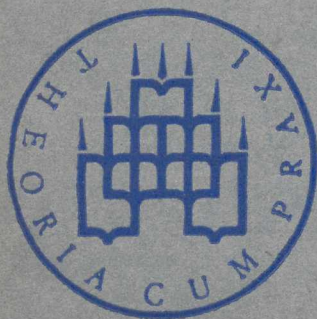


Rostocker
Mathematisches Kolloquium

Heft: 5



WILHELM-PIECK-
UNIVERSITÄT ROSTOCK
1977

Rostocker Mathematisches Kolloquium

Heft 5

**Tagung Digitalgraphik
in Rostock-Warnemünde
vom 30. 10. bis 5. 11. 1976**

Übersichtsvorträge

**1977
Wilhelm-Pieck-Universität Rostock
Sektion Mathematik**

**Redaktion: Abt. Wissenschaftspublizistik der Wilhelm-Pieck-Universität
Rostock**

DDR - 25 Rostock, Vogelsang 13/14

Fernruf 369 577

Verantw. Redakteur: Dipl.-Ges.-Wiss. Bruno Schrage

Herausgegeben von der Wilhelm-Pieck-Universität Rostock

Druck: IZ-Dok-Schiffbau, 117 Berlin

Druckgen. Nr.: BG 057/33/77 (204)

Vorwort	5
KIESEWETTER, Helmut: Zu einigen Problemen bei der Entwicklung und Anwendung von digitalgraphischen Programmsystemen	7
KOTZAUER, Adolf: Programmiersprachliche Aspekte in der Digitalgraphik und Probleme bei ihrer Realisierung	21
KUTSCHKE, Karl-Heinz: Einheitliches Konzept der Realisierung der Kommunikation Mensch/EDVA in Dialogsystemen	41
LUDWIG, Manfred: Ein Beitrag zur Realisierung eines ESER-orientierten interaktiven graphischen Terminals	59
ROGMANN, Horst: Anwendungen des Programmsystems GIPS für ein aktives graphisches Bildschirmgerät und eine Zeichenmaschine	75
SCHOLZE, Walter: Anwendung des Bildschirmdialoges für Entwurfs- und Gestaltungsprozesse in der Architektur	77
SCHUBERT, Dietrich: Datenbanksysteme und Digitalgraphik	87
SIMON, Vera: Industrielle Anwendungsprogramme für die Digitalgraphik	105
STAHN, Heinz: Zum Anwendungsbereich des algorithmischen Systems Entscheidungstabellentechnik	123

Vorwort:

Vom 1. bis 6. 11. 1976 veranstaltete die Sektion Mathematik der Wilhelm-Pieck-Universität Rostock, Forschungsgruppe Digitalgraphik, eine wissenschaftliche Tagung über Digitalgraphik. Die Tagung verfolgte das Ziel, einen Überblick über theoretische und praktische Arbeiten zur Digitalgraphik zu geben, die in den letzten Jahren durchgeführt wurden. Im Mittelpunkt standen folgende Themen:

- Grundlagen der Digitalgraphik,
- Digitalgraphische Programmsysteme,
- Anwendungen der Digitalgraphik.

Der Themenkreis wurde durch Übersichtsvorträge aus benachbarten Gebieten der Datenbanksysteme, der Entscheidungstabellentechnik und der Formelmanipulation abgerundet.

Die Teilnehmer kamen aus verschiedenen Bereichen des Hochschulwesens, der Akademie der Wissenschaften und der Industrie. Als Gäste der Tagung konnten Kollegen aus der UdSSR, CSSR und der VR Ungarn begrüßt werden. Die Tagung zeigte, wie wichtig es ist, die Grundlagenforschung eng mit den Bedürfnissen der Anwendung zu verbinden. Naturgemäß konzentrierte sich das Interesse der Tagungsteilnehmer besonders auf die Fragen des Einsatzes der Bildschirmtechnik und der Gestaltung von Dialogsystemen. In das Programm waren zwei Problem Diskussionen einbezogen:

Welche Probleme sind bei der Anwendung der Digitalgraphik zu lösen?

Was sind und was leisten Datenstrukturen?

Diese Diskussionen fanden starken Anklang und trugen wesentlich zum Gelingen der Tagung bei.

Mit dem vorliegenden Heft des Rostocker Mathematischen Kolloquiums werden die Übersichtsvorträge der Tagung veröffentlicht. Es ist vorgesehen, auch die Kurzvorträge in dieser Reihe herauszugeben.

Prof. Dr. H. Kieseewetter
Leiter der Tagung

Zu einigen Problemen bei der Entwicklung und Anwendung von digitalgraphischen Programmsystemen

Ausgehend von den Erfahrungen bei der Entwicklung des DIGRA 73-Systems /1/ soll versucht werden, einige allgemeingültige Standpunkte herauszuarbeiten, die für zukünftige Entwickler und Anwender von digitalgraphischen Programmsystemen von Bedeutung sein können. Dabei besteht die Gefahr, daß bestimmte Details, die unter Umständen für eine konkrete Aufgabe sehr wichtig sind, unterschätzt werden. Auch bestimmte allgemeine Probleme, wie beispielsweise der Aufbau großer Programmsysteme und die Gestaltung des Dialogs zwischen Nutzer und Automat, können hier nicht behandelt werden. Wir wollen drei Fragen in den Mittelpunkt stellen:

Welche spezifischen Probleme ergeben sich bei der Modellierung graphischer Objekte und Relationen?

