HMD Praxis der Wirtschaftsinformatik (2023) 60:1062–1076
https://doi.org/10.1365/s40702-023-01004-9

SCHWERPUNKT

Einsatz von Cloud-Technologien und -ressourcen in der Co-Simulation

Thorsten Wack · **Andreas Schröder**

Eingegangen: 3. Mai 2023 / Angenommen: 8. August 2023 / Online publiziert: 13. September 2023

Zusammenfassung Schnell veränderliche (gesetzliche) Rahmenbedingungen, volatile Märkte und immer stärker ausgeprägte Cross-industrielle Netzwerke machen den Einsatz von computergestützten Simulations- und Optimierungslösungen notwendig. Oftmals stellen die beteiligten Komponentenlieferanten ihre Teilmodelle jedoch nur als Black-Box bereit, um ihr geistige Eigentum zu schützen. In diesem Fall ist im Rahmen der Systemintegration eine monolithische Berechnung des Gesamtsystems nicht mehr möglich. Hier kann die Methode der Co-Simulation entscheidend weiterhelfen, mögliche Betriebszustände und Optima effizient zu ermitteln.

Im vorliegenden Artikel wird beschrieben, wie diese Methode unter Einsatz von Multi-Cloud-Technologien und -Ressourcen in einem skalierbaren System implementiert wurde. Dabei werden sowohl Privacy-Aspekte der beteiligten Akteure als auch eine optimale Ausnutzung lokaler und in der Cloud verfügbarer Ressourcen unter Berücksichtigung von Performance- und Kostenaspekten betrachtet. Der Einsatz der ausgewählten Technologien erlaubt es, eine Vielzahl an Berechnungen parallel durchzuführen und so komplexe Parameterräume durch entsprechende Variationsrechnungen in kürzester Zeit abzutasten.

Schlüsselwörter Co-Simulation · Waveform Relaxation · Multi-Cloud Technologien · Function as a Service

✉ Thorsten Wack · Andreas Schröder
Fraunhofer-Institut für Umwelt-, Sicherheits- und Energietechnik UMSICHT, Osterfelder Straße 3, 46047 Oberhausen, Deutschland
E-Mail: thorsten.wack@umsicht.fraunhofer.de

Use of cloud technologies and resources in co-simulation

Abstract Rapidly changing (legal) framework conditions, volatile markets and increasingly distinct cross-industry networks make the use of computer-aided simulation and optimization solutions necessary. Often, however, the component suppliers involved only make their partial models available as a black box to protect their intellectual property. In this case, a monolithic calculation of the entire system is no longer possible in the context of system integration. Here, the method of co-simulation can decisively help to efficiently determine possible operating states and optima.

This article describes how this method was implemented in a highly scalable system using multi cloud technologies and resources. Privacy aspects of the involved actors as well as an optimal utilization of local resources and resources available in the cloud are considered, considering performance and cost aspects. The use of the selected technologies makes it possible to carry out many calculations in parallel and to scan complex parameter spaces in the shortest possible time using corresponding variation calculations.

The use of the selected technologies allows calculations to be carried out in parallel and complex parameter spaces to be scanned in the shortest possible time through parallelized variation calculations.

Keywords Co-simulation · Waveform relaxation · Multi cloud technologies · Function as a service

1 Einleitung

Sowohl die zunehmende Volatilität der Rohstoff- und Energiemärkte als auch die intensivere politisch geforderte Sektorenkopplung und die Ausprägung cross-industrieller Netzwerke erfordern bei Fragestellungen der Systemintegration und dem Design sowie der Optimierung von Prozessen (Bruns et al. 2021), insbesondere im verfahrenstechnischen Bereich, einen integralen Ansatz, der gleichzeitig die verteilte und multidisziplinäre Entwicklung der beteiligten Teilsysteme ermöglicht.

Unter verfahrenstechnischer Systemintegration wird dabei allgemein die Komposition eines komplexen Gesamtsystems aus mehreren technischen Teilsystemen verstanden (Abb. 1). Häufig geht dem realen Bau einer verfahrenstechnischen Anlage die mathematische Modellierung und Simulation des Gesamtsystems voraus, um einen bestimmungsgemäßen Betrieb innerhalb der Betriebsparameter im Vorfeld sicherzustellen.

Bei der mathematischen Modellierung des Gesamtsystems ist die Berücksichtigung externer, geschlossener Modelle (sogenannte Black-Box-Modelle) für das Gesamtsystem von entscheidender Bedeutung, da Lieferanten oftmals die physikalisch detaillierten Wirkmechanismen ihrer Komponenten nicht offenlegen (Schutz des geistigen Eigentums). Zur Analyse derartig gekoppelter Modelle existieren zwei unterschiedliche Ansätze.

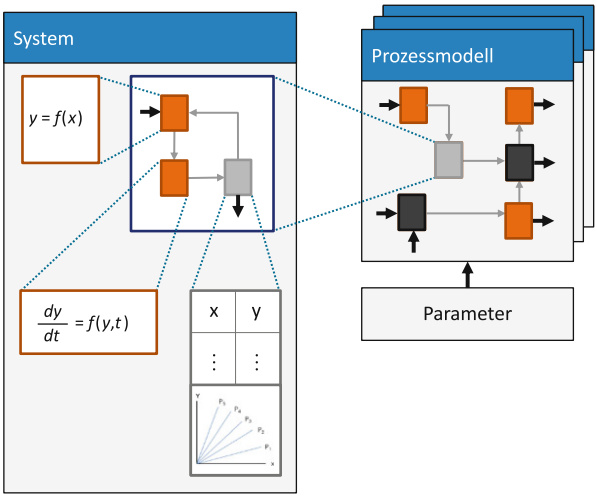

Abb. 1 Aggregation von Komponenten zu Systemen und anschließend zu komplexen Prozessmodellen

