TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Maschinenwesen

Institut für Verarbeitungsmaschinen und Mobile Arbeitsmaschinen
Professur für Baumaschinen- und Fördertechnik
Prof. Dr.-Ing. habil. Günter Kunze

Untersuchung und Bewertung von Tearing-Algorithmen

Von Waurich, Volker

Geboren am 03.01.1989 in Schönebeck

zur
Erlangung des akademischen Grades
Diplomingenieur
(Dipl. -Ing.)

vorgelegte Diplomarbeit.

Tag der Einreichung: ….....

1. Gutachter: ….....

2. Gutachter: .....

angenommen / nicht angenommen

Dresden, den..... .....

Institutsdirektor

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Maschinenwesen, Institut für Verarbeitungsmaschinen und Mobile Arbeitsmaschinen

Aufgabenstellung Diplomarbeit

Name:	**Waurich, Volker**
Matrikel-Nr.:	3428464
Studiengang:	Maschinenbau
Studienrichtung:	Allgemeiner und Konstruktiver Maschinenbau

Thema: **Untersuchung und Bewertung von Tearing Algorithmen**
Analysis and Evaluation of Tearing Algorithms

Modelle von mobilen Arbeitsmaschinen lassen sich sehr effizient mit Hilfe der Sprache Modelica voll physikalisch modellieren. Simulationsprogramme, die Modelica unterstützen, leiten daraus ein großes schwach besetztes differentiell algebraisches System von Differenzialgleichungen (DAE) ab. Um dies effektiv auswerten zu können, werden darauf verschiedene symbolische Optimierungen angewandt.
Mit Hilfe des Tearing Algorithmus ist es möglich, im Falle großer schwach besetzter nichtlinearer Gleichungssysteme die Anzahl der Unbekannten sowie die der Residuen drastisch zu verringern und damit die Effizienz sowie die Robustheit entscheidend zu erhöhen.
Ziel der Arbeit soll es sein, bekannte sowie eigene Tearing Algorithmen in den OpenModelica Compiler zu implementieren. Diese sollen im Anschluss anhand selbst gewählter Testprobleme verglichen und hinsichtlich ihrer Eignung für die Simulation von mobilen Arbeitsmaschinen bewertet werden.
Die Aufgaben umfassen im Einzelnen:

- Literaturrecherche zu Tearing Algorithmen
- Auswählen von Beispielen zum Vergleich der Algorithmen
- Auswahl und Implementierung bekannter Algorithmen in OpenModelica
- Implementierung eigener Algorithmen
- Bewertung und Vergleich der Algorithmen
- Dokumentation der Ergebnisse.

Betreuer: Dipl.-Ing. C. Schubert

Beginn der Arbeit: 20.12.2012
Abgabe der Arbeit: 20.04.2013 Bibl.-Nummer: DA-BFT-87

Die vom Lehrstuhl erlassenen Richtlinien zur Anfertigung der Arbeit sind zu beachten!

Prof. Dr.-Ing. habil. G. Kunze
Studienrichtungsleiter

Prof. Dr.-Ing. habil. G. Kunze
Betreuender Hochschullehrer

Kurzreferat

Die vorliegende Arbeit beschäftigt sich mit Tearing, einem symbolischen Verfahren zur effizienteren Berechnung großer Gleichungssysteme, wie sie in physikalischen Modellen auftreten. Das Konzept des Tearing wird in seinen Kontext eingeordnet und die Funktionsweise dargestellt. Es werden bereits bekannte Methoden aus der Literatur grundlegend erläutert, verglichen, klassifiziert und deren Vor- und Nachteile erarbeitet. Dazu werden die Algorithmen in den OpenModelica Compiler implementiert und bezüglich Effizienz und benötigter Rechenzeit getestet. Anhand der gewonnen Erkenntnisse werden Kriterien für einen optimalen Algorithmus beschrieben. Daraus wird eine eigene Tearing-Methode abgeleitet, die die Vorteile der untersuchten Algorithmen vereint und zusätzlich eine eigene Heuristik zur Auswahl der Tearing-Variablen nutzt. Die neu entwickelte Heuristik liefert eine zusätzliche Verbesserung verglichen zu den bekannten Algorithmen.

Eidesstattliche Erklärung

Hiermit bestätige ich mit meiner Unterschrift, dass ich die von mir am heutigen Tag eingereichte Diplomarbeit zum Thema:

Untersuchung und Bewertung von Tearing Algorithmen

vollkommen selbständig und nur unter Verwendung der in der Arbeit angegebenen Literatur und Abstimmungspartner angefertigt habe.

……..
Unterschrift mit Vor- und Zunamen

Dresden, den ……..

Inhaltsverzeichnis

Abbildungsverzeichnis

Formelzeichen

Formelzeichen (latein)	Bezeichnung
$\boldsymbol{A}$	Systemmatrix für $\boldsymbol{Ax} = \boldsymbol{b}$
$\boldsymbol{b}$	Parametervektor für $\boldsymbol{Ax} = \boldsymbol{b}$
$d(v)$	Grad eines Knoten v
d_i	Eingangsgrad
d_o	Ausgangsgrad
e	Kante eines Graphen
E	Menge aller Kanten
f	Funktion
$f'(x)$	Ableitung der Funktion f nach x
$\partial f / \partial x$	partielle Ableitung einer Funktion f nach x
G	Graph
$\boldsymbol{J}$	Jacobi-Matrix
m	Anzahl der inzidenten Kanten eines Knotens
n	Dimension einer quadratischen Matrix
p	Wichtung eines Knotens / einer Kante
v	Knoten eines Graphen
V	Menge aller Knoten

Indizes

b	fett-gedruckt
eq	Gleichung
ges	gesamt
i	Laufvariable
nb	nicht fett-gedruckt
var	Variable

Abkürzungen

BLTF	*block-lower-triangular-form* (untere Blockdreiecksform)
DAE	*differential-algebraic-equation*
resEq	*Residuengleichung*
tVar	Tearing-Variable

1 Einführung

1.1 Motivation

Die computergestützte Berechnung von physikalischen Systemen bietet zahlreiche Vorteile bei Forschung, Entwicklung und Überwachung. Virtuelle Simulation ermöglicht es, Anlagen, Konstruktionen und Prozesse zu untersuchen ohne das reale System vorliegen zu haben. Auch unwahrscheinliche oder gefährliche Anwendungsfälle können ohne teure und aufwendige reale Testumgebung nachgestellt werden. Die Qualität der Simulation hängt lediglich von der Güte des erstellten Modells ab. Dahingehend sind das Wissen und die Erfahrung des Ingenieurs gefragt. Je detaillierter ein System modelliert wird, desto exaktere Ergebnisse sind zu erwarten. Mit dem Grad der Detailtreue steigen jedoch der Berechnungsaufwand und somit auch die nötige Rechenzeit. In bestimmten Anwendungsfällen wie beispielsweise Echtzeitsimulation oder Anlagenüberwachung werden kurze Rechenzeiten benötigt. Das kann entweder erreicht werden, indem das Modell verkleinert oder umstrukturiert wird, was qualitative Einbußen nach sich ziehen könnte, oder durch eine effizientere Berechnung. Bei verringertem Rechenaufwand sinkt auch die Rechenzeit, die Berechnung wird robuster und es wird weniger Speicher benötigt. In komplexen physikalischen Modellen werden große Gleichungssysteme aus differentiellen und algebraischen Gleichungen aufgestellt. Beim automatischen Aufstellen der Gleichungen, wie beispielsweise in Modelica, sind die Systemmatrizen dieser sogenannten DAE-Systeme (engl.: *differential algebraic equation systems*) oftmals dünn besetzt. Die Verarbeitung dieser Matrizen benötigt viele Ressourcen, die es einzusparen gilt. Es wird angestrebt das Ausmaß der Berechnung weitestgehend zu verringern. Eine Möglichkeit bietet das Aufbrechen von algebraischen Schleifen innerhalb des Gleichungssystems. Diese müssten andernfalls gleichzeitig iteriert werden, was noch großes Verbesserungspotenzial birgt. Die Dimension des zu lösenden Systems kann sich dadurch verringern. Das Aufbrechen der Schleifen wird Tearing genannt und soll im Folgenden erläutert werden.

1.2 Präzisierung der Aufgabenstellung

Das Ziel der vorliegenden Arbeit besteht darin, das Verfahren des Tearing grundlegend zu analysieren. Es werden theoretische Grundlagen erörtert, die für das weitere Verständnis von Bedeutung sind und es wird ein Einblick in die Möglichkeiten des Lösens von Gleichungssystemen gegeben. Darauf aufbauend soll die prinzipielle Verfahrensweise des Tearing umfassend dargestellt werden und es wird versucht, eine systematische Einordnung möglicher Tearingkonzepte zu erarbeiten. Dazu sollen bereits existierende, ausgewählte Verfahren praktisch umgesetzt, miteinander verglichen und bestehende Gemeinsamkeiten, Unterschiede und Eigenarten erläutert werden. Die Auswahl der zu untersuchenden Tearingmethoden soll aus Algorithmen mit unterschiedlicher Arbeitsweise bestehen, um ein breites Spektrum der möglichen Tearingkonzepte zu beleuchten. Die Implementierung der gewählten Algorithmen erfolgt direkt im OpenModelica Compiler. Auf der Basis der erlangten Erkenntnisse werden Anforderungen, Bewertungskriterien und potenzielle Probleme formuliert. Neben dieser qualitativen Einschätzung wird eine quantifizierte Bewertung bezüglich Effizienz, Schnelligkeit etc. vorgenommen. Dazu werden die implementierten Algorithmen anhand ausgewählter Beispiele getestet und die jeweilige Größe ermittelt. Mit Hilfe dieser Erkenntnisse soll eine Empfehlung bezüglich des optimalen Tearingverfahrens gegeben werden und sofern nötig ein bestehendes Verfahren optimiert beziehungsweise ein eigener Algorithmus entwickelt werden.

2 Theoretische Grundlagen

Das folgende Kapitel enthält theoretische Grundlagen, die für das Verständnis der Arbeit bekannt sein müssen. Da Tearingverfahren mit Graphen arbeiten, wird ein kurzer Überblick über wesentliche graphentheoretische Elemente und Begriffe gegeben /Tit11/, /Bon08/. Es werden explizit nur jene Fakten erläutert, die für die vorliegende Arbeit von Bedeutung sind. Ebenso hat der Abschnitt bezüglich Lösungsverfahren von Gleichungssystemen keinen Anspruch auf Vollständigkeit, sondern soll lediglich dem Verständnis der nachfolgenden Kapitel dienen /Deu04/, /Kan05/, /Bär07/.

2.1 Das Gleichungssystem

In einem Element eines Mehrkörpersimulationsmodells (beispielsweise eine Masse, ein Kraftelement, ein Hydraulikzylinder etc.) sind Gleichungen definiert, die physikalische Größen miteinander in Beziehung setzen. Es existieren Abhängigkeiten zwischen den Zustandsgrößen der Elemente untereinander und Prozessgrößen, die die Elemente verknüpfen. Das Gleichungssystem, welches das System beschreibt, enthält alle Gleichungen $\boldsymbol{F}(\boldsymbol{x})$, gemäß [2.1] und sämtliche Variablen $\boldsymbol{x}$ der Modellelemente.

$$\mathbf{0} = \boldsymbol{F}(\boldsymbol{x}) \tag{2.1}$$

mit $\boldsymbol{x} = (x_1, x_2, \dots x_n)^T$

Um die Lösung dieses Gleichungssystems auf symbolischer Ebene zu optimieren, muss geklärt werden, welche Variablen in welcher Gleichung verknüpft sind. Um diese Information darzustellen wird die Jacobi-Matrix $\boldsymbol{J}$ genutzt, siehe [2.2].

$$\boldsymbol{J} = \frac{\partial \boldsymbol{F}(\boldsymbol{x})}{\partial \boldsymbol{x}} = \left[\frac{\partial F_i}{\partial x_j}\right]_{i=1\ldots n;\ j=1\ldots n} \quad [2.2]$$

Die Jacobi-Matrix beinhaltet sämtliche erste partielle Ableitungen der Gleichungen. Die Spalten stehen dabei für die Variablen und die Zeilen für die Gleichungen. In den Matrixeinträgen steht die jeweilige partielle Ableitung der Gleichung nach der Variablen. Ist die partielle Ableitung einer Gleichung nach einer Variablen Null, so liegt keine Abhängigkeit vor. Ist der Matrixeintrag der Jacobi-Matrix ein Nicht-Null-Element, enthält die Gleichung die Variable. Mit dieser Information kann eine, das Gleichungssystem beschreibende Matrix aufgestellt werden. Zur nachfolgenden symbolischen Aufbereitung mittels graphentheoretischen Algorithmen wird dann ein repräsentativer Graph erzeugt.

2.2 Grundlagen der Graphentheorie

Die Graphentheorie ist ein Teilgebiet der Mathematik, welches besonders in der Informatik Anwendung findet. Hierbei werden die Beziehungen und Eigenschaften von Graphen untersucht. Gleichungssysteme, wie sie in dieser Arbeit behandelt werden, lassen sich besonders gut mit Hilfe von Graphen darstellen und mit diversen Algorithmen umformen. Im Folgenden werden einige fundamentale Begriffe der Graphentheorie erläutert, die für das Verständnis der Arbeit notwendig sind.

2.2.1 Der Graph

Ein Graph $G = (V, E)$ ist ein geordnetes Paar, bestehend aus einer Menge von Knoten V (engl.: *vertex* bzw. *node*), einer Menge von Kanten E (engl.: *edge* bzw. *branch*), wobei jede Kante aus E zwei Knoten aus V verbindet /Tit11/. Mit anderen Worten ist ein Graph eine sinnbildliche Struktur, die Objekte V und ihre Verbindungen E zueinander darstellt. Steht ein Knoten in Beziehung mit einem anderen Knoten, so ist er über eine Kante mit ihm verbunden. Benachbarte Knoten werden auch als adjazente Knoten und die verbindenden Kanten auch zum Knoten inzidente Kanten genannt. In dieser Arbeit stellen die Knoten Gleichungen oder Variablen beziehungsweise eine paarweise Zuordnung beider dar. Die Kanten zeigen an, dass Gleichungen Variablen beinhalten beziehungsweise, dass Variablen in Gleichungen vorhanden sind. Für eine quadratische Matrix $\boldsymbol{B} \in \mathbb{R}^{\boldsymbol{n}\times\boldsymbol{n}}$ ergibt sich der zugehörige Graph $G(\boldsymbol{B})$ dementsprechend wie folgt. Eine Kante zwischen dem Knoten $v_i$und dem Knoten v_j existiert dann, wenn das Matrixelement $u_{ij} \neq 0$.

Eine weitere wichtige Größe ist der Grad $d(v)$ eines Knotens v. Er gibt an, wie viele inzidente Kanten ein Knoten besitzt /Tit11/, /Bon08/.

In Bild 1a ist ein einfacher Graph mit sechs Knoten dargestellt. Einfache Graphen sind ungerichtet und haben weder Mehrfachkanten noch Schleifen.

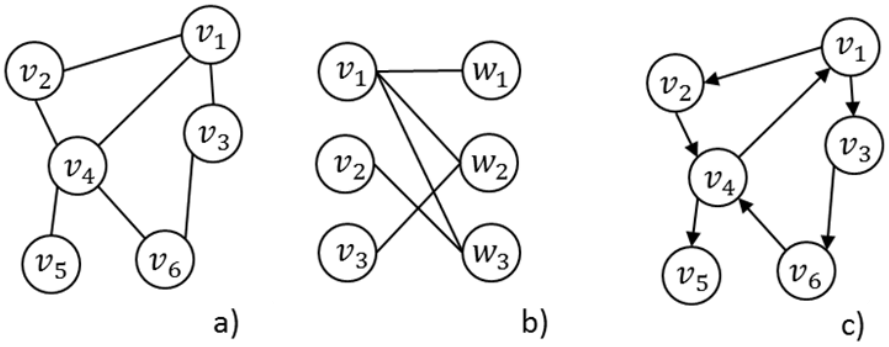


Bild 1 Graph (a), bipartiter Graph (b) und gerichteter Graph (c)

2.2.2 Bipartite Graphen

Ein bipartiter Graph $G([V,W],E)$ (engl.: *bipartite graph*) ist eine spezielle Form des Graphen, dessen Knotenmenge in zwei disjunkte Teilmengen aufgeteilt ist deren Elemente miteinander mittels einer Kante aus der Menge E verbunden sind. Jeder Knoten aus einer dieser Teilmengen darf nur zu einem Knoten der anderen Teilmenge adjazent sein. Dementsprechend hat jede Kante ein Ende mit einem Knoten aus je einer der zwei Mengen /Bon08/. Diese Graphen eignen sich insbesondere zur Veranschaulichung von Zuordnungsproblemen, in unserem Fall die Zuordnung zwischen Variablen und Gleichungen. Um den bipartiten Graphen aus einer Matrix $\boldsymbol{B} \in \mathbb{R}^{\boldsymbol{n\times n}}$ abzuleiten, wird eine neue Matrix $\boldsymbol{C} \in \mathbb{R}^{\boldsymbol{2n\times 2n}}$ gebildet mit $\boldsymbol{C} = \begin{bmatrix} 0 & \boldsymbol{B} \\ \boldsymbol{B}^T & 0 \end{bmatrix}$. Um das Gleichungssystem aus einem Modell als bipartiten Graphen abzuleiten, muss die Matrix $\boldsymbol{B}$ als Jacobi-Matrix $\boldsymbol{J}$ interpretiert werden. So erhält man die Matrix des bipartiten Graphen $\boldsymbol{C} = \begin{bmatrix} 0 & \boldsymbol{J} \\ \boldsymbol{J}^T & 0 \end{bmatrix}$,wobei die ersten n Zeilen und n Spalten für die Gleichungsknoten v und die letzten n Zeilen und Spalten für die Variablenknoten w stehen. Anhand der Matrixstruktur ist zu erkennen, dass Gleichungsknoten nur adjazent zu einem Variablenkoten sein können und umgekehrt. Eine Kante zwischen Gleichungsknoten v_i und dem Variablenknoten w_j existiert wiederum, falls das Matrixelement $u_{ij} \neq 0$. In Bild 1b ist ein bipartiter Graph mit den disjunkten Teilmengen V und W beispielhaft dargestellt.

2.2.3 Digraphen

Graphen können ungerichtet oder gerichtet sein. Einen gerichteten Graphen $D(V, E)$ bezeichnet man als Digraphen (engl.: *digraph*). Bei diesen speziellen Graphen erhalten die Kanten e_i eine zusätzliche Richtungsinformation, welche durch Pfeile dargestellt wird. Sie zeigen stets von einem Knoten zu einem Anderen und bilden somit, im Gegensatz zum gewöhnlichen Graphen, ein geordnetes Paar von Knoten. Es wird festgelegt, dass eine Kante $e = (v_1, v_2)$ von Knoten v_1 zu Knoten v_2 führt, wobei v_1 der Vorgänger von v_2 ist und v_2 der Nachfolger von v_1 /Bon08/. Digraphen sind nützlich, um beispielsweise den Informationsfluss innerhalb eines Modells oder eine Lösungsreihenfolge innerhalb eines Gleichungssystems wiederzugeben. Da es keine Unterteilung zwischen Gleichungs-und Variablenknoten gibt, stehen die Knoten in einem Digraphen für eine paarweise Zuordnung von Gleichung und zu lösender Variablen. Für Knoten in einem Digraphen kann sowohl der Ausgangsgrad d_o, als auch der Eingangsgrad d_i je nach Anzahl ein- oder ausgehenden Kanten angegeben werden. In Bild 1c ist ein gerichteter Graph dargestellt.

