Pacific Northwest
NATIONAL LABORATORY

PNNL-29898

Improving Cloud-based Performance of Distribution Planning

Improvements to GridLAB-D

April 2020

Frank Tuffner
Mark Eberlein
Kevin Atkinson
Randy Tran

U.S. DEPARTMENT OF ENERGY

Prepared for the U.S. Department of Energy
under Contract DE-AC05-76RL01830

DISCLAIMER

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC05-76RL01830

Printed in the United States of America

Available to DOE and DOE contractors from
the Office of Scientific and Technical Information,
P.O. Box 62, Oak Ridge, TN 37831-0062
www.osti.gov
ph: (865) 576-8401
fox: (865) 576-5728
email: reports@osti.gov

Available to the public from the National Technical Information Service
5301 Shawnee Rd., Alexandria, VA 22312
ph: (800) 553-NTIS (6847)
or (703) 605-6000
email: info@ntis.gov
Online ordering: http://www.ntis.gov

PNNL-29898

Improving Cloud-based Performance of Distribution Planning

Improvements to GridLAB-D

April 2020

Frank Tuffner
Mark Eberlein
Kevin Atkinson
Randy Tran

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

Pacific Northwest National Laboratory
Richland, Washington 99354

Summary

This report provides a high-level summary of the improvements to the GridLAB-D distribution analysis software, including work with GridUnity to focus the areas of improvement and test the final implementation. The improvements allow repetitive, time-series-based simulations, such as renewable hosting capacity studies, to show a 30% or more improvement in execution time. All improvements from this project are available on GridLAB-D's open-source repository, with most of the changes migrated into the upcoming version 4.3 release of the software.

Acknowledgments

The authors wish to thank the Technology Commercialization Fund office at the U.S. Department of Energy for funding this work, particularly through funding from the Office of Electricity Efficiency and Renewable Energy. The authors would also like to thank the staff at Grid Unity for their assistance on this project, with special thanks to Brian Fitzsimmons, Lulu Young, Alex Dinkel, and Jake Zhang. The authors also wish to acknowledge the contributions of former project member, Laura Leist, for her work in updating the code. Finally, the authors wish to thank Trevor Hardy with the Pacific Northwest National Laboratory for his review of this document.

Acronyms and Abbreviations

DER	Distributed Energy Resource
DOE	Department of Energy
GMLC	Grid Modernization Laboratory Consortium
LU	Lower-Upper – describes a type of matrix decomposition
PNNL	Pacific Northwest National Laboratory
TCF	Technology Commercialization Fund

Contents

Tables

1.0 Introduction

The evaluation of new equipment, operating scenarios, or new topologies on the power system is very important to the continued reliable operation. Newly deployed technologies can bring additional functionality, but may also strain existing grid resources or interact with older assets in unexpected ways. At the power system distribution level, new technologies and scenarios can often rapidly deploy, as a result of new technology trends or new policy incentives. Utility operators and planners often simulate and examine these new technologies, either directly, or through contractors and consultants. The ability to quickly obtain results from accurate simulation tools is often a key element of this examination.

While there are many distribution analysis tools available (Tang, et al. 2017), this project focused on improvements to the GridLAB-D distribution analysis package. GridLAB-D is an open-source distribution system analysis tool developed for the U. S. Department of Energy (DOE) (GridLAB-D Software 2020). GridLAB-D contains a variety of multi-domain models to evaluate distribution system projects, including distribution powerflow with electromechanical dynamics, renewable generation sources, residential house thermal and electrical modeling, and demand response market capabilities. This project focused primarily on renewable generation integration-type studies, especially those associated with carrying capacity of the distribution grid.

Partnering with GridUnity (formerly known as Qado Energy), engineers at the Pacific Northwest National Laboratory (PNNL) worked to improve the performance of GridLAB-D in these types of studies. This document summarizes some of the changes to the GridLAB-D software to achieve these performance improvements, including benchmarks demonstrating the overall improvement. The report also includes a brief summary of activities carried out to assist GridUnity, as well as some final thoughts and conclusions to the project.

