15. Symposium Energieinnovation, 14. bis 16. Februar 2018, Technische Universität Graz, www.EnInnov.TUGraz.at
ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

Vergleich kommerzieller und freier Solver bei der Durchführung von Netzbetriebssimulationen

Christian Bredtmann, Jan Kellermann, Albert Moser

Institut für Elektrische Anlagen und Energiewirtschaft (IAEW) an der RWTH Aachen,
Schinkelstr. 6, 52062 Aachen, br@iaew.rwth-aachen.de, www.iaew.rwth-aachen.de

Kurzfassung: Im Übertragungsnetz kommt es durch die fortschreitende Integration Erneuerbarer Energien zu einer stetig steigenden Anzahl von Netzengpässen, denen mit netzbezogenen Maßnahmen, Redispatch oder Einspeisemanagement begegnet wird. Der optimale Einsatz der verfügbaren Maßnahmen kann mithilfe von Optimierungsansätzen, die einem Optimal Power Flow Ansatz ähneln, bestimmt werden. Für typische Anwendungen im europäischen Übertragungsnetz ergeben sich Probleme mit über 5.000 Freiheitsgraden, 10.000 Nebenbedingungen und Matrizen mit Millionen von Einträgen ungleich Null. Für rechenzeitintensive Operationen wie dem Lösen linearer Gleichungssysteme sowie dem Lösen (gemischt ganzzahliger) linearer Probleme bietet sich die Nutzung bestehender und leistungsfähiger Bibliotheken an. In diesem Beitrag werden ausgewählte kommerzielle und freie Bibliotheken untersucht und für realitätsnahe Optimierungsprobleme und Problemgrößen im Kontext von Netzbetriebssimulationen gegenübergestellt. Die Ergebnisse zeigen signifikante Unterschiede der Rechenzeiten zwischen den getesteten Bibliotheken und die Grenzen der mit frei zugänglichen Bibliotheken lösbaren Problemgrößen auf.

Keywords: Redispatch, Übertragungsnetz, Solver, Gleichungssystem, Optimierung, GGLP

1 Hintergrund und Motivation

Die Struktur der Energieerzeugung in Deutschland hat sich in den letzten Jahren, unter anderem aufgrund gesetzlicher Vorgaben zur Förderung regenerativer Erzeugungsanlagen, stark gewandelt. Viele dargebotsabhängige Erzeugungsanlagen auf Basis Erneuerbaren Energien (EE-Anlagen) speisen vermehrt dezentral und lastfern ein. Die Folge ist ein ständiger Zuwachs des Transportbedarfs im Übertragungsnetz. Der dafür nötige Ausbau des Netzes geht jedoch nur langsamer voran als der Ausbau der EE-Anlagen. Dadurch kommt es im Übertragungsnetz zu einer steigenden Anzahl von Netzengpässen, die durch netzbezogene Maßnahmen, Redispatch und Einspeisemanagement behoben werden. Im Jahr 2016 hat die Gesamtmenge an Redispatch in Deutschland dabei etwa 11.475 GWh entsprochen [1].

Mit Hilfe von Netzbetriebssimulationen können Redispatch- und Einspeisemanagement-einsätze optimiert werden. Für das Aufstellen und Lösen des Optimierungsproblems sind in der Regel zwei rechenzeitintensive Vorgänge erforderlich. Im Rahmen einer Leistungs-flussberechnung erfolgt eine iterative Linearisierung der Systemgleichungen (numerisches Lösen). Das so gewonnene lineare Gleichungssystem (LGS) beschreibt auf Basis der Funktionalmatrix $\boldsymbol{J}$ die Zusammenhänge zwischen den Leistungseinspeisungen/-entnahmen je Netzknoten und den komplexen Knotenspannungen. Durch vielfaches Lösen des LGS für verschiedene rechte Seiten können nun Sensitivitäten der Leistungseinspeisung je Netzknoten auf die Knotenspannungen im gesamten Netz und damit auch auf die Leistungsflüsse je Zweig

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

abgeleitet werden. Diese Sensitivitäten werden für die Formulierung des Optimierungsproblems benötigt, welches in der Regel als „Lineares Problem“ (LP) oder bei Berücksichtigung von Einschaltentscheidungen als „Gemischt Ganzzahlig Lineares Problem“ (GGLP, englisch MILP) formuliert wird. Die zu minimierende Zielfunktion kann dabei bspw. die durch Redispatch angepasste Energiemenge oder die Redispatchkosten enthalten. Nebenbedingungen sind in der Regel die Stromgrenzen von Zweigen im ungestörten Betrieb und in Ausfallsituationen, minimale und maximale Einspeisungen je Erzeugungsanlage sowie bei zeitkoppelnden Betrachtungen maximale Leistungsgradienten sowie Mindestbetriebs- und Mindeststillstandszeiten. Die Freiheitsgrade umfassen die Wirkleistungseinspeisungen von Kraftwerken und EE-Anlagen, können aber auch um Blindleistungsbereitstellung oder Transformatorstufenstellungen sowie Ein- und Ausschaltentscheidungen ergänzt werden.

