Constraint-Programmierung
Grundlagen und Anwendungen

Thom Frühwirth und Slim Abdennadher
Institut für Informatik
Ludwig-Maximilians-Universität München

Die ersten Seiten aus dem Lehrbuchs im Springer Verlag
165 S. 34 Abb., Brosch. ISBN 3-540-60670-X, DM 32,-, September 1997

Vorwort

Was?

Die Constraint-Programmierung ist eine der spannendsten Entwicklungen in der Anwendung von Computern in den letzten zehn Jahren. Dieses junge Gebiet hat von Anfang an in Forschung und Praxis gleichermaßen für Aktivität gesorgt. Kein Wunder, daß sich damit Forscher begeistern und ebenso Millionen verdienen lassen – handelt es sich bei der Constrainttechnologie doch um eine allgemeine Methode für elegantes, effizientes, deklaratives Problemlösen mit höheren Programmiersprachen. Lesen Sie das Einführungskapitel dieses Buches, und Sie werden merken, wie allgegenwärtig Constraints (Bedingungen) sind – von Fahrradschlössern über Versicherungsbestimmungen bis zur Mathematik.

Das:

Es galt, Pionierarbeit zu leisten: Dieses Lehrbuch ist die erste umfassende und einheitliche Einführung in die Constraint-Programmierung weltweit. Es bietet eine kurze und prägnante aktuelle Darstellung der wesentlichen Aspekte und Entwicklungen in der constraintbasierten logikorientierten Programmierung, von der theoretisch fundierten Beschreibung der unterschiedlichen Klassen von Programmiersprachen zu Constraintsystemen und Constraintlösern bis zu konkreten Anwendungsbeispielen aus der Praxis.

Wir haben versucht, unnötigen Formalismus und Anglizismen zu vermeiden, ohne ungenau zu werden. Sie finden in diesem Buch illustrative Beispiele, viele tabellarische Zusammenfassungen, Abbildungen und einen informativen Anhang mit Übungsaufgaben mit ausgewählten Lösungsvorschlägen sowie weiterführender Literatur.

Für:

Dieses Buch wendet sich vor allem an Dozenten und Studierende mit Vorwissen in Logik bzw. Programmierung in Prolog. Es ist darüber hinaus für alle geeignet, die an Themen aus den Gebieten Künstliche Intelligenz, Programmierung und Problemlösen interessiert sind, insbesondere an der gegenseitigen Befruchtung von Theorie und Praxis. Dem innovativen Praktiker sei das letzte Kapitel empfohlen.

Dank!
Unser Dank gilt Thoms Frau Andrea und seiner Tochter Anna sowie Slims Frau Nabila für ihre Geduld und ihren Beistand; François Bry und Martin Wirsing, die unsere Beschäftigung mit dem Thema gefördert haben; unseren Kollegen, vor allem Norbert Eisinger, Tim Geisler, Alexander Knapp, Holger Meuss und Heribert Schütz, die mit Ihren Kommentaren und Korrekturhinweisen zum Buch beigetragen haben; den Studenten der Vorlesungen an der LMU München und den Teilnehmern an den Tutorien, auf denen das Buch zum Teil basiert, und Hans Wössner sowie Ruth Abraham vom Springer-Verlag für ihre Unterstützung.

Wir wünschen viel Spaß mit Constraint-Programmierung!

München, im Juli 1997 Thom Frühwirth und Slim Abdennadher

PS: Wie lange wird dieses Lehrbuch Gültigkeit haben, wenn die Entwicklung in diesem Gebiet so schnell verläuft? Wir haben vorgesorgt: Online im Internet, auf den Webseiten des Springer-Verlages, `http://www.springer.de/`, finden Sie laufend weitere Informationen, ergänzendes Lehrmaterial (z.B. Overheadfolien) sowie Software zum Ausprobieren und Herunterladen.