2.0 GridLAB-D Performance Improvements

The primary mechanism for improving the performance of GridLAB-D in distribution analysis simulations was through changes and updates to the underlying source code. While some underlying algorithmic changes were implemented, most of the improvements were associated with changing the tools used to compile the code and enacting better practices in the source code. This section will provide some detail into the improvements and provide metrics for comparing the improvement at the end of the section.

It is useful to note that while most of the performance improvements described here are already implemented in the current branch of GridLAB-D (moving toward version 4.3 [Navajo] at the time of this report), all of the fixes are available on the GridLAB-D GitHub repository, under the link provided in (GridLAB-D Feature 1091 2020). These changes are expected to be fully migrated into the main version of GridLAB-D in a future release.

2.1 Source Code Programming and Mechanism Updates

This section will describe some of the main updates and changes to GridLAB-D to achieve the improved performance. This list is by no means exhaustive, but will capture some of the primary mechanisms. For detailed changes, interested parties are encouraged to view the source code directly (GridLAB-D Feature 1091 2020).

2.1.1 Support Library Updates

One of the first methods to improving the performance of GridLAB-D was to update some of the support libraries to more modern implementations. GridLAB-D has been in development for over 10 years, so some libraries have not been revisited since their original implementation, often years ago. For performing renewable integration studies, support libraries associated with the powerflow solvers were updated.

2.1.1.1 superLU

superLU (Demmel et al. 1999) is the primary LU matrix decomposition library used by GridLAB-D, utilized by the Newton-Raphson powerflow solver to iteratively solve the voltage and current characteristics of the power system. The library was updated from version 3.0 to version 5.2 to incorporate performance improvements and bugfixes. The updated library did not result in significant performance improvements on its own, but did help enabled some of the programming updates mentioned later in Section 2.

2.1.1.2 KLU

The KLU library (Davis and Nataranjan 2009) is an alternative LU decomposition library to superLU. KLU is not as robust or generic as superLU, but is better tailored toward the types of sparse matrices that Newton-Raphson powerflow equations produce. As a result, KLU often produces significant performance improvements to powerflow simulations. Compared against the baseline version, performance improvements associated with KLU were accomplished through two mechanisms. Much like superLU, the underlying KLU library was updated to a more current version. Furthermore, the interface through which KLU (and other future LU decomposition libraries) interfaces with GridLAB-D was also updated to enable great platform compatibility and usability. Due to some licensing considerations with a prior version, KLU is built separately. As part of this project, the external library interface utilized was updated to be more generic and easier to build, which allows easier incorporation of KLU, including on multiple platforms (it was primarily available for Microsoft Windows-based versions of GridLAB-D in the past).

2.1.2 Source Code Update

The primary mechanism for the performance improvements to GridLAB-D came through improvements to the underlying source code. These improvements were less algorithmic in nature than to make the software compatible with more modern compiler methods, and to enable modern code optimization techniques to be applied.

2.1.2.1 Change to C++

The baseline version of GridLAB-D, version 4.0 (Keeler) includes a mix of C and C++ code. Much of the core and one older module were still almost exclusively C-based (as opposed to C++), primarily due to the age of the code. Unfortunately, this was causing several warnings, and eventually errors, when trying to compile GridLAB-D with new compiler standards and newer libraries.

The first step in this process was to fix underlying coding problems that were warnings in older versions of C and C++, but had become errors in newer versions of the compiler. These were typically antiquated or less stable approaches to things such as character string definitions. Fixing these warnings and errors enabled the compilers supporting more modern C/C++ standards to be utilized.

The second aspect of the C++-change was to convert all of the underlying C-based files to a more C++-based style. Note that this was primarily to enable compiler compatibility for newer versions of C/C++; the underlying C-based code was only minimally changed, not updated to reflect modern C++ coding approaches. This transition, along with the warning and error fix just mentioned, was primarily to enable more modern compilers and techniques to be utilized.

