Investigating the time complexity of monolithic modelling to generate alternatives (MGA) analysis on electric grid optimization linear programming problems

Saad Ayub

May 2024

Submitted to fulfill the final project requirement for APC 523

The internet was used while completing this project to find reference materials post-office hours, and in assisting with generating matrices in latex, etc.

This project was done in collaboration with Richard Zhu.

1 Introduction

The energy sector of any modern scale is an impressive coordination between thousands of connection points, transmission lines, generators, management, private and public organizations, and consumers of electricity. In planning capacity expansion and operation, there are thousands to tens of millions of variables to consider, and choosing an "optimal" route for effective investment, planning, and regulation is a ripe area for numerical optimization models. These problems generally fall under two categories, linear programming (LP) and mixed-integer linear programming (MILP) implementations of an optimization model.

For this project, we sought to improve the GenX open-source model, which is a highly-configurable, open source capacity expansion model which uses constrained LP or MILP models to construct a cost-optimal portfolio of electricity investments and operational decisions that meets electricity demand for one or several years into the future. The model also allows for constraints related to environmental impact, and is intended to help policymakers design low-carbon or carbon-free expansions to the electricity sector.

A characteristic example of such a grid optimization problem (in MILP form), from [3], is the problem described by the following objectives and constraints,

$$\begin{aligned}
\text{minimize } & c_I^T y + \sum_{w\in W} c_w^T x_w \\
\text{subject to } & A_w x_w + B_w y \le b_w, \forall w \in W \\
& \sum_{w\in W} Q_w x_w \le e \\
& Ry \le r \\
& x_w \ge 0, \forall w \in W \\
& y \ge 0 \\
& y \in \mathbb{Z}_m
\end{aligned}$$

where $y \in \mathbb{R}$ is the vector of investment decision variables, c_I^T the vector of parameters representing the fixed cost objective, x_w the vector presenting operational decision variables, c_w the vector representing the additional terms of the objective function, $w \in W$ each of the (in this case, weekly) subperiods in W, and matrix Q_w and vector e the policy constraints.

Benders decomposition is a method for solving MILPs that involves a divide-and-conquer approach to solving the optimization problems. First, the master problem is solved (a "rough" solution is found) and then subproblems are found ("refining the solution"). This is somewhat analogous to the multigrid approach that we studied in class. However, notably, we iteratively solve the master problem, then move to the subproblem, then go back to the master problem and so forth. This allows us to gradually converge to a potential solution. The reason this approach works is because we can freeze complicating variables found during the master problem and take them as constants in the subproblem, significantly simplifying the computation and mathematics. Since this optimization step via the various sub-problems is done so many times, and the solution finding is largely independent between the other subproblems, parallelizing and distributing these large number of instances is a natural step for the development of the codebase. Since Benders Decomposition is a novel method for MGA in grid optimization, we seek to compare it to the standard monolithic MGA optimization technique. However, since the Benders implementation is both distributed and multithreaded, we seek to construct a similarly distributed and multithreaded version of the monolithic MGA algorithm for comparison.

2 MGA Parallelization and Distributed Models

The broader work in improving the GenX model is in seeking to explore a Modeling to Generate Alternatives (MGA) analysis, presenting multiple nearly optimal alternatives with various subtle trade-offs. These are helpful in the context of infrastructure optimization problems where planners may want a solution within 10%, say, of the most optimal cost (cost relaxation), but would be willing to trade off cost differences within this range to reduce, say, the number of hours required for a network expansion. The original work already has implemented Benders Decomposition, adjusted to apply to energy system MILPs in [3], to perform the optimization and then MGA to present multiple feasible options that form a "basis" of the space of potential solutions under relaxed constraints.

GenX seeks to distribute and multi-thread the *Monolithic* MGA method (i.e., a version not using Benders Decomposition) as well, with the goal to compare the performance between the distributed and multithreaded versions of the Monolithic and Benders decomposed problems. The points of analysis would be the relationships between solution time and compute cost (in \$), solution time and MILP dimension, and solution time and memory under different hyperparameters of parallelization (including the number of cores used, etc.). The existing codebase is capable of multi-threaded processing for the Benders decomposition, but full distribution has been identified as a means of improving the metrics noted for analysis above. The basis of the distribution would be to allow isolated tasks to be dispatched to multiple worker computational units, each tasked with a largely similar task to optimize independently. The workers will each return an array of optimized parameters, and the main dispatching server will process these asynchronous returns to proceed with the computations in a manner similar to the non-distributed codebase. We will be developing upon a Benders + MGA variant of the open-source energy expansion modelling software Gen-X[1] being developed by in conjunction with our co-authors.