2.2.4 Zusammenhang

Zwei Knoten eines Graphen heißen zusammenhängend, wenn beide Knoten durch eine Kantenfolge miteinander verbunden sind. Ein Graph ist somit zusammenhängend, wenn es zwei beliebige Knoten gibt, die durch eine Folge von Kanten miteinander verbunden sind. Jede Teilmenge zusammenhängender Knoten eines Graphen, die nicht zusammenhängend mit einer anderen Teilmenge ist, nennt man Zusammenhangskomponente des Graphen. Innerhalb eines gerichteten Graphen sind zwei Knoten stark zusammenhängend, sofern sie über eine Kantenfolge über einen gerichteten Weg in beide Richtungen miteinander verbunden sind /Tit11/, /Bon08/. Der Graph selbst ist als stark zusammenhängend zu bezeichnen, wenn dies für jeden Knoten zutrifft. In Bild 2a ist ein Graph dargestellt, der einen stark-zusammenhängenden Teilgraphen beinhaltet. Der Knoten v_5 ist nicht Bestandteil der stark-zusammenhängenden Komponente. Ein schwach-zusammenhängender Digraph wird als solcher bezeichnet, wenn sein ungerichtetes Äquivalent zusammenhängend ist. Mit Hilfe der Kenntnis über die einzelnen Zusammenhangskomponenten innerhalb eines Graphen kann dieser in

Teilkomponenten zerlegt werden. Bei den Anwendungsfällen, wie sie in dieser Arbeit betrachtet werden, kann somit die Bearbeitung der Teilsysteme, also der stark zusammenhängenden Komponenten (engl.: *strongly connected components*), separat erfolgen anstatt das System im Ganzen zu lösen.

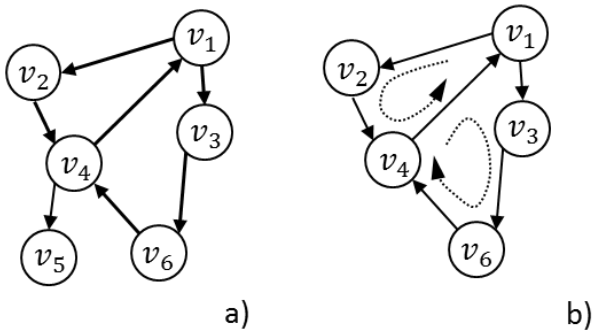


Bild 2 Digraphen mit stark zusammenhängendem Teilgraphen (a) und algebraischen Schleifen in der stark zusammenhängenden Komponente (b)

2.2.5 Kreise und algebraische Schleifen

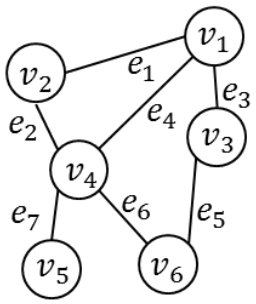
In einem zusammenhängenden Graphen bzw. bipartiten Graphen existiert ein Kreis, wenn es eine Kantenfolge gibt, deren Start- und Zielknoten derselbe ist /Titt11/. Jede Kante darf in dieser Kantenfolge nur einmal vorkommen. In einem stark zusammenhängenden Graphen existiert zwischen allen Knoten ein Weg. Also sind alle möglichen Paare von Knoten beiderseits verbunden. Stark zusammenhängende Graphen enthalten demnach immer Kreise. Mit Hilfe dieser Information kann ein Kreis gefunden werden. Im vorliegenden Anwendungsfall der Repräsentation eines Gleichungssystems sprechen wir nicht von einem Kreis sondern von algebraischen Schleifen. In Bild 2b sind die zwei Schleifen innerhalb der stark-zusammenhängenden Komponente dargestellt. Für den Anwendungsfall des Tearing gilt es, diese algebraischen Schleifen in einem Gleichungssystem aufzubrechen. Es müssen demnach nur die Schleifen in den Zusammenhangskomponenten dem Tearing unterzogen werden. Diese Partitionierung in stark zusammenhängende Komponenten erfolgt während der Permutation der Strukturinzidenzmatrix (siehe Abschnitt 2.3.1) zu einer *block-lower-triangular-form* (engl. untere Blockdreiecksform), der sogenannten BLT-Transformation.

2.3 Algebraische Repräsentation eines Graphen

Grundlegende graphentheoretische Probleme werden meist zuallererst mittels einer graphischen Repräsentation beschrieben. Das ist sehr anschaulich und hilft, das Problem zu erfassen und Algorithmen nachvollziehbarer darzustellen. Bei großen und komplexen Graphen wird solch eine graphische Beschreibung jedoch schnell unübersichtlich und schwer zu handhaben. Es ist hilfreich, eine weitere mathematische Repräsentation zu nutzen, um Graphen effizient zu beschreiben und zu modellieren. Hierzu bietet sich eine algebraische Beschreibung mittels Matrizen an. Diese wird auch für die computerunterstützte Aufbereitung der Graphen benutzt. Es existieren im Allgemeinen zwei grundlegende Möglichkeiten, einen Graph vollständig darzustellen. Die Inzidenzmatrix und die Adjazenzmatrix. Welche Form benutzt werden sollte, hängt im Einzelfall von den nachfolgenden Umformungen ab.

2.3.1 Die Inzidenzmatrix

Die Inzidenzmatrix (lat.: *incidere* „fallen“) gibt an, welche Knoten und welche Kanten miteinander verbunden (inzident) sind. Die Spalten der Matrix stehen für die Kanten und die Zeilen für die Knoten. Für einen einfachen, ungerichteten Graphen ist ein Eintrag 1, wenn die zugehörige Kante und der Knoten inzident sind. Ansonsten ist der Eintrag 0 /Bon08/. Pro Spalte treten also zwei 1-Einträge auf. In Bild 3b ist die Inzidenzmatrix zum Graphen aus Bild 3a dargestellt. Im Falle eines gerichteten Graphen sind die Matrixeinträge -1 bei Kanten die vom Knoten weg gerichtet sind und bei zum Knoten hin gerichteten Kanten sind die Einträge 1. Für den Anwendungsfall des Tearing wird jedoch eher selten eine Inzidenzmatrix gemäß der zuvor genannten Definition benutzt.



a)

	e_1	e_2	e_3	e_4	e_5	e_6	e_7
v_1	1	0	1	1	0	0	0
v_2	1	1	0	0	0	0	0
v_3	0	0	1	0	1	0	0
v_4	0	1	0	1	0	1	1
v_5	0	0	0	0	0	0	1
v_6	0	0	0	0	1	1	0

b)

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	0	1	1	1	0	0
v_2	1	0	0	1	0	0
v_3	1	0	0	0	0	1
v_4	1	1	0	0	1	1
v_5	0	0	0	1	0	0
v_6	0	0	1	1	0	0

c)

Bild 3 Graph (a) mit zugehöriger Inzidenzmatrix (b) und Adjazenzmatrix (c)

2.3.2 Die Adjazenzmatrix und Strukturinzidenzmatrix

Bei der Adjazenzmatrix wird der Graph nur mittels der Knoten beschrieben. Es handelt sich um eine quadratische Matrix, bei der die Zeilen und Spalten für die Knoten stehen. Sind zwei Knoten benachbart (adjazent), so ist der entsprechende Eintrag der Matrix 1, ansonsten 0. In Bild 3c ist die Adjazenzmatrix zum Graphen aus Bild 3a abgebildet. Bei ungerichteten Graphen entstehen symmetrische Matrizen. Bei gerichteten Graphen steht nur in der Zeile der Vorgängerknoten und in der Spalte der Nachfolgeknoten der gerichteten Kante eine 1. Digraphen können aus einem bipartiten Graphen abgeleitet werden indem beispielsweise die Elemente beider Teilmengen der Knoten paarweise einander zugewiesen werden. Im Fall des Tearing repräsentieren die Knoten dann paarweise Zuordnungen aus Gleichung und zu lösender Variable. Zur Darstellung des bipartiten Graphen dient jedoch eine andere Matrixform, die ebenso wie die Adjazenzmatrix nur Informationen über die Knoten bereitstellt und die Kanten unspezifiziert lässt. Es wird, basierend auf dem bipartiten Graphen eine so genannte Strukturinzidenzmatrix aufgebaut /Cel06/. Sie stellt in den Matrixzeilen die Gleichungen und in den Matrixspalten die Variablen dar. Ist das Gleichungssystem eindeutig lösbar, so handelt es sich demnach um eine quadratische Matrix. Um die Matrix zu füllen, wird folgendermaßen vorgegangen: Ist eine Variable in einer Gleichung enthalten, so wird in der entsprechenden Zelle eine 1 eingetragen, andernfalls eine 0.

2.4 Lösungsverfahren für lineare und nichtlineare Gleichungssysteme

2.4.1 Lösung linearer Gleichungssysteme

Betrachtet man ein lineares Gleichungssystem der Form $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ so kann dieses im Allgemeinen durch simples Umstellen und Einsetzen der Gleichungen ineinander gelöst werden. Äquivalent dazu existiert das Gauss'sche Eleminationsverfahren /Kan05/, welches die Systemmatrix in eine Stufenform umwandelt. Anhand des Gleichungssystems [2.3] wird das Gauss-Verfahren erläutert. Die linke Seite der Gleichung zeigt die Koeffizientenmatrix $\boldsymbol{A}$ und den Variablenvektor $\boldsymbol{x} = (x_1, x_2, x_3)^T$, die rechte Seite den konstanten Vektor $\boldsymbol{b}$.

x_1	x_2	x_3	b
1	-1	1	0
-1	-1	0	-10
0	0	1	5

[2.3]

Das Verfahren nach Gauss basiert auf grundlegenden algebraischen Operationen wie Reihen- und Spaltenpermutation als auch auf der Skalierung und Addition von Zeilen. Ziel ist es, eine Stufenform zu erreichen. Das bedeutet, dass sich die Anzahl der Nulleinträge von der ersten zur letzten Spalte jeweils um eins erhöht. Bei einer quadratischen $n \times n$ Koeffizientenmatrix besitzt die erste Zeile demzufolge keine 0-Einträge und die letzte Zeile $(n-1)$ 0-Einträge. Für das System [2.3] kann das erreicht werden, indem die erste und zweite Zeile addiert werden. Man erhält das Gleichungssystem [2.4].

x_1	x_2	x_3	b
1	-1	1	0
0	-2	1	-10
0	0	1	5

[2.4]

Nun kann schrittweise, beginnend mit der Letzten, jede Gleichung in die darüber liegende Gleichung eingesetzt werden. Es bleibt die Hauptdiagonale, die auf 1 skaliert, die Lösung des Variablenvektors liefert. Siehe dazu das System [2.5].

x_1	x_2	x_3	b
1	0	0	2.5
0	1	0	7.5
0	0	1	5

[2.5]

Möchte man einen Algorithmus zur Lösung eines linearen Gleichungssystems programmieren, bietet es sich an, eine LR-Zerlegung durchzuführen. Dabei wird die Koeffizientenmatrix $\boldsymbol{A}$ in das Produkt einer oberen Dreiecksmatrix $\boldsymbol{R}$ und einer unteren Dreiecksmatrix $\boldsymbol{L}$ umgewandelt /Bär07/, /Kan05/. Bei kleinen bis mittleren Systemen ist der Rechenaufwand dafür noch zu rechtfertigen.

2.4.2 Lösung nichtlinearer Gleichungssysteme

Für nichtlineare Gleichungssysteme kann keine allgemeingültige Lösungsmethodik angegeben werden. Analytische Lösungen können nur im Einzelfall ermittelt werden, weswegen gemeinhin eine numerische Lösung favorisiert wird. Die Lösung einer Gleichung, welche durch die Nullstelle der impliziten Gleichung gegeben ist, wird dabei schrittweise an den wahren Wert angenähert, bis ein Abbruchkriterium erfüllt wird. Das Newton-Verfahren hat sich für viele Anwendungen als gut geeignet herausgestellt. Aufgrund der Verbreitung und der Verwendung speziell zur Lösung der modellgenerierten Gleichungssysteme wird es exemplarisch für alle möglichen Verfahren vorgestellt. Es ist im Mehrdimensionalen anwendbar und setzt lediglich stetig differenzierbare Funktionen voraus, deren erste Ableitungen ohne Probleme zu berechnen sind. Der Grundgedanke der Newton-Iteration basiert auf einer Linearisierung der Funktion im Startpunkt mit einer iterativen Annäherung der Linearisierungspunkte an die Nullstelle. Für die Linearisierung der Funktion $\boldsymbol{f}(\boldsymbol{x})$ gemäß einer Taylor-Entwicklung, wird deren erste, differentielle Ableitung $\boldsymbol{f}'(\boldsymbol{x})$ im Mehrdimensionalen die sogenannte Jacobi-Matrix $\boldsymbol{J}(\boldsymbol{x})$, benötigt. Der Startwert wird im Folgenden $\boldsymbol{x_0}$ genannt und die Nullstelle $\boldsymbol{x}^*$ /Deu04/, /Bär07/.

$$0 = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{J}(\boldsymbol{x}) \cdot (\boldsymbol{x}^* - \boldsymbol{x_0}) \qquad [2.4]$$

$\boldsymbol{x}^*$ soll ermittelt werden. Deshalb wird Gleichung [2.4] folgendermaßen umgestellt.

$$\boldsymbol{x}^* = \boldsymbol{x_0} - \boldsymbol{J}(\boldsymbol{x_0}))^{-1}\boldsymbol{f}(\boldsymbol{x_0}) \qquad [2.5]$$

Da $\boldsymbol{x_0}$ nur ein geschätzter Startwert ist, ist der errechnete Wert $\boldsymbol{x}^*$ ebenfalls nur eine Korrektur des angenommenen Startpunkts. In einem iterativen Schema gemäß Gleichung [2.6] kann $\boldsymbol{x_i}$ dem wahren Wert schrittweise angenähert werden.

$$\boldsymbol{x_i} = \boldsymbol{x_{i-1}} - (\boldsymbol{J}(\boldsymbol{x_{i-1}}))^{-1}\boldsymbol{f}(\boldsymbol{x_{i-1}}) \qquad [2.6]$$

Um die Berechnung der inversen Jacobi-Matrix und die anschließende Multiplikation mit $\boldsymbol{f}(\boldsymbol{x_{i-1}})$ zu umgehen, kann die Iteration aus Formel [2.6] so umgestellt werden, dass ein lineares Gleichungssystem gemäß [2.7] entsteht.

$$\boldsymbol{J}(\boldsymbol{x_{i-1}}) \cdot \boldsymbol{\Delta x} = -\boldsymbol{f}(\boldsymbol{x_{i-1}}) \qquad [2.7]$$

Das Gleichungssystem der Form $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ wird dann nach $\boldsymbol{\Delta x}$ gelöst und dann $\boldsymbol{x_i}$ gemäß $\boldsymbol{\Delta x} = \boldsymbol{x_i} - \boldsymbol{x_{i-1}}$ gebildet /Bär07/.

Zur visuellen Veranschaulichung der Newton-Methode ist in Bild 4 das Diagramm für eine beliebige Funktion dargestellt. Es sind vier Iterationsschritte abgebildet. Der angenommene Startwert sei x_0. Für dieses Argument wird die Tangente am Funktionsgraphen konstruiert. Der Schnittpunkt der Tangente mit der Abszisse legt den korrigierten Wert x_1 für die nächste Iteration fest. Mit jedem neuen Schritt rückt der korrigierte Wert näher an die eigentliche Nullstelle x^*. Abbruchkriterium für das Beenden der Iteration kann beispielsweise die Unterschreitung einer zulässigen maximalen Differenz zwischen den errechneten Argumenten oder aber das Erreichen einer vorgegebenen Anzahl an Iterationsschritten sein/Bär07/.

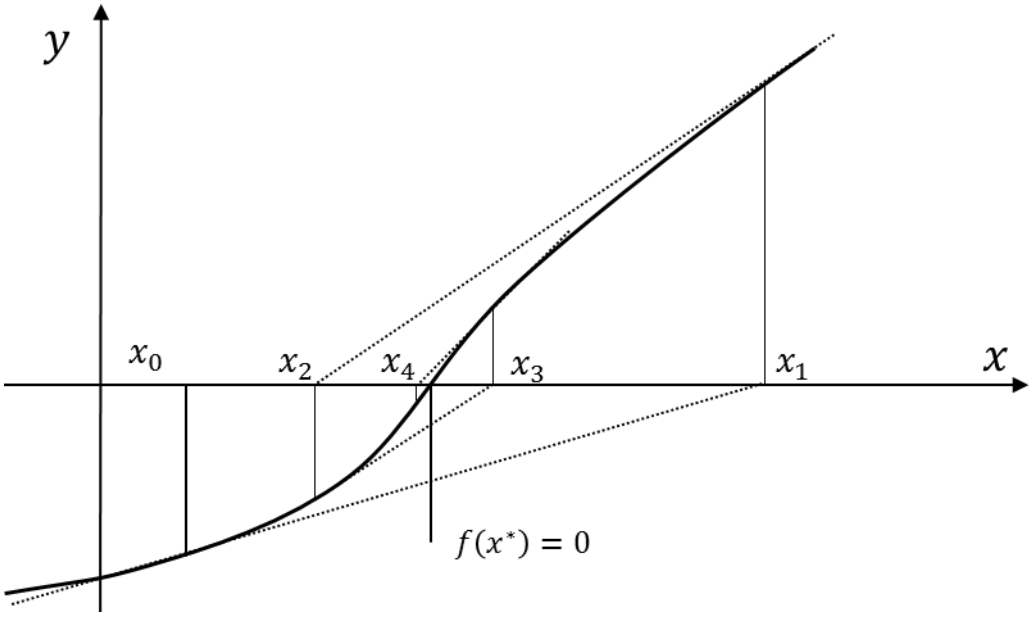


Bild 4 Newton-Verfahren

3 Methoden zur Auswahl von Tearing-Variablen

3.1 Das Konzept des Tearing

Dieses Kapitel soll einen ersten Überblick über die Grundlagen des Tearing bieten. Im Folgenden wird darauf eingegangen, wozu Tearing genutzt wird, welche Vorteile es bietet und wie es im Allgemeinen abläuft. Danach werden konkrete Tearing-Algorithmen erläutert.