2.1.2.2 Compiler Optimization Enabled

With the underlying code updated to be natively C++ compiled, and compatible with more modern compilers, the largest performance increase could finally be realized. During the migration to a more generalized open-source implementation in GridLAB-D version 4.0, the ability to enable optimization during the code compilation was lost. Compiler-based optimizations are ways for the code to be streamlined and speed improved while converting from the human readable-source code to something the computer understands how to run. While it was easier to now consistently compile GridLAB-D across multiple hardware platforms, the overall performance suffered. With the C++-compiler compatibility fixes above, as well as some general coding fixes, the ability to regain optimization, and its associated performance improvements, was once again possible with GridLAB-D. Furthermore, this was retained across all of the supported platforms, ensuring the improvements could be utilized for multiple analysis conditions and scenarios.

2.1.2.3 CMake Build Generation Process

GridLAB-D v4.0 made use of the GNU autotools package to generate build files, resulting in a functional, but slow and burdensome, build process, which complicated dependency management. The GNU autotools also required an extremely high learning curve before maintenance and updates can be reasonably performed.

To resolve the complexity of building GridLAB-D, a new, modernized, and simplified build process was developed using the CMake toolkit, which facilitates a number of quality-of-life improvements to the process of preparing and compiling GridLAB-D. The most notable improvements within the CMake process are full and stable support for parallel builds, significantly reducing build times; process managed platform, architecture, and optimization properties; and graphical interface support for adjusting configuration.

2.2 Performance Benchmarks

With the high-level fixes and improvements stated above, it is useful to note the magnitude of a performance improvement these changes produced. To prevent issues with any proprietary models, one of the models from the taxonomy of prototypical feeders was selected (Schneider et al. 2008). In conjunction with population scripts developed as part of other DOE-funded efforts (Schneider et al. 2009, DOE GridAPPS-D 2020), a complex model representing a moderate-sized distribution feeder can be created.

The feeder selected represents a baseline distribution power system, as would typically exist before significant distributed energy resources (DERs), such as solar photovoltaic or micro-wind turbines, are deployed. For renewable integration studies, such as a carrying capacity analysis, this base feeder would slowly be re-simulated with more and more DERs on the system, evaluating at which points characteristics like the voltage level and overall loading level are exceeded. This information can then provide the host utility with either points to focus infrastructure improvements, or limitations on where customer and commercial DERs may be deployed.

For this baseline model, the selected feeder has the following characteristics:

- Simulated for Region 1, representing the west coast of the United States
- Contains 1594 residential homes of varying sizes
- Detailed end-use appliances and end-use schedules are included for each house
- Simulations in August

The performance comparisons for this section include three different sets of GridLAB-D values, with pairs representing the superLU solver and KLU solver results. The individual versions of GridLAB-D are:

- Baseline – Version 4.0, representing GridLAB-D prior to the improvements of this project
- TCF Build – Effectively version 4.3, representing GridLAB-D with all of the improvements of this project
- Microgrid Build – Effectively version 4.3, representing the mainline development branch of GridLAB-D, incorporating many of the improvements of this project, as well as other improvements

Note that the last version is a mix of fixes from this project, from the main GridLAB-D development branch, and from a microgrid-specific build; it includes many other improvements that were beyond the scope of this work. Some of the more fundamental changes from this project are expected to be incorporated in the next major version of GridLAB-D (version 5.0), anticipated to be released by the end of December 2020. The inclusion of the "microgrid" build is to demonstrate how the improvements of this project are aligning with other aspects of GridLAB-D to improve overall performance. All results were obtained by running 15 simulations of the file on an Intel Core i7 processor running at 2.6 GHz, with 32 GB of RAM , running under the Windows 10 operating system.