Wo sind digitalgraphische Programmsysteme anwendbar?

Welche methodischen Richtlinien können für die Entwicklung und Nutzung von digitalgraphischen Programmsystemen aufgestellt werden?

1. Modellierung graphischer Objekte und Relationen

Wenn wir für einen neuen Anwendungsbereich Aufgaben mit Hilfe von Rechenautomaten lösen wollen, müssen wir als erstes Modelle für die zu bearbeitenden Objekte und Relationen entwickeln. Die Modelle sind abstrakte Beschreibungsformen, mit deren Hilfe wir die zu lösenden Aufgaben präzisieren und für eine anschließende automatische Bearbeitung auf Rechenautomaten aufbereiten. Dabei ergeben sich in natürlicher Weise Informationsstrukturen. Deren Elemente stellen Abstraktionen der konkreten Objekte dar, und die strukturellen Zusammenhänge spiegeln die Relationen wider, die zwischen

den Objekten bestehen. In zunehmendem Maße werden die Informationsstrukturen als algebraische Strukturen formuliert, denn algebraische Strukturen sind die adäquaten mathematischen Modelle für die Widerspiegelung von Objekten und Relationen /2,3/. Die algebraischen Strukturen bestehen aus Mengen von Elementen und Operationen, wobei die Operationen in der Regel nur für gewisse Typen von Elementen definiert werden.

Zuerst müssen die Informationsstrukturen festgelegt und auf dieser Grundlage Aufgaben formuliert werden. Dann hat man einen soliden Ausgangspunkt, um Algorithmen zur Lösung dieser Aufgaben aufzustellen.

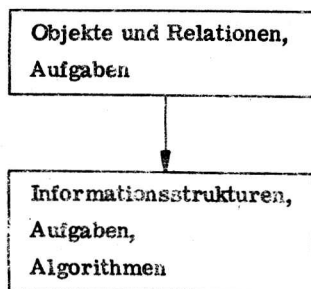
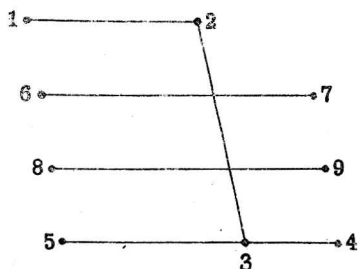


Abb. 1: Modellierung

Von dieser allgemeinen Position aus betrachtet, kann man sagen, daß das Problem der Modellierung graphischer Objekte und Relationen (s. Abb. 1) gelöst ist. Diese können genauso bearbeitet werden wie beispielsweise arithmetische Größen und Relationen.

Wir unterscheiden drei Klassen von graphischen Objekten: Kurven, Flächen und Körper. Die entsprechenden algebraischen Strukturen werden wir anhand von einfachen Beispielen erläutern.

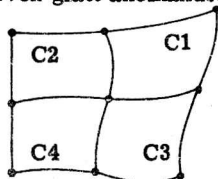
Als Beispiel für Kurven betrachten wir eine einfache Rohrleitungsanlage A (s. Abb. 2), bestehend aus den beiden Leitungen L 1 und L 2, die ihrerseits aus je zwei Strängen S1 und S2, bzw. S3 und S4 zusammengesetzt sind (_ bedeutet Verbindung, . bedeutet Anreihung).



$$\begin{aligned}
 S1 &= 1 \quad 2 \quad 3 \quad 4 \\
 S2 &= 3 \quad 5 \quad - \quad - \\
 S3 &= 6 \quad 7 \\
 S4 &= 8 \quad 9 \\
 L1 &= S1 \cdot S2 \\
 L2 &= S3 \cdot S4 \\
 A &= S1 \cdot S2 \\
 A &= L1 \cdot L2
 \end{aligned}$$

Abb. 2: Rohrleitungsanlage A

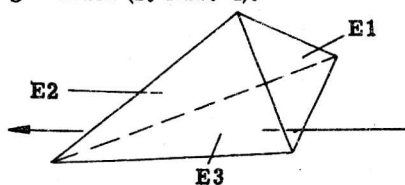
Als Beispiel für eine Fläche wählen wir einen Ausschnitt BUG aus einer Schiffsoberfläche (s. Abb. 3). BUG wird aus vier Flecken C1, C2, C3 und C4 zusammengesetzt. Die Flecken werden so definiert, daß sie längs ihrer Randkurven glatt aneinander anschließen (Coons-Flecken).



$$BUG = C1 \vee C2 \vee C3 \vee C4$$

Abb. 3: BUG

Körper sind räumliche Punktmengen, die als Durchschnitt und Vereinigung von elementaren Körpern (Halbräume, Kugeln, Kegel, Zylinder, usw.) erzeugt werden (s. Abb. 4).



$$TET = E1 \wedge E2 \wedge E3 \wedge E4$$

Abb. 4: Tetraeder

Aus den genannten Grundtypen lassen sich mit Hilfe der graphischen Operationen hinreichend allgemeine graphische Objekte aufbauen.

