Introduction

Recently processors are coming with multiple cores. So parallel applications can take advantage of this for faster performance. Intel has also developed Xeon Phi co-processor with Intel Many Integrated Core Architecture (Intel MIC). Xeon Phi co-processor consists of 60 cores and works as a co-processor with Intel Xeon Processor. Intel has developed Intel Cilk Plus to Intel's C++ compiler. Cilk Plus allows programmer to improve the application performance by adding the parallelism to existing or new C, C++ programs.

It is well known that Cilk Plus shows better performance for Regular algorithms, which operate on vectors or matrices and we can statically predict the behaviour of the program. Programs like matrix multiplication, BLAS come in regular program category. This programs have very high computational demands and very less serial part in it. It also accesses the memory in streaming fashion. Parallel version of these applications shows tens times better performance than the serial version.

Some applications use irregular data structures like trees, graphs, queues. Many applications from domains like data mining, n-body simulation, satisfiability, compilers are irregular applications. It is difficult to predict the behavior of the irregular programs before execution. Memory access and control flow in irregular programs are irregular. Getting parallelism out of irregular programs is a challenging job.

Lonestar has implemented some irregular applications in CUDA on GPU, where they have shown that irregular programs gives speedup on GPUs. GPU and multicore system shows the SIMD way of behaviour. Multicore also supports the vectorization. So we have taken the Lonestar implemenation of irregular programs and re-written them in CIlk Plus for multicore processors. We have implemented Lonestar benchmarks in Cilk Plus to see the performance of irregular programs on multicore systems. We have compared the performance of cilk plus implemenation on the basis of speedup, lines of code, ease of programming.