2.2.1 Large Model Static Simulation

The first comparison involved just doing a static powerflow analysis of the selected model. This is simply executing the GridLAB-D simulation until it reaches an initial converged powerflow. For comparison, the baseline GridLAB-D, the updates of this project (TCF build), and the current GridLAB-D microgrid-development build, are compared. For each case, two simulations are shown: one using the superLU LU decomposition package, and one using the KLU LU decomposition package. Recall from Section 2.1.1.2 that KLU is better suited for the types of LU decomposition powerflow solvers create, but had some licensing issues in the past that prevented it from being directly included. It is also useful to note that version 4.0 (baseline) results are with the older versions of superLU and KLU, representing the performance at the beginning of this project. Table 1 below shows the comparison between the three different simulation results.

Table 1 - Static/Single Powerflow Execution Time for GridLAB-D Versions

Baseline (v4.0)		**TCF (v4.3)**		**Microgrid (v4.3)**	
superLU (old)	KLU (old)	superLU	KLU	superLU	KLU
9.9 s	7.8 s	3.7 s	3.7 s	4.6 s	4.5 s

As Table 1 demonstrates, the improvements from this project are significant. Incorporating the fixes above, the large feeder executes in 3.7 s, as opposed to 9.9 s or 7.8 s with the baseline software. It is important to note that this is for a single timestamp or powerflow solution – much of this time is related to parsing the GLM and overall software overhead. The next section, based around time-series analysis, will focus more on how the improvements of this project have increased the performance of more computationally-intensive simulations.

2.2.2 Large Model Time Series Simulation

The second comparison is to examine how more detailed renewable integration (and carrying capacity) studies may be conducted. For this scenario, the same feeder is run for one week with a 1-minute time resolution (10080 timesteps), examining how the power of the loads changes throughout the day and week. This test is meant to highlight the improvements to the actual solver as it creates multiple solutions, moving through time. The static simulation is still heavily subjected to some high-computational-overhead tasks, such as parsing the text file and linking the objects together internally. The time series shows that after that initial "common cost", the performance improvements are more pronounced.

Table 2 shows the comparison between the same three simulation scenarios. As with the static case, timing information is obtained as an average of 15 individual trials.

Table 2 – Week-long Time-series Execution Time for GridLAB-D Versions

Baseline (v4.0)		**TCF (v4.3)**		**Microgrid (v4.3)**	
superLU (old)	KLU (old)	superLU	KLU	superLU	KLU
40 m 29.8 s	26 m 56.6 s	27 m 14.4 s	21 m 28.6 s	22 m 54.2 s	17 m 54.5 s

Table 2 shows that the week-long simulations show a slightly different result from the static powerflow results. In this simulation scenario, most of the computation time is spent on the model algorithms and powerflow solutions, rather than overhead tasks like reading the model file and organizing the memory. As such, it provides better insight into mechanisms that receive a greater benefit from the optimization and support library updates. The improvements of this project show a significant reduction in the simulation times, with even the superLU results nearly taking less time to complete than the fastest baseline times. For this particular scenario, the "microgrid" build shows even greater improvements, highlighting how these improvements complement other improvements to the GridLAB-D source code, obtaining even faster simulation results.

3.0 GridUnity Collaborative Activity

Most of the direct impacts of this project have been the improvements to GridLAB-D to help support distribution analysis, especially renewable integration and DER evaluation analysis. GridUnity provided scenarios for the analysis, which focused many of the improvements on specific aspects of GridLAB-D. GridUnity's current approach to much of the DER analysis is leveraging the capabilities of commercial software, such as Eaton's CYME CYMDIST software (Eaton Corporation 2020). GridUnity utilizes Amazon Web Services and other cloud-based computing firms to perform the simulations. As such, faster execution times and less server-time have the potential to not only produce results in a timelier fashion, but also reduce usage fees associated with the cloud-based computing services.