Für typische Anwendungen im europäischen Übertragungsnetz können sich hierbei schnell Problemformulierungen mit über 5.000 Freiheitsgraden, 10.000 Nebenbedingungen und Matrizen mit Millionen von Einträgen ungleich Null (Non-Zeros) ergeben. Für beide rechenzeitintensive Operationen, das Lösen großer LGS sowie das einmalige Lösen eines LP/GGLP bietet sich die Nutzung bestehender, leistungsfähiger Bibliotheken und Optimierungsumgebungen an. Ein wichtiges Kriterium für den Einsatz in der Wissenschaft aber auch bei der gewerblichen Nutzung sind Verfügbarkeit und Kosten der Solver. Diese reichen von kostenlosen Open Source Bibliotheken bis hin zu kommerziellen Produkten, deren Lizenzen mehrere Tausend Euro kosten. In diesem Beitrag werden ausgewählte kommerzielle und frei verfügbare Bibliotheken für realitätsnahe Optimierungsprobleme bei Netzbetriebssimulationen für das Übertragungsnetz gegenübergestellt und bewertet.

2 Problembeschreibung

Der allgemeine Ablauf des Optimierungsverfahrens ist in Abbildung 1 dargestellt. Eine detaillierte mathematische Formulierung des Optimierungsproblems findet sich in [2] und [3].

Basierend auf einem initialen AC-Lastfluss (AC-LF) für das zu untersuchende System erfolgt im ersten Schritt eine Ausfallsimulation, welche die Auswirkungen von Betriebsmittelausfällen mittels einer Anpassung der Funktionalmatrix aus der Lastflussberechnung sowie Leistungsinjektionen an den Anschlussknoten der ausgefallenen Betriebsmittel approximiert [2], [3]. Dazu muss für jeden zu simulierenden Ausfall ein LGS gelöst werden.
Nachfolgend werden Sensitivitäten bestimmt, die den linearisierten Einfluss von Wirk- und/oder Blindleistungsänderungen an einem Netzknoten auf die Knotenspannungen im gesamten Netz abbilden. Dazu muss für jeden Knoten, an dem die Leistungsbilanz im Rahmen der Optimierung angepasst werden kann, ein LGS je Situation (ungestörter Betrieb und Betriebsmittelausfälle) gelöst werden.
Unter Nutzung der Sensitivitäten wird anschließend ein LP oder GGLP formuliert. Die Zielfunktion umfasst Kosten für den Einsatz von Redispatch sowie Strafkosten für überlastete Leitungen oder Spannungsbandverletzungen. Unter Einsatz der Freiheitsgrade (z.B. Leistungsanpassung von Einspeisungen) wird die Zielfunktion minimiert.