Einerseits können die Systemintegrierenden eine multidisziplinäre Modellierungssprache verwenden und mit Hilfe geeigneter Werkzeuge, wie z. B. Modelica (Modelica Association 2018) oder Advanced Continuous Simulation Language (ACSL) (Mitchell und Gauthier 1976), ein monolithisches Gesamtmodell erstellen, in dem die unterschiedlichen Domänen in einer einheitlichen abstrakten Form zusammengeführt werden. Die resultierenden mathematischen Strukturen – in der Regel differentiell-algebraische Gleichungssysteme – werden anschließend durch entsprechende Algorithmen simultan zur Lösung gebracht. Zwar erlauben entsprechende Werkzeuge eine ergonomische und effiziente Modellierung des Gesamtsystems, jedoch können geschlossene Modelle nur bedingt berücksichtigt werden, da ihre systeminternen physikalischen Mechanismen in der Regel unbekannt sind oder nicht zur Verfügung gestellt werden.

Andererseits können die Systemintegrierenden eine gekoppelte Simulation (Co-Simulation) durchführen, bei der die verschiedenen Teilsysteme separat gelöst werden und die Verkopplung über die Ein- und Ausgangsvektoren der Teilsysteme realisiert wird. Dazu werden in der Regel Relaxationsverfahren angewendet. Die Lösung für das Gesamtsystem ergibt sich dabei aus der iterativen Lösung der Teilmodelle, wobei die Kopplungsvektoren bei jedem Iterationsschritt bis zu einem festgelegten Konvergenzkriterium interpoliert werden. Dieser Ansatz ermöglicht einerseits die Einbettung geschlossener Modelle, andererseits können die Entwickelnden der Subsysteme die eigenen Modelle im Gesamtkontext analysieren und optimieren. Im Sinne eines „*Rapid Prototyping*“ besitzen sie so die Möglichkeit, den beteiligten Kollaborierenden verschiedene Varianten und Revisionsstände des eigenen Modells zur Verfügung zu stellen.

Der ursprünglich iterative und sequenzielle Prozess der Systemintegration kann auf diese Weise parallelisiert und somit insbesondere unter Nutzung von Cloud-Ressourcen erheblich effizienter und skalierbarer (Die Skalierbarkeit ist das Maß dafür wie gut das System auf Veränderungen reagiert, indem es Ressourcen hinzufügt oder entfernt, um den Anforderungen gerecht zu werden) gestaltet werden, da jedes Sub-

system in unterschiedlichen Ausprägungen simultan in verschiedenen Prozesskonzeptvarianten untersucht werden kann. Im vorliegenden Artikel wird beschrieben, wie eine solche Lösung unter Einsatz von Cloud-Technologien und -Ressourcen zu einem skalierbaren System ausgebaut wurde. Dabei werden sowohl Privacy-Aspekte der beteiligten Akteure als auch eine optimale Ausnutzung lokaler und in der Cloud verfügbarer Ressourcen unter Berücksichtigung von Performance- und Kostenaspekten betrachtet. Der Einsatz der ausgewählten Technologien erlaubt es, eine Vielzahl an Berechnungen parallelisiert durchzuführen und so komplexe Parameterräume durch entsprechende Variationsrechnungen gleichzeitig in kürzester Zeit abzutasten. So können mögliche Betriebszustände und Optima effizient ermittelt werden.

2 Lösung gekoppelter mathematischer Modelle

Bei der Systemintegration ist häufig die zeitabhängige Dynamik der Systeme – z. B. bei Start- oder Shut-Down-Prozessen – von besonderem Interesse (Deerberg et al. 2003). Daher ist die Anwendung instationärer Modelle erforderlich. Auf verteilte Parameter und damit auf die Berücksichtigung raumabhängiger Effekte wird aufgrund der resultierenden Rechenkomplexität und der damit verbundenen Laufzeiten der entsprechenden Simulationswerkzeuge in der Regel zunächst verzichtet.

Aus diesem Grund konzentriert sich die modellbasierte Systemintegration oftmals im ersten Schritt auf transiente Modelle mit konzentrierten Parametern, welche im Allgemeinen als differentiell-algebraisches Systeme (DAE System) formuliert sind (Deerberg et al. 2003). Diese Systeme können in allgemein impliziter Form geschrieben werden als:

$$F(t, x, \dot{x}) = 0 \text{ mit } x = x(t) \text{ und } \dot{x} = \frac{dx}{dt} \tag{1}$$

Dabei sind x und F Vektoren, t die Zeit x repräsentiert den systeminternen Zustand. Oft liegen diese Systeme als quasilinear implizites System (QI-System) in der folgenden Form vor:

$$B(t, x)\dot{x} = f(t, x) \tag{2}$$

Die Matrix B ist in vielen praktischen Fällen dünn besetzt und in der Regel singulär. Liegt ein gekoppeltes, multidisziplinäres, monolithisches Gesamtsystem vor, so kann dieses entsprechend in seine monodisziplinären Subsysteme zerlegt werden:

$$B_i(t, x_i)\dot{x}_i = f_i(t, x_i, u_i) \text{ mit } y_i = g_i(x_i, u_i) \text{ und } i = 1...N \tag{3}$$

Hierbei sind die Vektoren x_i die internen Zustandsvektoren, u_i die Eingangsvektoren sowie y_i die Ausgangsvektoren des jeweiligen Subsystems i. Die Funktionen f_i und g_i sind meist nichtlineare Vektorfunktionen.

Werden alle relevanten Subsysteme in einem monolithischen Modell formuliert und anschließend mit Hilfe eines entsprechenden Solvers ausgewertet, spricht man von einer starken Kopplung (Busch 2012). Dieses Vorgehen wird häufig bei der klassischen Systemintegration eingesetzt.

