Optimizing Large-Scale Linear Energy System Problems with Block Diagonal Structure by Using Parallel Interior-Point Methods

Thomas Breuer[1], Michael Bussieck[2], Karl-Kiên Cao[3], Felix Cebulla[3], Frederik Fiand[2], Hans Christian Gils[3], Ambros Gleixner[4], Dmitry Khabi[5], Thorsten Koch[4], Daniel Rehfeldt[4], and Manuel Wetzel[3]

[1] Jülich Supercomputing Centre (JSC), Forschungszentrum Jülich GmbH
[2] GAMS Software GmbH
[3] German Aerospace Center (DLR)
[4] Zuse Institute Berlin/Technical University Berlin
[5] High Performance Computing Center Stuttgart (HLRS)

Abstract. Current linear energy system models (ESM) acquiring to provide sufficient detail and reliability frequently bring along problems of both high intricacy and increasing scale. Unfortunately, the size and complexity of these problems often prove to be intractable even for commercial state-of-the-art linear programming solvers. This article describes an interdisciplinary approach to exploit the intrinsic structure of these large-scale linear problems to be able to solve them on massively parallel high-performance computers. A key aspect are extensions to the parallel interior-point solver PIPS-IPM originally developed for stochastic optimization problems. Furthermore, a newly developed GAMS interface to the solver as well as some GAMS language extensions to model block-structured problems will be described.

Keywords: energy system models, linear programming, interior-point methods, parallelization, high performance computing

1 Introduction

Energy system models (ESMs) have versatile fields of application. For example they can be utilized to gain insights into the design of future energy supply systems. Increasing decentralization and the need for more flexibility caused by the temporal fluctuations of solar and wind power lead to increasing spatial and temporal granularity of ESMs. In consequence, state-of-the-art solvers meet their limits for certain model instances.

A distinctive characteristic of many linear programs (LPs) arising from ESMs is their block-diagonal structure with both linking variables and linking constraints. This article sketches extensions of the parallel interior-point solver PIPS-IPM [6] to handle LPs with this characteristic. The extended solver is designed to make use of the massive parallel power of high performance computing (HPC) platforms.

Furthermore, this article introduces an interface between PIPS-IPM (including its new extension) and energy system models implemented in GAMS. In particular, it will be described how users can communicate the model's problem structure to PIPS-IPM. Since finding a proper block structure annotation for a complex ESM is not trivial, we will exemplify the annotation process for the ESM REMix [4]. With many ESMs implemented in GAMS, the new interface between GAMS and PIPS-IPM makes the solver available to the energy modeling community.

2 A Specialized Parallel Interior Point Solver

When it comes to solving linear programs (LPs), the two predominant algorithmic approaches to choose from are Simplex and interior-point, see e.g. [7]. Since interior-point methods are often more successful for large problems, in particular for ESM [1], this method was chosen for the LPs at hand. Mathematically, a salient characteristic of these LPs is their block-diagonal structure with both linking constraints and linking variables, as depicted below

$$
\begin{array}{lllllll}
 & \min & c^T x & & & & \\
\text{s.t.} & T_0 x_0 & & & & = h_0 & (eq_0) \\
 & T_1 x_0 & + W_1 x_1 & & & = h_1 & (eq_1) \\
 & T_2 x_0 & + \quad W_2 x_2 & & & = h_2 & (eq_2) \\
 & \vdots & & \ddots & & \vdots & \\
 & T_N x_0 & + & & W_N x_N & = h_N & (eq_N) \\
 & F_0 x_0 & + F_1 x_1 + F_2 x_2 & \cdots & F_N x_N & = h_{N+1}, & (eq_{N+1})
\end{array}
$$

with $x = (x_0, x_1, ..., x_N)$. The linking variables are represented by the vector x_0, whereas the linking constraints are described by the matrices $F_0, ..., F_N$ and the vector h_{N+1}. The approach to solve this LP is based on the parallel interior-point solver PIPS-IPM [6] that was originally developed for solving stochastic linear programs. Such problems also exhibit a block-diagonal structures, although only with linking variables and without linking constraints. In this way, PIPS-IPM in its original form cannot handle problems with linking constraints. In the last months, the authors of this paper have extended PIPS-IPM in order to handle LPs with both linking constraints and linking variables.

PIPS-IPM and also its new extension make use of the Message Passing Interface (MPI) for communication between their (parallel) *MPI-processes*. An important feature of PIPS-IPM is the distribution of the LP among the MPI-processes with no process needing to store the entire problem. This allows to tackle problems that are too large to even be stored in the main memory of a single desktop machine. The main principle is that for each index $i \in \{0, 1, ..., N\}$ all x_i, h_i, T_i, and W_i (for $i > 0$) need to be available in the same MPI-process—h_{N+1} needs to be assigned to the MPI-process handling $i = 0$. Moreover, each MPI-process needs access to the current value of x_0. The distribution is in the

following exemplified for the case of the information to both $i = 0$ and $i = 1$ being assigned to the same MPI-process (in gray). The vectors and matrices that need to be processed together are marked in gray, black, and bold, respectively.