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Was sind Constraints? 1
1.2 Constraint-Programmierung 3
1.3 Inhaltsübersicht 4

2 Prädikatenlogik und Kalküle **7**

2.1 Syntax 7
2.2 Semantik 12
2.3 Logische Kalküle 14

3 Logikprogrammierung **17**

3.1 LP-Kalkül 19
3.2 Deklarative Semantik 24
3.3 Negation durch Scheitern 29
3.4 Prolog 30

4 Constraint-Logikprogrammierung **35**

4.1 Constraintsysteme 38
4.2 Constraintlöser 41
4.3 CLP-Kalkül 43
4.4 Deklarative Semantik 49

5 Constrainterweiterungen **53**

5.1 Implikation 54
5.2 Disjunktion 58
5.3 Kardinalität 60
5.4 Meta-Constraint 62

6 Nebenläufige CL-Programmierung **65**

6.1 NCLP-Kalkül 67
6.2 Deklarative Semantik 74

7 Constraint Handling Rules **77**
7.1 CHR-Kalkül . . . 79
7.2 Deklarative Semantik . . . 84

8 Constraintsysteme **89**
8.1 Terme T . . . 90
Anwendung: Typüberprüfung . . . 94
8.2 Boolesche Algebra B . . . 95
Anwendung: Digitale Schaltungen . . . 99
8.3 Endliche Bereiche FD . . . 100
Anwendung: Zeitplanung . . . 108
8.4 Lineare Gleichungssysteme R . . . 110
Anwendung: Finanzwesen . . . 114
8.5 Nichtlineare Gleichungssysteme I . . . 115
Anwendung: Mietspiegel . . . 118

9 Anwendungen **119**
9.1 Marktüberblick . . . 119
9.2 Der Münchner Mietspiegel Online . . . 123
9.3 Planung drahtloser Systeme . . . 129

A Übungsaufgaben und Lösungsvorschläge **139**
A.1 Übungsaufgaben Logikprogrammierung . . . 139
A.2 Übungsaufgaben Constrainterweiterungen . . . 141
A.3 Übungsaufgaben Constraintsysteme . . . 142
A.4 Lösungsvorschläge . . . 149

Kapitel 1

Einleitung

1.1 Was sind Constraints?

Das englische Wort „Constraint“ bedeutet Einschränkung oder (Wert-, Rand-, Neben-)Bedingung. Constraints[1] eignen sich zur Darstellung unvollständiger Information, also zur Beschreibung der Eigenschaften und Beziehungen von teilweise unbekannten Objekten. Als recht allgemeiner und abstrakter Begriff haben Constraints die verschiedensten Ausprägungen und Arten. (Doch haben sie alle wichtige Gemeinsamkeiten, wie wir sehen werden.)

Ein Beispiel: Ein Mathematiker hat ein Fahrrad mit Zahlenschloß. Er kann sich nicht mehr an die letzte Ziffer der Zahlenkombination erinnern. Er weiß nur noch: Sie ist ungerade, natürlich eine einstellige Zahl, außerdem keine Primzahl und nicht 1. Indem er die unvollständigen Informationen über die Zahl kombiniert, kann er die gesuchte Zahl, nämlich 9, ermitteln. Dabei sind *ungerade, einstellig, keine Primzahl* und *nicht 1* die Constraints, die die Zahl beschreiben. Man beachte, daß das Constraint *ungerade* für sich allein eine unendliche Menge von Lösungen besitzt. Im allgemeinen reichen Constraints allein nicht aus, um ein Problem ganz zu lösen. Man muß zwischendurch immer wieder suchen. Würde uns bei diesem Beispiel die letzte Information fehlen, so wären wir darauf angewiesen, die Zahlen 1, 3, 5, 7 und 9 auszuprobieren.

Auch die Bestimmungen der Gepäckversicherung, die festlegen, wann der Versicherungsfall vorliegt, sind Constraints: Danach hat man seine Handtasche ständig im Auge zu behalten, sein Gepäck im verschlossenen Kofferraum des Autos zu lagern und das Auto nachts in einer abgeschlossenen Garage zu parken, die nur einem eng begrenzten Kreis von Benutzern zugänglich ist.

Ein zeitliches Constraint ist z.B. die Tatsache, daß die Vorlesung „Constraint-Programmierung“ dienstags von 14 bis 16 Uhr stattfindet. Ein räumliches Constraint ist die Nummer des Raumes, in dem diese Vorlesung stattfindet, nämlich 114.

[1] Dies ist einer der wenigen Fachausdrücke, die wir nicht übersetzen.