Aufgrund der Stabilität dieser hoch optimierten Solver ist diese Form der Kopplung auch bei der Integration unterschiedlich steifer Subsysteme numerisch sehr stabil, da z. B. die Schrittweiten der der Steifheit der jeweiligen Subsysteme angepasst werden. Ohne Nutzung von sogenannten Sparse-Algorithmen steigt die Komplexität quadratisch mit der Anzahl der Gleichungen.

2.1 Implizite Kopplung durch Waveformrelaxation

Im Gegensatz zur starken Kopplung werden bei der schwachen Kopplung die jeweiligen Subsysteme isoliert betrachtet und gelöst.

Dabei wird in der Regel der Betrachtungszeitraum in mehrere Teilabschnitte oder auch „*Segmente*“ zerlegt, so dass lediglich zu festgelegten Makrozeitpunkten t_i ein Abgleich der Kopplungsvektoren stattfindet. Bei den sogenannten impliziten Verfahren wird die Berechnung für jeden Makrozeitschritt nun so lange wiederholt, bis ein entsprechendes Konvergenzkriterium erreicht wird. Ein Beispiel für ein solches implizites Verfahren stellt die die Waveformrelaxation dar.

Hier wird zunächst ein sogenannter Gauss-Seidel Schritt (sequenzielle Lösung der Subsysteme) oder aber ein Jacobi Schritt (parallele Lösung der Subsysteme) durchgeführt (siehe auch Wack et al. 2018). Anstelle der Berechnung des nächsten Makrozeitschritts wird die Berechnung des aktuellen Zeitschritts wiederholt. Dabei entspricht der Eingangsvektor u_i für den Zeitschritt $k+1$ für das Subsystem M_i dem Ausgangsvektors y_j des verkoppelten Subsystems M_j aus dem vorherigen Iterationsschritt k:

$$u_{1,k+1}(t) = y_{2,k}(t) \tag{4}$$

Mit diesem Eingangsvektor u wird die Berechnung m-mal wiederholt. Typischerweise wird dabei nach jeder Berechnung die relative Abweichung zwischen zwei Iterationsschritten ermittelt und gegen einen Toleranzwert ε geprüft:

$$\varphi_{k+1}(u_{k+1}, u_k) \leq \varepsilon \tag{5}$$

Die Berechnung der relativen Abweichung φ für einen Iterationsschritt $k+1$ gemäß Gl. 5 wird mit Hilfe unterschiedlicher Normen der Kopplungsvektoren realisiert. Meist wird die Darstellung gemäß Gl. 6 gewählt:

$$\varphi_{k+1}(u_{k+1}, u_k) = \frac{\|u_{k+1} - u_k\|_2}{\|u_k\|_{\max}} \tag{6}$$

mit

$$||u_{k+1}-u_k||_2 = \sqrt{\sum_{i=1}^{m}(u_{i,k+1}-u_{i,k})^2} \text{ und } m = \dim(u) \tag{7}$$

sowie

$$||u_k||_{\max} = \max_{i=1,\cdots,m} |u_{i,k}| \text{ und } m = \dim(u) \tag{8}$$

Die 2-Norm (Bronstejn et al. 2020) des Differenzvektors der Kopplungsvektoren u_{k+1} und u_k aus den aufeinanderfolgenden Iterationsschritten k und $k+1$ aus Gl. 7 entspricht dabei der Wurzel aus der Summe der Differenzenquadrate und wird durch die max-Norm (Walz 2001) des Kopplungsvektors u_k dividiert. Die max-Norm entspricht dem Maximum der Einzelkomponenten u_i des Kopplungsvektors u aus dem Zeitschritt k und wird gemäß Gl. 8 berechnet.

Gleichzeitig erfolgt die Begrenzung der Iterationstiefe auf einen Maximalwert m_{max}:

$$m \leq m_{\max} \tag{9}$$

Wird das Konvergenzkriterium erfüllt oder die maximale Iterationstiefe m_{max} erreicht, so wird die Iteration für den betrachteten Zeitschritt i beendet und das Verfahren für den nächsten Zeitschritt $i+1$ erneut gestartet.

2.2 Adaptive Segmentierung

Neben der Anzahl der Stützstellen pro Segment und der Festlegung der ε-Umgebung hat die Größe der Makrozeitschritte der Waveformrelaxation einen erheblichen Einfluss auf die Genauigkeit der Ergebnisse der Co-Simulation. Daher erfolgt eine adaptive Anpassung der Segmentierung auf Basis Iterationstiefe m_i und der in Gl. 5 beschriebenen Funktion $\varphi(u)$ zur Bewertung der Kopplungsvektoren hinsichtlich der Konvergenz bei aufeinanderfolgenden Iterationsschritten. Die Auswertung dieser beiden Parameter aus einem Iterationslauf für ein Segment erlaubt die gezielte Anpassung der Schrittweiten der Segmente unabhängig von etwaigen Kenntnissen der Dynamik des Gesamtsystems oder der Dynamiken der konstituierenden Teilsysteme (Abb. 2).

Wird das Konvergenzkriterium für den Zeitschritt $i-1$ nicht in einer vorgegebenen Anzahl an maximal zur Verfügung stehenden Iterationsschritten m_{max} erreicht, so wird das Segment um einen Kontraktionsfaktor β verkleinert und anschließend

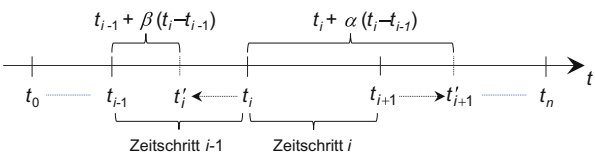

Abb. 2 Anpassung der Segmente auf Basis der Konvergenzeigenschaften

wiederholt berechnet. Falls das Konvergenzkriterium für das Segment $i-1$ innerhalb einer vorgegebenen minimalen Anzahl an Iterationsschritten, gebildet aus der Differenz der maximalen Anzahl an Iterationen m_{max} und einem Schwellenwert m_t, erreicht wird, so wird das nachfolgende Segment i um einen Expansionsfaktor α vergrößert. In allen übrigen Fällen bleibt das Intervall für das Folgesegment i unverändert.

