Title: Parallel Integer Linear Program Solvers

Summary:
We aim to parallelize the Branch and Bound (B&B) algorithm for Integer Linear Programming (ILP) using the Message Passing Interface (MPI) and C++. This endeavor will focus on optimizing the algorithm's workload distribution and synchronization across multi-core processors to cut down computation times for complex ILP cases significantly. Given adequate progress, we plan to extend our parallelization efforts to GPU implementation or leverage OpenMP for comparison.

URL: https://teddyliang.github.io/15418-final-project/

Background
Solving an ILP is known to be NP-hard, indicating that no known algorithm can solve all ILP problems in polynomial time. This complexity arises because solving an ILP requires exploring a potentially exponential number of variable combinations to find the optimal solution. The Branch and Bound (B&B) algorithm systematically explores the solution space by branching into subproblems and bounding their potential to contain an optimal solution, effectively pruning the search space. However, it is computationally expensive due to the extensive search process and the complexity of managing and pruning the search tree, especially as the size of the problem increases. Parallelizing this process could make solving large-scale ILPs more computationally feasible.

The Challenge
Parallelizing the B&B algorithm presents several challenges:
- Dynamic Workload Distribution: Ensuring even workload distribution is complicated by the algorithm's variable branch sizes and computation times.
- Synchronization and Communication: Implementing efficient communication through MPI to synchronize the global state among processors is crucial and challenging, given the overhead it can introduce.

Resources
This project will be developed on a multi-core CPU system, employing MPI for processor communication and C++ for its high performance and flexibility.

Goals and Deliverables
Plan to Achieve:
- Successfully implement a parallel B&B algorithm using MPI and C++ that demonstrates substantial performance improvements over its sequential counterpart.
- Analyze and report on the performance metrics, emphasizing speedup and efficiency across various processor configurations.

Hope to Achieve:

  - Implement advanced dynamic workload balancing and synchronization techniques for performance optimization.
  - Extend the project to include GPU parallelization with CUDA or multi-threading with OpenMP, comparing these approaches against our initial MPI implementation in terms of performance and scalability.

The demonstration will feature a performance comparison between the parallel and sequential versions of the B&B algorithm, illustrating the efficiency gains and detailing the parallelization strategies used. If extended to GPU or OpenMP, we will also present comparative analyses of these implementations.

Platform Choice
MPI and C++ are chosen for their comprehensive support for parallel and distributed computing, ideal for the complex task of parallelizing the B&B algorithm. MPI excels in managing communication in distributed systems, while C++ provides the necessary control over computation and data management. Should time permit, exploring GPU parallelization with CUDA or multi-threading with OpenMP will offer insights into different parallel computing paradigms and their suitability for ILP problem-solving.

Schedule

- Week 1: Write a sequential B&B version for baseline.
- Week 2: Begin developing the parallel B&B algorithm using MPI and C++.
- Week 3: Continue development and start initial performance testing.
- Week 4: Focus on dynamic workload balancing, synchronization enhancements, and continue optimization processes. Prepare groundwork for potential GPU or OpenMP extensions.
- Week 5: Conduct final performance assessments on PSC and complete optimizations.