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

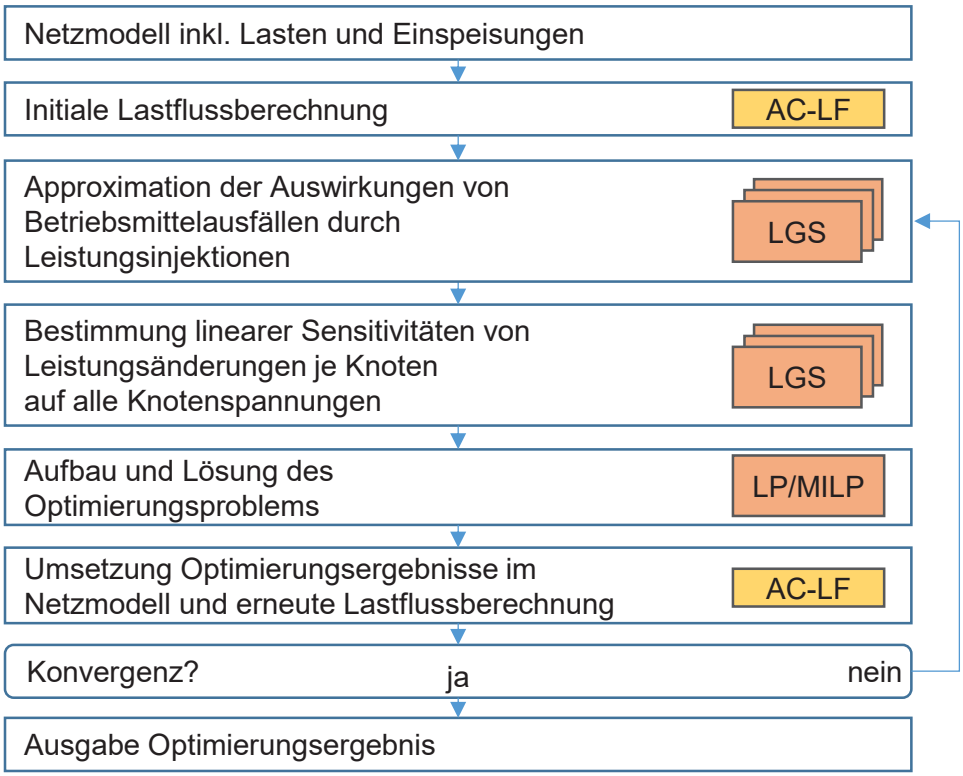

Abbildung 1: Allgemeiner Ablauf des Optimierungsverfahrens

Das Optimierungsproblem sowie die Bestimmung der Sensitivitäten, für welche die meisten rechenzeitintensiven Operationen stattfinden, werden im Folgenden genauer beschrieben.

2.1 Formulierung des Optimierungsproblems

Das Optimierungsproblem wird in der Standardform für LP bzw. GGLP zur Minimierung einer linearen Zielfunktion z formuliert:

$$min\, z(\vec{x}) = \{\vec{c}^{\,T} \cdot \vec{x} \;\; | \;\; \vec{b}_l \leq \boldsymbol{A} \cdot \vec{x} \leq \vec{b}_u\} \tag{2.1}$$

Dabei können die einzelnen Lösungsvariable x_i entweder reelle oder bei einer Formulierung als GGLP ganzzahlige (z.B. Transformatorstufenstellungen) bzw. binäre (z.B. Ein-/Ausschaltentscheidungen) Werte annehmen. Der Vektor $\vec{c}$ beschreibt dabei die Kostenanteile der einzelnen Lösungsvariablen, $\vec{b}_l$ und $\vec{b}_u$ beschreiben die Nebenbedingungsgrenzen.

Die in (2.1) einzuhaltenden Nebenbedingungen umfassen im Wesentlichen die Stromgrenzen relevanter Zweige, minimale und maximale Leistungseinspeisungen je Erzeugungsanlage, maximal zulässige Leistungsgradienten zwischen aufeinanderfolgenden Zeitschritten und Mindestbetriebs- und Mindeststillstandszeiten. Sie können je nach Problemformulierung jedoch auch um Füllstandsnebenbedingungen bei Speichern oder Grenzen der Transformatorstufenstellungen ergänzt werden, sofern diese mit optimiert werden sollen.

Die Zielfunktion z kann je nach Ziel der Optimierung unterschiedlich gewichtete Zielfunktionsbeiträge berücksichtigen. Die Wesentlichen Beiträge sind in Formel (2.2) dargelegt.

$$z(\vec{x}) = C_{Redispatch}(\vec{x}) + C_{Netz}(\vec{x}) + C_{GWV}(\vec{x}) \tag{2.2}$$

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

Der Kostenterm $C_{Redispatch}$ bildet Kosten für die Wirkleistungsanpassung an Erzeugungsanlagen ab. Ziel ist es, mit möglichst geringen Kosten für Leistungsänderungen einen Systemzustand ohne Grenzwertverletzungen zu erreichen. Der Kostenterm C_{Netz} ergänzt einen Zielfunktionsbeitrag für die Nutzung netzbezogener Freiheitsgrade, um eine möglichst geringe Anzahl an Zustandsänderungen im Netzbetrieb zu erreichen. Der Kostenterm C_{GWV} bildet Strafkosten für Grenzwertverletzungen wie Leitungsüberlastungen oder Spannungsbandverletzung ab, da diese in Form von Slack-Variablen in die Optimierung einfließen.

2.2 Bestimmung der Sensitivitäten und Ausfallsimulation

Ein wichtiger Bestandteil der linearen Problemformulierung sind Sensitivitäten der Leistungsbilanzen je Knoten auf alle Knotenspannungen. Zur Bestimmung der Sensitivitäten wird die im Rahmen einer AC-Lastflussberechnung bestimmte Funktionalmatrix bzw. Jacobimatrix $\boldsymbol{J}$ genutzt, welche die Zusammenhänge zwischen den Leistungseinspeisungen/-entnahmen je Netzknoten $\underline{\vec{S}}$ und den Knotenspannungen $\underline{\vec{U}}$ der nichtlinearen Lastflussgleichungen (2.3) für einen Arbeitspunkt linearisiert (2.4). Ziel der Sensitivitätsbestimmung ist es, für jeden Knoten j den Einfluss einer Änderung der Wirk-/Blindleistung auf alle Knotenspannungen linear abzuschätzen. Dies kann wie in (2.5) angedeutet durch Leistungsinjektionen je Netzknoten und die Bestimmung der resultierenden Spannungsänderungen erfolgen. Anders als in (2.5) angedeutet muss dafür nicht zwingend die Jacobimatrix invertiert werden, sondern es kann das in (2.4) formulierte LGS für viele rechte Seiten $\vec{S}_j^{inj}$ gelöst werden. Ein ähnliches Vorgehen wird im Rahmen der Ausfallapproximation angewandt, um die Umverteilung der Leistungsflüsse durch ausfallende Betriebsmittel berechnen zu können.