Nehmen wir an, daß bei der Kontrolle der Rohrleitungsanlage A festgestellt wird, daß sich die Leitungen L1 und L2 kreuzen. Dann besteht die folgende Aufgabe: Leitung L2 soll kreuzungsfrei um die Leitung L1 herumgeführt werden (s. Abb. 5).

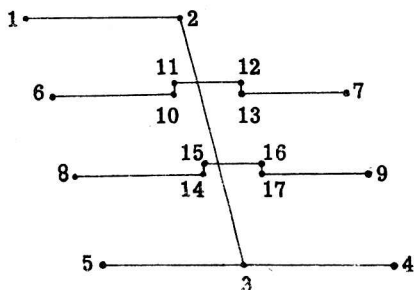


Abb. 5: Kreuzungsfreie Rohrleitungsanlage

Die Aufgabe wird durch den folgenden Algorithmus gelöst:

$$S3 = 6 _ 10 _ 11 _ 12 _ 13 _ 7$$

$$S4 = 8 _ 14 _ 15 _ 16 _ 17 _ 9$$

Dabei soll verabredungsgemäß der alte Wert einer Variablen automatisch gelöscht werden, wenn dieser Variablen ein neuer Wert zugewiesen wird.

An der einfachen Aufgabe für die Rohrleitungsanlage A lassen sich viele Wesenszüge der graphischen Datenverarbeitung demonstrieren. Der strukturelle Aufbau der Rohrleitungsanlage A kann durch einen Graph veranschaulicht werden. Die Knoten des Graphen sind die graphischen Objekte und die Bögen sind Verbindungen von einem Objekt nach allen Unterobjekten, aus denen dieses Objekt zusammengesetzt ist. Der untenstehende Graph (s. Abb. 6) zeigt den strukturellen Aufbau der Rohrleitungsanlage A, nachdem die Leitungen L1 und L2 kreuzungsfrei gemacht worden sind.

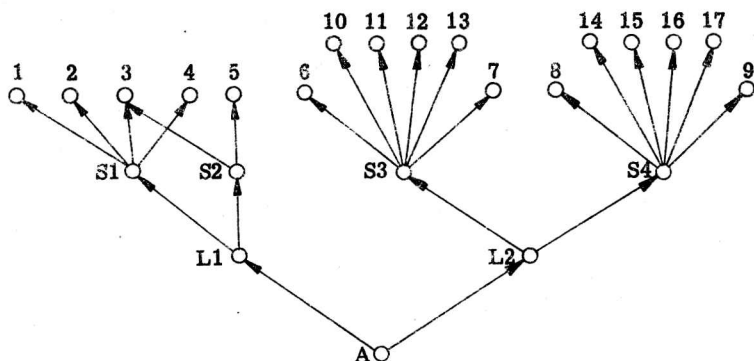


Abb. 6: Struktureller Aufbau der Rohrleitungsanlage A

2. Realisierung digitalgraphischer Programmsysteme

Wie das Beispiel (s. Abb. 6) zeigt, müssen wir die strukturellen Beziehungen in graphischen Gebilden zugriffsbereit zur Verfügung stellen, wenn wir Veränderungen in diesen Gebilden vornehmen wollen. Dabei müssen wir zwischen den Anforderungen bei der Stapelverarbeitung und der Dialogverarbeitung grundsätzlich unterscheiden.

Bei der Stapelverarbeitung (s. Abb. 7) haben wir es mit einem determinierten Prozeß zu tun. Die Ergebnisse sind bereits in eindeutiger Weise bestimmt, wenn die Eingabedaten vorgegeben sind. Von der Eingabe der Daten bis zur Ausgabe der Ergebnisse steuert das Programm den Ablauf der Verarbeitung. Willkürliche Veränderungen und Eingriffe sind nicht vorgesehen. Der Programmierer kann seine graphischen Gebilde so aufbauen, daß das geplante Ergebnis in möglichst wenig Zwischenstufen erzeugt wird. Die Teilinformationen werden im allgemeinen sofort ausgewertet, wenn sie entstehen. Beispielsweise können wir die benötigten graphischen Objekte mit Hilfe von Unterprogrammen aufbauen. Der Aufruf eines Unterprogramms bewirkt die Erzeugung des graphischen Objekts. Indem wir mehrere Aufrufe in geeigneter Weise kombinieren, schaffen wir die gewünschte Zusammenstellung der graphischen Objekte zu einem Bild, das wir anschließend beispielsweise auf einer Zeichenmaschine darstellen können. Viele Programm-

systeme zum automatischen Zeichnen arbeiten nach diesem Prinzip. Um Aufgaben in Stapelverarbeitung zu lösen, genügen bereits relativ starre Speicherungsformen für die strukturellen Beziehungen.