Dieser Algorithmus führt dazu, dass bei stark verkleinerten Segmenten die Berechnung der einzelnen Teilsysteme sehr häufig durchgeführt wird. Besteht ein Gesamtsystem aus einer Vielzahl an Subsystemen führt dies zu einem erhöhten Kommunikationsaufwand und somit zu entsprechend hohen Anforderungen an die Bandbreite.

Weiterführende Informationen zur Co-Simulation sowie den eingesetzten Algorithmen sind bei Busch (2012) zu finden.

3 Beschreibung des Referenzsystems

Gemäß Abb. 3 wird im Referenzsystem als zentrales Element M_1 ein ideal durchmischter Rührkesselreaktor verwendet, der kontinuierlich betrieben wird. Die Kinetik im Reaktor wird von einer chemischen Reaktion erster Ordnung dominiert. Der Reaktor wird durch einen Mantel gekühlt. Das Kühlmedium wird dabei über die Pumpe M_3 im Kreis geführt und aus dem Vorlagebehälter M_4 entnommen, nach der Aufheizung im Reaktormantel in dem Wärmeübertrager M_2 heruntergekühlt und anschließend wieder in den Behälter gepumpt. Mit Hilfe des PID-Reglers M_6 soll der Volumenstrom im gesamten Kühlkreislauf so eingestellt werden, dass die Reaktortemperatur, die über das PT 100 Widerstandsthermometer (DIN EN 60751:2009-05) ermittelt wird, auf einem definierten Niveau gehalten wird.

Diese Art der Reaktionsführung wird häufig im großtechnischen Kontext beispielsweise in der Pharmaindustrie eingesetzt. Gerade die korrekte Temperaturfüh-

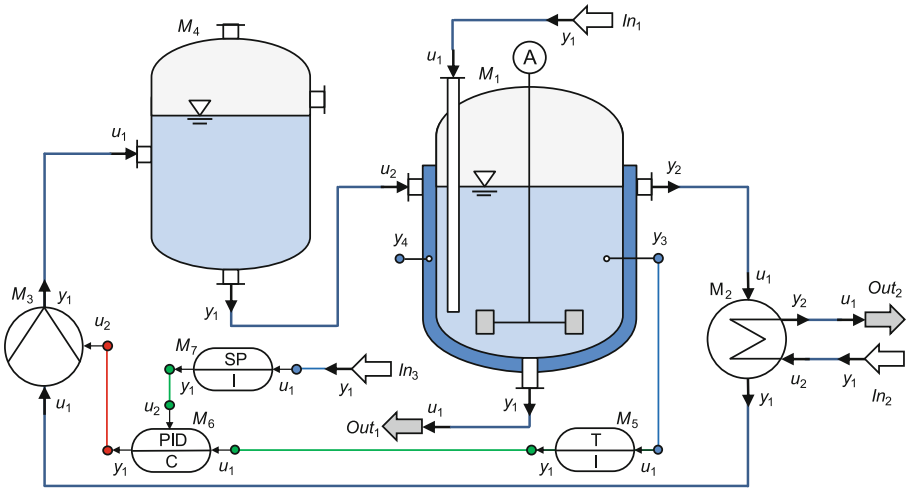

Abb. 3 Darstellung des Referenzsystems

rung ist hier in der Regel entscheidend für die Qualität der Produkte. Oftmals werden die Rührkessel dann auch in Kaskaden betrieben, um den Umsatz zu optimieren. Die transiente Simulation eines solchen Systems wird in der Verfahrenstechnik unter anderem angewendet, um das Zusammenspiel und die zeitabhängige Dynamik der beteiligten Teilsysteme zu untersuchen und zu optimieren.

Die gemessenen Temperaturen sowie die Sollwerte und Stellsignale werden gemäß DIN IEC 60381-1 (DIN IEC 60381-1:1985-11) über die entsprechenden Transmitter M_5 und M_7 auf einen Bereich von 4,0 bis 20,0 mA abgebildet. Die Sollwertvorgabe des Systemeingangs In_3 soll nach $t = 200\,\mathrm{s}$ einen diskreten Sprung von 390,0 K auf 420,0 K aufweisen, um so die Sprungantwort und das Verhalten des Systems bei Beteiligung einer diskreten Variablen zu untersuchen. Die Simulation des Referenzsystems, bestehend aus 12 gekoppelten Subsystemen mit insgesamt 54 Differentialgleichungen, einer Differenzengleichung und fünf algebraischen Gleichungen, soll im Zeitfenster [0 s, 300 s] durchgeführt werden.

Die Herleitungen sowie die theoretischen Grundlagen beteiligten Teilmodelle mit Benennung entsprechender weiterführender Literaturstellen sind bei Wack (2022) uu finden.

4 Szenarienrechnung

Häufig entstehen durch entsprechende Variationen in der Systemintegrationsaufgabe komplexe Szenarienräume. Werden beispielsweise D_{in} unterschiedliche Eduktvariationen (z. B. variierende Einsatzmengen und Feedströme) mit D_{out} verschiedenen Szenarien für die Produktmärkte (z. B. variierende Erlöse für das Produkt) kombiniert und dazu D_S verschiedene Varianten für einen Anlagenverbund untersucht, welche jeweils mit D_{CE} unterschiedlichen Szenarien für die Investitionskosten (CAPEX „Capital Expenditures“) und D_{OE} Szenarien für die Betriebskosten (OPEX „Operational Expenditures“) betrachtet werden, so entstehen in Summe D Varianten mit:

$$D = \prod_i D_i = D_{in} \cdot D_{\mathrm{out}} \cdot D_S \cdot D_{CE} \cdot D_{OE} \tag{10}$$

Je engmaschiger dieser Szenarienraum abgetastet wird, also je größer D_i ist, umso genauer kann der Korridor ermittelt werden, in dem ein Anlagenkonzept wirtschaftlich betrieben werden kann. Schon bei $D_i = 10$ entstehen in Summe 100.000 Variationen. Bei einer Rechenzeit von 10 min benötigte die komplette Berechnung aller Varianten bei sequenzieller Bearbeitung ca. 394 Tage.