$$\underline{\vec{S}} = 3 \cdot \boldsymbol{diag}\left(\underline{\vec{U}}\right) \cdot \underline{\boldsymbol{Y}}^* \cdot \underline{\vec{U}}^* \tag{2.3}$$

$$\boldsymbol{J} \cdot \Delta\underline{\vec{U}} = \Delta\underline{\vec{S}} \;\Leftrightarrow\; \boldsymbol{J} \cdot \begin{pmatrix} \Delta U \\ \Delta\theta \end{pmatrix} = \begin{pmatrix} \Delta P \\ \Delta Q \end{pmatrix} \tag{2.4}$$

$$\begin{pmatrix} \Delta U \\ \Delta\theta \end{pmatrix} = \boldsymbol{J}^{-1} \cdot \vec{S}_j^{inj}, \qquad \vec{S}_j^{inj} = \left(0 \cdots 0, P_j^{inj}, Q_j^{inj}, 0 \cdots 0\right)^T \tag{2.5}$$

Typischerweise gliedert sich das Lösen von linearen Gleichungssystemen in drei Schritte:

1. Symbolische Faktorisierung
2. Numerische Faktorisierung
3. Lösen für eine oder mehrere rechte Seiten (engl. Right-Hand-Sides, RHS)

Für die Sensitivitätsbestimmung kann für alle rechten Seiten die symbolische und numerische Zerlegung weiterverwendet werden, da sich die Matrix nicht verändert. Für die Berechnung von Ausfallsituationen gilt dies nicht, da sich die Admittanz- und damit auch die Jacobi-Matrix in jeder Ausfallsituation ändern können. Typischerweise bleibt in den Ausfallsituationen jedoch die Struktur der Matrix erhalten, d.h. es verändern sich nur die numerischen Zahlenwerte in der Matrix. In diesem Fall kann die symbolische Zerlegung der Matrix beibehalten werden.

Die hier zu lösenden Matrizen haben eine Größe, die der doppelten Knotenanzahl des Netzes entspricht. In realitätsnahen Netzmodellen sind es typischerweise über 20.000 Zeilen/Spalten. Die Form der Jacobi-Matrix kann sich implementierungsabhängig unterscheiden, ist jedoch im Allgemeinen quadratisch, sehr dünnbesetzt und hat keine spezielle Form (keine Bandmatrix,

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

nicht symmetrisch, etc.). Oftmals kann die Jacobi-Matrix der Leistungsflussgleichungen jedoch in eine Block-Dreiecksmatrix überführt werden. LGS mit Matrizen dieses allgemeinen Typs werden mit LU-Zerlegung gelöst. Daher werden im Folgenden nur solche Solver untersucht.

Sind weitere Matrixeigenschaften wie Symmetrie oder Definitheit bekannt, können speziellere Solver Anwendung finden, die durch Ausnutzung der speziellen Eigenschaften einen entsprechenden Rechenzeitvorteil haben können.

3 Vorstellung Solver

In diesem Abschnitt werden ausgewählte Bibliotheken vorgestellt, welche auf die zuvor beschriebenen Berechnungen spezialisiert sind.

3.1 Solver für lineare Gleichungssysteme

Tabelle 1 listet eine Auswahl bekannter Solver auf, die zum Lösen der LGS bzw. zur Bestimmung der Sensitivitäten für die Netzbetriebssimulation geeignet sind. Alle vorgestellten Solver sind spezialisiert auf dünnbesetzte Matrizen. Sie stehen entweder unter einer Open Source oder eine Community Lizenz und sind damit insbesondere für nicht-kommerzielle Zwecke verwendbar. Die einzelnen Lösungsalgorithmen entstammen drei verschiedenen Bibliotheken. Alle Bibliotheken sind gut dokumentiert, werden aktiv weiterentwickelt und von einer größeren Community genutzt.