$$
\begin{array}{lccccccc}
\min & c_0^T x_0 & + & c_1^T x_1 & + & c_2^T x_2 + & \cdots & \mathbf{c_N^T x_N} \\
\text{s.t.} & T_0 x_0 & & & & & & = h_0 \\
& T_1 x_0 & + & W_1 x_1 & & & & = h_1 \\
& T_2 x_0 & + & & W_2 x_2 & & & = h_2 \\
& \vdots & & & & \ddots & & \vdots \\
& \mathbf{T_N x_0} & + & & & & \mathbf{W_N x_N} & = \mathbf{h_N} \\
& F_0 x_0 & + & F_1 x_1 & + & F_2 x_2 & \cdots \quad \mathbf{F_N x_N} & = h_{N+1}
\end{array}
$$

The maximum of MPI processes that can be used is N; in the opposite border case the whole LP is assigned to a single MPI-process

The extension of PIPS-IPM has already been successfully tested on medium-scale ESM problems with up to a million constraints and variables and up to 90 blocks. Since the number of MPI-processes is bounded by the number of blocks, the maximum number of MPI-processes we have used so far is also 90.

3 Communicating Block Structured GAMS Models to PIPS-IPM

A recently implemented GAMS/PIPS-IPM interface that considers the special HPC platform characteristics makes the solver available to a broader audience. This section is twofold. It outlines how users can annotate their GAMS models to provide a processable representation of the model block structure and provides insights in some technical aspects of the GAMS/PIPS-IPM-Link.

3.1 Annotating GAMS Models to Communicate Block Structures

Automatic detection of block structures in models is challenging [3], hence, a processable block structure information based on the user's deep understanding of the model is often preferable. It is important to note that there is no unique block structure in a model but there are many of them, depending on how rows and columns of the corresponding matrix are permuted. For ESMs blocks may for example be formed by regions or time steps as elaborated in section 4.

GAMS provides facilities that allow complex processable model annotations [2]. The modeler can assign stages to variables via an attribute `<variable name>.stage`. That functionality originates from multistage stochastic programming and can also be used to annotate the block structure of a model to be solved with PIPS-IPM. Once the block membership for all variables is annotated, the block membership of the constraints can in principle be derived from that annotation. However, manual annotation of constraints in a similar fashion is also possible and

allows to run consistency checks on the annotation to detect potential mistakes. The annotation assignment can be demonstrated with a simple example based on the block structure introduced in section 2. The following pseudo-annotation would assign stages to all variables x_i to indicate their block membership.

$$x_i.stage = i \qquad \forall i \in \{0, 1, ..., N\}$$

Linking variables are those assigned to stage 0. Similarly, constraints could also be annotated where stage 0 constraints are those containing only linking variables. Constraints assigned to stages 1,..,N are those incorporating only variables from the corresponding block plus linking variables and finally constraints assigned to stage N+1 are the linking ones. Note that the exemplary pseudo-annotation may seem obvious and simple but finding a good block structure annotation for a complex model is not trivial. The challenge is not mainly to find an annotation that is correct in the mathematical sense but to find one where the power of PIPS-IPM is exploited best. A desirable annotation would reveal a block structure with many independent blocks of similar size while the set of linking variables and linking constraints is small.

3.2 The GAMS/PIPS-IPM-Link

Currently, the GAMS/PIPS-IPM-Link implements the connection between modeling language and the solver in a two-phase process. Phase 1, the model generation, is followed by phase 2 where PIPS-IPM pulls the previously generated model via its callback interface and solves the problem.

So far, model generation used to be a sequential process where GAMS generates one constraint after another. For the majority of applications this is fine as model generation is usually fast and the time consumption is negligible compared to the time consumed to solve the actual problem. However, some ESMs may result in sizeable LPs where model generation time becomes relevant. Hence, it is worthwhile to mention that the previously introduced annotation can also serve as a basis to generate the model in a distributed fashion. Instead of generating one large monolithic model, many small model blocks can be generated in parallel to exploit the power of HPC architectures already during model generation.

4 Structuring Energy System Models for PIPS-IPM

In order to distribute all blocks of the full-scale ESM to the computing nodes of a HPC architecture a problem-specific model annotation has to be provided. Based on the modeler's knowledge about the problem at hand the number of blocks and block structure has to be decided upon corresponding directly to the assignment of variables to blocks.

The concurrency of supply and demand of electrical energy necessitates a balancing for every region and time step. While in theory these balancing constraints can be solved independently, transport of energy between regions and storage

of energy require a integrated optimization of all regions and time steps. The number of variables and constraints linked by the annotation depends strongly on these spatial and temporal interconnections. Transport of energy between two regions is typically represented by dispatch variables leading to linking variables if their respective regions have been assigned to different blocks. State of charge variables for energy storages consider the state of charge in the previous time step and therefore lead to a large number of linking constraints if each time step is represented by a single block. Typically, ESM also comprise boundary conditions that link both regions and time steps, e.g. by the consideration of global and annual emission limits. The high number of linking variables and constraints lead to a trade-off between speed-up and parallelism that needs to be studied systematically in future numerical experiments.