5 Implementierung

Das in Abschn. 2 beschrieben Waveformrelaxations-Verfahren wurde als Python-Package implementiert und steht unter https://pypi.org/project/simunetcore (Wack 2023) öffentlich zur Verfügung. Es bildet den sogenannten simunet-core und bietet die Möglichkeit, beliebige eigene Modelle zu erstellen, diese zu Systemen zu ver-

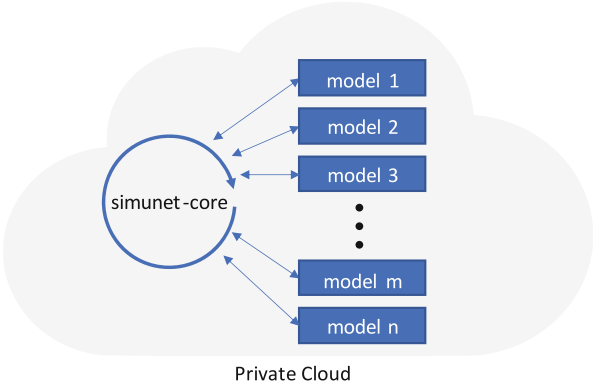

Abb. 4 Betrieb des Waveform-relaxations-Verfahrens in der Private Cloud

schalten und wie beschrieben mittels Gauss-Seidel- oder Jacobi-Verfahren zu lösen. Die beispielhafte Anwendung ist unter https://simunet.org/getting-started (Fraunhofer UMSICHT 2023) veranschaulicht.

Darauf aufbauend wurde im Rahmen des Großforschungsprojektes Carbon2Chem® (Deerberg et al. 2018) eine Simulationsplattform entwickelt, um eine ergonomische Benutzeroberfläche und ein skalierbares Compute-Backend zu schaffen. Die webbasierte Plattform besteht aus einer Modellregistrierung, einem Fließbildeditor, Komponenten zur Konfiguration und Durchführung einzelner Simulationen und zur Visualisierung der Berechnungsergebnisse. Für das „Abtasten" ganzer Parameterräume wurde zusätzlich ein Szenario-Manager implementiert, der es erlaubt, komfortabel Wertebereiche oder -listen für ausgewählte Parameter anzulegen und die Berechnung der sich daraus kombinatorisch ergebenden Varianten durchzuführen. Die Visualisierungskomponente wurde entsprechend erweitert, um den Vergleich einzelner Varianten ergonomisch und effizient zu ermöglichen. Das Frontend ist durchgängig webbasiert und kommuniziert mit entsprechenden Microservices im Backend.

Bei der Umsetzung wurde a priori der Cloud-native Ansatz verfolgt (Abb. 4). Alle Komponenten werden als Container in einer Private Cloud auf einem lokalen Kubernetes Cluster unter Verwendung gängiger DevOPs Methoden und Tools über eine CI/CD Pipeline bereitgestellt und betrieben. Der Einsatz einer Private Cloud war zunächst durch die projektspezifischen Anforderungen vorgegeben.

Alternativ könnten einzelne oder alle Komponenten auch in beliebigen Public Clouds betrieben werden, sofern Kubernetes zur Container-Orchestrierung verfügbar ist. Die definierten Systeme sowie die Simulationsergebnisse werden in einem S3-kompatiblen (Simple Storage Service) Objektspeicher abgelegt. Für die Kommunikation zwischen den Microservices werden sowohl dedizierte REST-Aufrufe (Representational State Transfer) zur Übertragung synchroner Kommandos als auch Events (Apache Kafka als Event-Streaming-Plattform) zum asynchronen Datenaustausch verwendet. Die konsequente Verwendung von gängigen Datenformaten, hier vor allem JSON (JavaScript Object Notation), Kommunikationsprotokollen und Backend-Komponenten ermöglicht jederzeit den Betrieb der Plattform in unterschiedlichen Clouds sowie in hybriden bzw. Multi-Cloud Konstellationen. Entscheidend

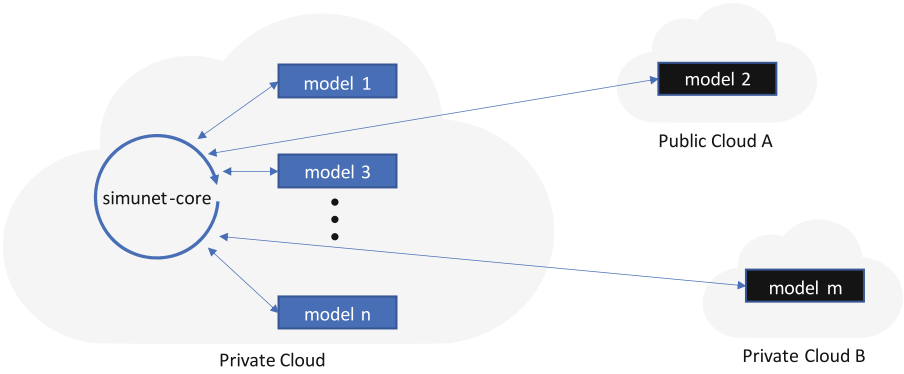

Abb. 5 Einbindung von remote betriebenen Black-Box-Modellen

für die konkrete Wahl des Ansatzes sind vor allem zugrundeliegende Compliance-Anforderungen.