The project sought to distribute this MGA system to improve on the large memory requirements needed for the model to function. In previous work, solving the problem monolithically (i.e., with no decomposition into subproblems) sometimes ran into memory issues despite using a large computing cluster to perform the operations [3]. The multithreading is done through implementing Gurobi, which will automatically allocate suitable non-sequential tasks to different

[1]https://github.com/GenXProject/GenX.jl

threads as possible. Although multithreading has drastically reduced the compute time of the each Benders solution [3], it is not expected to impact the Monolithic model as significantly due to the fact that there are fewer non-sequential tasks to be done. Multiple Monolithic MGAs could run at once, but as each instance would be necessarily large and not many parallel versions could run on one core's allocation of memory, mapping out any part of the parameter space near optimality would take quite some time. The distribution of the monolithic system shouldn't change the time needed for a single alternative solution, but spreading out the generation of alternatives to multiple workers would allow us to map out the solution space much faster.

3 Related Works

Benders decomposition Benders Decomposition, which involves taking an optimization problem and dividing it to form subproblems, has been applied to a range of challenging optimization problems however is dependent on having closed-form analytical formulas for subproblems [6]. Recent work has applied Benders Decomposition to grid optimization problems with significant gains in runtime efficiency for larger-scale optimization systems [3]. Studies also investigate the effect of regularization on these Benders algorithms to further improve performance [5].

Optimization for grid design Optimization for electric grid design has also been prevalent, with authors using probabilistic multiobjective frameworks for distributed energy resource planning though lacking rigorous testing on larger, more realistic systems [8], a MILP-based distributed system optimization to decrease the need for transmission expansion [4], and transmission expansion planning for renewable power with promising real-world results [1]. Optimization algorithms have also been used to study household-level electricity patterns, such as the effect of conventional versus super-efficient appliances on PV systems both attached and not attached to the grid [2], and on the building-level by leveraging building-models to improve energy efficiency [7].

4 Methods and Results

Progress was made in two avenues for this project, the multithreading and the distribution, but these were in separate instances. That is to say, we were successful in creating a distributed Julia program, but this could not run the multithreaded monolithic model using the Gurobi multithreading manager. We also succeeded in making a fully working GenX model for a monolithic, multithreaded case, but this was not able to be distributed. The final step for this aspect of the GenX development would be to combine these two separate points of achievement together, which will be done in future work.

Richard Zhu will give detail to the multithreaded monolithic model and observations thereupon, while here we will further investigate the distribution, and gains that could be expected to be made in the computation time and exploration of the near-optimal parameter space. The Julia program made for this purpose is based on the code for the multithreaded Benders decomposition code for MGA, with the present work swapping out the MGA for a multithreaded Monolithic approach as well as adding in the distribution.

For the distribution specifically, the Julia program `Run_distributed_locally.jl` was written as a test case to "distribute" workers to other CPU threads, rather than to other nodes, to narrow down the potential reasons the model would not fully run. The program `Run_distributed.jl` was the intended final code, which was to be bugfixed by the other. Here, a number of workers and processes per worker can be set automatically based on the number of tasks expected for the MGA, the set number of CPUs to be used per worker, and the number of threads Gurobi was allowed to allocate to. As the calculation and handling for the number of tasks of the

MGA differ greatly between the Benders and Monolithic methods, various other functions were changed to reflect the shift away from multithreading the sub-problems of the alternative being generated. This would include some files in the `src/additional_tools` folder as well. Effort notwithstanding, the distributed model was not able to run although the distribution of the tasks to workers succeeded, meaning that what follows is necessarily theoretical.

Taking for reference the work done by [3], we can arrive at an estimate for the expected effect of distributing the Monolithic MGA. Figure 1 plots the results from previous GenX work, comparing the time needed to perform the initial fully-optimized optimization for the energy system at various scales, in a MILP formulation of the problem. Additional computations were done in the original work, but the simgle-thread Monolithic model was not able to converge within the allotted time (at least > 32 hours) for the majority of the cases for greater than six electricity grid zones in the model.