Arithmetische Constraints lassen sich formal einfach darstellen:

$$X + Y = 7$$

ist ein Constraint, das eine Aussage über die Beziehung der Objekte (in diesem Fall Variablen) X und Y, und damit über deren mögliche Werte, macht.

Charakteristisch für den constraintbasierten Ansatz ist das Lösen von Problemen, indem man solche Constraints angibt, die von einer Lösung erfüllt werden müssen. Dann lösen wir Constraints, damit das Zahlenschloß sich öffnet, damit der Versicherungsschutz gewahrt bleibt, wir rechtzeitig zur Vorlesung kommen und der Wert der Variablen in einem Gleichungssystem bekannt wird. Dabei können zwar spezielle Algorithmen zum Einsatz kommen, sie müssen aber gewissen allgemeinen Prinzipien gehorchen.

Damit „Constraintlösen" in Computerprogramme integriert werden kann und damit Constraints und ihre Lösungen einen Einfluß auf den Ablauf von Programmen haben können, muß es ein Programmstück geben, das die Constraints verwaltet und löst. Das ist der Constraintlöser.

Bei arithmetischen linearen Constraints könnte der Constraintlöser das Gauß'sche Eliminationsverfahren anwenden, um z.B. folgende Constraints zu lösen:

$$X + Y = 7, \quad X - Y = 3$$

Man möchte, daß ein Constraintlöser die Gleichungen möglichst so weit vereinfacht, daß die Wertebelegungen der Variablen explizit werden:

$$X = 5, \quad Y = 2$$

1.2 Constraint-Programmierung

Die Familie der „Constraint-Logikprogrammiersprachen" entstand Mitte der achtziger Jahre als eine natürliche Fusion zweier deklarativer Paradigmen, nämlich Constraintlösen und Logikprogrammierung.

Die Idee der Logikprogrammierung ist, Probleme logisch zu beschreiben. In diesen Programmiersprachen (z.B. Prolog) werden das zum Problem gehörige allgemeine Wissen und die konkreten Annahmen durch Regeln und Fakten, eine eingeschränkte Klasse von logischen Formeln, ausgedrückt. Eine Lösung wird durch Anwendung passender Regeln auf eine Anfrage auf der Basis der Fakten gesucht. Wegen ihrer abstrakten deklarativen Natur eignen sich Logikprogrammiersprachen gut für die schnelle Erst- und Weiterentwicklung von Prototypen auf der Basis unvollständiger Spezifikationen (engl. rapid prototyping). Mit der Einbettung von Constraints in Logikprogrammiersprachen wurde es möglich, schnell und elegant komplexe Probleme durch eine Verbindung aus Constraintlösen und Suche zu lösen.

Constraint-Programmierung kann vorteilhaft eingesetzt werden zum Schließen mit sowohl unvollständiger, ungenauer bzw. unsicherer als auch vollständiger Information (z.B. Finanzanalyse) und zum Lösen kombinatorischer Probleme (z.B. Zeitplanung) in Entscheidungsunterstützungssystemen (auch: Expertensysteme, intelligente Agenten). Seit Anfang der neunziger Jahre wird Constraint-Programmierung mit großem Erfolg von mehreren Firmen weltweit kommerziell eingesetzt, ihr gemeinsamer Umsatz mit Constrainttechnologie wurde 1996 auf 100 Millionen US-Dollar geschätzt.[2]

Das System Daysy zum Beispiel adaptiert für die Lufthansa den Einsatz von Personal nach Störungen im Flugbetrieb (Verspätungen, Erkrankung usw.), so daß die Änderungen im Personalplan und die Kosten minimiert werden. Nokia Mobile Phones, der zweitgrößte Mobiltelefonhersteller der Welt, verwendet Constraints zur automatischen Konfiguration von Software für Mobiltelefone. Renault setzt Constraint-Programmierung seit 1995 zur Optimierung der Zulieferung und Fertigung von Varianten eines Autotyps ein.

1.3 Inhaltsübersicht

Dieses Lehrbuch gibt einen einführenden Überblick über die Grundlagen der Constraint-Programmierung und die verschiedenen Constraintsysteme, ihre Entwicklungsgeschichte und ihre Anwendungsmöglichkeiten. Während sich der erste Teil des Buches mit Klassen von Constraint-Programmiersprachen beschäftigt, stellt der zweite Teil Arten von Constraints und ihre Anwendungen vor.