Name	**Lizenz**	**Letzte Aktualisierung** ***(Stand 01/2018)***
Eigen SparseLU	Open Source, Mozilla MPL2	Eigen3-beta, 17.01.2018
Eigen BiCGSTAB	Open Source, Mozilla MPL2	Eigen3-beta, 17.01.2018
SuiteSparse UMFPACK	Open Source, GNU GPL 2.0 Modul CHOLMOD: GNU LGPL 2.1 Modul AMD: BSD 3-clause	v5.7.6, 05/2016
SuiteSparse KLU	Open Source, GNU LGPL 2.1 Mod. AMD/COLAMD: BSD 3-clause	v1.3.8, 05/2016
Intel MKL PARDISO	Kommerziell, Community und Academic Lizenzen	v2018.0.1, 10/2017

Tabelle 1: Übersicht Solver für lineare Gleichungssysteme

3.2 Solver für das Optimierungsproblem

Zur Lösung der in Kapitel 2.1 formulierten LP/GGLP kann eine Vielzahl frei verfügbarer oder kommerzieller Solver genutzt werden. In Voruntersuchungen wurden mehrere Solver identifiziert, die für die vorliegenden Problemgrößen in Frage kommen. Ein Auszug dieser Solver wird in Tabelle 2 dargestellt. Es zeigt sich, dass eine Vielzahl kommerzieller und frei verfügbarer Solver existiert, die zurzeit aktiv weiterentwickelt werden und von einer größeren Community genutzt werden.

Es wurden bereits Vergleiche und Benchmarks durchgeführt, die eine weitere Einschränkung der geeignetsten Solver für die vorliegenden Optimierungsprobleme zulassen. So wurden in

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

[12] und [13] Benchmarks für LP durchgeführt, die einen Großteil der genannten Solver abdecken. Insbesondere die kommerziellen Solver CPLEX, Gurobi und Xpress zeigten hier die kürzesten Rechenzeiten. Von den frei verfügbaren Lösungen schnitten SCIP und Cbc/Clp am besten ab. Ein ähnliches Ergebnis zeigte sich bei den gelösten Probleminstanzen. Auch hier lagen die kommerziellen Solver vor SCIP und Cbc/Clp. Deutlich weniger Instanzen konnten von LP_SOLVE und GLPK gelöst werden.

Name	**Lizenz**	**Letzte Aktualisierung** ***(Stand 01/2018)***	**Quelle**
CPLEX	kommerziell	v12.8, 10/2017	[4]
Gurobi	kommerziell	v7.5.1, 2017	[5]
Xpress	kommerziell	v8.4.5, 01/2018	[6]
MINOS	Kommerziell	v5.5, 2013	[7]
COIN.OR (CLlp,Cbc,Cgl)	Open Source, Eclipse Public License	Clp v1.16.11, 06/2017 Cbc v2.9.9, 06/2017 Cgl v0.59.10, 06/2017	[8]
GLPK (GNU Linaer Programming Kit)	Open Source, GNU GPL	v4.64, 12/2017	[9]
SCIP Optimization Suite	Open Source, ZIB Academic License	v5.9, 12/2017	[10]
LP_SOLVE	Open Source, GNU LGPL	v5.5, 09/2016	[11]

Tabelle 2: Auswahl häufig genutzter Solver für LP/GGLP

Ein weiterer Vergleich in [14] zieht ebenfalls den Schluss, dass die verfügbaren freien Solver nicht an die Performance der kommerziellen Lösungen heranreichen, jedoch trotzdem prinzipiell in der Lage sind, Probleme in den vorliegenden Größenordnungen zu lösen. Von den getesteten freien Solvern wurde Clp/Cbc als der leistungsfähige identifiziert. Weiterhin zeigte der Solver MINOS schlechtere Ergebnisse als CPLEX. Weitere Benchmarks werden auch von den Herstellern der kommerziellen Solver selber, z.B. in [15], veröffentlicht.
Auf Basis der Literaturrecherche sowie der verfügbaren Lizenzen kommerzieller Software wurden für diesen Beitrag die kommerziellen Solver CPLEX und Gurobi sowie die Open Source Lösung Coin-OR (Clp bzw. Cbc) ausgewählt.

4 Methodisches Vorgehen

Im Folgenden werden das methodische Vorgehen und die Konfiguration der einzelnen Solver vorgestellt.

Alle linearen Solver verwenden als BLAS/LAPACK-Bibliothek die Intel Math Kernel Library (MKL). Die einzige Ausnahme bildet KLU, da die Implementierung intern keine BLAS nutzt. Die MKL wird in der Version 2018.0.1 verwendet. Für alle durchgeführten Untersuchungen wird als Lineare-Algebra/Matrix-Bibliothek Eigen3 in der Version vom 17.01.2018 verwendet. Die für die Untersuchungen relevanten dünnbesetzten Matrizen liegen in Compressed-Sparse-Column-(CSC)-Form vor.

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

Die Solver für die LGS werden in der neusten Version entsprechend Tabelle 1 genutzt. Im Falle der Optimierer werden Gurobi und Cbc/Clp/Cgl in der neusten Version entsprechend Tabelle 2 eingesetzt. Abweichend dazu wird CPLEX in der Vorgängerversion 12.7.0 genutzt. Da Gurobi und CPLEX für die Bestimmung der GGLP-Lösung Parallelisierung nutzen, ist dementsprechend auch Cbc mit einer Thread-Parallelisierung kompiliert worden.