Eine besondere Bedeutung kommt den Aufrufen der jeweiligen Teilmodelle (bzw. deren numerische Implementierung) durch die zentrale Waveformrelaxation zu. Wie eingangs beschrieben, werden einzelne Modelle häufig von Akteuren bereitgestellt, die keine detaillierte Einsicht in die numerische Umsetzung ihres Modells, und damit in die zugrunde liegenden physikalischen Details, gewähren. Daher erlaubt es der simunet-Core, einzelne Modelle

1. lokal
2. remote per MQTT (Message Queuing Telemetry Transport)
3. remote per REST-Protokoll
4. als serverless Function (Function as a Service) bei einem Cloud-Provider, aktuell implementiert Amazon Web Services (AWS)

aufzurufen.

Bei den Varianten 2.), 3.) und 4.) trägt derjenige, der das Modell bereitstellt, die alleinige Verantwortung für die Sicherheit, die korrekte Ausführung und die ggf. erforderliche Skalierung der numerischen Lösung (Abb. 4). Vorausgesetzt wird lediglich, dass die Ein- und Ausgabedaten der Aufrufe im JSON-Format übergeben werden. Zur Absicherung steht ein verbindliches JSON-Schema zur Verfügung, das jederzeit eine syntaktische Prüfung erlaubt. Der grundsätzliche Aufbau ist in Abb. 5 dargestellt.

Grundsätzlich ist beim Einsatz von Remote-Modellen aufgrund von Latenzen, Bandbreitenlimitierungen und Protokoll-Overhead mit Performance-Einbußen zu rechnen. Zudem hängt die reine Laufzeit einer Modellberechnung neben der spezifischen numerischen Implementierung von den remote verfügbaren Rechenressourcen ab. Diese Einbußen müssen jedoch in Kauf genommen werden, wenn Black-Box-Modelle ermöglicht werden sollen.

Insbesondere bei den oben beschriebenen Szenariorechnungen ist eine derartige Architektur notwendig, da hier durch die Kombination mehrerer Parameter mit jeweils mehreren Werten schnell eine sehr große Anzahl von Varianten ent-

steht, die eine entsprechend große Anzahl an Modellberechnungen erfordern. Beispielhaft werden für das gegebene Referenzsystem von den zahlreichen Modellparametern nur drei in jeweils 10 Schritten variiert, daraus ergeben sich insgesamt $D = \prod_i D_i = 10 \cdot 10 \cdot 10 = 1000$ Varianten.

Soll ein solcher Parameterraum berechnet werden, generiert der Szenario-Manager alle notwendigen Varianten (Eingabedaten) und sendet diese an einen oder mehrere simunet-Kernel. Diese werden typischerweise lokal in Form mehrerer Container (Waveform-Worker) bereitgestellt, von denen jeder wiederum durch Multiprocessing mehrere Simulationsrechnungen gleichzeitig durchführen kann. In der hier betrachteten Referenzarchitektur stehen 64 Waveform-Worker mit jeweils 16 Prozessen zur Verfügung, so dass maximal 1024 Berechnungen gleichzeitig durchgeführt werden können.

Für das Referenzsystem wurde die maximale Iterationstiefe auf $m_{\max} = 15$ festgelegt, so dass ein Makrozeitschritt nach maximal 15 Iterationen beendet wird. Für jede Iteration werden $N_{mod} = 12$ Modellberechnungen aufgerufen. Diese Aufrufe erfolgen im Jacobi Modus parallel, im Gauss-Seidel Modus sequenziell. Der Adaptionsmechanismus (Abb. 2) zerlegt das Zeitfenster der Gesamtsimulation $t = [0\,\text{s}, 300\,\text{s}]$ abhängig von der Parameterkonstellation in bis zu $N_t = 100$ Makrozeitschritte. Die Anzahl der Aufrufe für eine Variante beträgt also im Worst-Case:

$$N_{\text{call}} = m_{\max} \cdot N_{\text{mod}} \cdot N_t = 15 \cdot 12 \cdot 100 = 18.000 \tag{11}$$

Für eine Berechnung eines Szenarios mit 1000 Varianten bedeutet dies dann bis zu 18.000.000 Modellberechnungen. Es ist leicht zu erkennen, dass für eine effiziente Berechnung eines Szenarios dieser Größenordnung eine parallele Abarbeitung der Einzelberechnungen zwingend erforderlich ist.

6 Vergleich der Rechenzeiten

Das eingangs beschrieben Referenzsystem wurde in simunet modelliert und mit Jacobi-Basisschritten simuliert. Der Simulationszeitraum beträgt 360 s und es werden insgesamt 1000 Zeitschritte berechnet.

Die zwölf verwendeten Modelle werden als Function as a Service beim Cloud-Anbieter AWS betrieben. Dies entspricht der im Abschn. 5 beschrieben Variante 4. Die Varianten 1 bis 3 hängen bezüglich ihrer Performanz sehr stark von der Implementierung des jeweiligen Backends ab. Daher wird im Weiteren lediglich Variante 4 untersucht, da diese Variante ein reproduzierbares und nachvollziehbares Benchmark zulässt.

Die Berechnung des Referenzsystems benötigt circa 50 s. Es sind insgesamt 40 Segmentberechnungen und 416 Iterationen erforderlich, jedes verwendete Modell wurde also 416-mal aufgerufen. Daraus ergeben sich in Summe circa 5000 Funktionsaufrufe für eine einzelne Berechnung.