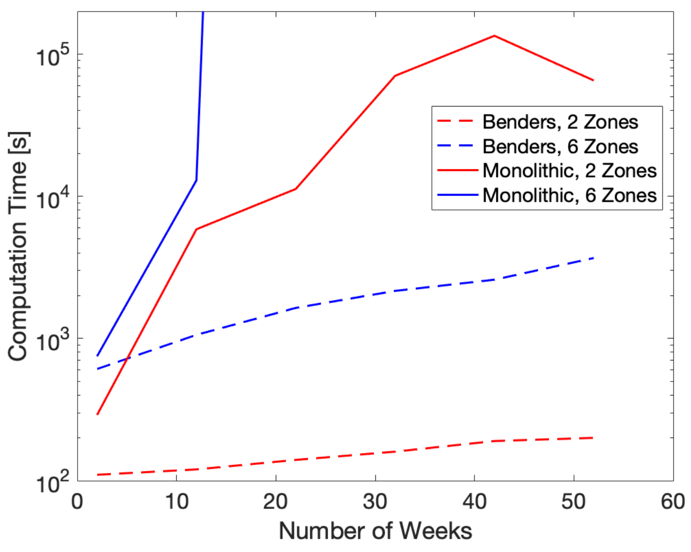

Figure 1: Computation time comparison between Benders and Monolithic formulation as a function of the number of computed weeks for the MILP formulation of the problem. Results for Monolithic, 6 Zones became intractable (did not converge within time limit) after 10 weeks of simulation. Number of zones refers to a number of grid regions present in simulated model. [3].

Multithreading the Monolithic model would still somewhat decrease the computation time needed for the Monolithic MGA despite the associated memory problems and dearth of tasks to multithread, which should allow more of these Monolithic MILP optimizations to converge. As that the method for MGA may be generally less computationally intensive than solutions to the full optimization, we can assume most formulations of the Monolithic MGA will not be as intractable. Figure 1 can thus be understood as a proxy for the time taken to explore the near-optimal parameter space. Taking the example of the two-zone computations, allowing for parallel computation increases the speed of the process by a factor of around 2 to 700 times. The proportion of overhead computation for each parallel task would be less or equally complex for the creation and collection of workers in the distributed system, compared to the generation and splitting of tasks for the multithreading, and then stitching together the solution along the Benders decomposition cuts. In other words, we could predict that the distribution of the Monolithic model would be expected to improve the speed of MGA to a comparable degree to how multithreading the Monolithic model (via Benders decomposition) had on the computation of a single pass of the full optimization.

Another possibility for consideration is represented by Figure 2, in which the computation times for the LP formulation of the optimization are plotted. In this, we see that the Benders

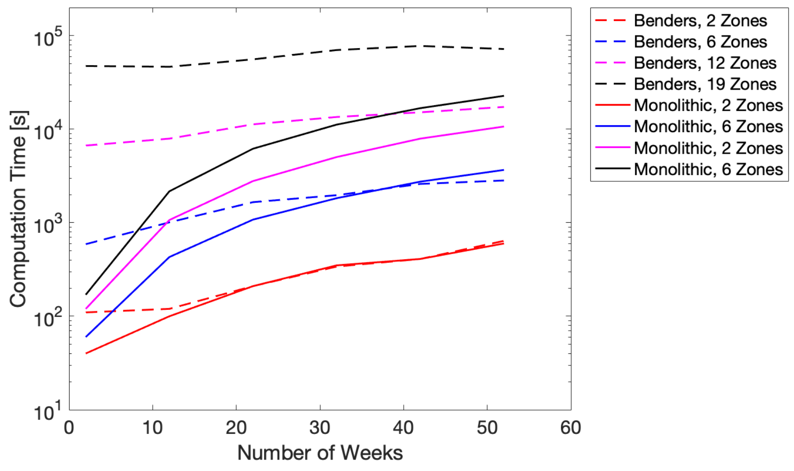

Figure 2: Computation time comparison between Benders and Monolithic formulation as a function of the number of computed weeks for the LP formulation of the problem. Benders decomposition generally takes longer than the Monolithic model due to the overhead costs of setting up the decomposition. [3].