Die modulare Struktur des Optimierungsframeworks ermöglicht es, den eingesetzten Solver auszutauschen, ohne dabei die restlichen Teile der Berechnungen zu beeinflussen. Dies sichert eine gute Vergleichbarkeit der verschiedenen Solver bei sonst gleichen Bedingungen.

Zur Beurteilung der Leistungsfähigkeit der Solver wird im Rahmen dieses Beitrags die real verstrichene Zeitdauer („wall-clock time“) für die Berechnung der Rechenschritte gemessen. Die eingesetzte Zeitmessungsfunktion hat eine Auflösung von einer Nanosekunde und ist daher für die Auswertung geeignet. Zur Sicherstellung eines robusten Ergebnisses sind die Laufzeitmessungen mehrfach durchgeführt und ein entsprechender Mittelwert gebildet worden. Zur Bestimmung der Güte wird bei den Solvern für LGS eine Berechnung des Residuums durchgeführt, da die exakte Lösung nicht bekannt ist: $r = \|b - A \cdot x_0\|$

Um Einflüsse eines bereits „heißen“ Caches etc. zu vermeiden und ein möglichst realitätsnahes Ergebnis zu erhalten, sind die Untersuchungen nicht mittels Mikrobenchmarks, sondern im Rahmen eines normalen Optimierungslaufs durchgeführt worden.

5 Exemplarische Ergebnisse

Basierend auf dem vorgestellten methodischen Vorgehen werden anhand eines realen Problems die Leistungsfähigkeit und Robustheit der Solver für verschiedene Problemgrößen und Eigenschaften untersucht.

5.1 Untersuchtes System

Bei dem untersuchten System handelt es sich um ein Modell des europäischen Übertragungsnetzes, bestehend aus etwa 11.000 Knoten, 13.000 Zweigen und 600 steuerbaren Elementen (darunter insbesondere Kraftwerke und Phasenschiebertransformatoren).

Die Untersuchungen wurden auf einem Linux System mit Intel Xeon CPU E5-2670 v3 mit insgesamt 24 Kernen mit 2 Threads je Kern (Hyperthreading) mit 30MB L3-Cache und 256 GB RAM durchgeführt.

5.2 Vergleich der Solver linearer Gleichungssysteme

Zunächst werden die Untersuchungen bezogen auf die Ermittlung der Sensitivitäten diskutiert. Die gemittelten Größen aller untersuchten Jacobi-Matrizen sind in Tabelle 3 dargestellt. Wie zu erkennen ist, sind die Matrizen zwar relativ groß bezogen auf die Anzahl an Zeilen/Spalten, es sind jedoch nur unter einem Promille der Einträge in der Matrix ungleich Null.

System	Anzahl Zeilen / Spalten	Anzahl Non-Zeros	Anteil Non-Zeros	Anzahl zu lösender rechte Seiten
Jacobi-Matrix	21.702	138.180	0,029 %	637

Tabelle 3: Eigenschaften Jacobi-Matrix

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

In Abbildung 2 sind neben den auf den schnellsten Solver skalierten Gesamtrechenzeiten auch die Residuen der einzelnen Solver gezeigt.

Wie zu erkennen ist, bietet KLU für das vorliegende Problem die höchste Performance. Alle anderen Implementierungen sind mindestens um den Faktor 3 langsamer. Neben dem iterativen Solver BiCGSTAB zeigt der Intel PARDISO Solver die geringste Performance. Letzterer bietet jedoch die Möglichkeit, mehrere Threads parallel zu nutzen, wodurch sich die Performance bereits ab zwei Threads ([nT] entsprechen n Threads) deutlich verbessert. Das Residuum ist für alle Solver in einem ähnlich guten Bereich von maximal 1.35E-12.