3.1.1 Zweck des Tearing

Grundlegend ist Tearing ein symbolisches Verfahren, um die Lösung von Gleichungssystemen effektiver und schneller zu gestalten. Soll ein DAE-System mit großen stark zusammenhängenden Komponenten gelöst werden, so müssen alle Gleichungen, die nicht explizit eine Variable lösen können, sondern von mehreren Variablen gelichermaßen abhängen, gleichzeitig iterativ gelöst werden. Das bedeutet großen Rechenaufwand. Mit Hilfe des Tearing soll nun die Dimension des Problems reduziert werden. Es wird versucht, das System auf eine minimale Anzahl an iterativ bestimmbaren Gleichungen zu beschränken. Die restlichen Gleichungen werden dann in die explizite Form überführt um die zugewiesenen Variablen berechnen zu können. Die stark zusammenhängende Komponente stellt eine algebraische Schleife dar. Es muss eine Variable gesucht werden, die die Schleife aufbricht. Das bedeutet, dass bei dieser Tearing-Variablen die Lösung der nachfolgenden Gleichungen beginnen kann. Prinzipiell kann jede Variable als Tearing-Variable deklariert werden. Oftmals ist eine einzelne Variable jedoch nicht ausreichend, um das gesamte System zu lösen. Es müssen also noch weitere Tearing-Variablen benutzt werden. Jede einzelne muss jedoch wieder iterativ an die wahre Lösung angenähert werden, was wiederum zusätzlichen Rechenaufwand bedeutet. Es gilt die minimale Menge an Tearing-Variablen zu finden, die die Schleifen vollständig aufbrechen kann. Dieser Aufgabe kommt der Tearing-Algorithmus nach. Um die Anzahl an Iterationen minimal zu halten, muss auch die Anzahl der Tearing-Variablen mit Hilfe einer effektiven Tearing-Heuristik minimiert werden.

3.1.2 Prinzipieller Ablauf

Tearingverfahren setzen zusätzliche symbolische Operationen voraus, beziehungsweise laufen simultan zu diesen ab. Das eigentliche Tearing, welches in dieser Arbeit betrachtet wird, ist somit nur ein Teil einer Abfolge von Methoden zur Lösung von Gleichungssystemen. Die übergebenen Eingabedaten müssen zuerst in eine Darstellungsform überführt werden, auf die effektiv die graphentheoretischen Operationen angewandt werden können. Die Gesamtmenge wird dann, sofern möglich, in Teilsysteme stark zusammenhängender Komponenten unterteilt. Diesen Vorgang bezeichnet man als *partitioning* (engl. Aufteilung). Die Matrix hat nun eine untere–Blockdreicksform (engl.: *BLT-form, block-lower-triangular-form*) Statt das gesamte Gleichungssystem zu untersuchen, können nun kleinere Subsysteme nacheinander bearbeitet werden, die durch die Blöcke dargestellt sind. Diese beinhalten stark zusammenhängende Komponenten, die algebraische Schleifen bilden. Es gilt nun, die algebraischen Schleifen zu brechen und die Gleichungen zu lösen. Das Aufbrechen der Schleifen (engl. *tearing*) und das Zuordnen der Variablen zu den Gleichungen (engl. *output assignment*) geschieht je nach verwendetem Tearing-Algorithmus simultan oder nacheinander. Nachfolgend ist es möglich, die Variablen iterativ mittels Newton-Verfahren zu bestimmen. Bild 5 zeigt wie das Tearing chronologisch in den Lösungsablauf einzuordnen ist.

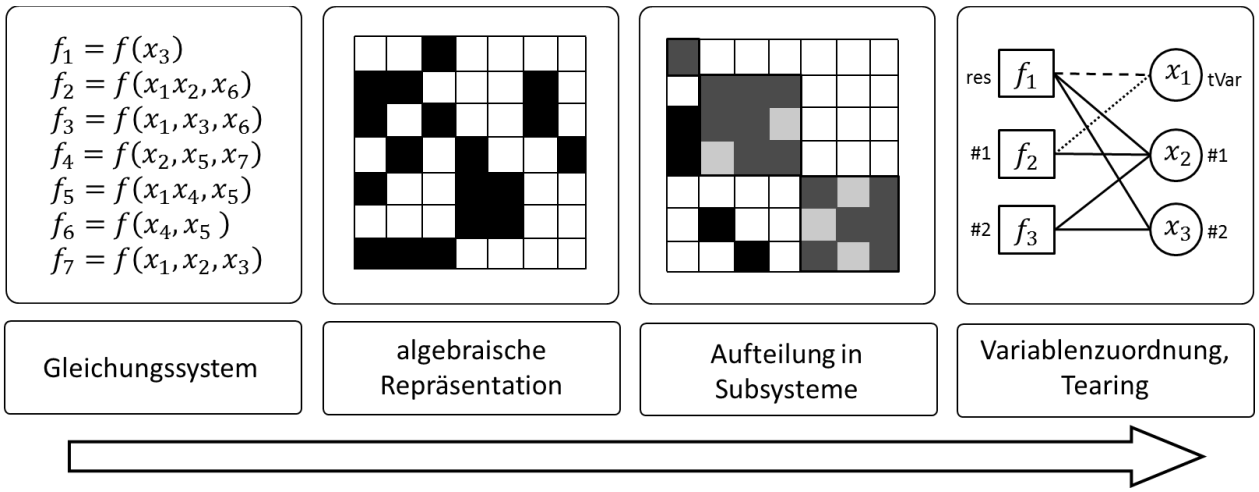


Bild 5 Einordnung des Tearing-Verfahren

Das übergebene Gleichungssystem kann sowohl algebraische Gleichungen als auch Differentialgleichungen enthalten. Es handelt sich demnach um ein DAE-System

(*differential-algebraic-equation-system,* engl.: differentiell–algebraisches-Gleichungssystem). Die Gleichungen sind impliziert notiert, da noch keine Variablenzuweisung für die Gleichungen getroffen ist.

Zur weiteren Bearbeitung wird der Zusammenhang zwischen Gleichungen und enthaltenen Variablen in Matrixform dargestellt. In diesem Fall wurde die Strukturinzidenzmatrix als geeignete Repräsentation gewählt. Da für die graphentheoretischen Umformungen vorerst nur die Inzidenzstruktur von Nöten ist (bei weiterführenden Betrachtungen kann die Lösbarkeit von Gleichungen nach einer Variable von Interesse sein), wird lediglich die Existenz einer Variablen in der jeweiligen Gleichung erfasst.

Die Strukturinzidenzmatrix des bipartiten Graphen kann durch Permutation der Reihen und Spalten in eine untere Blockdreiecksform gebracht werden. Die Blöcke repräsentieren die stark zusammenhängenden Komponenten. Diagonalelement außerhalb von Blöcken stellen explizite Zuordnungen von Variablen zu Gleichungen dar. Diese können, beginnend mit der Variablen in der linken, oberen Ecke nacheinander gelöst werden. Die Blöcke hingegen müssen einzeln mittels Tearing behandelt werden.

3.2 Systematisierung von Tearing-Algorithmen

Um einen Überblick über die Arbeitsweise von Algorithmen zur Auswahl von Tearing-Variablen zu erlangen, ist es nützlich, sie systematisch nach verschiedenen Kriterien zu unterteilen. Hier soll nun eine Übersicht über mögliche Klassifizierungen gegeben werden.

3.2.1 Knoten- und Kanten-Tearing

Tearing bedeutet vereinfacht, dass Schleifen in einem Graphen aufgebrochen werden. Dies kann einerseits dadurch geschehen, dass Knoten entfernt werden (engl. *node tearing*), so wie es am häufigsten in der Literatur dargestellt wird, oder dass nur einzelne Kanten (engl. *branch tearing*), ausgehend von der Tearing-Variable entfernt werden. Beim Knoten-Tearing werden alle inzidenten Kanten zu einem Knoten entfernt, wohingegen beim Kanten-Tearing zumindest eine inzidente Kante zum Knoten bestehen bleibt. Die Konsequenz beider Verfahren liegt darin, wie die Tearing-Variable iteriert wird. Beim Knoten-Tearing werden die Iterationsgrößen mittels Newton-Iteration berechnet. Die notwendige Gleichung, deren erste Ableitung nach der Tearing-Variable gebildet werden muss, ist die Residuengleichung. Beim Kanten-Tearing werden die Tearing-Variablen mittels Gauss-Seidel Verfahren (Einzelschrittverfahren) oder Jacobi-Verfahren (Gesamtschrittverfahren) iteriert. Bild 6 zeigt die unterschiedlichen Verfahren anhand eines Digraphen. Beim Kanten-Tearing wird die Tearing-Variable in die nachfolgenden Gleichungen des Kreises eingesetzt und das System solange aufgelöst, bis die Residuengleichung erreicht ist. Dann kann der nächste Iterationsschritt der Tearing-Variable berechnet werden. Es kann unterschieden werden, ob man die Variable in einer oder in mehreren Gleichungen als bekannt setzt. Wird die Variable nur in einer Gleichung bekannt gesetzt und in den Anderen berechnet, handelt es sich um ein Mehrschrittverfahren (Jacobi-Verfahren). Wird die Tearing-Variable in alle Gleichungen, in der sie vorkommt eingesetzt und das System gelöst, spricht man von einem Einzelschrittverfahren (Gauss-Seidel-Verfahren). Beim Knoten-Tearing muss die Residuengleichung nach der Tearing-Variable abgeleitet werden. Alle unbekannten Variablen in der Residuengleichung sind mit Hilfe der Tearing-Variablen

zu bestimmen. Der nächste Iterationsschritt wird mittels Newton-Verfahren ermittelt /Elm94/.

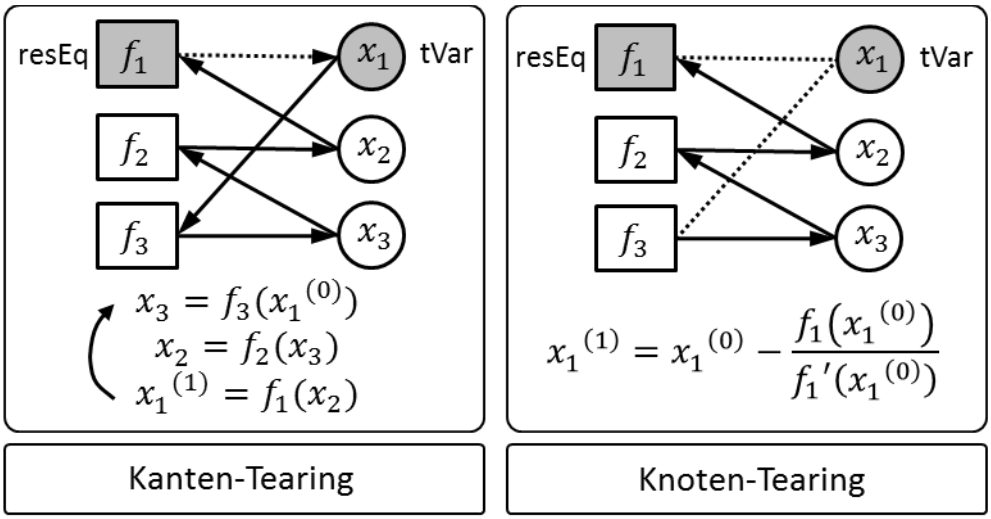


Bild 6 Kanten- und Knoten-Tearing

3.2.2 Vorangehende und partielle Variablenzuordnung

Mit dem Tearing-Prozess geht auch immer eine Variablenzuordnung (engl.: *matching*) einher. Jeder Gleichung wird eine Variable zugeordnet, nach der die Gleichung umgestellt wird, um sie zu lösen. Diese Gleichung wird als kausalisierte Gleichung bezeichnet. Ein kausalisiertes Gleichungssystem ist demnach ein System mit vollständiger Variablenzuordnung. Tearing-Methoden können nun dahingehend unterschieden werden, ob die Auswahl der Tearing-Variablen während der Variablenzuordnung ablaufen oder erst nach einer kompletten Variablenzuordnung erfolgen soll. Diese Unterscheidung ist grundlegend für die Funktionsweise eines Tearing-Algorithmus. In Bild 7 ist eine schematische Einordnung des Tearing mit vorangehendem und partiellem Matching gegeben.

Beim Tearing mit vorangehender Variablenzuordnung besteht die Idee darin, den Informationsfluss innerhalb des Gleichungssystems zu erfassen, die Schleifen zu identifizieren und sukzessiv aufzubrechen. Die Richtung der Information kann jedoch nur in einem gerichteten Graphen, einem Digraphen, wiedergegeben werden. Dazu muss das System jedoch vollständig kausalisiert werden.

Hierzu kann beispielsweise der Steward-Matching-Algorithmus genutzt werden (siehe Kapitel 3.3.2). Mit Hilfe der Information über das kausalisierte System wird dann der Digraph erzeugt. Die Knoten repräsentieren jeweils eine Gleichung mit zugeordneter Variablen. Die Anzahl der Knoten halbiert sich und die bipartite Struktur geht verloren. Auf dieser Grundlage können weitere Operationen durchgeführt werden, um alle auftretenden Schleifen aufzubrechen. Der Ollero-Amselem-Algorithmus (siehe Kapitel 3.6) und der Steward-Algorithmus (siehe Kapitel 3.7) funktionieren nach diesem Prinzip.

Tearing mit partieller Variablenzuordnung basiert auf einer schrittweise-unterbrochenen Kausalisierung des Systems. Der Matching-Algorithmus (zum Beispiel Tarjan-Matching, siehe Kapitel 3.3.1) arbeitet solange eine eindeutige Zuordnung getroffen werden kann. Besteht keine Möglichkeit mehr das System weiter eindeutig zu kausalisieren, wird das Matching abgebrochen und es muss eine Tearing-Variable ausgewählt werden. Im Anschluss wird die Kausalisierung fortgeführt. Sollte die Kausalisierung wieder abbrechen, muss eine neue Tearing-Variable gewählt werden. Dieser Zyklus wird solange durchlaufen, bis ein komplett kausalisiertes System entstanden ist. Alle Variablen, die währenddessen für das Tearing gewählt wurden, bilden die vollständige Menge der Tearing-Variablen, das sogenannte Tearingset.

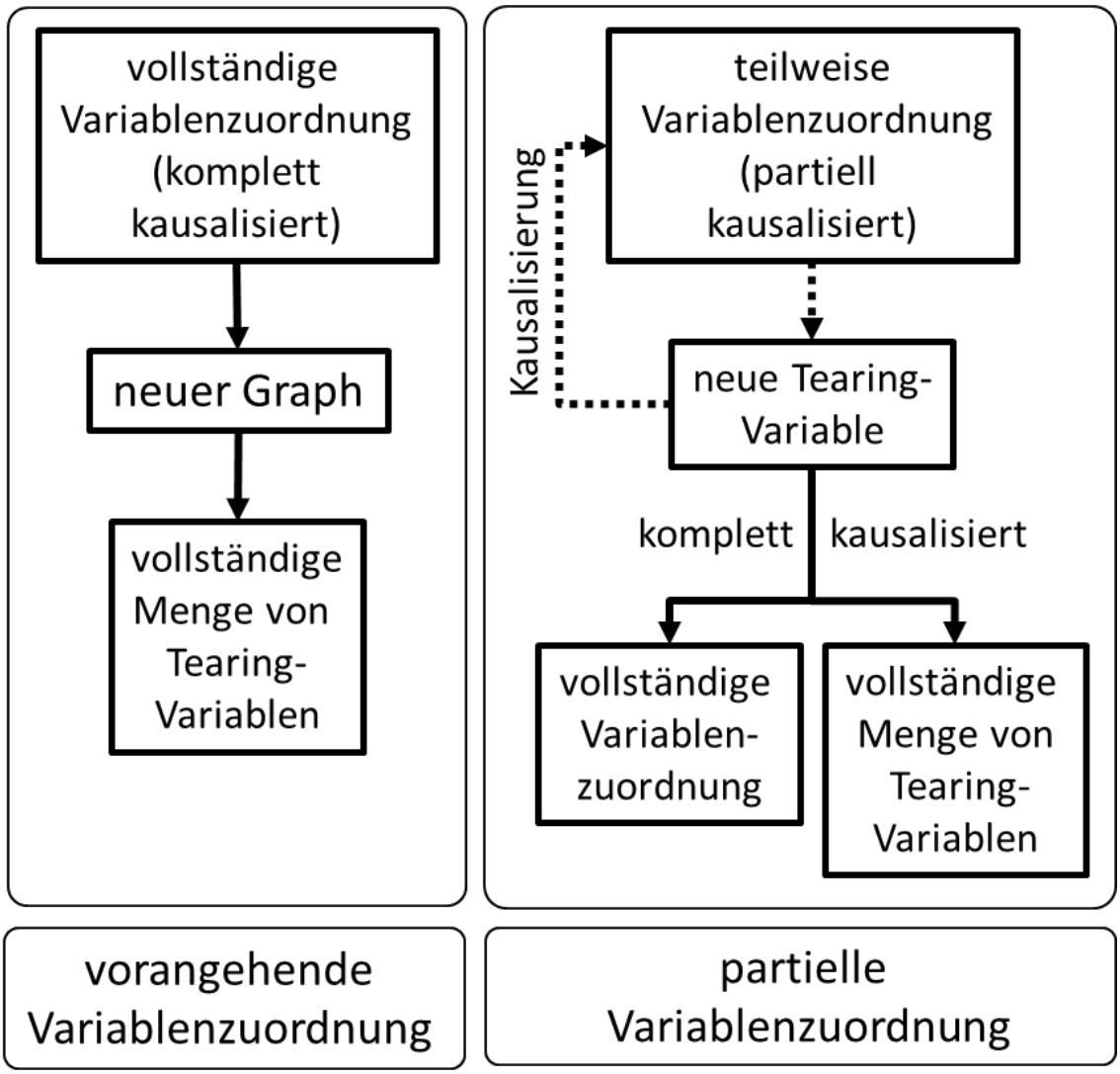


Bild 7 Unterteilung in Verfahren mit vorangehender und partieller Variablenzuordnung

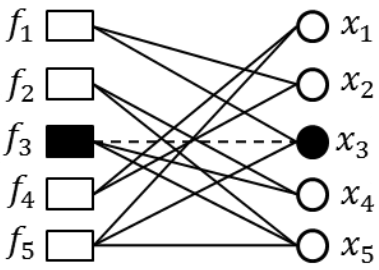
3.3 Algorithmen zur Variablenzuordnung

3.3.1 Matching nach Tarjan

Die Variablenzuordnung nach Tarjan, /Cel06/ kausalisiert das System schrittweise. Gleichzeitig zur Zuweisung der Variablen zu den lösenden Gleichungen wird auch die Lösungsreihenfolge der Gleichungen festgelegt. Das Matching nach Tarjan arbeitet mit dem bipartiten Graphen beziehungsweise mit der zugehörigen Strukturinzidenzmatrix. Für das Matching nach Tarjan ist es sowohl wichtig zu wissen, welche Gleichung welche Variablen enthält, als auch die Information zu kennen, welche Variable in welcher Gleichung vorkommt. Für die Implementierung wird demnach die Strukturinzidenzmatrix sowie deren Transponierte ausgewertet. Das zunächst vollständig akausale System soll Schritt für Schritt kausalisiert werden, so dass jede Gleichung einer Variablen zugeordnet wird und diese löst. Außerdem wird festgelegt in welcher Abfolge die Gleichungen gelöst werden können. Eine Gleichung, die eine Variable löst, wird *assignment* (engl. Zuweisung) genannt. Für die Kausalisierung nach Tarjan gibt es zwei grundlegende Regeln /Cel06/.