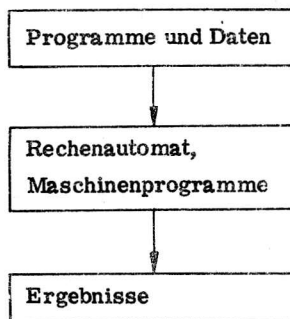


Abb. 7: Stapelverarbeitung

Bei der Dialogverarbeitung (s. Abb. 8) gibt es wesentlich stärkere Anforderungen:

- Ständiger Zugriff zu möglichst allen Teilinformationen.
- Ständiges Erzeugen, Löschen und Verändern von Teilinformationen.

Um diese Anforderungen zu erfüllen, brauchen wir grundsätzlich neue Speicherungsformen. Insbesondere müssen wir den strukturellen Aufbau, wie er beispielsweise in Abb. 6 dargestellt wird, in sehr variabler Weise speichern, um die Voraussetzungen für dynamische Strukturveränderungen zu schaffen. Das zwingt dazu, Datenstrukturen aufzubauen, auf denen die obengenannten Operationen effektiv ausgeführt werden können.

Daraus leiten sich weitere Anforderungen ab:

- Strukturprogramme für Aufbau, Belegung und Veränderung von Datenstrukturen,
- Problemprogramme für ständig wiederkehrende graphische Grundoperationen,
- Dialogsteuerung, um den Wechsel zwischen den Aktionen des Nutzers und den Reaktionen des Automaten zu ermöglichen,

- Menüs von Lichttasten, die dem Nutzer die nächsten möglichen Aktionen anbieten, Zusammenstellung aller Programme zu einem Programmsystem.

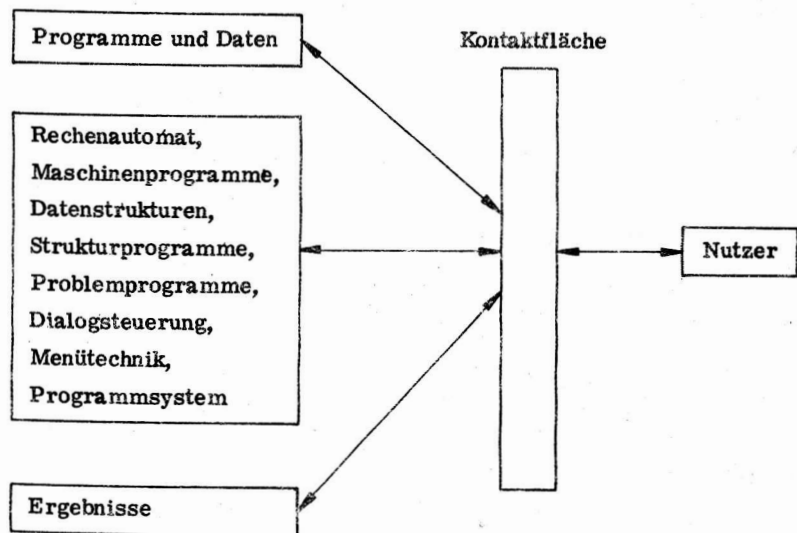


Abb. 8: Dialogverarbeitung

Ausgehend von den genannten Grundkomponenten sind weitere Probleme zu lösen:

- Datenstrukturen und Problemprogramme müssen optimal aufeinander abgestimmt werden, dergestalt daß die am häufigsten auftretenden graphischen Grundoperationen schnell ausgeführt werden.
- Die strukturellen Informationen müssen automatisch aus- und eingeladen werden, sobald der zur Verfügung stehende Kernspeicherplatz voll ausgelastet ist.
- Die strukturellen Informationen müssen konserviert werden, damit bei einer Unterbrechung der Dialogsitzung der aktuelle Stand der Bearbeitung des Problems gegebenenfalls über einen längeren Zeitraum aufbewahrt werden kann und in dieser Form für eine spätere Dialogsitzung zur Verfügung steht. Außerdem dient die Konservierung dem Aufbau von Datenbanken.

- Für die Dialogverarbeitung sind effektive Realisierungsprinzipien auszuarbeiten.
- Es ist notwendig, große Programmsysteme in Segmente zu zerlegen, um den Kernspeicherbedarf in bestimmten Grenzen zu halten.
- Programmiersprachen und Programmsysteme müssen offen gestaltet werden, so daß Erweiterungen bezüglich der Objekte und Operationen in einfacher Weise vorgenommen werden können.

3. Anwendungen digitalgraphischer Programmsysteme

Bei den Anwendungen der Digitalgraphik sind die verschiedenen Typen graphischer Ein- und Ausgabegeräte (z. B. Digitalisiergeräte, Zeichenmaschinen, alphanumerische Bildschirmgeräte, graphische Bildschirmgeräte) und die entsprechenden Zusatzeinrichtungen problemabhängig einzusetzen. Nicht für jede Anwendung braucht man unbedingt ein graphisches Bildschirmgerät und Dialogverarbeitung. Insbesondere kann man auf den Dialog verzichten, wenn leistungsfähige Algorithmen vorliegen, mit denen eine Aufgabe ohne zusätzliche Eingriffe des Nutzers vollständig gelöst werden kann. Dann ist die Stapelverarbeitung vorzuziehen.