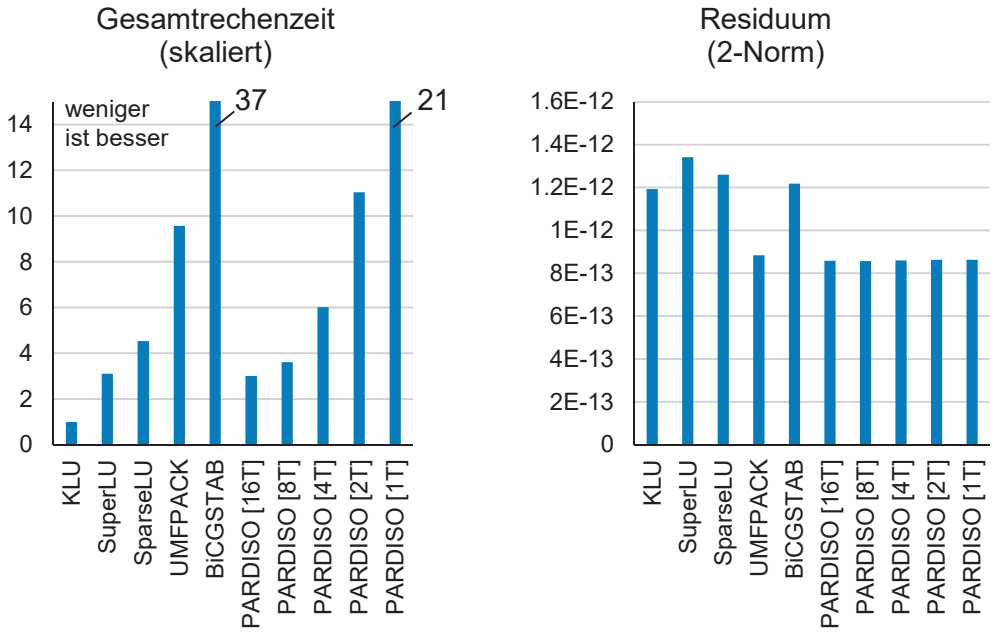

Abbildung 2: Skalierte Gesamtrechenzeit und Residuum

5.3 Vergleich der Solver für das Optimierungsproblem

Nachfolgend werden die Ergebnisse der verschiedenen Solver für das LP/GGLP diskutiert. Es werden drei GGLP mit einer unterschiedlichen Anzahl konsekutiver Stunden mit Zeitkopplungen sowie drei LP ohne Zeitkopplungen und ohne Ganzzahligkeitsentscheidungen untersucht. Die Anzahl an Variablen und Nebenbedingungen kann Tabelle 4 entnommen werden.

Problem	Anzahl betrachteter Stunden	Anzahl Variablen gesamt	Anzahl Variablen ganzzahlig / binär	Anzahl Nebenbedingungen	Anzahl Non-Zeros
LP1	1	610	0	1.133	338.410
LP12	12	7.037	0	14.019	3.737.693
LP24	24	13.798	0	28.565	7.155.304
GGLP1	1	2.264	974	2.093	740.198
GGLP12	12	32.211	10.860	32.535	8.484.389
GGLP24	24	63.340	21.000	65.907	16.664.620

Tabelle 4: Systemeigenschaften

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

Für die Untersuchungen ist bei allen Optimierungsalgorithmen das relative MIP Gap einheitlich auf 0,01 und die Anzahl an nutzbaren Threads auf 32 Threads gesetzt worden.

Die Ergebnisse der Untersuchungen sind in Abbildung 3 dargestellt. Auch hier sind die Rechenzeiten auf den jeweils schnellsten Solver je Problem skaliert. Bei den LP zeigt sich Gurobi bei dem vorliegendem Optimierungsproblem als schnellster Solver, CPLEX ist bei dem gleichen Problem um den Faktor 1.5 - 2 langsamer. Die Performance des freien Solvers Cbc ist gegenüber den proprietären Produkten nicht vergleichbar. Dies zeigt sich insbesondere bei den GGLP bei denen Gurobi und CPLEX eine annähernd gleiche Performance bieten, Cbc jedoch bei dem größeren Problem „GGLP12“ um den Faktor 165 langsamer ist und bei der 24-stündigen Optimierung „GGLP24“ keine Lösung in praxistauglicher Rechenzeit gefunden hat. Die Zielfunktionswerte sind für die LP exakt identisch, für die GGLP sind diese in Abbildung 3 dargestellt und, wie zu erkennen, bei gefundener Lösung sehr ähnlich.