1. Wenn eine Gleichung nur eine unbekannte Variable enthält, nach der noch keine andere Gleichung umgestellt wird, so wird diese Gleichung nach dieser Unbekannten gelöst.
2. Wenn eine Variable nur in einer Gleichung auftritt, so muss diese Gleichung nach der Variablen gelöst werden.

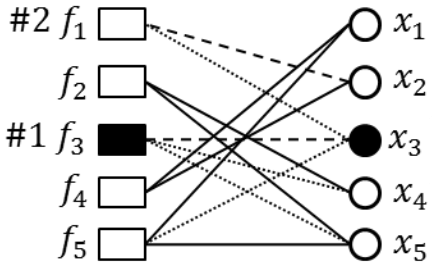
Diese Methode soll anhand des Beispiels aus Bild 8 veranschaulicht werden.



	x_1	x_2	x_3	x_4	x_5
f_1	0	1	1	0	0
f_2	0	0	0	1	1
f_3	0	0	1	1	1
f_4	1	1	0	0	0
f_5	1	0	1	0	1

Bild 8 bipartiter Graph und Strukturinzidenzmatrix eines akausalen Systems

Da hier die Darstellung des Tearing mit simultaner Variablenzuordnung präsentiert werden soll, wird eine Beispiel gewählt, welches nicht von vornherein komplett kausalisierbar ist. Das wäre beispielsweise eine stark zusammenhängende Komponente, wie man sie in einem Block aus der BLT-Transformation erhält. Kann innerhalb des Graphen keine Zuweisung getroffen werden, weil eine algebraische Schleife enthalten ist, so muss eine Tearing-Variable ausgewählt werden, die die weitere Kausalisierung ermöglicht. Wie die Variable gewählt werden kann, wird in späteren Kapiteln erläutert. Hier wird nur das Matching betrachtet. Im Beispiel wird x_3 als Tearing-Variable deklariert. Dieser Variablen wird die Gleichung f_3 zugeordnet, welche Residuengleichung genannt wird. Die Zuordnung von x_3 zu f_3 wird mit einer gestrichelten Linie gekennzeichnet. Nach der Bestimmung der Tearing-Variable kann der Tarjan-Algorithmus fortgesetzt werden. In Bild 9 ist eine weitere Zuweisung getroffen. Da x_3 nun bekannt ist, wird jede weitere inzidente Kante zum Variablenknoten gepunktet dargestellt. Ebenso wird jede inzidente Kante zu f_3 gepunktet, um zu veranschaulichen, dass diese Gleichung nicht mehr zugeordnet werden darf.



	x_3	x_2	x_1	x_4	x_5
f_3	1	0	0	1	1
f_1	1	1	0	0	0
f_2	0	0	0	1	1
f_4	0	1	1	0	0
f_5	1	0	1	0	1

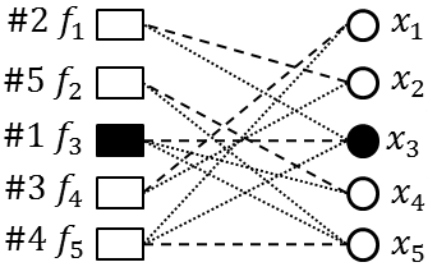
Bild 9 bipartiter Graph und Strukturinzidenzmatrix eines partiell kausalisierten Systems

Da nun die Regeln zur Kennzeichnung der Kanten eingeführt sind, können die Grundsätze des Tarjan-Matching dahingehend erweitert werden /Cel06/:

1. Unter allen nicht-kausalisierten Gleichungen, wird eine Gleichung mit nur einer schwarzen, inzidenten Kante gesucht. Die Variable zu dieser Kante wird der Gleichung zugeordnet. Die Kante wird gestrichelt und alle übrigen inzidenten Kanten zu Gleichung und Variable werden gepunktet. Die Gleichung bekommt die nächstkleinste freie Laufnummer beginnend bei 1.

2. Unter allen nicht-zugewiesenen Variablen, wird Eine gesucht, die nur eine schwarze inzidente Kante besitzt. Die Gleichung zu dieser Kante wird der Variablen zugeordnet. Die Kante wird gestrichelt und alle übrigen inzidenten Kanten zu Gleichung und Variable werden gepunktet. Die Gleichung bekommt die nächstgrößte, freie Laufnummer beginnend bei n, wobei n die Anzahl aller Gleichungen angibt.

Um fortzufahren wird nun eine Gleichung mit nur noch einer schwarzen Kante gesucht. Da f_1 nur noch abhängig von x_2 ist, wird sie dieser Variablen zugewiesen. Da es sich um die zweite getroffene Variablenzuordnung handelt, bekommt die Gleichung die Laufnummer #2. Diese legt die Lösungsreihenfolge fest. Die Strukturinzidenzmatrix des Systems kann äquivalent umgeformt werden, so dass die Kausalisierung ebenfalls abgebildet wird. Dazu müssen die Reihen und Spalten permutiert werden. Die Gleichungen in den Reihen werden entsprechend ihrer Laufnummer von oben nach unten angeordnet. Die Variablen in den Spalten werden so vertauscht, dass die Einträge der zugeordneten Gleichung die Hauptdiagonale der Matrix bilden. Mit Ausnahme der Tearing-Variablen und der Residuengleichung bildet das kausale System eine untere Dreiecksmatrix mit einer vollbesetzten Hauptdiagonalen. Nach Abschluss des Tarjan-Algorithmus ergibt sich ein vollständig kausalisiertes Gleichungssystem, wie in Bild 10 dargestellt.



	x_3	x_2	x_1	x_5	x_4
f_3	1	0	0	1	1
f_1	1	1	0	0	0
f_4	0	1	1	0	0
f_5	1	0	1	1	0
f_2	0	0	0	1	1

Bild 10 bipartiter Graph und Strukturinzidenzmatrix eines vollständig kausalisierten Systems

3.3.2 Matching nach Steward

Um einen Digraphen zu bilden, ist es notwendig, die Richtung des Informationsflusses innerhalb des Gleichungssystems zu bestimmen. Dafür muss jedoch zunächst geklärt werden, welche Variablen von welcher Gleichung gelöst werden und welche Variablen nötig sind, um eine Gleichung zu berechnen. Die Auswahl der Variablenzuordnung ist dabei nicht eindeutig. Je nach verwendeter Zuordnungsmethode kann das Matching-Ergebnis unterschiedlich ausfallen. Im vorliegenden Anwendungsfall ist jede beliebige, eindeutige Variablenzuordnung ausreichend. Allerdings ist der resultierende Digraph in jedem Fall ein anderer. Das ermittelte Tearingset ist demnach von der vorangehenden Variablenzuordnung abhängig. Für ein eindeutiges Matching wird aber in jedem Fall ein vollständiges Tearingset ermittelt. Eine eindeutige Zuordnung ist möglich, solange genauso viele Gleichungen wie Variablen vorliegen. Sind mehr oder weniger Gleichungen als Variablen vorhanden, ist das System über- beziehungsweise unterbestimmt. Sollte dies der Fall sein, so wird das während des Steward-Algorithmus erkannt. Grundlage für das Steward-Matching ist die Strukturinzidenzmatrix. In Bild 11 wird das Verfahren beispielhaft dargestellt. Prinzipiell gestaltet sich der Algorithmus in zwei Phasen: *assignment* (engl. Zuweisung) und *reassignment* (engl. Neuzuweisung). Während eines assignments wird eine Gleichung einer Variablen zugeordnet. Dieser Schritt wird solange wiederholt bis eine bereits zugewiesene Gleichung erneut einer Variablen zugeordnet werden soll. In diesem Fall tritt ein Reassignment ein. Die bereits zugewiesene Gleichung muss einer anderen Variablen zugewiesen werden. Wie im vorliegenden Beispiel zu erkennen, wird der ersten Variablen x_1 die erste abhängige Gleichung f_4 zugewiesen. Die zugehörige Reihe und Spalte wird mit einem * markiert. Das bedeutet, dass diese Reihe beziehungsweise Spalte nicht mehr für ein Reassignment in Frage kommt. Diese Markierung dient dazu, eine unendliche Schleife von Reassignments zu verhindern. In Bild 11c wird die Variable x_2 der Gleichung f_1 zugewiesen. Vor jedem gültigen assignment werden die bisherigen Markierungen entfernt und die neuen Markierungen gesetzt. Im nächsten Schritt wird eine Variable der bereits zugewiesenen Gleichung f_1 zugeordnet. Es ist ein Reassignment notwendig, welches die Variable x_2 der nächsten abhängigen Gleichung f_4 zuordnet. Markierte Reihen sind für ein Reassignment gesperrt.

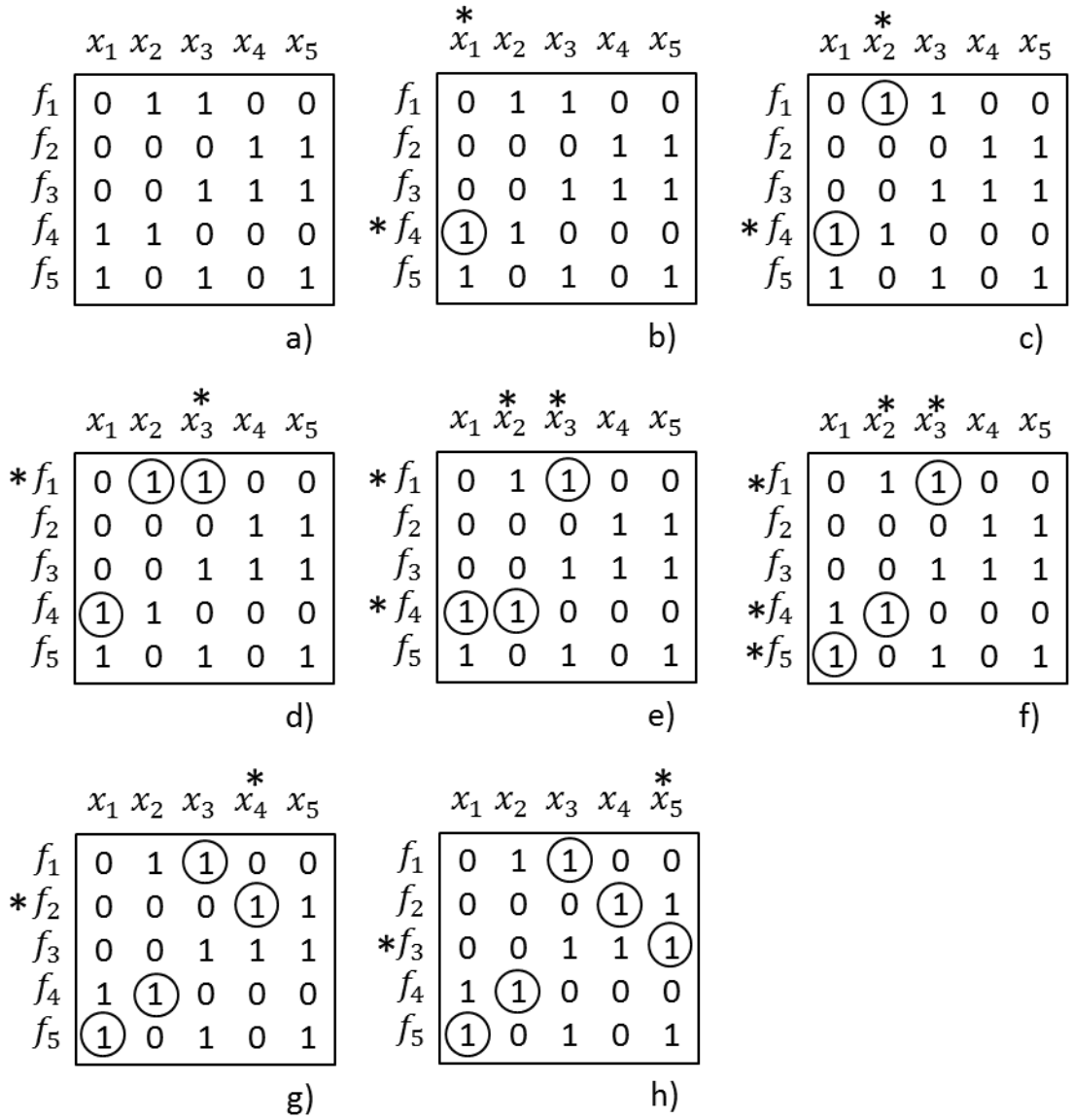


Bild 11 Steward-Matching anhand der Strukturinzidenzmatrix

Der Algorithmus läuft solange weiter bis eine vollständige Variablenzuordnung getroffen ist. Sollte es nicht mehr möglich sein, eine Variable zuzuordnen, enthält das Gleichungssystem weniger Variablen als Gleichungen.

3.4 Heuristik nach Cellier

François E. Cellier beschreibt eine Tearing-Heuristik mit partieller Variablenzuordnung, basierend auf dem Tarjan-Matching (siehe Kapitel 3.3.1) /Cel06/. Die Auswahl der Tearing-Variablen erfolgt also erst, nachdem das Matching gestoppt hat. Die Tearing-Heuristik arbeitet stets mit dem partiell kausalisierten System. Das bedeutet, dass nach einer unvollständigen Variablenzuordnung, gemäß des Tarjan-Matching, bereits zugewiesene Variablen und Gleichungen nicht mit in die Handlungsvorschrift einbezogen werden. Anhand des nachfolgenden Beispiels aus Bild 12a soll die Arbeitsweise der Cellier-Heuristik erläutert werden.

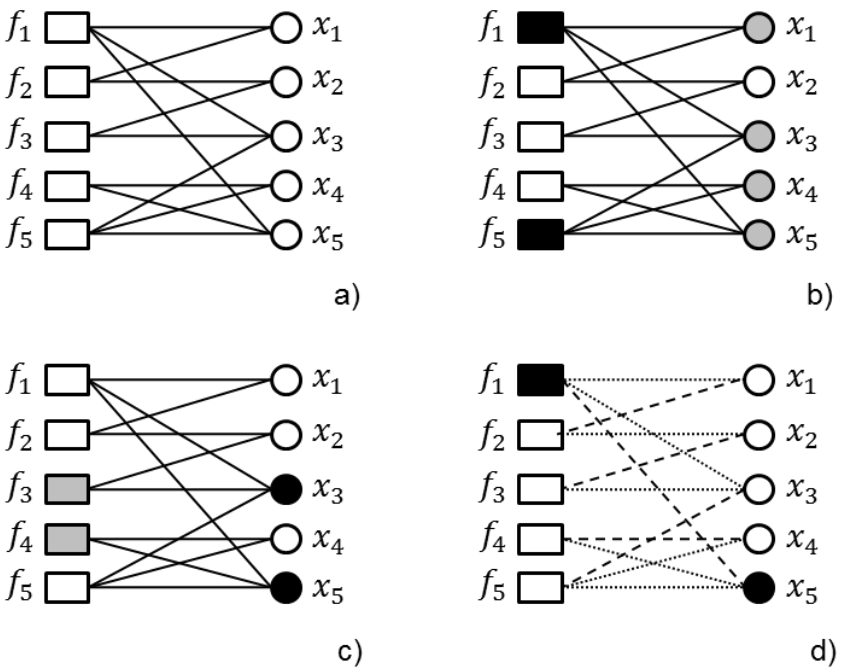


Bild 12 Beispielauswahl von Tearing-Variablen gemäß der Cellier-Heuristik

Das Cellier-Verfahren zur Tearing-Variablenbestimmung kann in vier Schritten abgearbeitet werden /Cel06/.

1. Wähle die Gleichungen aus dem bipartiten Graphen aus, die die meisten adjazenten Variablen besitzen.
2. Von diesen adjazenten Variablen, wähle jene aus, zu denen wiederum die meisten Gleichungen adjazent sind.

3. Für jede dieser Variable wird die Anzahl Gleichungen ermittelt, die kausalisiert werden können, wenn die jeweilige Variable als bekannt gilt.
4. Es wird eine von den Variablen, die die meisten Gleichungen kausalisieren würde, als Tearing-Variable ausgewählt.

Im Beispiel besitzen Gleichung f_1 und f_5 je drei adjazente Variablen. Bild 12b zeigt diese Gleichungen als schwarz markiert und die adjazenten Variablen als grau markiert. Von diesen Gleichungen besitzen x_1 und x_4 nur zwei adjazente Gleichungen, wohingegen Variable x_3 und Variable x_5 in drei Gleichungen vorkommen. Es werden im nächsten Schritt (Bild 12c) die von x_3 und x_5 kausalisierbaren Gleichungen bestimmt. Diese entsprechen den adjazenten Gleichungen mit nur zwei Kanten. Wird eine dieser Kanten entfernt, so kann die Gleichung eine neue Variable lösen. Gleichungen mit mehr als zwei Kanten werden demnach nicht mitgezählt. Die betrachteten Variablen grenzen an jeweils eine kausalisierbare Gleichung, nämlich für x_3 f_3 und für x_5 f_4. Die Gleichungen f_1 und f_5 sind zwar adjazent zu den gewählten Variablen, sind aber nicht kausalisierbar. Beide Variablen besitzen die gleiche Anzahl an kausalisierbaren Gleichungen. Da es in der Cellier-Heuristik keine weiteren Auswahlkriterien gibt, kann eine beliebige Variable gewählt werden. Im Beispiel aus Bild 12d wird Variable x_5 für das Tearing gewählt. Im Zuge der Variablenauswahl muss für die spätere Iteration auch eine Residuengleichung bestimmt werden. Der Cellier-Algorithmus gibt dafür keine konkreten Vorschläge. Hier wird aus allen adjazenten Gleichungen mit den meisten adjazenten Variablen die erste Gleichung f_1 ausgewählt. Bild 12d zeigt das mittels Tarjan-Matching vollständig kausalisierte System mit der schwarz-markierten Tearing-Variablen und Residuengleichung.

3.5 Heuristik nach Carpanzano

Die Methodik von Emanuele Carpanzano verfolgt genau wie das Verfahren von Cellier eine Tearing-Auswahl während der Variablenzuordnung /Car00/. Ziel der Heuristik, die der Tearing-Auswahl zu Grunde liegt, ist es mit einer Tearing-Variablen so viele *assignments* (engl. Zuweisungen) zu erzeugen, wie möglich. Ein Assignment ist eine Gleichung, die nur eine Variable enthält und nach dieser aufgelöst werden kann. Dies führt zu einem wichtigen Fakt bei der Auswahl der Tearing-Variablen. Im Gegensatz zu den anderen, vorgestellten Tearing-Algorithmen, die nur auf der symbolischen Ebene des Graphen arbeiten, berücksichtigt Carpanzano auch die Lösbarkeit. Für die praktische Umsetzung des Tearing ist die Lösbarkeit einer Gleichung nach einer Variablen von großer Bedeutung. Ist eine Variable einer Gleichung zugewiesen, nach der sie nicht aufgelöst werden kann, so ist eine numerische Lösung sehr aufwendig beziehungsweise aufgrund einer möglichen Division durch Null nicht möglich. Deswegen ist es notwendig, Gleichungen, die nach keiner Variablen aufgelöst werden können, als Tearing-Variable zu deklarieren. Um den Aspekt der Lösbarkeit in die Tearing-Heuristik einfließen zu lassen, führt Carpanzano die *reduced incidence matrix* (engl. reduzierte Inzidenzmatrix) ein (siehe Bild 13).

