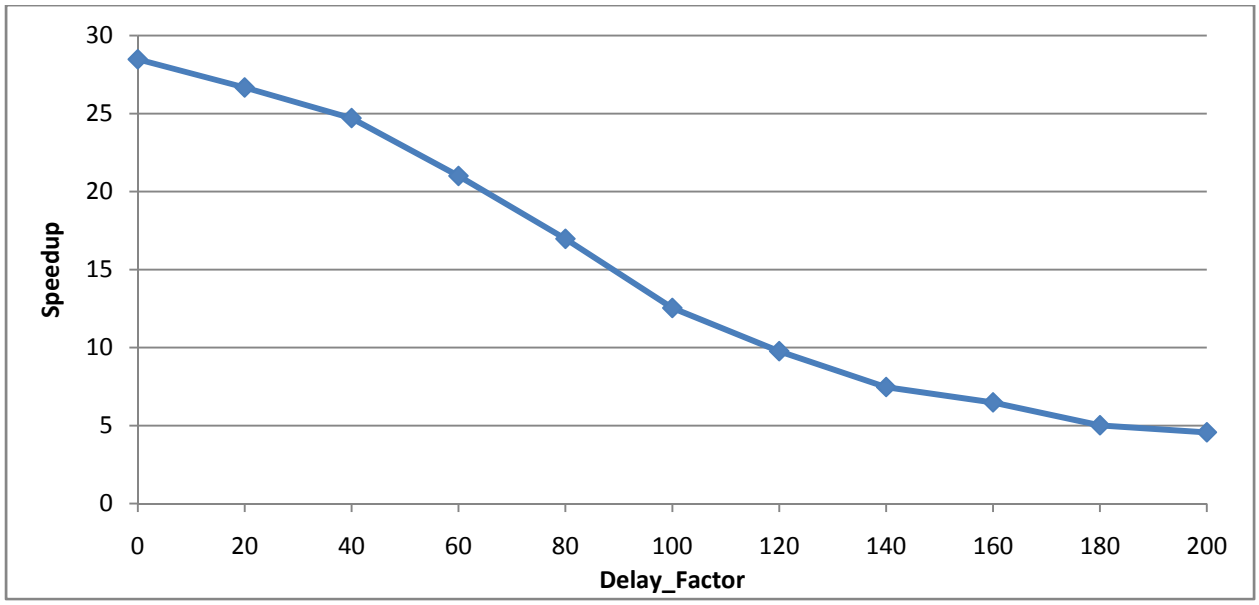

Figure 5: Effects of delay factor on speedup (without L3 and L4)