Projects in Heterogeneous Computing

Plans for next semester

At the beginning of the semester, we set out to not only learn about and understand resource allocation heuristics used in heterogeneous computing, but to also implement the heuristics and solve a real-world problem. Chris worked on the Genitor genetic algorithm and Kody on the Two-phase Iterative Minimization. Both programs are currently running and providing satisfactory results.

Although the heuristics are working, there is still much work to be done. Each project can be tweaked and modified in order to improve it. For example, the Genitor has many input parameters that can still be altered to provide better results. Though genetic algorithms are versatile heuristics and can be used to find optimum solutions to many applications, the parameters are usually very specific to the current task and will most likely change depending on many aspects of the problem statement. Current parameters still have much room for variation should be tested and explored. For the Two-phase Minimization heuristic, Kody is currently designing a three-phase heuristic that will hopefully yield even better results by computing the α of the chromosomes and optimizing based on that. Within the code of both programs, there are many improvements that can be done. These heuristic implementations are only recently successfully working and still in the first stages of their optimization.

Apart from these two heuristics, there are many other heuristics that can be used to solve this resource allocation problem. These two were chosen because they were not only a good starting point into the world of Heterogeneous Computing, but also historically yielded good solutions to similar problems. There are many other heuristics that can be implemented and tested, such as Tabu Search, RT Iterative Minimization, A*, and Ant Colony Optimization. We will probably test some of these algorithms next semester and compare them to results from the Genitor and Min-Min.

Another possibility for future work involves changing the problem into a dynamic environment where players can join, play, and leave whenever they want. The current programs, since they are static resource allocation heuristics, do not account for dynamic changes during game play. This would greatly increase the complexity of the system but would perhaps more accurately simulate real-world applications. No matter where we choose to go with improvements or new heuristics, there are many possibilities for how to continue this project and we look forward to the challenges they will bring.