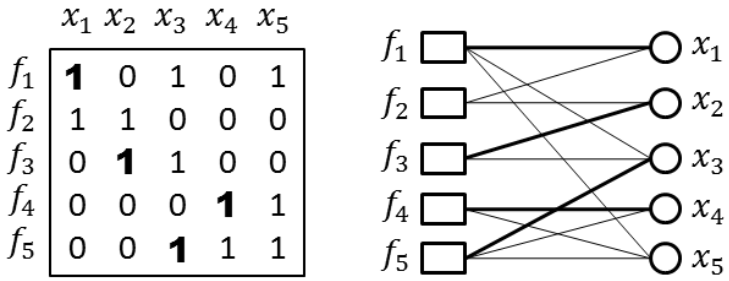


Bild 13 reduzierte Inzidenzmatrix und zugehöriger bipartiter Graph

Die Matrixstruktur entspricht dem Aufbau einer Strukturinzidenzmatrix. Die Reihen repräsentieren die Gleichungen, die Spalten die Variablen und ein 1-Eintrag steht für das Vorkommen einer Variablen in einer Gleichung. Ist eine Gleichung nach einer Variablen auflösbar, so ist die zugehörige Kante im bipartiten Graphen fett-gedruckt. Der Eintrag in der reduzierten Inzidenzmatrix ist ebenfalls fett-gedruckt. Dünn

gedruckte Kanten und Matrixeinträge signalisieren demnach eine Nicht-Lösbarkeit Wie bestimmt wird, ob eine Gleichung nach einer Variablen umgestellt werden kann, hängt von den Möglichkeiten ab, die der Löser bereitstellt. Darauf, wie die Bewertung der Lösbarkeit durchgeführt werden kann, wird hier nicht eingegangen. Im vorliegenden Beispiel gilt alles als lösbar, solange es analytisch umformbar ist. Die Variablenzuordnung nach Carpanzano gestaltet sich prinzipiell so wie es im Matching-Algorithmus nach Tarjan beschrieben wird. Zum einen werden Gleichungen, die nur noch eine inzidente Kante besitzen (assignments) dieser Variablen zugeordnet. Diese Gleichungen werden dann zusammen mit der zugeordneten Variablen und allen inzidenten Kanten aus dem bipartiten Graphen entfernt. Andernfalls muss eine Tearing-Variable bestimmt werden, welche dann mit den inzidenten Kanten aus dem Graphen entfernt wird. Die Residuengleichungen ergeben sich automatisch dadurch, dass pro deklarierte Tearing-Variable eine nicht zugeordnete, implizite Gleichung am Ende des matching-Prozesses übrig bleibt. Für die Auswahl der Tearing-Variablen gibt Carpanzano drei Heuristiken an, die auf den teilweise kausalisierten Graphen angewendet werden /Car00/.

1. Wähle eine Variable, die über eine nicht fett-gedruckte Kante mit einer Gleichung verbunden ist. Unter all diesen möglichen Variablen wird die gewählt, die die geringste Anzahl an inzidenten Kanten aufweist.
2. Jeder Variablen x_i wird ein Gewicht p_i gemäß Gleichung [3.1] gegeben.

$$p_i = \left[\left(1 + \frac{1}{m}\right) \cdot m_{nb} + m_b\right] \qquad [3.1]$$

   Dabei ist m die Anzahl aller Variablen, m_{nb} die Anzahl aller nicht fett-gedruckten inzidenten Kanten und m_b die Anzahl der fett-gedruckten Kanten.

3. Jeder Kante wird entsprechend der reziproken Anzahl der zum zugehörigen Gleichungsknoten inzidenten Kanten gewichtet. Jede Variable erhält ein Gewicht, welches sich aus der Summe der Gewichte der inzidenten Kanten zusammensetzt. Gewählt wird die Kante, die das höchste Gewicht besitzt.

Diese Regeln zur Tearing-Auswahl sind universell anwendbar und domänenunspezifisch. Zudem weißt Carpanzano darauf hin, dass es in Spezialfällen, wie zum Beispiel bei Mehrkörpersystemen mit Baumstruktur möglich ist, das

Tearingset über domänenspezifische Heuristiken zu ermitteln. Allerdings ist in einem domänenübergreifenden Modell eine automatische und universelle Tearing-Auswahl, wie die oben vorgestellte Heuristik, notwendig.

3.6 Tearing-Auswahl nach Ollero-Amselem

Das Verfahren nach Ollero-Amselem ist ein Verfahren, welches eine komplette Variablenzuordnung voraussetzt. Für die vorangehende Variablenzuordnung wird das Steward-Matching (siehe Kapitel 3.3.2) verwendet. Mit der Information über die Zuordnung wird aus dem ursprünglichen bipartiten Graphen zunächst der Digraph des kausalisierten Systems und dann der sogenannte Signalflussgraph (engl.: *signal-flow-graph*) aufgebaut. Aus diesem werden solange Knoten kontrahiert und gelöscht, bis der Graph leer ist. Währenddessen werden bestimmte Knoten in das sogenannte *essential set* (engl. notwendige Menge) gelegt. Alle Knoten des essential sets können die Schleifen des Systems vollständig lösen. Bevor der eigentliche Tearing-Algorithmus angewendet werden kann, muss zunächst der Signalflussgraph hergeleitet werden /Yan07/. Für den Signalflussgraphen wird der Digraph des gematchten Systems benötigt. Im nachfolgenden Beispiel wird dargestellt, wie sich der Signalflussplan aufbauen lässt.

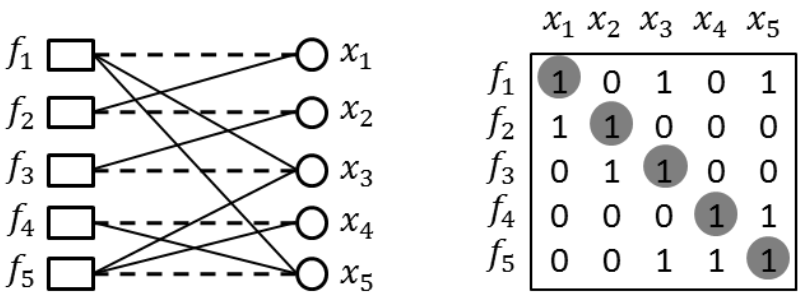
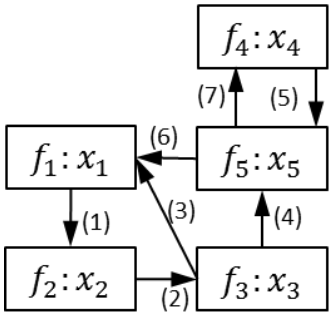


Bild 14 bipartiter Graph und Strukturinzidenzmatrix mit vollständiger Variablenzuordnung

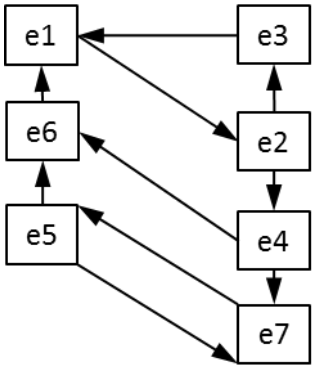
Bild 14 zeigt das Anfangssystem in Form eines bipartiten Graphen sowie als Strukturinzidenzmatrix. Die Gleichungen f_i werden als rechteckige Knoten, die Variablen x_i als kreisförmige Knoten dargestellt. Das System wurde bereits gematcht und die getroffene Zuordnung ist einerseits durch gestrichelte Linien im Graphen, sowie durch grau markierte Elemente in der Strukturinzidenzmatrix gekennzeichnet. In einem Zwischenschritt wird das System in einen Digraphen überführt (Bild 15), der in Form einer Adjazenzmatrix algebraisch repräsentiert wird.



	$f_1: x_1$	$f_2: x_2$	$f_3: x_3$	$f_4: x_4$	$f_5: x_5$
$f_1: x_1$	0	1	0	0	0
$f_2: x_2$	0	0	1	0	0
$f_3: x_3$	1	0	0	0	1
$f_4: x_4$	0	0	0	0	1
$f_5: x_5$	1	0	0	1	0

Bild 15 Digraph und Adjazenzmatrix des gematchten Systems

Die Knoten stehen für die zugeordneten Paare von Gleichung und zu lösender Variable. Im Digraphen zeigen die negativ inzidenten Kanten eines Knoten $f: x$ die Menge der notwendigen Variablen, die nötig ist um die Gleichung f nach x zu lösen. Die positiv inzidenten Kanten zeigen auf die Gleichungen, in denen die Variable x auftaucht. Im Signalflussplan hingegen werden die Knoten aus den Kanten des gerichteten Graphen gebildet. Deswegen werden die gerichteten Kanten im Digraphen nummeriert. Die Nummerierung erfolgt hierbei gemäß dem Auftreten der 1-Einträge in der Adjazenzmatrix. Der Signalflussgraph wird dementsprechend genauso viele Knoten enthalten, wie es Kanten im Digraphen des gematchten Systems gibt /Mah90/.



	e1	e2	e3	e4	e5	e6	e7
e1	0	1	0	0	0	0	0
e2	0	0	1	1	0	0	0
e3	1	0	0	0	0	0	0
e4	0	0	0	0	0	1	1
e5	0	0	0	0	0	1	1
e6	1	0	0	0	0	0	0
e7	0	0	0	0	1	0	0

Bild 16 Signalflussgraph und zugehörige Adjazenzmatrix

In Bild 16 sieht man den fertigen Signalflussgraphen. Ein Knoten e_i hat eine gerichtete Kante in den Knoten e_j, wenn die Kante i zu einem Knoten im zugehörigen Digraphen negativ inzident und die Kante j zum selben Knoten positiv inzident ist.

Dieser Graph bildet die Grundlage für die Tearingmethode nach Ollero-Amselem. Auf den Signalflussgraphen werden folgende graphentheoretische Operationen angewandt /Fun89/, /Tit11/:

Entfernen von Knoten: Wird ein Knoten v aus einem Graphen G entfernt, so wird der Knoten mit allen inzidenten Kanten entfernt (siehe Bild 17b).

Kontraktion von Knoten: Wird ein Knoten v aus einem Graphen G kontrahiert, so wird der Knoten entfernt. Es wird eine neue Menge von Kanten hinzugefügt, die die unmittelbaren Vorgänger mit den unmittelbaren Nachfolgern des kontrahierten Knotens verbinden (siehe Bild 17c).

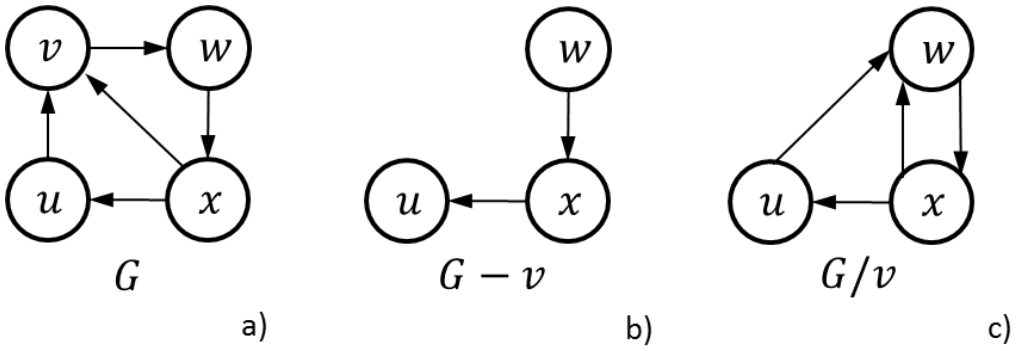


Bild 17 Vereinfachungen von Graphen

Diese direkten Graphenvereinfachungen werden solange auf den Signalflussgraphen angewandt, bis ein leerer Graph entsteht. Das Entfernen von Knoten wird angewandt bei:

1. Knoten ohne eine negativ inzidente oder positiv inzidente Kante. Das bedeutet der Ein- oder der Ausgangsgrad des Knotens v ist 0 ($d_i(v) = 0$ oder $d_o(v) = 0$).
2. Knoten mit reflexiven Kanten (engl. *self-loops*). Eine Kante ist reflexiv, wenn der Start- und Endknoten der Selbe ist ($e = (v, v)$).

Können keine Knoten aus dem Graphen entfernt werden, so werden Knoten kontrahiert, um den Graphen zu vereinfachen und das weitere Entfernen von Knoten zu ermöglichen. Die Auswahl der Knoten erfolgt über einen Abgleich der Ein- bzw. Ausgangsgrade. Es wird einer von den Knoten kontrahiert, die die geringsten Ein-

oder Ausgangsgrade besitzen. Die Handlungsvorschrift kann wie folgt aufgeschrieben werden /Mah90/:

1. Ausgangspunkt ist der Signalflussgraph. Der Kontrollwert ES, der den aktuellen Vergleichswert des Aus- bzw. Eingangsgrads enthält wird ES=1 gesetzt.
2. Untersucht wird ein Knoten v:
    a. Wenn $d_i(v) = 0$ oder $d_o(v) = 0$ dann wird der Knoten gelöscht und mit Punkt 3 fortgesetzt.
    b. Wenn v eine reflexive Kante (v, v) besitzt, so wird der Knoten v gelöscht und dem essential set hinzugefügt. Es wird mit Punkt 3 fortgesetzt.
    c. Wenn $d_i(v) = \text{ES}$ oder $d_o(v) = \text{ES}$, dann wird der Knoten v kontrahiert. Falls das nicht zutrifft, wird mit Punkt 4 fortgesetzt.
3. Beende den Algorithmus wenn der Graph leer ist.
4. Beginne Punkt 2 mit einem anderen Knoten, solange bis alle Knoten untersucht wurden.
5. Wenn mindestens ein Knoten gemäß Punkt 2b entfernt wurde, wird ES=1 gesetzt. Andernfalls wird ES = ES+1 gesetzt und erneut mit Punkt 2 fortgesetzt.

Die Knoten werden dabei gemäß ihrer numerischen Bezeichnung nacheinander durchlaufen /Mah90/. In Bild 18 wird dargestellt wie für den Signalflussgraphen aus Bild 16 gemäß der Methode nach Ollero-Amselem das essential set bestimmt wird. Im Signalflussgraphen besitzt jeder Knoten mindestens einen direkten Vorgänger und Nachfolger. Reflexive Kanten sind ebenfalls noch nicht vorhanden. Daher besteht der erste Schritt in der Kontraktion des Knotens $e1$, da dieser einen Ein- als auch einen Ausgangsgrad von eins hat. Da Knoten $e2$ einen Grad von (2,2) hat, wird Knoten $e3$ untersucht und entsprechend kontrahiert. Daraus resultiert in Bild 18c eine reflexive Kante bei Knoten $e2$. Da die Suche entsprechend der numerischen Reihenfolge fortgesetzt wird, ist Knoten $e4$ mit nur einer positiv inzidenten Kante der nächste Kandidat für die Knotenkontraktion. Ebenso werden Knoten $e5$ und $e6$ nacheinander kontrahiert, da der Eingangsgrad beziehungsweise Ausgangsgrad von

beiden Knoten eins beträgt. Als nächstes wird Knoten $e7$ untersucht. Dieser besitzt eine reflexive Kante und wird daher für das essential set gewählt. Es bleibt lediglich Knoten $e2$ der ebenfalls in das essential set kommt. Der Algorithmus terminiert nun, da der Graph leer ist.

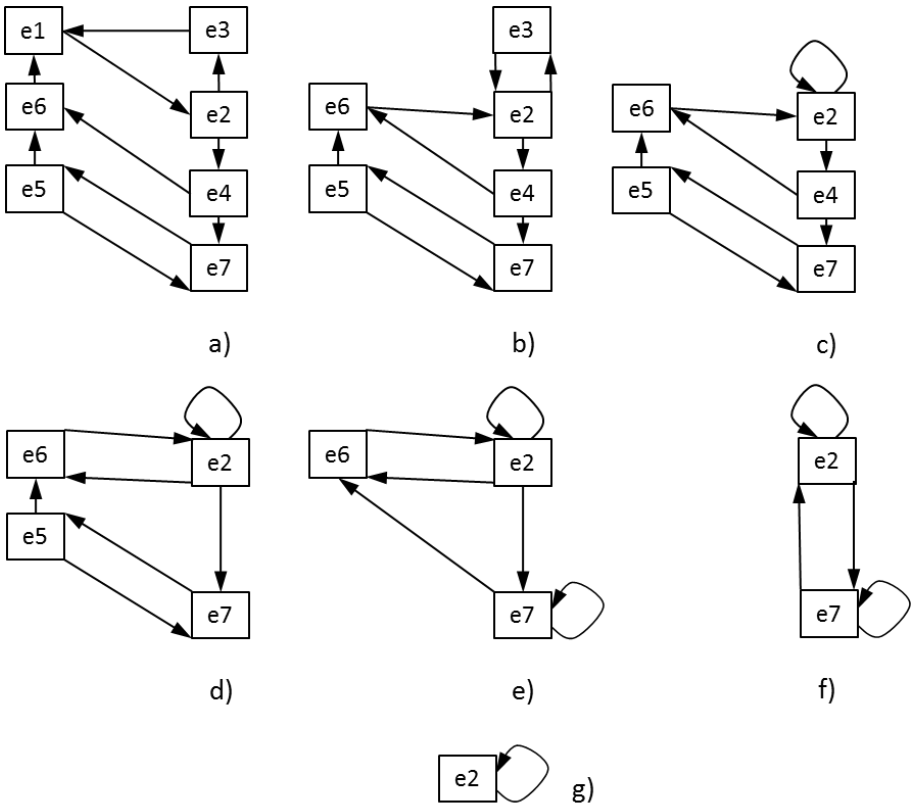


Bild 18 Vereinfachung eines Graphen nach der Methode von Ollero-Amselem zur Bestimmung des essential set

Das essential set besteht nun aus den Knoten $e2$ und $e7$. Es bleibt noch das essential set in ein Tearingset umzuwandeln. Der Knoten des Signalflussgraphen steht für eine Kante im Digraphen des gematchten Systems. Die Kantennummerierung ist in numerischer Ordnung von oben links nach unten recht in der Adjazenzmatrix des Digraphen vorgenommen worden. Wird Kante (2) gesucht, so findet man den zweiten Eintrag in der Adjazenzmatrix an der Position $(2,3)$ und die Kante (7) an der Stelle $(5,4)$. Das bedeutet die Kante (2) verbindet den Knoten $f_2: x_2$ mit dem Knoten $f_3: x_3$ und die Kante (7) den Knoten $f_5: x_5$ mit dem Knoten $f_4: x_4$. Diese Kante soll aufgebrochen werden. Das geschieht, in dem man die zu lösende Variable des ersten Knotens als Tearing-Variable deklariert. In diesem Beispiel sind die Tearing-Variablen x_2 und x_5.