method takes between three to a hundred times longer for short optimization periods, though it begins performing as well as the Monolithic optimization as the period of analysis gets longer. The LP formulation is much less complex than that of the MILP setup, and this inversion from the pattern established in Figure 1 is because the overhead cost of splitting and rejoining the Benders subproblems is greater than the benefits gained by performing the process in the first place [3]. Although it is expected that this overhead cost would be generally larger than the overhead in simply distributing instances of the monolithic MGA and recording the re-optimized parameter values, it is good to note that an overcomplicated distribution system might make the computations longer instead of improving the MGA analysis. Although, the time needed in setting up the existing distribution system in `Run_distributed.jl` is on the order of seconds, and thus not expected to expand into a relevant length of time unless the communication rate between workers and the dispatch program, to transmit the models and parameter information, is effectively slow.

5 Conclusion and Future Work

The distribution and multithreading of the Monolithic MGA scheme will be used by the GenX team at Princeton to compare the improvements made to the MGA model through Benders decomposition of the problem. The multithreading of the model covered by Richard Zhu would be expected to reduce the solution time somewhat, but depending on the intensity of the overhead costs and communication times of performing the final distribution, the distribution of this system would be expected to allow for much faster exploration of the near-optimal parameter space through MGA.

The implementation was partially successful and yielding a fully functional multithreaded, distributed version of the Monolithic MGA model. Despite our best efforts and communications with the GenX team over the term of the course, we ran into recurring issues with combining the multithreading with the distributed model, as well as generally having the distributed system working as desired. In lieu of using a real example system to test the performance of the MGA model against either the multithreaded, distributed Benders, or against the unimproved Monolithic model, we are able to glean logical expectations for performance from comparisons between the single-threaded Monolithic and the multithreaded Benders models. Such an analysis shows that the computation costs of the MILP MGA would stand to be improved by up to a few hundred times if the distribution would allow for the parallel processing of the Monolithic model without having a proportionally large overhead cost in either setup or communication of/to the worker cores.

Future work would first include the completion of the multithreaded, distributed Monolithic model. After that work would include running simulation diagnostics akin to those presented in [3], and comparing those results to those of a multithreaded, distributed Benders model. Past this would be an exploration of methods to further improve the Benders model, such as iterating on the method to divide the monolithic problem into Benders subproblems. The goal of GenX has always been to facilitate the planning of investments and policy for greening the electricity grid, and thus additional work could be done to consider more varied approaches to decarbonization, such as co-location of facilities or carbon capturing and storage.

References

[1] Fatemeh Bayatloo and Ali Bozorgi-Amiri. A novel optimization model for dynamic power grid design and expansion planning considering renewable resources. *Journal of Cleaner Production*, 229:1319–1334, 2019.

[2] Chaouki Ghenai and Maamar Bettayeb. Design and optimization of grid-tied and off-grid solar PV systems for super-efficient electrical appliances. *Energy Efficiency*, 13(2):291–305, February 2020.

[3] Anna Jacobson, Filippo Pecci, Nestor Sepulveda, Qingyu Xu, and Jesse Jenkins. A computationally efficient benders decomposition for energy systems planning problems with detailed operations and time-coupling constraints. *INFORMS Journal on Optimization*, 6(1):32–45, 2024.

[4] Boran Morvaj, Ralph Evins, and Jan Carmeliet. Optimization framework for distributed energy systems with integrated electrical grid constraints. *Applied Energy*, 171:296–313, June 2016.

[5] Filippo Pecci and Jesse D. Jenkins. Regularized benders decomposition for high performance capacity expansion models, 2024.

[6] Ragheb Rahmaniani, Teodor Gabriel Crainic, Michel Gendreau, and Walter Rei. The benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817, 2017.

[7] Michaël Salvador and Stéphane Grieu. Methodology for the design of energy production and storage systems in buildings: Minimization of the energy impact on the electricity grid. *Energy and Buildings*, 47:659–673, April 2012.

[8] Vahid Vahidinasab. Optimal distributed energy resources planning in a competitive electricity market: Multiobjective optimization and probabilistic design. *Renewable Energy*, 66:354–363, 2014.