Im nächsten Kapitel werden die im Buch verwendeten, grundlegenden logischen Formalismen kurz beschrieben: Syntax und Semantik der Prädikatenlogik erster Stufe sowie Kalküle als Zustandsübergangssysteme mit Kongruenzen.

In Kapitel 3, Logikprogrammierung, geht es um das Programmieren in einer ausführbaren Untermenge der Prädikatenlogik. Wir definieren ihre operationale (prozedurale) Semantik in Form eines Kalküls, ihre deklarative Semantik durch Prädikatenlogik und erklären den formalen Zusammenhang zwischen operationaler und deklarativer Semantik. Mit Prolog stellen wir schließlich kurz den bekanntesten Vertreter und Klassiker dieser Familie von Programmiersprachen vor.

Diese beiden Grundlagenkapitel können keinen Kurs in Logik bzw. Prolog ersetzen.

Ausgehend von der Logikprogrammierung erweitern wir in den nächsten Kapiteln diese Klasse von Programmiersprachen Schritt für Schritt. Die Beschreibungsweise für eine Programmiersprache — Kalkül, deklarative Semantik, Zusammenhänge — sowie einige Beispiele behalten wir auch in den folgenden Kapiteln bei.

[2]Ein Vergleich: Der Umsatz mit Data Mining betrug 1996 ca. 120 Mill. Dollar, der von Microsoft fast 10 Milliarden Dollar.

In Kapitel 4, Constraint-Logikprogrammierung, erweitern wir die Logikprogrammierung um Constraints, die wir als spezielle logische Prädikate auffassen. Wir definieren in diesem Kapitel auch, was ein Constraintsystem ist und was ein Constraintlöser macht.

In Kapitel 5, Constrainterweiterungen, behandeln wir Erweiterungen der Ausruckskraft und Kombinierbarkeit von Constraints. Dieses Kapitel kann beim ersten Lesen ausgelassen werden.

In Kapitel 6, Nebenläufige Constraint-Logikprogrammierung, stellen sich Constraints als mächtige Grundlage für die Kommunikation und Synchronisation von nebenläufigen Prozessen dar.

In Kapitel 7 lernen wir mit Constraint Handling Rules eine nebenläufige Programmiersprache zum Schreiben von Constraintlösern kennen.

Wenn wir in Kapitel 8 die gängigen Constraintsysteme, d.h. Arten von Constraints, vorstellen, werden Constraint Handling Rules nützlich sein, um deklarativ und prägnant die zugehörigen Constraintlöser sowohl prinzipiell zu beschreiben als auch zu implementieren. Für jedes Constraintsystem geben wir auch ein größeres Beispiel an, das sein typisches praktisches Anwendungsgebiet illustriert.

In Kapitel 9, Anwendungen, kommen wir zur kommerziellen Praxis der Constraint-Programmierung: Wir beschreiben kurz den Markt für diese Technologie, die beteiligten Softwarefirmen, Klassen von Anwendungen und konkrete Projekte für namhafte Firmenkunden aus aller Welt. Dann stellen wir zwei examplarische Anwendungsstudien für die Constrainttechnologie näher vor, die sich nichtsdestoweniger durch ihren innovativen Charakter auszeichnen. Der Anfang dieses Kapitels kann auch als erstes gelesen werden.

Im Anhang finden sich Übungsaufgaben zu den einzelnen Abschnitten und ausgewählte Lösungsvorschläge. Hinweise auf weiterführende Literatur und ein ausführliches Stichwortverzeichnis schließen das Lehrbuch ab.

Als Einführung und erster Überblick kann dieses Lehrbuch, auch aus Platz- und Aktualitätsgründen, auf viele Themen nicht eingehen: Auf Implementierungsmethoden und Programmiermethodologie, auf Optimierungsverfahren, dynamische Constraints, die zur Laufzeit zurückgenommen werden können, Prioritäten zwischen Constraints, nicht-logische Programmiersprachen mit Constraints (funktionale, objektorientierte, imperative), und Datenbanken mit Constraints. Hier helfen die Hinweise auf weiterführende Literatur.