3.7 Tearing-Auswahl nach Steward

Der Tearing-Algorithmus nach Steward ist das älteste der dargestellten Verfahren zur Auswahl einer Tearing-Variable. Er arbeitet, genau wie das Ollero-Amselem-Verfahren mit dem vollständig zugeordneten System. Aus dem bipartiten Graphen, muss mittels der Matching-Information ein Digraph erzeugt werden, dessen Knoten v eine nach der zugeordneten Variablen x umgestellte, explizite Gleichung f verkörpert. Grundgedanke des Steward-Verfahrens ist es, in dem Digraphen alle Schleifen zu bestimmen /Dre70/. Kennt man alle Schleifen und die Gleichungen, durch die sie gebildet werden, so kann auch bestimmt werden, welche Schleife von welcher Gleichung aufgebrochen werden kann. Mit dieser Kenntnis kann der Tearing-Knoten so ausgewählt werden, dass er möglichst viele Schleifen auflöst und das Tearingset so klein wie möglich wird. Anhand des Beispiels aus Bild 14 soll das Steward-Verfahren erläutert werden. Der Digraph und die zugehörige Adjazenzmatrix ergeben sich wie in Bild 15 dargestellt. In der Adjazenzmatrix können nun sämtliche Schleifen gesucht werden, die im gematchten System auftauchen. Dies kann beispielsweise mittels Tiefen- oder Breitensuche geschehen. In Bild 19 ist zum einen der Suchbaum für die Schleifensuche dargestellt, zum anderen die Adjazenzmatrix und der entsprechende Digraph mit eingetragenen Schleifen.

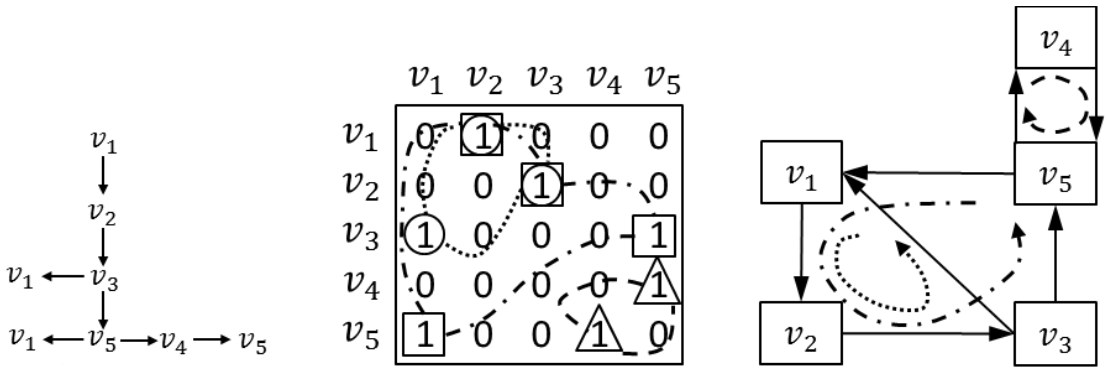


Bild 19 Wegebaum zur Schleifensuche, Adjazenzmatrix und zugehöriger Digraph mit Schleifen

Um eine Schleife aufzuspüren, beginnt man im ersten Knoten und folgt den gerichteten Kanten bis ein Knoten doppelt passiert wird. Die erste Schleife ist in diesem Fall (v_1, v_2, v_3, v_1). Hat man einen Kreis geschlossen, so wird im Suchbaum der nächst frühere Abzweig verfolgt, bis wiederum ein Kreis geschlossen ist. Gibt es keine Abzweige mehr, die untersucht werden könnten, ist die Suche beendet. Im Beispielsystem sind noch zwei weitere Schleifen zu finden, nämlich $(v_1, v_2, v_3, v_5, v_1)$ und (v_5, v_4, v_5). Da nun bekannt ist, welche Knoten in welchen Schleifen enthalten sind, kann ein Knoten gewählt werden. Um die Knoten auszuwählen, die mehr als eine Schleife aufbrechen, wird eine Matrix aufgestellt, die in den Reihen die Schleifen und in den Spalten die vorkommenden Knoten beinhaltet. Diese Matrix wird *loop-occurrence matrix* (engl.: Schleifenmatrix) genannt. Wie in Bild 20a dargestellt enthält sie einen 1-Eintrag, wenn der Knoten in der jeweiligen Schleife auftritt.

	v_1	v_2	v_3	v_4	v_5
A	1	1	1	0	0
B	1	1	1	0	1
C	0	0	0	1	1

a)

	v_1	v_2	v_3	v_4	v_5
A	3	1	2	0	0
B	5	1	2	0	3
C	0	0	0	5	4

b)

Bild 20 Schleifenmatrix (a) und erweiterte Schleifenmatrix (b)

Es ist ersichtlich, wie viele Schleifen aufgebrochen werden, wenn ein Knoten entfernt wird. Allerdings ist es nicht zielführend, einen kompletten Knoten zu löschen, denn in diesem Fall würde die zu lösende Variable und alle zur Gleichung inzidenten Kanten ebenfalls entfernt werden. Für das Tearing ist es nicht erwünscht, einen ganzen Knoten zu entfernen, sondern lediglich eine einzelne Kante aus dem Digraphen. Um das zu realisieren wird die Schleifenmatrix erweitert. Statt 1-Einträge werden die Vorgänger der jeweiligen Knoten entlang des Kreises eingetragen. Da in Schleife A dem Knoten v_1 der Knoten v_3 vorrangeht, steht im Matrixeintrag (A, v_1) eine 3 für v_3. Die vollständig erweiterte Schleifenmatrix ist in Bild 20b dargestellt. Es ist nun die Spalte auszuwählen, die die meisten gleichen nicht-Null-Einträge besitzt.

In diesem Beispiel wären Spalte 2 oder Spalte 3 möglich. Für Spalte 2 bedeutet dies, dass die Variable nach der Gleichung 1 gelöst wird und die notwendig für die Lösung von Gleichung 2 ist, als Tearing-Variable deklariert wird. Damit sind die Schleifen A und B gebrochen. Es bleibt noch Schleife C, die aus zwei Knoten besteht, zu lösen. Hier ist es gleich, ob Knoten 4 oder Knoten 5 gewählt wird. Da hier eine Kante aus dem Digraphen ausgewählt wurde, muss diese noch in die zugehörige Variable im bipartiten Graphen rücktransformiert werden. Die Kante (v_1, v_2) , welche für $(f_1 : x_1, f_2 : x_2)$ steht, ergibt demnach x_1 als Tearing-Variable. Eine gültige Menge an Tearing-Variablen für dieses Bespiel wäre demnach (x_1, x_5).

Im Original arbeitet die Steward-Methode mit Kantentearing. Deshalb werden auch nur einzelne Kanten aus der Schleifenmatrix ausgewählt. Möchte man den Algorithmus für das Knotentearing verwenden, muss der Eintrag in der Schleifenmatrix gesucht werden, der am häufigsten vorkommt. Dadurch werden gleichzeitig alle Kanten mit gleichem Vorgängerknoten gebrochen.

4 Implementierung und Ergebnisse

4.1 Implementierung in den OpenModelica Compiler

Für die automatisierte Bestimmung des Tearingsets wurden alle in Kapitel 3 erläuterten Algorithmen in den Compiler von OpenModelica implementiert. Das sind zum einen die Tearing-Methoden mit vorangehender Variablenzuordnung von Steward und Ollero-Amselem. Diese wurden zuerst mittels des Steward-Matching vom bipartiten Graphen in den gerichteten Graphen überführt. Zum anderen die Methode nach Cellier und die nach Carpanzano, basierend auf der zweiten genannten Auswahlheuristik. Hier wird die Variablenzuordnung mit dem ebenfalls vorgestellten Matching nach Tarjan bestimmt. Um zu untersuchen, welcher Algorithmus welche Tearing-Variablen auswählt, wurden alle Algorithmen auf verschiedene Testbeispiele angewendet. Diese enthalten einerseits kleinere Gleichungssysteme aus der Literatur, die dazu dienen, die Funktionsweise zu überprüfen /Cel06/. Weiterhin wurden Modelle von Pendeln mit unterschiedlicher Körperanzahl untersucht und außerdem Modelle aus der *modelica standard library*, vorrangig Mehrkörpersysteme. Es wurde die Größe des Tearingsets erfasst und die Rechenzeit für die vollständige Zuordnung und die Tearing-Auswahl gemessen.

4.2 Anforderungen an eine Tearing-Methode

Um eine Tearing-Methode qualitativ einzuschätzen, müssen Kriterien für einen guten Tearing-Algorithmus entwickelt werden. Grundsätzlich zeichnet sich ein guter Algorithmus, unabhängig von der Problematik des Tearing durch Robustheit und Schnelligkeit aus. Mit Robustheit ist gemeint, dass er universell ist, also auf möglichst alle denkbaren Systeme reagieren kann. Wie schnell ein Algorithmus ist hängt oft davon ab, wie viele Operationen er ausführt und wie er skaliert. Da es mit deterministischen Algorithmen nicht möglich ist eine Optimallösung in nicht-polynomieller Zeit zu finden, ist es Voraussetzung, dass eine Tearing-Heuristik zumindest in Polynomialzeit eine zulässige Lösung errechnet /Car00/.

Eine gute Tearing-Auswahl ist durch eine minimale Anzahl an Tearing-Variablen gekennzeichnet. Je näher die Größe des heuristisch ermittelten Tearingset an der Größe des minimalen Tearingset liegt, desto effizienter kann die Heuristik eingestuft werden. Dieser Aspekt ist wohlmöglich der entscheidendste bei der Bewertung von Tearing-Algorithmen. Je weniger Tearing-Variablen es gibt, desto weniger Variablen müssen später iterativ ermittelt werden. Das verringert den Rechenaufwand in jedem einzelnen Zeitschritt und ist somit von eminenter Bedeutung. Die Tearing-Auswahl erfolgt einmalig während der symbolischen Programmoptimierung, wohingegen die Anzahl der Tearing-Variablen bei jeder Zeitschrittberechnung ins Gewicht fällt. Zugunsten einer effizienteren Heuristik können auch Abstriche bei der Rechenzeit des Algorithmus gemacht werden. Weiterhin sind in den theoretischen Betrachtungen, die in der Literatur zu finden sind, keine praxisrelevanten Probleme erwähnt, wie beispielsweise die Lösbarkeit. Nur Carpanzano lässt die Lösbarkeit von Gleichungen mit in die Tearingauswahl einfließen. Theoretisch lässt sich jedes eindeutig definierte System mittels Tearing in eine Menge von Zuweisungen in Form expliziter Gleichungen und in eine Menge von Residuengleichungen zur iterativen Lösung der Tearing-Variablen aufteilen. Problematisch wird es, sobald eine Gleichung, die einer Variablen zugeordnet wird, nicht mit den verfügbaren Mitteln umgestellt werden kann oder die Umformung der Gleichung zu ressourcenintensiv wird. Es muss demnach entschieden werden, wie eine Gleichung aussehen darf, damit sie nach einer Variablen umgestellt werden soll. Die Entscheidung welche Termstrukturen als lösbar gelten oder nicht, ist während der Analysephase im Compiler zu klären. Der Tearing-Algorithmus muss mit dieser Information umgehen können und entscheiden wann eine Gleichung nicht nach einer Variablen umgestellt werden kann und wann diese Variable für das Tearing herangezogen werden muss.

4.3 Vergleich der Tearing-Methoden

In Bild 21 ist die erreichte Anzahl an Tearing-Variablen für die vorgestellten Algorithmen über der Anzahl der Gleichungen aufgetragen. Der vollständige Datensatz ist im Anhang zu finden. Jeder Punkt steht für die Menge der errechneten Tearing-Variablen für eine stark zusammenhängende Komponente, welche mit Hilfe der BLT-Transformation ermittelt wurde. Zudem wurde für jeden Algorithmus eine lineare Interpolationskurve eingetragen. Durch diese Trendlinie kann die Effizienz der einzelnen Algorithmen einfacher verglichen werden. Es ist anzumerken, dass nicht alle Algorithmen für jedes Gleichungssystem funktionieren. Das Verfahren nach Steward ist ab einer gewissen Größe des zu untersuchenden Systems zu speicherintensiv. Die Suche nach allen Schleifen in der Adjazenzmatrix des gerichteten Graphen benötigt übermäßig viel Speicher. Diese Tatsache wurde auch schon in der Literatur erwähnt /Dre70/. Der Steward-Algorithmus ist nur für kleine Systeme gut geeignet. Bei mittleren bis großen Problemen benötigt das Verfahren zu viel Speicherplatz und Rechenzeit, um praktikabel zu sein. Der originale Algorithmus nach Cellier gelingt bei fast allen Testbeispielen. Lediglich in zwei der untersuchten Systeme versagt die Variablenzuordnung. Das Matching kann nicht beendet werden, da notwendige Gleichungen bereits als Residuengleichungen deklariert wurden. Nach der Tearing-Wahl wird eine Residuengleichung unter allen adjazenten Gleichungen bestimmt. Diese Auswahl wird in gewissem Maße zufällig getroffen und kann deshalb nicht dem Anspruch einer eindeutigen Zuweisung, die Vorraussetzung für das Tarjan-Matching ist, erfüllen. Das Verfahren wie es von Cellier beschrieben ist (Cel06], erhebt keinen Anspruch auf universelle Anwendbarkeit, sondern dient vorrangig der Veranschaulichung der Funktionsweise eines Tearing-Algorithmus. Unabhängig von der Tatsache, dass es prinzipielle Schwächen bezüglich der Residuenwahl gibt, ist die Heuristik von Cellier in allen untersuchten Anwendungsfällen gültig. Die Methode nach Ollero-Amselem und Carpanzano ist in allen Fällen erfolgreich. Carpanzanos Algorithmus findet jedoch im Verlgeich zum Ollero-Amselem das weitaus kleinere Tearingset.

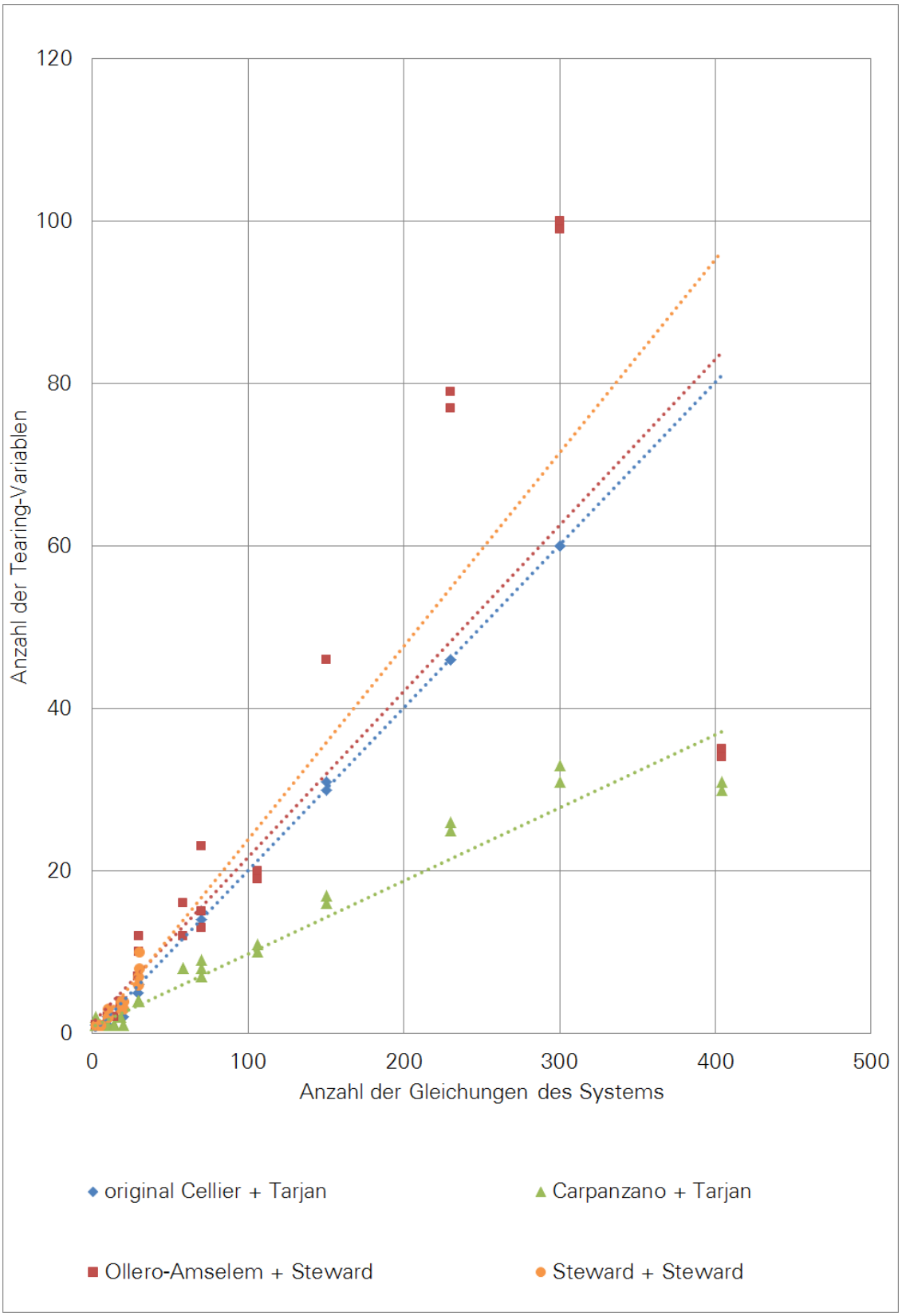


Bild 21 Anzahl der Tearing-Variablen über der Größe des Gleichungssystems für verschiedene Algorithmen, gepunktete Linien zeigen die lineare Näherung

Als weiteres Untersuchungskriterium wurde die Rechenzeit der Tearing-Algorithmen gemessen. Hierbei wurde für die größte stark zusammenhängende Komponente eines Testsystems die mittlere Rechenzeit aus fünf Messungen erfasst. Die Ergebnisse sind in Bild 22 dargestellt.