Sollten nun drei Parameter in jeweils 10 Schritten variiert werden, so ergeben sich nach (11) insgesamt 1000 Berechnungen, mit in Summe circa 5 Mio. Funkti-

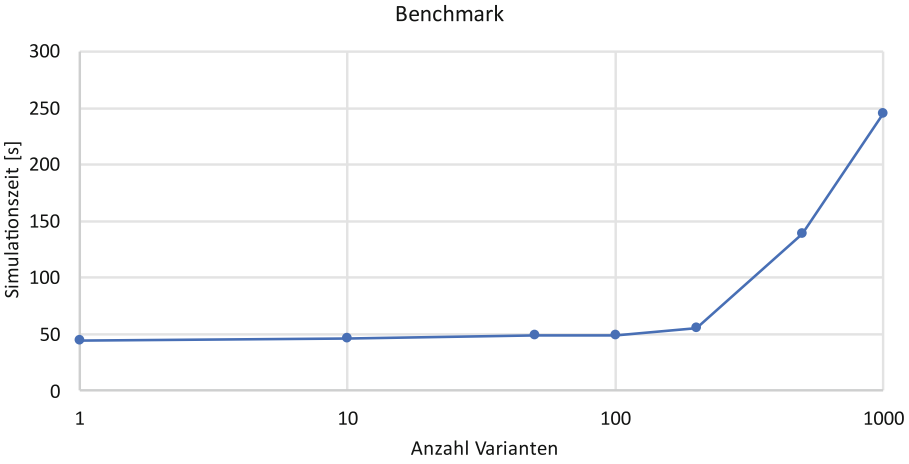

Abb. 6 Simulationszeit in Abhängigkeit der Anzahl an Varianten

onsaufrufen. Müssten diese Berechnungen sequenziell ausgeführt werden, so würde die Gesamtrechenzeit mindestens $1000 \cdot 50\,\text{s} \approx 14\,\text{h}$ betragen.

In nächsten Schritt wurde mit Hilfe des Szenarienmanagers sukzessive die Anzahl der Varianten erhöht und die Gesamtrechenzeit ermittelt. Abb. 6 zeigt die Ergebnisse dieser Untersuchung.

Im Bereich von 1 bis 100 Varianten bleibt die Gesamtrechenzeit in etwa konstant bei circa 50 s. Danach steigt sie auf bis zu 245 s bei 1000 Varianten an. Nähere Untersuchungen haben ergeben, dass dieser Effekt durch eine Bandbreitenlimitierung zwischen der Private Cloud und AWS zustande kommt. Selbst die Rechenzeit von ca. 4 min bei 1000 Varianten bedeutet jedoch gerade einmal einen Faktor 5 bezogen auf eine einzelne Berechnung.

Bei ausreichender Bandbreite zwischen allen beteiligten Instanzen ist somit davon auszugehen, dass sich die Gesamtrechenzeit für n Varianten näherungsweise von $O(n)$ bei sequenzieller Berechnung auf $\approx O(1)$ reduziert und ebenso keine Limitierung bei der Anzahl gleichzeitiger Rechnungen zu erwarten ist.

Die notwendige Rechenzeit für ein Modell hängt stark von den verfügbaren Ressourcen ab. Bei remote betriebenen Modellen in der AWS-Cloud beispielsweise, wird über die Zuweisung von Speicher ebenfalls die Anzahl der verwendeten virtuellen Prozessoren vorgegeben. Mittels verfügbarer Tools des Anbieters lassen sich darüber die Laufzeit und/oder die Kosten pro Aufruf optimieren (Amazon Web Services 2023) Die Ausführung der Funktionen auf ARM (Acorn RISC Machines) – anstatt x86-Prozessoren führt laut Anbieter zu einem Laufzeit- und Kostenvorteil von bis zu 34 % (Amazon Web Services 2021).

Die Positionierung der einzelnen Systemkomponenten in derselben Cloud würde die beschriebene Bandbreitenlimitierung weitestgehend aufheben und Latenzen minimieren. Hier sind lediglich Compliance-Vorgaben der beschränkende Faktor, da innerhalb der Simulationsplattform und in den Waveform-Workern alle Informationen über die Verschaltung der Modelle, Stoffdaten, Regelalgorithmen, Kosten usw.

hinterlegt sind. Dieses spezifische Wissen darf häufig nicht in öffentlichen Clouds gehandhabt werden.

7 Zusammenfassung

Die zunehmende Komplexität verfahrenstechnischer Gesamtsysteme im Zuge der vermehrten cross-industriellen Kopplung stellt sehr hohe Anforderungen an die Systemintegration und damit an die mathematische Modellierung und computergestützte Simulation solcher Systeme. Dabei sind die algorithmisch effizientesten Ansätze häufig aufgrund der vielen beteiligten Mitwirkenden und deren nachvollziehbarem Anspruch, ihre Daten sowie ihr Wissen über Prozessdetails und damit ihr geistiges Eigentum zu schützen, nicht einsetzbar. Weiterhin erschweren diese Algorithmen ein effizientes paralleles Arbeiten. Sie führen häufig zu einem zeitaufwändigen iterativ-sequenziellen Vorgehen. Das Prinzip der Co-Simulation bietet hier eine Möglichkeit, diesen Anforderungen gerecht zu werden und eine verteilte und simultane Entwicklung der Submodule unter Wahrung des jeweiligen Fachwissens zu ermöglichen.

Die vorgestellte Implementierung einer Co-Simulationsplattform, die auf etablierten Cloud-Technologien basiert, ermöglicht den Einsatz von Black-Box-Modellen, die von verschiedenen Akteuren bereitgestellt werden können. Jeder Anbieter kann seine spezifischen Modelle an verschiedenen Standorten, in der Regel in seiner bevorzugten Cloud (Private oder Public), betreiben. Mit Hilfe des Szenario-Managers können umfangreiche Parametervariationen orchestriert werden. Das vorgestellte System bietet die Möglichkeit, die Berechnungen parallel auszuführen und die Ergebnisse zentral zu aggregieren. Auch hier ist der Einsatz von Cloud-Technologie, insbesondere die elastische Skalierbarkeit von „Function as a Service“-Lösungen, hervorragend geeignet, um die Rechenzeit bei einem hohen Maß an Kostenkontrolle zu reduzieren. Beliebig viele Varianten eines Szenarios können parallel berechnet werden und die Gesamtrechenzeit des Szenarios liegt in derselben Größenordnung wie die Rechenzeit der langsamsten Variante. Limitiert wird dies gegebenenfalls durch Bandbreitenbeschränkungen und/oder zu knapp bemessenen Ressourcen auf der remote-Seite.