Trotzdem muß natürlich gesagt werden, daß der Dialog eine große Bedeutung für die graphische Datenverarbeitung hat. Die graphischen Bildschirmgeräte mit den bekannten Möglichkeiten der graphischen Aus- und Eingabe fordern geradezu zum Dialog heraus. Sie bieten die weitaus stärkeren Möglichkeiten für die Anwendungen der Digitalgraphik. Viele Aufgabenkomplexe der Digitalgraphik können überhaupt erst auf der Grundlage der Bildschirmtechnik erfolgreich bearbeitet werden.

Die Anwendungen der Digitalgraphik erfassen nahezu alle Bereiche produktiver Tätigkeit des Menschen, weil graphische Informationen von der Grundlagenforschung bis hin zur unmittelbaren Produktion entscheidend in die Problemlösungsprozesse eingehen. Besonders starke Effekte können durch den Einsatz der Bildschirmgeräte bei der Automatisierung von Entwurfsprozessen erzielt werden. Hier werden durch die Anwendungen der Digitalgraphik

in den volkswirtschaftlich wichtigen Bereichen der technischen Vorbereitung, bei der Projektierung, Konstruktion, Ausarbeitung der Technologie und bei der Durchführung der Produktion qualitativ neue und leistungsfähigere Lösungsverfahren erschlossen. Solche komplexen Lösungsverfahren bedürfen aber auch umfangreicher Vorbereitungsarbeiten.

Wie kommen wir zu einer Einschätzung der möglichen Anwendungsgebiete der Digitalgraphik?

Der Wunsch allein, graphische Informationen in die automatische Bearbeitung einzubeziehen, ist nicht hinreichend, um eine effektive Anwendung zu begründen. Zuerst muß der Problemlösungsprozeß gründlich analysiert werden, um die erzielbaren Effekte exakt zu ermitteln. Es muß gesichert sein, daß die graphischen Informationen entscheidend in den Problemlösungsprozeß eingehen.

Bei der schrittweisen Lösung der gestellten Aufgabe muß der mit dem Problem vertraute Fachmann immer wieder durch die graphischen und weitere ergänzende Informationen in seinen Entscheidungen unterstützt werden.

Besonders günstige Verhältnisse liegen vor, wenn die Problemobjekte selbst graphische Objekte sind (Leiterplatten, integrierte Schaltungen, Rohrleitungen u. a.).

Die Effektivität einer Anwendung hängt aber auch davon ab, wie es gelingt, die Arbeit mit dem Programmsystem nutzerfreundlich zu gestalten. Für die Dialogarbeit am Bildschirm muß dem Nutzer ein Menü von Lichttasten angeboten werden, das ihm erlaubt, seine Aufgaben mit möglichst wenig elementaren Aktionen möglichst schnell zu lösen. Die am häufigsten auftretenden Fälle werden zum Standard erhoben. Vom Nutzer werden nur Aktionen erwartet, wenn er von diesen Standards abweicht. Dadurch kann die Anzahl der notwendigen Nutzeraktionen wesentlich reduziert werden. Die Zusammenstellung des Menüs für eine konkrete Aufgabe ist von großer Bedeutung für den praktischen Einsatz der Digitalgraphik.

Unabhängig von den konkreten Problemsituationen bei den Anwendungen der Digitalgraphik können einige generelle Aussagen über die Effektivität gemacht werden:

- Die graphische Auswertung von Ergebnissen ist effektiver als die Auswertung mit Hilfe von Tabellen.
- Die graphische Kontrolle ist effektiver als andere Kontrollformen.
- Bei der automatischen Verarbeitung graphischer Informationen werden wesentlich kürzere Verarbeitungszeiten erzielt. Nicht in jedem Fall ist das gleichzeitig mit einer Senkung der Kosten verbunden.
- Die Dialogarbeitsweise führt zu einer wesentlich höheren Effektivität des Nutzers in der Phase der schöpferischen Bearbeitung eines Problems.

In der Tabelle 1 wird versucht, die wichtigsten Aufgabenkomplexe für die Anwendungen der Digitalgraphik zusammenzustellen und anhand einiger Kriterien zu bewerten. Die Rolle der graphischen Informationen im Problemlösungsprozeß, die Möglichkeiten für den Einsatz der Stapel- bzw. Dialogverarbeitung und besondere Probleme bei der Realisierung müssen auf alle Fälle berücksichtigt werden. Die Aufgabenkomplexe sind Zusammenfassungen, die in verschiedenen Stufen auf der Basis verschiedener digitalgraphischer Geräte ohne und mit Dialogverarbeitung und für vielschichtige spezielle Problemsituationen in Angriff genommen werden müssen. Teilweise existieren schon ausgereifte Lösungen für solche Aufgaben.