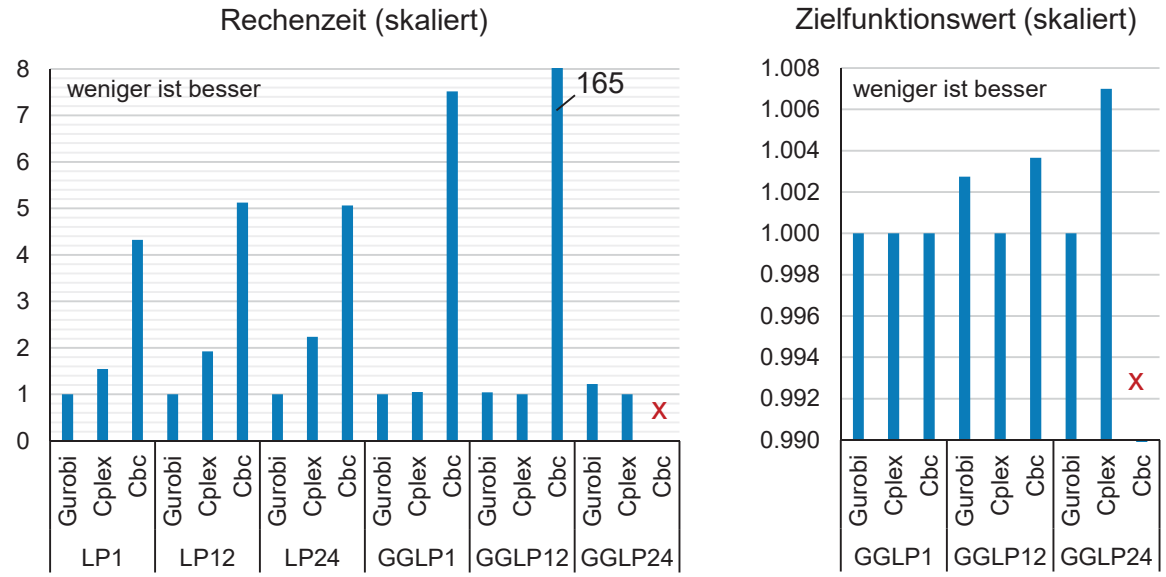

Abbildung 3: Skalierte Rechenzeit und GGLP-Zielfunktionswerte

6 Zusammenfassung und Fazit

Für die Bestimmung und Optimierung von netzbezogenen Maßnahmen, Redispatch- und Einspeisemanagement bieten sich durch die gezielte Auswahl geeigneter Bibliotheken große Potentiale zur Reduktion der Rechenzeit bei unveränderter Problemformulierung.

Die Ergebnisse der Untersuchungen zeigen große Unterschiede zwischen den getesteten Bibliotheken. Für das Lösen von LGS zeigt KLU die beste Performance, die weiteren Open Source Solver bieten auch noch akzeptable Rechenzeiten. Beim Lösen von LP/GGLP zeigen die kommerziellen Solver eine deutlich höhere Leistung als die Open Source Alternative. Die Rechenzeiten in den Untersuchungen haben Unterschiede bis zu einem Faktor von ca. 165 gezeigt. Bei sehr großen GGLP konnte die Open Source Lösung teilweise keine Lösung bestimmen.

Es ist davon auszugehen, dass die Modellgröße und die Anzahl steuerbarer Elemente und damit auch die Problemgrößen der Optimierungen zukünftig weiter ansteigen, wodurch der Reduzierung der Rechenzeiten eine zunehmend wichtige Bedeutung zukommt.

ISBN 978-3-85125-586-7 DOI 10.3217/978-3-85125-586-7

7 Literaturverzeichnis

[1] Bundesnetzagentur für Elektrizität, Gas, Telekommunikation, Post und Eisenbahnen, „Monitoringbericht 2017," Bonn, 2017.

[2] J. Eickmann, C. Bredtmann und A. Moser, „Security-Constrained Optimization Framework for Large-Scale Power Systems Including Post-Contingency Remedial Actions and Inter-Temporal Constraints," in *International Symposium on Energy System Optimization 2015*, Heidelberg, 2015.

[3] J. Eickmann, Simulation der Engpassbehebung im deutschen Übertragungsnetzbetrieb, Dissertation RWTH Aachen, Aachener Beiträge zur Energieversorgung, Band 164, Aachen: print production Verlag, 2015.

[4] „CPLEX," IBM®, [Online]. Available: https://www.ibm.com/analytics/data-science/prescriptive-analytics/cplex-optimizer. [Zugriff am 30 01 2018].

[5] „GUROBI," [Online]. Available: http://www.gurobi.com/products/whats-new/whats-new-in-the-latest-version. [Zugriff am 30 01 2018].

[6] „XPRESS," Fico, [Online]. Available: http://www.fico.com/de/products/fico-xpress-optimization. [Zugriff am 30 01 2018].

[7] „MINOS," Stanford Business Software Inc., [Online]. Available: http://www.sbsi-sol-optimize.com/asp/sol_product_minos.htm. [Zugriff am 30 01 2018].

[8] „COIN-OR," COIN-OR, [Online]. Available: https://www.coin-or.org/. [Zugriff am 30 01 2018].

[9] „GLPK," [Online]. Available: https://www.gnu.org/software/glpk/. [Zugriff am 30 01 2018].

[10] „SCIP," [Online]. Available: http://scip.zib.de. [Zugriff am 30 01 2018].

[11] „LP_SOLVE," [Online]. Available: http://lpsolve.sourceforge.net/. [Zugriff am 30 01 2018].

[12] B. Meindl und M. Templ, „Analysis of commercial and free and open source solvers for linear optimization problems," Wien, 2012.

[13] B. Meindl und M. Templ, „Analysis of Commercial and Free and Open Source Solvers for the Cell Suppression Problem," in *Transactions on Data Privacy 6*, 2013.

[14] J. L. Gearhart, K. L. Adair, R. J. Detry, J. D. Durfee, K. A. Jones und N. Martin, „Comparison of Open-Source Linear Programming Solvers," 2013.

[15] GUROBI Optimization, *Gurobi 7.5 Performance Benchmarks,* 2017.