8 Anhang

Tab. 1 Lateinische Symbole

Symbol	Einheit	Bedeutung
B	[–]	QI-Darstellung eines DAE
D	[–]	Anzahl an Variationen
F	[–]	Implizite Darstellung eines DAE
f	[–]	Nichtlineare Vektorfunktion
g	[–]	Nichtlineare Vektorfunktion
m	[–]	Iterationstiefe
m	[–]	Komponentenanzahl der Eingangsvektoren
N	[–]	Anzahl
t	[s]	Zeit
u	[–]	Eingangsvektor
x	[–]	Systeminterner Zustandsvektor
y	[–]	Ausgangsvektor
x	[–]	Systeminternen Zustand

Tab. 2 Griechische Symbole

Symbol	Einheit	Bedeutung
α	[–]	Expansionsfaktor
β	[–]	Kontraktionsfaktor
ε	[–]	Konvergenzkriterium
φ	[–]	Konvergenzbewertungsfunktion
$\sum$	[–]	Summe
Π	[–]	Produkt

Tab. 3 Indizes

Index	Bedeutung
call	Aufruf
CE	CAPEX „Capital Expenditures“
i	Laufindex
in	Edukt
k	Laufindex
n	Anzahl an Makrozeitschritten
max	Maximum
mod	Modelle
OE	OPEX „Operational Expenditures“
out	Produkt
S	Anlagenverbund
t	Grenzwert (threshold)

Danksagung Wir danken allen Partnern des Projekts Carbon2Chem® für die angenehme und erfolgreiche interdisziplinäre Zusammenarbeit. Außerdem danken wir dem Bundesministerium für Bildung und Forschung (BMBF) für die Förderung des Projekts „Carbon2Chem®“ (Projektnummer 03EK3037D).

Funding Open Access funding enabled and organized by Projekt DEAL.

Literatur

Amazon Web Services (2021) AWS Lambda Functions Powered by AWS Graviton2 Processor—Run Your Functions on Arm and Get Up to 34 % Better Price Performance | Amazon Web Services. https://aws.amazon.com/de/blogs/aws/aws-lambda-functions-powered-by-aws-graviton2-processor-run-your-functions-on-arm-and-get-up-to-34-better-price-performance/. Zugegriffen: 03. Mai 2023

Amazon Web Services (2023) Profiling functions with AWS Lambda Power Tuning—AWS Lambda. https://docs.aws.amazon.com/lambda/latest/operatorguide/profile-functions.html. Zugegriffen: 03. Mai 2023

Bronstejn IN, Mühlig H, Musiol G (2020) Taschenbuch der Mathematik, 11. Aufl. Edition Harri Deutsch. Verlag Europa-Lehrmittel Nourney Vollmer GmbH & Co. KG, Haan-Gruiten

Bruns B, Herrmann F, Grünewald M, Riese J (2021) Dynamic Design Optimization for Flexible Process Equipment. Ind Eng Chem Res 60(20):7678–7688. https://doi.org/10.1021/acs.iecr.1c00306

Busch M (2012) Zur effizienten Kopplung von Simulationsprogrammen. Dissertation, Universität Halle

Deerberg G, Grän-Heedfeld J, Wack T, Schwichtenberg H, Winter G (2003) Computergestützte verfahrenstechnische Modellierung mit SimCARE. Chem-Ing-Tech 75(8):1144. https://doi.org/10.1002/cite.200390389

Deerberg G, Oles M, Schlögl R (2018) The Project Carbon2Chem. Chem-Ing-Tech 90(10):1365–1368. https://doi.org/10.1002/cite.201800060

DIN EN 60751:2009-05, Industrielle Platin-Widerstandsthermometer und Platin-Temperatursensoren (IEC_60751:2008); Deutsche Fassung EN_60751:2008. Beuth Verlag GmbH, Berlin. https://doi.org/10.31030/1507857

DIN IEC 60381-1:1985-11, Analoge Signale für Regel- und Steueranlagen; Analoge Gleichstromsignale; Identisch mit IEC 60381-1, Ausgabe_1982. Beuth Verlag GmbH, Berlin. https://doi.org/10.31030/1057334

Fraunhofer UMSICHT (2023) What is simunet®?—Getting started. https://simunet.org/getting-started/_content/0.html. Zugegriffen: 03. Mai 2023

Mitchell EEL, Gauthier JS (1976) Advanced Continuous Simulation Language (ACSL). SIMULATION 26(3):72–78. https://doi.org/10.1177/003754977602600302

Modelica Association Modelica®—A Unified Object-Oriented Language for Systems Modeling. https://www.modelica.org/documents/ModelicaSpec34.pdf. Zugegriffen: 24. April 2018

Wack T (2022) Zur Co-Simulation heterogener mathematischer Modelle in der Verfahrenstechnik. Ruhr-Universität Bochum

Wack T (2023) simunetcore. https://pypi.org/project/simunetcore/. Zugegriffen: 03. Mai 2023

Wack T, Schlüter S, Hennig T, Wagner H, Diekmann A, Zheng Q, Grundler J (2018) Application of Waveform Relaxation in Distributed Process Co-Simulation. Chem-Ing-Tech 90(10):1559–1567. https://doi.org/10.1002/cite.201800047

Walz G (Hrsg) (2001) Lexikon der Mathematik. In sechs Bänden. Spektrum Akad. Verl., Heidelberg