Unfortunately, due to the nature of the models (protected by nondisclosure agreements), explicit details of the improvement on the GridUnity side are not available to report. GridUnity continues to evaluate the use of GridLAB-D in their analysis, particularly to help reduce CPU-time costs. PNNL provided some support to help convert the CYME-based models into GridLAB-D format, leveraging the techniques developed under a DOE Grid Modernization Laboratory Consortium (GMLC) project – GridAPPS-D. However, most of the interaction with GridUnity was to focus on which aspects of GridLAB-D to improve (to provide the most benefit), as well as determining model sizes to benchmark the results (see Section 2.2.2).

4.0 Conclusions and Future Work

The performance of GridLAB-D for use in analyzing distribution systems, particularly to evaluate renewable carrying capacity or other DER impacts on the grid, was significantly improved through the efforts of this project. For the standard, superLU-based simulations of a week-long time series, a performance improvement of 32% was realized. Coupled with additional fixes from other GridLAB-D projects, the superLU-based execution improvement increases to 43% for this scenario. For heavily repetitive simulations, such as those for hosting capacity analysis, this can allow for more scenarios to be evaluated for an identical processing time, or a significant savings in fees associated with server time. This not only helps power system consulting companies like GridUnity perform more efficient analysis, but can also enable a cost savings that opens the analysis to more utilities. Furthermore, with all of the improvements pushed to the open-source repository, other DOE programs and research focuses can benefit from the improved performance.

Even with the significant performance improvements from the work of this project, some obvious further improvements are still possible. Items such as more efficient load parsers, better parallelization capabilities, and simulation “check-pointing” (ability to save the exact state at any point in the simulation and pick up exactly where left off) would further improve the usability and performance of GridLAB-D in a variety of power system analysis studies. During the work of this project, it was revealed many of the core-coding concepts from over 10 years ago no longer are compatible with some of these modern capabilities. Work has begun on updating the core programming of GridLAB-D to be compatible with the desirable features. Building off the basis provided by this TCF project, further performance and usability improvements are expected once this is completed.

References

Davis T. and E. P. Nataranjan, "Algorithm 907: KLU, A Direct Sparse Solver for Circuit Simulation Problems," in *ACM Transactions on Mathematical Software*, vol. 37, no. 6, pp 27:1-27:23, 2009.

Demmel J.W., J. R. Gilbert, and X. S. Li, "An Asynchronous Parallel Supernodal Algorithm for Sparse Gaussian Elimination", in *SIAM J. Matrix Analysis and Applications*, vol. 20, no. 4, pp 915-952, 1999.

Eaton Corporation, CYMDIST Distribution System Analysis. Available. [Online] http://www.cyme.com/software/cymdist/. Accessed April 21, 2020.

GridLAB-D Contributors, *GridLAB-D™ Distribution Software*, Available. [Online] https://www.gridlabd.org/, Accessed April 21, 2020.

GridLAB-D Contributors, GridLAB-D Feature 1091, Available. [Online] https://github.com/gridlab-d/gridlab-d/tree/feature/1091, Accessed April 21, 2020.

Schneider K., J. Fuller, F. Tuffner, and Y. Chen, *Modern Grid Strategy: Enhanced GridLAB-D Capabilities Final Report*, Technical Report, PNNL-18864, September 2009.

Schneider K., Y. Chen, D. Chassin, R. Pratt, D. Engel, and S. Thompson, *Modern Grid Initiative Distribution Taxonomy Final Report*, Technical Report, PNNL-18035, November 2008.

Tang Y., J. S. Homer, T. E. McDermott, M. Coddington, B. Sigrin, and B. Mather, *Summary of Electric Distribution System Analyses with a Focus on DERs*, Technical Report, PNNL-26272, April 2017

U.S. Department of Energy, "GridAPPS-D Conceptual Design, v1.0", Available. [Online]. https://gmlc.doe.gov/resources/gridapps-d-conceptual-design-v1.0. Accessed April 21, 2020.

Pacific Northwest
National Laboratory

902 Battelle Boulevard
P.O. Box 999
Richland, WA 99354
1-888-375-PNNL (7665)

www.pnnl.gov