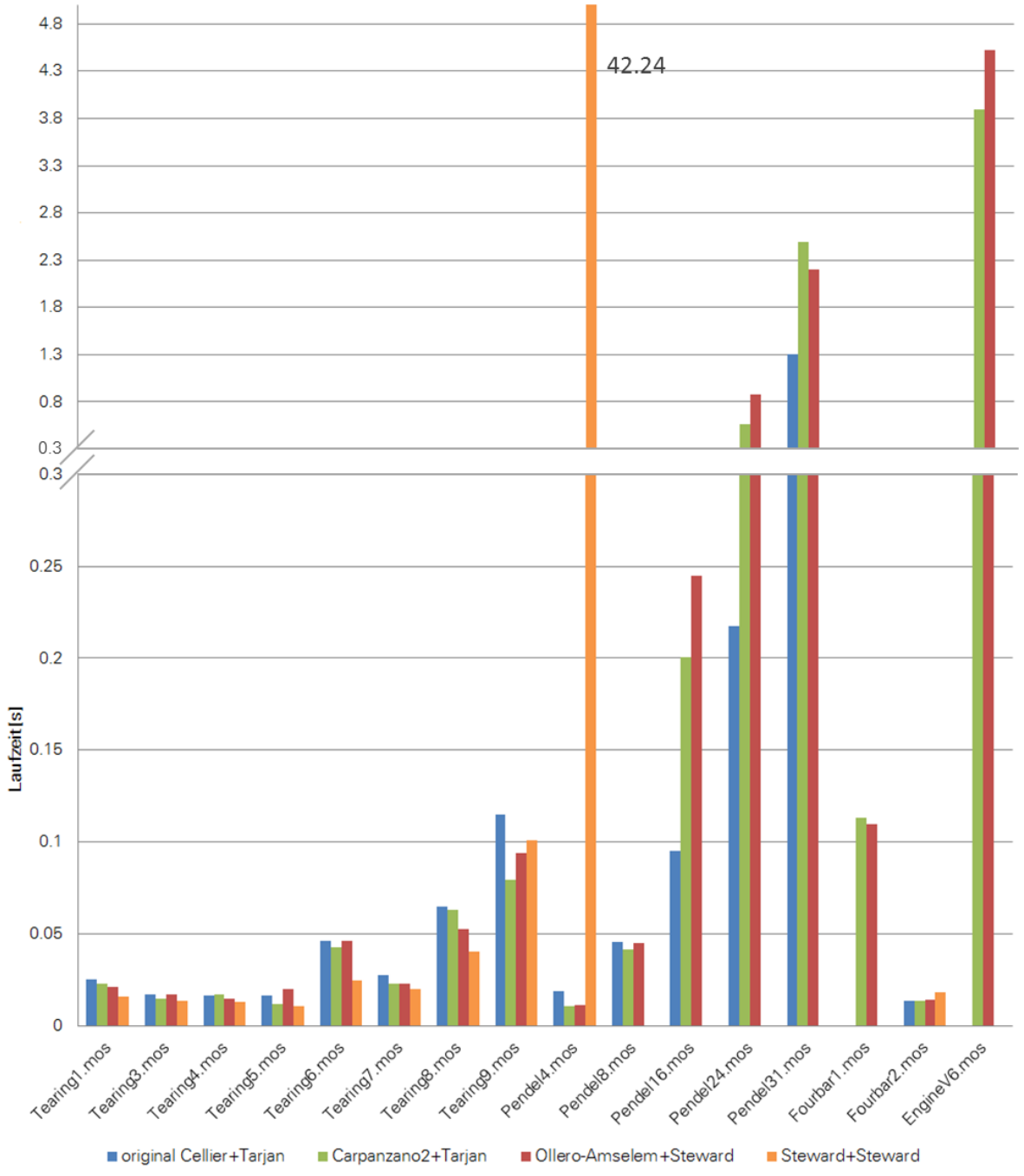


Bild 22 Rechenzeitverhalten der Tearing-Algorithmen für verschiedene Testbeispiele

Es ist festzustellen, dass sich in den Beispielen mit kleineren Gleichungssystemen kaum signifikante Unterschiede in der Rechenzeit feststellen lassen. Je größer die stark zusammenhängende Komponente wird, desto mehr Rechenzeit wird benötigt und desto mehr Unterschiede ergeben sich zwischen den Algorithmen. Das Verfahren nach Steward ist bei kleinen Systemen noch relativ schnell. Bei größeren Systemen als Pendel4 reicht der Speicher nicht aus, um die Schleifenfindung zu beenden. In Pendel4 ist zu sehen, dass die Rechenzeit schon enorm hoch ist und demnach hier die Grenze der Systemgröße erreicht ist. Bei den größeren Beispielen sind Ollero-Amselem und Carpanzano die langsamsten Algorithmen. Zu Carpanzanos Verfahren ist zu sagen, dass die verminderte Geschwindigkeit durch das ständige Abgleichen der erweiterten Adjazenzmatrix während des Matching und der Variablenauswahl bedingt sein kann. Die erweiterte Adjazenzmatrix beinhaltet neben der Strukturinformation des Graphen noch eine weitere Information zur Lösbarkeit jeder Variablen in jeder adjazenten Gleichung. Das Verfahren nach Ollero-Amselem ist aufgrund der aufwändigen Matrizengenerierung recht langsam. Aus dem bipartiten Graphen wird zunächst mittels Zuordnungsinformation der gerichtete Graph aufgebaut. Dieser wird dann in den Signalflussgraphen umgerechnet, auf den dann der eigentliche Prozess der Variablenauswahl angewandt wird. Die ermittelten Knoten müssen zudem noch in die Variablen des Ausgangssystems umgerechnet werden. Diese vielen Schritte ergeben ein relativ rechenintensives Verfahren. Die Methode nach Cellier stellt sich bei den großen Systemen als schnellste Methode heraus. Dies ist damit zu begründen, dass dieses Verfahren lediglich auf dem originalen System des bipartiten Graphen arbeitet. Es wird keine neue Matrix generiert. Der Algorithmus gleicht nur die Strukturinzidenzmatrix ab und kontrolliert auch keine erweiterte Adjazenzmatrix wie bei dem Verfahren nach Carpanzano, was die Rechnung langwieriger macht.

4.4 Auswertung der Ergebnisse

4.4.1 Einfluss der Variablenzuordnung

Die verwendeten Algorithmen wurden in zwei unterschiedliche Klassen unterteilt. Während Cellier und Carpanzano die Tearing-Variablen während der Kausalisierung auswählen, wird bei Ollero-Amselem und Steward das Tearingset nach einem vollständigen Matching ermittelt. Es ist deutlich zu sehen, dass die Algorithmen mit vorangehendem Matching eine größere Anzahl an Tearing-Variablen ermitteln, als die Methoden mit partiellem Matching. Beim Ollero-Amselem- und Steward-Tearing wird der ursprüngliche bipartite Graph entweder durch den Signalflussgraph oder mittels einer Schleifenmatrix ersetzt. Diese zusätzliche Abstrahierung löst das Problem von der eigentlichen Struktur des Gleichungssystems. Die Tearing-Auswahl muss demnach das neugeschaffene System mit einem kompletten Tearingset lösen. Eine kontinuierliche Überprüfung der Kausalisierbarkeit des originalen Systems wie bei den Methoden mit partiellem Matching ist nicht möglich. Dabei wird bei jedem Tearing-Schritt das System weiterkausalisiert und es ist sofort ersichtlich sobald ein vollständiges Tearingset gefunden wurde. Zudem ist anzumerken, dass die Tearing-Auswahl von Ollero-Amselem und Steward davon abhängig sind, wie das vorangehende Matching abläuft. Ist das Gleichungssystem anders sortiert oder wird ein anderer Zuweisungsalgorithmus verwendet, kann das Tearingset komplett anders aussehen und auch eine andere Anzahl von Tearing-Variablen besitzen.

4.4.2 Rechenzeit

Die Rechenzeit ist bei Verfahren die auf partiellem Matching beruhen kürzer als bei Methoden mit vollständiger Variablenzuordnung. Da keine zusätzlichen Matrizen generiert werden, ist auch die Anzahl der nötigen Rechenschritte niedriger. Wird jedoch wie bei Carpanzanos Methode eine ständige Kontrolle der Lösbarkeit mittels erweiterter Adjazenzmatrix getroffen, verlangsamt dies den Algorithmus zusätzlich.

4.4.3 Wahl der Residuengleichung

Weiterhin ist noch zu erwähnen, dass es nicht trivial ist, welche Residuengleichung zu welcher Tearing-Variablen zugewiesen wird. Für Methoden mit vorangehender Variablenzuordnung, ist jeder Variablen von vornherein eine Gleichung zugewiesen. Hierbei ist die Residuengleichung der Tearing-Variablen durch das Matching eindeutig zugeordnet. Für Tearing-Verfahren mit partiellem Matching ist die Frage nach der Wahl Residuengleichung ungeklärt. Es ist in der Literatur nicht explizit erwähnt, ob eine Residuengleichung die Tearing-Variable direkt enthalten muss. Jedoch wird in Celliers Algorithmus die Residuengleichung direkt nach Auswahl der Tearing-Variablen aus der Menge aller adjazenten Gleichungen ausgewählt /Cel06/, /Yan07/. Das ist dahingehend korrekt, da die Residuengleichung während der Newton-Iteration nach der Tearing-Variablen abgeleitet wird. Allerdings ist auch eine indirekte Abhängigkeit der Gleichung von der Tearing-Variablen ausreichend. Da es sich bei den untersuchten Systemen um stark zusammenhängende Komponenten handelt, ist jede Gleichung indirekt von jeder Variablen abhängig und kann durch sie ausgedrückt werden. Die Terme müssen dafür nur ineinander eingesetzt werden. Es ist somit auch möglich, Residuengleichungen als solche zu deklarieren, auch wenn sei keine Tearing-Variable enthalten. Eine Restriktion dahingehend ist also nicht notwendig. Carpanzanos Algorithmus funktioniert nach diesem Prinzip. Die Tearing-Variablen werden ausgewählt, jedoch keiner Gleichung zugewiesen. Der Zuordnungsprozess läuft weiter bis sämtliche Variablen entweder als Tearing-Variable deklariert oder durch eine explizite Gleichung ausgedrückt werden. Nach vollendetem Matching bleiben genauso viele Residuengleichungen, wie Tearing-Variablen übrig. In Bild 23 sind zwei Graphen dargestellt, die die Anzahl der Tearing-Variablen über der Anzahl der Gleichungen in einer stark zusammenhängenden Komponente veranschaulichen. Es handelt sich zum einen um den originalen Cellier-Algorithmus, der die Residuengleichung im Zuge der Tearing-Auswahl bestimmt. Der zweite Datensatz stellt eine modifizierte Version des Cellier-Algorithmus dar. Hierbei wird, genauso wie bei der Methode von Carpanzano, die Residuengleichungen erst nach vollständiger Zuordnung aller Variablen ermittelt. Die übrigen Gleichungen, die nicht Teil des Matching waren, werden den Tearing-Variablen zugewiesen.

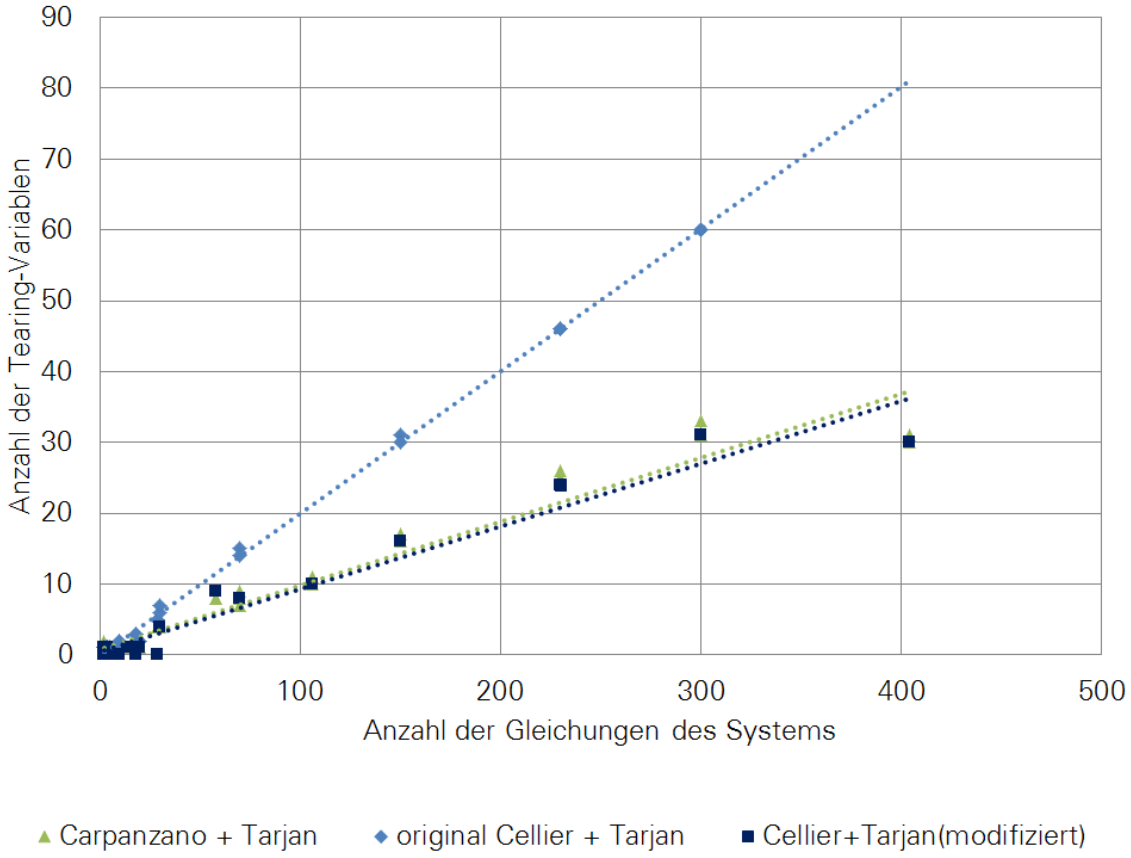


Bild 23 Vergleich unterschiedlicher Methoden zur Residuenbestimmung anhand des Cellier-Algorithmus im Vergleich zum Carpanzano-Algorithmus

Es ist deutlich zu erkennen, dass die modifizierte Cellier-Methode kleinere Tearingsets findet. Durch die Einschränkung der frühzeitigen Gleichungszuweisung, geht die Möglichkeit verloren, die zugewiesene Residuengleichung für ein weiteres Matching zur Verfügung zu halten.

5 Entwicklung einer eigenen Tearing-Heuristik

5.1 Erkenntnisse aus den Ergebnissen

Die vorangehenden Untersuchungen haben Vor- und Nachteile der verschiedenen Tearing-Algorithmen herausgestellt. Zudem wurden diverse Kriterien erarbeitet, die einen guten Tearing-Algorithmus charakterisieren. Anhand dieser erlangten Erkenntnisse kann nun die Struktur und der Aufbau eines optimalen Algorithmus zur Auswahl der Tearing-Variablen eingegrenzt werden.

Es hat sich gezeigt, dass die Methoden mit partiellem Matching besser abschneiden, also ein kleineres Tearingset finden, als die Verfahren mit vorangehendem Matching. Außerdem ergab sich, dass es geschickter ist, die Residuengleichungen nach der kompletten Zuordnung der Variablen zu bestimmen. Dadurch ist auch eine eindeutige Variablenzuordnung möglich und das Matching kann in jedem Fall beendet werden. Der Algorithmus wird dadurch robuster. Ein weiterer eminenter Punkt ist die Beachtung der Lösbarkeit bei der Variablenzuweisung. Ohne diesen Aspekt ist der Algorithmus nicht praxistauglich. Es ergibt sich somit, dass der Algorithmus von Carpanzano die besten strukturellen Voraussetzungen erfüllt. Allerdings wurde die Heuristik zur Auswahl der nächsten Tearing-Variablen an sich noch nicht betrachtet. Die Carpanzano-Methode liefert zwar sehr gute Ergebnisse, jedoch soll eine weitere Heuristik entwickelt werden, die eventuelles Verbesserungspotenzial hinsichtlich der Tearing-Auswahl ausschöpft. Eine weitere Reduzierung der Größe des Tearingsets ist denkbar Die Rechenzeitunterschiede zwischen den einzelnen Verfahren haben keinen ausschlaggebenden Einfluss auf die Entwicklung einer eigenen Methode genommen.

5.2 Grundgedanke der Heuristik

Nach partieller Kausalisierung der stark zusammenhängenden Komponente muss die nächste Tearing-Variable bestimmt werden. Um eine günstige Variable für das Tearing zu wählen, muss geklärt werden, was man mit dieser Variablen im nächsten Schritt erreichen will. Sie sollte nämlich möglichst viele Gleichungen lösen können. Demzufolge sollte die nächste Tearing-Variable bestenfalls folgendes erfüllen:

1. Sie sollte in möglichst vielen Gleichungen vorkommen, um großen Einfluss im Gleichungssystem auszunutzen.
2. Sie sollte, wenn möglich eine weitere Kausalisierung ermöglichen.

Überträgt man diese Anforderungen auf einen Graphen, so ergeben sich folgende Kriterien:

1. Der Variablenknoten ist günstig, der die meisten adjazenten Gleichungsknoten besitzt.
2. Eine Kausalisierung wird ermöglicht, wenn eine Variable von einer Gleichung mit nur zwei adjazenten Variablen gewählt wird. Gibt es keine Gleichung, die nur von zwei Variablen abhängt, so sollte die Gleichung gewählt werden, die die wenigsten adjazenten Variablen besitzt.

Zur Beurteilung wie gut eine Variable die Kriterien erfüllt, wird jede Variable gewichtet. Die Gewichtungsfunktion beruht auf folgenden Überlegungen. Um zu erreichen, dass eine Variable gewählt wird, die in möglichst vielen günstigen Gleichungen vorkommt, muss das Gewicht p_{var} proportional zur Anzahl n_{eq} der adjazenten Gleichungen sein. In den Kriterien wurde außerdem festgestellt, dass Gleichungen mit nur zwei adjazenten Variablen günstiger sind als Gleichungen, die mehr Variablen enthalten. Die Gleichungen werden demnach auch einer Wichtung unterzogen. Je weniger adjazente Variablen eine Gleichung besitzt, desto größer ist die Wichtung p_{eq}. Umso mehr Variablen eine Gleichung enthält, desto kleiner ist deren Gewichtung. Da die Anzahl der adjazenten Variablen n_{var} indirekt proportional zur Wichtung sein soll, muss eine Berechnungsvorschrift für die Gleichungswichtung gefunden werden. Eine Bewertung im Sinne einer Umkehrfunktion

$$p_{eq} = \frac{1}{n_{var}} \quad [5.1]$$

ist ungünstig, da die Gewichtungen zu stark mit der Anzahl abnehmen. Es wird eine Gewichtungsfunktion mit linearem Anstieg bevorzugt. Die Funktion lautet:

$$p_{eq} = n_{ges} - n_{var} \quad [5.2]$$

Wobei n_{ges} die Anzahl aller Gleichungen, also die Größe des Gleichungssystems angibt. Komplett ausformuliert gestaltet sich die entwickelte Heuristik wie folgt:

1. Gib jeder Gleichung eine Gewichtung gemäß Gleichung [5.2].
2. Summiere für jede Variable die Gewichtungen der adjazenten Gleichungen auf. Wähle die Variable als nächste Tearing-Variable, die das größte Gewicht besitzt.