Tabelle 1: Aufgabenkomplexe für Anwendungen der Digitalgraphik

Aufgabenkomplex	Problem- objekte	Graphische Objekte u. ihre Bewer- tung	Stapel(S)- bzw. Dialog(D)- verarbeitung	Besondere Probleme bei der Realisierung
1. Berechnung und Darstellung von Diagrammen, Höhenlinien und Funktionsge- birgen	allgemei- ne Größen	2d-Kurven, 3d-Kurven, Flächen, Mittel für die Auswer- tung	S: stark D: normal	Kurvensteuerung bei unterschied- lichen Anforderungen, Bereitstel- lung der Eingabedaten, Verarbeitung großer Datenmengen, Aufwand bei anspruchsvollen Dar- stellungen.
2. Entwurf und Kontrolle von Leiterplatten	Leiter- platten	2d-Kurven, Problem- objekte	D: stark S: normal	Automatische Bearbeitung von Näherungslösungen
3. Entwurf und Kontrolle von integrierten Schaltungen	Schaltun- gen	2d-Kurven, 2d-Gebiete, Problem- objekte	S: normal D: stark	Kopplung zwischen Gebieten und ihren Berandungen
4. Entwurf und Kontrolle von Rohrleitungen	Rohrlei- tungen	3d-Kurven, Problem- objekte	S: normal D: stark	Automatische Bereitstellung von Näherungslösungen, Erfassung von Details
5. Entwurf und Kontrolle von Maschinenauf- stellungen	Maschinen (Position)	2d-Kurven, 3d-Kurven, Körper, Problem- objekte	S: normal D: stark	Aufwand bei anspruchsvollen Dar- stellungen, Erfassung von Details

6. Entwurf und Kontrolle von Gebäudezusammenstellungen	Baukörper (Position)	3d-Kurven, Körper, Problemobjekte	S: normal D: stark	Bewertung des Entwurfsprozesses, Verarbeitung großer Datenmengen, Erfassung von Details
7. Entwurf und Berechnung von Maschinenkonstruktionen	Maschinen (Funktionen)	2d-Kurven, 3d-Kurven, i. a. Hilfsobjekte	S: normal D: stark	Anteil der graphischen Informationen an den funktionellen Beziehungen, Kopplung graphischer und nichtgraphischer Informationen, Modellierung dynamischer Vorgänge
8. Entwurf und Berechnung von Baukonstruktionen	Bauteile (Funktionen)	2d-Kurven, 3d-Kurven, i. a. Hilfsobjekte	S: normal D: stark	Anteil der graphischen Informationen an den funktionellen Beziehungen, Kopplung graphischer und nichtgraphischer Informationen
9. Entwurf und Kontrolle von Oberflächen, u. a. im Schiffbau, Automobilbau, Flugzeugbau	Oberflächen	Flächen, Problemobjekte	S: schwach D: stark	Approximation und Interpolation von Flächen, Glatte Verheftung von Flecken, Aufwand bei anspruchsvollen Darstellungen

4. Methodische Richtlinie für die Entwicklung und Anwendung von digitalgraphischen Programmsystemen

Bei der Konzeption und dem Einsatz digitalgraphischer Programmsysteme muß man von den zu lösenden Aufgaben ausgehen. Die dreifache Stufung: Modellierung, Realisierung, Anwendung entspricht den bisherigen Erfahrungen. Hieraus lassen sich Einschätzungen über den Umfang der zu lösenden Aufgaben, eine sinnvolle Planung des Arbeitsablaufes bei der Programmentwicklung und die Möglichkeiten für den Einsatz ableiten. Aus den vorangehenden Abschnitten ergibt sich zusammenfassend eine methodische Richtlinie, die in Abb. 9 dargelegt ist.