Figure 1 shows the non-zero entries matrix of the ESM REMix [4] on the left side and the revealed underlying block structure after permutation of the matrix on the right side. Linking variables and constraints are marked in dark gray while PIPS-IPM blocks are marked in light gray. The ESM represents the electricity sector for Germany with 21 spatial regions, 17 technologies per region and 168 time steps respectively 7 blocks of 24 time steps in the annotated case.

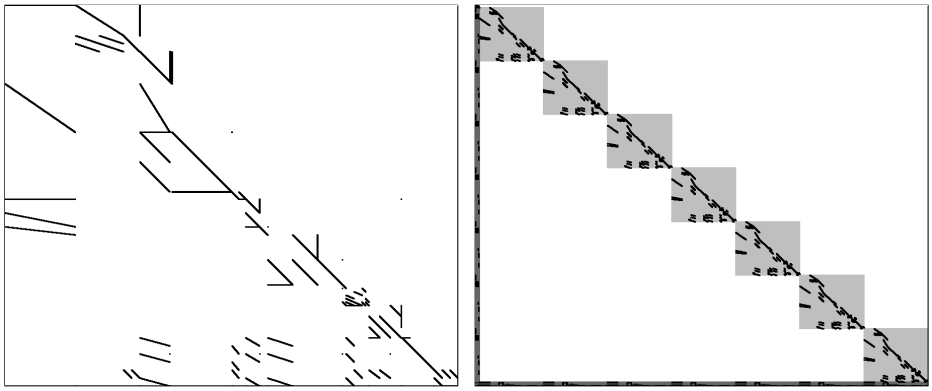

Fig. 1. Non-zero entries of the ESM and permuted matrix with block structure

5 Summary and Outlook

Large-scale LPs emerging from ESMs that are computationally intractable for today's state-of-the-art LP solvers motivate the need for new solution approaches. To serve those needs, extensions to the parallel interior point solver PIPS-IPM that exploits the parallel power of high performance computers have been implemented. In the future, the solver will be made available to the ESM community by a GAMS/PIPS-IPM interface.

The integration of HPC specialists in the development process ensures consideration of peculiarities of several targeted HPC platforms at an early stage of development. PIPS-IPM is developed and tested on several target platforms like the petaflops systems Hazel Hen at HLRS and JURECA at JSC as well as on many-core platforms like JUQUEEN and modern Intel Xeon Phi Processors. Workflow automation tools explicitly designed for HPC applications like JUBE [5] support the development and execution by simplifying the usage of workflow managers like PBS and Slurm.

Initial computational experiments already show the capability of the extended PIPS-IPM version to solve the ESM problems at hand, although so far only on a small scale. However, the good scaling behavior and the results of the original PIPS-IPM in solving large-scale problems [6] suggest that the approach described in this article might ultimately lead to a solver that can tackle currently unsolvable large-scale ESMs. Extensions to the GAMS/PIPS-IPM-Link will finally integrate the current multi-phase workflow (see section 3.2) into one seamless process to give energy system modelers a similar workflow compared to the use of conventional LP solvers.

Acknowledgements

The described research activities are funded by the Federal Ministry for Economic Affairs and Energy within the BEAM-ME project (ID: 03ET4023A-F). Ambros Gleixner was supported by the Research Campus MODAL *Mathematical Optimization and Data Analysis Laboratories* funded by the Federal Ministry of Education and Research (BMBF Grant 05M14ZAM).

References

1. Cao, K., Gleixner, A., Miltenberger, M.: Methoden zur Reduktion der Rechenzeit linearer Optimierungsmodelle in der Energiewirtschaft - Eine Performance-Analyse. In: EnInnov 2016: 14. Symposium Energieinnovation (2016)
2. Ferris, M.C., Dirkse, S.P., Jagla, J., Meeraus, A.: An Extended Mathematical Programming Framework. In: Computers & Chemical Engineering, vol. 33, pp. 1973-1982 (2009), doi:10.1016/j.compchemeng.2009.06.013
3. Ferris, M.C., Horn, J.D.: Partitioning mathematical programs for parallel solution. In: Mathematical Programming, vol. 80, pp. 35-61 (1998), doi:10.1007/BF01582130
4. Gils, H.C. et al.: Integrated modelling of variable renewable energy-based power supply in Europe. In: Energy 123, 173-188 (2017), doi:10.1016/j.energy.2017.01.115
5. Luehrs, S. et al.: Flexible and Generic Workflow Management. doi:10.3233/978-1-61499-621-7-431
6. Petra, C.G., Schenk, 0., Anitescu, M.: Real-time Stochastic Optimization of Complex Energy Systems on High Performance Computers. In: Computing in Science & Engineering (CiSE) 16(5), pp. 32-42 (2014)
7. Vanderbei, R.J.: Linear Programming: Foundations and Extensions. Springer (2014)
8. Schenk, O, Gartner, K.: On Fast Factorization Pivoting Methods for Sparse Symmetric Indefinite Systems. Technical Report, Department of Computer Science, University of Basel (2004)