5.3 Bewertung der eigenen Heuristik

In Bild 24 ist ein Diagramm zusehen, in dem die eigens entwickelte Heuristik mit der Methode nach Carpanzano und der von Cellier mit verbesserter Residuenwahl verglichen wird. Dargestellt wird die Größe des ermittelten Tearingsets in Relation zur Größe der betrachteten stark zusammenhängenden Komponente. Zur besseren Vergleichbarkeit enthält das Diagramm interpolierte Trendlinien für die jeweiligen Algorithmen. Carpanzanos und Celliers Methoden liegen dabei sehr nah beieinander. Die Punkte des eigenen Algorithmus liegen jedoch für jedes untersuchte System unterhalb der anderen Punkte. Die Trendlinie zeigt ebenfalls, dass im Durchschnitt ein kleineres System gefunden wird, als mit den anderen beiden Methoden. Besonders zu beachten ist das größte Beispiel mit 404 Gleichungen in dem die erreichte Anzahl von 31 beziehungsweise 30 Tearing-Variablen mit 25 ermittelten Variablen nochmals unterboten wurde.

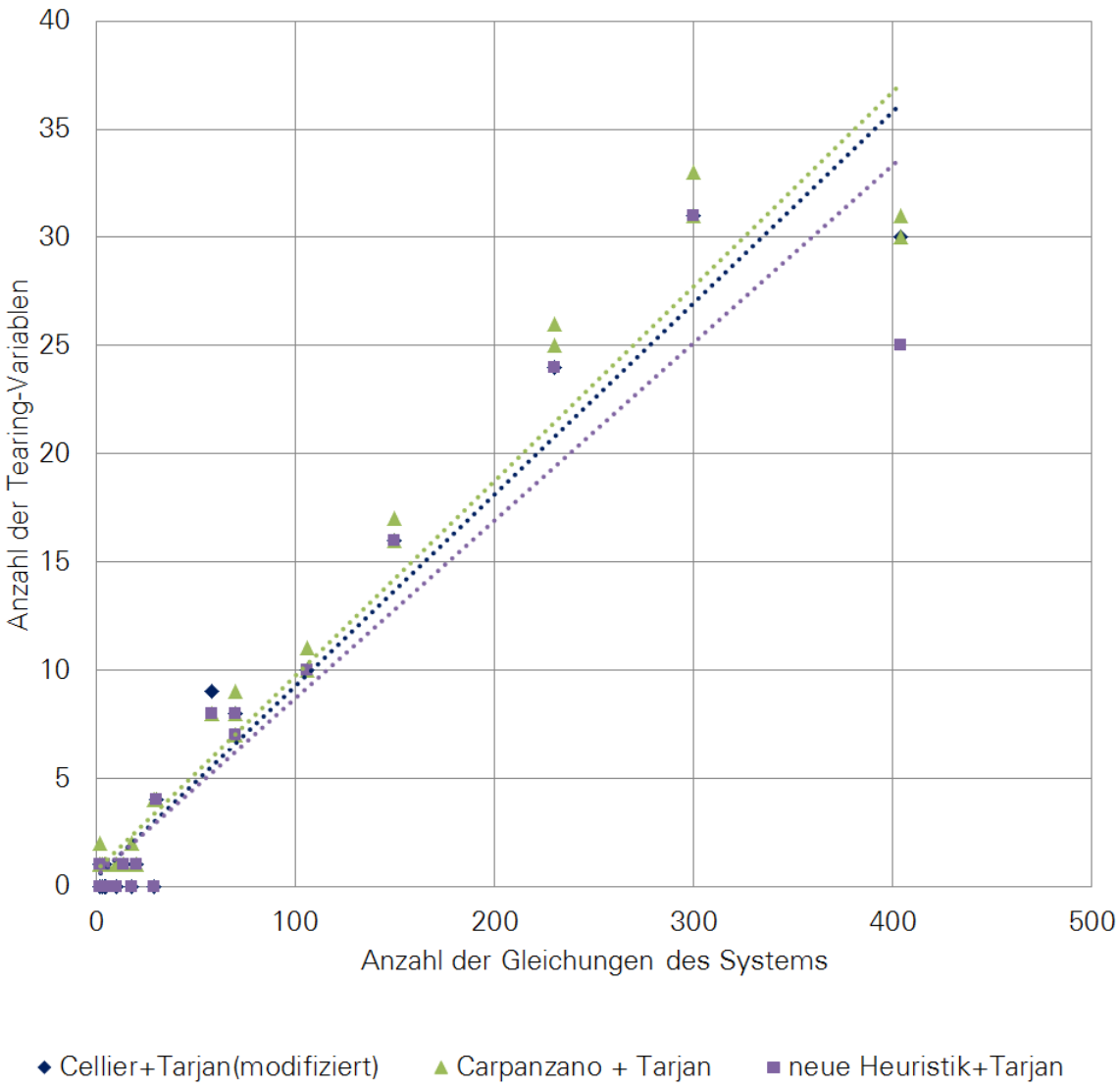


Bild 24 Vergleich der eigenen Heuristik mit modifiziertem Cellier-Verfahren und Carpanzanos Methode

Weiterhin wurde auch das Rechenzeitverhalten der drei Algorithmen untersucht. Bild 25 stellt die nötige Rechenzeit für das Tearing für die größten stark zusammenhängenden Komponenten in jedem Teilsystem dar. Im Allgemeinen ist zu erkennen, dass Carpanzanos-Verfahren das Langsamste ist und das von Cellier am Schnellsten. Da bei der Carpanzano-Methode die erweiterte Adjzazenzmatrix herangezogen wird, um die Gewichtungen für die Variablenauswahl zu ermitteln ist hier erhöhter Rechenaufwand zu verzeichnen. Die erweiterte Adjazenzmatrix wird benötigt um die Lösbarkeit der Gleichungen bei der Zuordnung und Variablenwahl zu berücksichtigen. Da der Cellier-Algorithmus die Lösbarkeit nicht beachtet entsteht

hier ein Rechenzeitvorteil, da keine kontinuierliche Prüfung der erweiterten Adjazenzmatrix notwendig ist. Beim eigenen Algorithmus beläuft sich die globale Wichtung für zunächst alle Gleichungen und danach alle Variablen auf mehr Operationen als die sukzessive Wichtung beim Cellier-Verfahren. Dadurch ist die eigene Heuristik langsamer als Celliers-Methode.

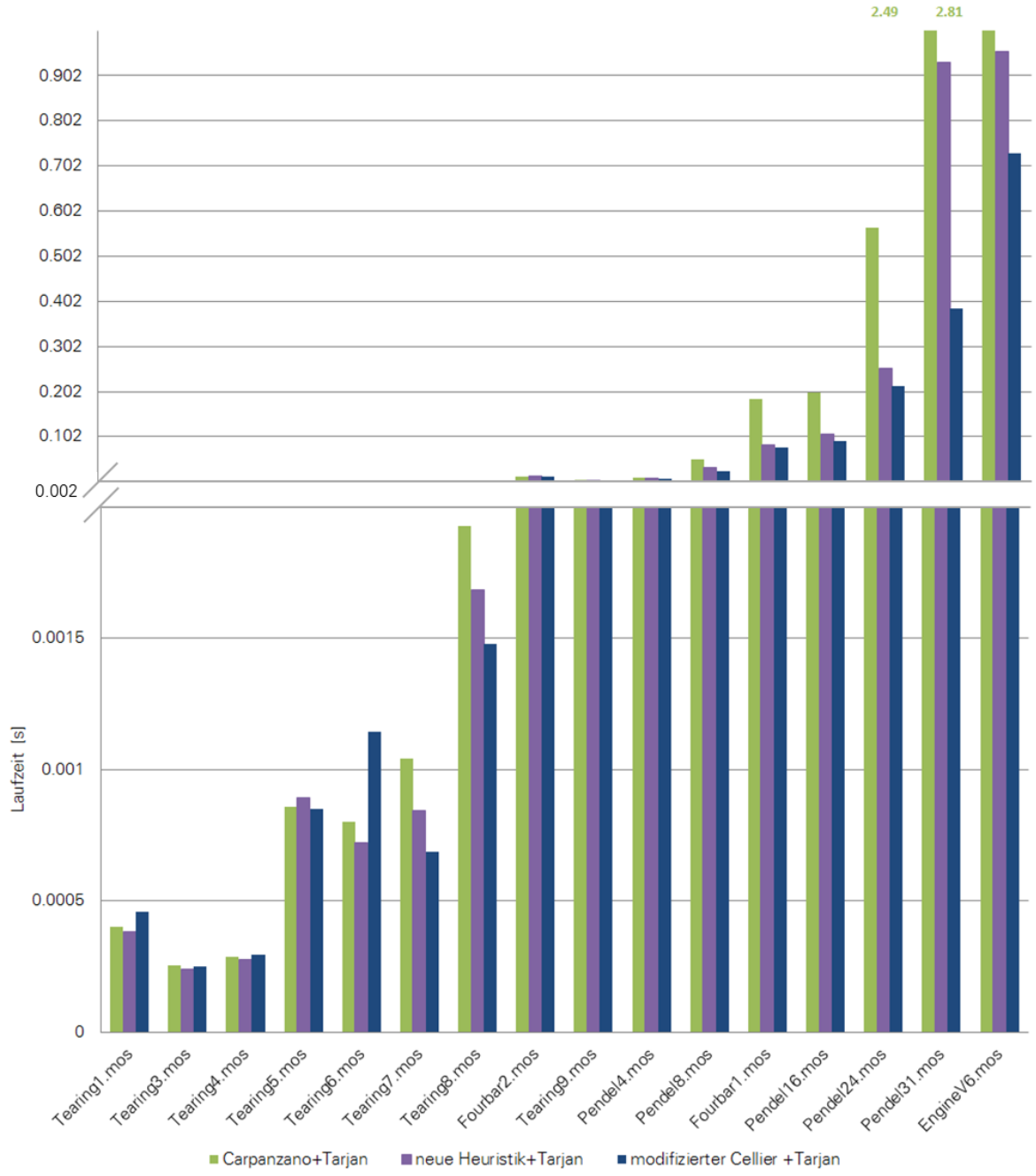


Bild 25 Rechenzeitvergleich für die Tearing-Verfahren von Carpanzano, der eigenen Heuristik und dem modifizierten Cellier-Algorithmus

6 Fazit

6.1 Zusammenfassung der Arbeit

Für eine effizientere Berechnung großer DAE-Systeme wurde die Tearing-Methode näher untersucht. Dabei werden algebraische Schleifen innerhalb einer stark zusammenhängenden Komponente aufgebrochen, um die Anzahl der iterativ bestimmten Variablen zu reduzieren. Anstatt die gesamte stark zusammenhängende Komponente auf einmal zu lösen, müssen nach Anwendung des Tearing nur noch einzelne Residuengleichungen iterativ nach der Tearing Variablen gelöst werden. Die Kernfrage beim Tearing ist, welche Variablen ausgewählt werden müssen, um das System vollständig zu kausalisieren. Die Anzahl der gewählten Variablen soll dabei minimal sein. Eine eindeutige, optimale Lösung über einen deterministischen Algorithmus ist nachweislich nicht in polynomieller Zeit ermittelbar. Deswegen werden Heuristiken herangezogen, welche die Optimallösung möglichst gut annähern sollen. Mit der Auswahl der Tearing-Variablen geht auch stets die Zuordnung der Variablen einher. Dabei wird jede Gleichung einer zu lösenden Variablen zugewiesen. Die impliziten Gleichungen werden in die explizite Form überführt. Dahingehend können Tearing-Verfahren in zwei prinzipielle Gruppen eingeteilt werden, nämlich Tearing-Verfahren mit vorangehender und mit partieller Variablenzuordnung. Bei der vorangehenden Zuordnung wird vor der eigentlichen Tearing-Auswahl das Gleichungssystem vollständig kausalisiert. Das geschieht mit dem Matching-Algorithmus nach Steward. Dabei wird eine eindeutige Zuweisung zwischen Gleichung und Variable für das gesamte System getroffen. Anhand dieser Information kann der bipartite Graph in einen Digraphen überführt werden. Mit dem gerichteten Graphen lassen sich der Informationsfluss im System und somit auch die auftretenden Schleifen darstellen. Die Verfahren von Ollero-Amselem und Steward arbeiten auf diesem Prinzip. Beim Algorithmus nach Ollero-Amselem wird die Dimension des Graphen schrittweise vereinfacht, bis der Graph leer ist. Die Schleifen werden dabei soweit reduziert, bis sie nur noch aus einem Knoten bestehen, der für das Tearing herangezogen wird. Der Steward-Algorithmus hingegen sammelt mittels eines Suchalgorithmus alle im System vorkommenden

Schleifen in einer Schleifenmatrix. Darin werden die Knoten ausgewählt, die die meisten Schleifen aufbrechen. Andere Algorithmen wie beispielsweise von Cellier und Carpanzano funktionieren mit einem partiellen Matching. Dabei wird das Gleichungssystem abwechselnd kausalisiert oder es wird eine Tearing-Variable bestimmt. Somit ist die Zuordnung eindeutig und es erfolgt eine sofortige Kontrolle über die Wirkung jeder einzelnen Tearing-Variablen. Die teilweise Kausalisierung erfolgt mit Hilfe des Tarjan-Algorithmus. Die Auswahl der Tearing-Variablen wird mit verschiedenen Heuristiken getroffen. Prinzipieller Unterschied liegt jedoch in der Wahl der zugehörigen Residuengleichungen. Celliers Methode trifft die Wahl sofort nach der Bestimmung jeder einzelnen Tearing-Variablen, wohingegen Carpanzano erst nach vollständiger Kausalisierung die übrigen Gleichung für die Berechnung der Residuen heranzieht. Außerdem ist anzumerken, dass Carpanzanos Methode, die einzige ist, die die Lösbarkeit einer Gleichung nach einer Variablen mit in Betracht zieht. Bezüglich der Effizienz der Algorithmen lässt sich sagen, dass Verfahren mit partiellem Matching ein kleineres Tearingset finden, als Verfahren mit vollständigem Matching. Der Algorithmus von Steward ist zudem noch so ressourcenintensiv, dass eine robuste Lösung nur für kleine Systeme garantiert werden kann. Ollero-Amselems Verfahren hingegen ist sehr robust und findet stets ein valide Menge an Tearing-Variablen. Carpanzanos Heuristik bestimmt immer das kleinste, aller ermittelten Tearingsets. Celliers Methode ist aufgrund der Residuenwahl nicht für alle Testbeispiele anwendbar. Wird die Bestimmung der Residuen so umgestellt, dass erst nach vollständiger Kausalisierung eine Wahl getroffen wird, so ist die Menge an ermittelten Tearing-Variablen vergleichbar zu Carpanzanos Algorithmus. Anhand der vorliegenden Ergebnisse wird festgelegt, dass ein gutes Tearing-Verfahren auf partiellem Matching und einer Residuenwahl nach erfolgreicher Kausalisierung beruht. Zudem muss eine Beachtung der Lösbarkeit mit einfließen. Da noch Verbesserungspotenzial in der Auswahlheuristik bestand, wurde ein eigener Algorithmus entwickelt. Dieser ist nach den zuvor erwähnten Kriterien konstruiert. Die neue Heuristik brachte verglichen zu Carpanzanos Methode eine weitere Verbesserung bezüglich der ermittelten Anzahl an Tearing-Variablen.

6.2 Ausblick

Die ermittelten Kriterien für einen Tearing-Algorithmus haben die strukturellen Möglichkeiten bei der Entwicklung eines optimalen Tearing-Verfahrens eingegrenzt. Natürlich ist nicht ausgeschlossen, dass auch mit einer Methode basierend auf vollständigem Matching ein minimales Tearingset gefunden werden kann. Hinsichtlich der Berücksichtigung der Lösbarkeit sind diese Verfahren, wie sie hier betrachtet wurden jedoch nicht praxistauglich. Der eigens entwickelte Algorithmus und der von Carpanzano ziehen die Lösbarkeit in Betracht. Beide funktionieren unabhängig davon, was als lösbar definiert ist. Im vorliegenden Fall sind alle analytisch lösbaren Gleichungen als lösbar definiert. Dahingehen gibt es noch Potenzial für weitere Analysen Es ist zu untersuchen, wie sich ein Algorithmus verhält, wenn mehr oder weniger der vorhandenen Gleichungen als unlösbar gekennzeichnet sind. Wie sieht das Ergebnis aus, wenn besonders viele Variablen als unlösbar deklariert sind und wie kann eine günstige Variablenzuordnung getroffen werden, um das Tearingset klein zu halten?

Da sich in dieser Arbeit lediglich auf die Methode des Tearing konzentriert wurde, könnten zukünftige Untersuchungen sich mit anderen Verfahren zur Behandlung dünnbesetzter Matrizen beschäftigen. Es existieren weitere symbolische und numerische Methoden, die anstelle des Tearing angewandt werden können. Auf diesem Gebiet ist noch Potenzial für weitere Forschung vorhanden.

Quellenverzeichnis

/Bär07/ **Bärwolff, Günter. 2007.** *Numerik für Ingenieure, Physiker und Informatiker.* München: Elsevier GmbH, 2007

/Bon08/ **Bondy, J.A. und U.S.R., Murty. 2008.** *Graph Theory.* USA : Springer, 2008.

/Car00/ **Carpanzano, Emanuele. 2000.** *Order Reduction of General Nonlinear DAE Systems by Automatic Tearing.* London: Taylor & Francis, 2000.

/Cel06/ **Cellier, E. François und Kofman, Ernesto. 2006.** *Continous System Simulation.* New York: Springer Science+Business Media Inc., 2006.

/Deu04/ **Deuflhard, Peter. 2004.** *Newton methods for Nonlinear Problems, Affine Invariance and Adaptive Algorithms.* Berlin: Springer, 2004.

/Dra07/ **Drábek, Pavel und Milota, Jaroslav. 2007.** *Methods of Nonlinear Analysis, Applications to Differential Equations.* Basel: Birkhäuser Verlag AG, 2007.

/Dre70/ **Drew, Thomas B., et al. 1970.** Advances in Chemical Engineering. London: Academic Press, 1970. Bd. Volume 8.

/Elm94/ **Elmqvist, Hilding und Otter, Martin. ca. 1994.** Methods for Tearing Systems of Equations. ca. 1994

/Fun98/ **Funke, Meinrad. 1998.** *Allgemeine Feedback Vertex Set Probleme.* München: Herbert Utz Verlag, 1998.

/Kan05/ **Kanzow, Christian. 2005.** *Numerik linearer Gleichungssysteme - Direkte und iterative Verfahren.* Heidelberg: Springer, 2005.

/Mah90/ **Mah, Richard S. 1990.** *Chemical Structures and Information Flow.* Stoneham: Butterworth Publishers, 1990.

/Tit11/ **Tittmann, Peter. 2011.** Graphentheorie – Eine anwendungsorientierte Einführung. München: Carl Hanser Verlag, 2011

/Yan07/ **Yan, Yuhan. 2007.** *Bewertung und Analyse von Tearing-Algorithmen.* Paderborn: Universität Paderborn, 2007.