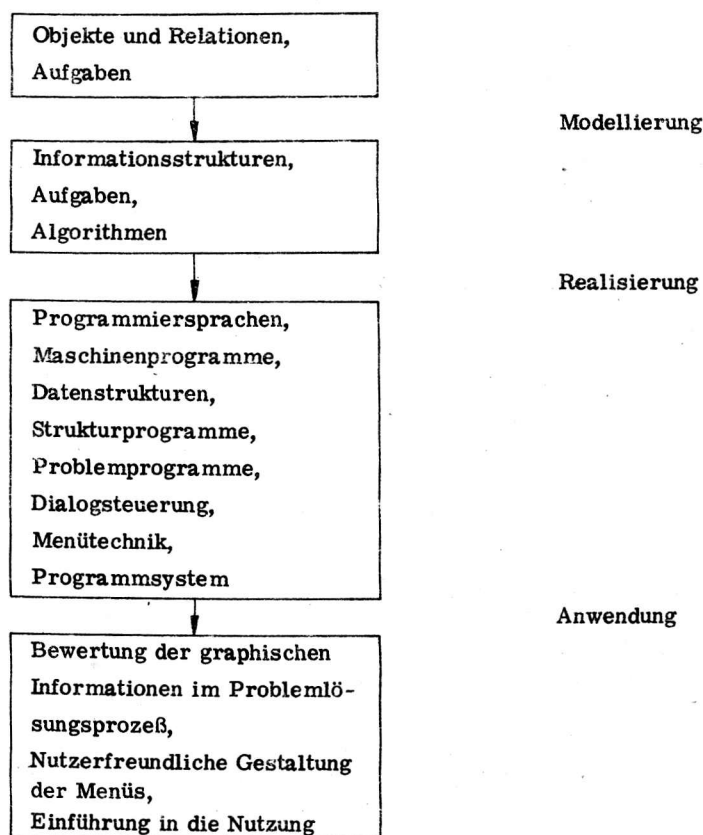


Abb. 9: Methodische Richtlinie

Literatur

- /1/ Programmierhandbuch DIGRA 73. Schriftenreihe des Instituts für Schiffbau, Rostock, 1974
- /2/ Kiesewetter, H.: Beiträge zur Digitalgraphik
Allgebraische Strukturen in der Digitalgraphik.
Wiss. Zeitschrift Rostock, 1972, S. 705 - 714
- /3/ Глушков, Цейтлин, Ющенко,
Алгебра, языки, программирование,
Киев 1974

Eingegangen: 30. 11. 1976

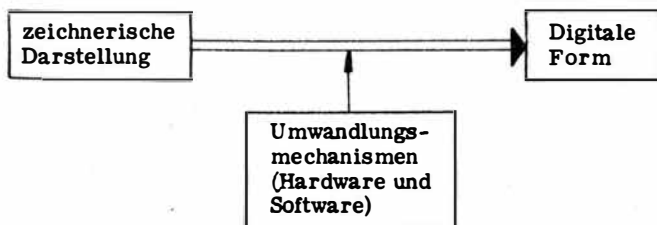
Anschrift des Verfassers

Prof. Dr. habil. Helmut Kiesewetter
Wilhelm-Pieck-Universität Rostock
Sektion Mathematik
DDR - 25 Rostock, Universitätsplatz 1

Programmiersprachliche Aspekte in der Digitalgraphik und Probleme bei ihrer Realisierung

1. Einführende Bemerkungen

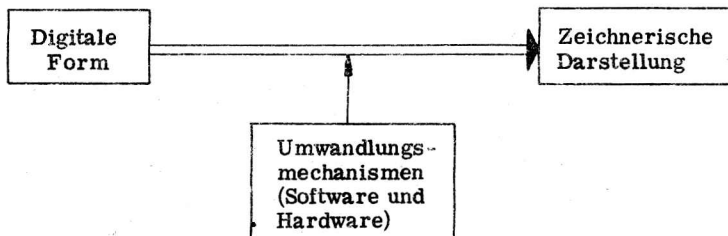
Durch die Schaffung graphischer Geräte für die Informationsverarbeitung (genauer: graphischer peripherer Ein- und Ausgabegeräte) ist die Verarbeitung graphischer Informationen durch Rechenautomaten möglich geworden. Dadurch ist nicht ein neuer Typ von Rechnern notwendig geworden, der mit graphischen Größen (wie Punkt, Strecke, Ebenenstück, Dreieck usw.) arbeiten kann, sondern nur die Möglichkeit gegeben, dem Rechner die Informationen in graphischer Form mitzuteilen bzw. vom Rechner die Ergebnisse in graphischer Form z. B. als Zeichnung von der Zeichenmaschine oder als Bild vom Bildschirmgerät zu erhalten. Die Rechner arbeiten nach wie vor auf digitaler Basis, so daß keine Spezialrechner für die graphische Datenverarbeitung erforderlich sind. Somit findet bei der Eingabe von graphischen Informationen ein Umwandlungsprozeß statt, der durch die folgende Skizze wiedergegeben werden kann:



Dieser Prozeß der Umwandlung ist insofern problematisch, da bei der automatischen Abtastung einer zeichnerischen Darstellung die geometrischen Zusammenhänge kaum berücksichtigt werden können bzw. sonst der Mensch in den Eingabeprozess eingeschaltet werden muß. Dies ist neben technischen

Problemen sicherlich mit ein Grund, warum die graphischen peripheren Eingabegeräte nicht in großem Umfang eingesetzt wurden und werden. Das bedeutet nicht, daß diese Geräte nicht ihre Berechtigung und Zukunft haben (z. B. Digitalisiergeräte). Es gibt jetzt schon konkrete Anwendungsbereiche, in denen sie einen festen Platz einnehmen (s. auch Vorträge von V. Dobrowolny und von R. Plewka auf dieser Tagung).

