Milestone Report

Make sure your project schedule on your main project page is up to date with work completed so far, and well as with a revised plan of work for the coming weeks. As by this time you should have a good understanding of what is required to complete your project, I want to see a very detailed schedule for the coming weeks. I suggest breaking time down into half-week increments. Each increment should have at least one task, and for each task put a person's name on it.

Schedule:
12/7 - CUDA implementation (Laura)
12/8 - Create tests and begin profiling (Devin/Laura)
12/10 - Improved sequential implementation (Laura)
12/11 - Analysis of profiling (Laura/Devin)
12/12 - Speedup Graphs (Laura)
12/14 - Finish paper (Laura/Devin)
12/15 - Print poster (Laura/Devin)

One to two paragraphs, summarize the work that you have completed so far. (This should be easy if you have been maintaining this information on your project page.)

We have implemented the basic sequential algorithm and an OpenMP-based parallel algorithm. Both are based on the standard pseudo-polynomial dynamic programming algorithm, where we maintain a set of possible subset sums and consider the additional sums that can be created by adding another element of the set. The parallel algorithm is based on the observation that since the order of adding elements of the set doesn't matter, we can compute the possible sums of smaller subsets in parallel using the sequential algorithm before combining them together. To combine the possible sums of two subsets, we use a parallelized iterative FFT to avoid the auxiliary arrays and poor locality of the traditional recursive FFT.

We have also begun analyzing the speedup and timings of the parallel implementation as compared to the sequential implementation, testing the performance across different tests.

Describe how you are doing with respect to the goals and deliverables stated in your proposal. Do you still believe you will be able to produce all your deliverables? If not, why? What about the ”nice to haves”? In your milestone writeup we want a new list of goals that you plan to hit for the poster session.

In terms of the goals and deliverables from the proposal, we think that we should still be able to produce all of the deliverables. For the “nice to haves” we probably won't have enough time to implement an OpenMPI solution but will continue to improve the parallel implementations we have.

Here are the original deliverables:
Sequential solution

CUDA solution
OpenMP solution
Speedup graphs of the parallel implementations over several different workloads
Profiling of parallel code and identification of bottlenecks

We have already created a naive sequential solution and an OpenMP solution as well as calculated the speedups of the OpenMP solution over different numbers of threads on the GHC machines.

New goals:
Develop different tests that test different aspects of the implementation/amounts of data (number and range of elements, target value) and results (true or false)
CUDA solution
Improved sequential solution (Pisinger implementation) – could be a “nice to have”
Speedup graphs over different numbers of threads for the different implementations on GHC machines (for all tests)
Speedup graphs over different numbers of threads for the cpu implementations on Bridges machines (for all tests)
Profiling of parallel code and identification of bottlenecks

What do you plan to show at the poster session? Will it be a demo? Will it be a graph?

At the poster session we plan to show graphs of the speedup and execution time of our parallel implementations over the sequential implementations for various scenarios and thread counts.

Do you have preliminary results at this time? If so, it would be great to included them in your milestone write-up.

Here are some of the preliminary results from the openMP implementation on one of the tests that we created. This specific test has 50k elements with an upper bound of 1k and a target sum of 10k.
We got the following speedups for the different amounts of threads on the GHC machines:
8 threads:
total simulation time: 0.035123s
speedup over naive: 11.527831
4 threads:
total simulation time: 0.060753s
speedup over naive: 6.654881
2 threads:
total simulation time: 0.113746s
speedup over naive: 3.556072

List the issues that concern you the most. Are there any remaining unknowns (things you simply don’t know how to solve, or resource you don’t know how to get) or is it just a matter of coding and doing the work?

One issue is the implementation of the CUDA implementation as we still need to decide how to solve that. I think that should be the most challenging task remaining and we need to find/develop a correct algorithm and adapt it to CUDA.