Bei der Ausgabe graphischer Informationen findet ein Umwandlungsprozeß statt, den die folgende Skizze wiedergibt:



Hier ergibt sich eine neue Qualität für den Einsatz des Rechenautomaten und seine Beziehung zum Menschen, d. h. zum Nutzer. Abgesehen vom bedeutend komplizierteren Umwandlungsprozeß gegenüber der digitalen Ausgabe, was größerer Softwareaufwand und höhere Rechenzeiten bedeutet, und von der komplizierteren Gerätetechnik für die graphische Ausgabe ergeben sich folgende Forderungen:

1. Dadurch, daß der Mensch die graphischen Zusammenhänge sehr schnell erfaßt (im Vergleich zu digitalen), entsteht beim Nutzer das starke Bedürfnis einer sofortigen Änderung. Das bedeutet Angabe von Informationen, mit deren Hilfe der Rechenautomat Veränderungen in der digitalen Form vornimmt, die die gewünschten Verbesserungen in der zeichnerischen Darstellung bringen, d. h. interaktive Arbeit, graphischen Dialog. Darauf wird im weiteren noch eingegangen.
2. Da die digitale Form für eine Zeichnung in vielen Fällen nicht aus einer zeichnerischen Darstellung, also nicht über die graphische Eingabe, entsteht, müssen noch andere Möglichkeiten für ihre Erstellung geschaffen werden. Hier bieten sich als bestes Hilfsmittel die graphischen Sprachen an. Denn

die digitale Form durch Angabe von Punktfolgen, wie sie für die graphische Ausgabe benötigt werden, zu erstellen, ist eine vergleichsweise ähnlich aufwendige Tätigkeit wie in den 50er Jahren die Programmierung in Maschinensprache. Natürlich sind für die Erstellung der digitalen Form auch Zwischenstufen möglich, wie z. B. Schaffung einer Menge von Unterprogrammen aufrufbar in einer üblichen Programmiersprache. In speziellen Anwendungsfällen sind damit gute Ergebnisse zu erreichen und schon erreicht worden.

Zusammenfassend kann gesagt werden: Für den Einsatz graphischer Arbeits- und Denkweisen in der Informationsverarbeitung sind für lange Sicht erforderlich

- ein modernes Rechnersystem,
- graphische periphere Geräte, die den graphischen Dialog ermöglichen (z. B. graphisches Bildschirmgerät),
- zusätzliche Software (als graphische Software bezeichnet), die die Erzeugung der zeichnerischen Darstellung aus der digitalen Form, die Erzeugung der digitalen Form aus einer nutzerfreundlichen Form (z. B. graphische Sprache) und die Manipulation mit der zeichnerischen Darstellung (graphischer Dialog) ermöglicht.

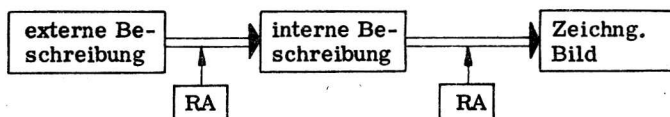
Bemerkung: Dies soll nicht heißen, um mit der graphischen Datenverarbeitung in einem Anwendungsbereich zu beginnen, müssen alle genannten Komponenten erfüllt sein. Man kann schon sehr gute Ergebnisse mit sehr bescheidenden Mitteln erzielen.

2. Graphische Sprachen

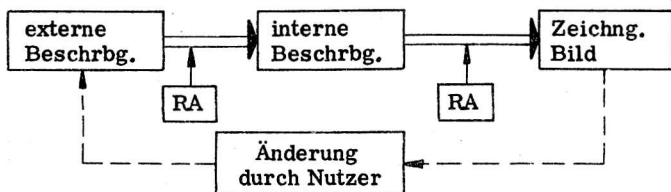
Auf die Bedeutung und Notwendigkeit der graphischen Sprachen wurde schon im 1. Abschnitt hingewiesen. Somit wird mit Hilfe der graphischen Sprache die Beschreibung der geometrischen Gestalt eines Objektes vorgenommen. Sie wird vom Menschen durch einen Abstraktionsprozeß erstellt und soll als externe Beschreibung des Objektes bezeichnet werden.

Aus der externen Beschreibung erzeugt der Rechenautomat mit Hilfe der entsprechenden Software die digitale Form, die interne Beschreibung der geometrischen Gestalt des Objektes. Die interne Beschreibung wird im Rechner durch Datenstrukturen realisiert.

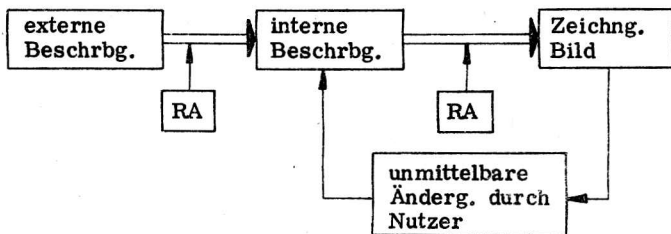
Der Prozeß der automatischen Erzeugung einer Zeichnung bzw. eines Bildes kann durch die folgende Skizze veranschaulicht werden:



Damit ist gleichzeitig die passive Arbeitsweise charakterisiert (s. Abb. 1a), die durch Stapelverarbeitungssysteme im Rechenautomaten realisiert wird. Im Gegensatz dazu liefert die aktive Arbeitsweise (auch interaktive Arbeitsweise) eine Änderungsmöglichkeit in der internen Beschreibung (s. Abb. 1b). Dies verlangt jedoch eine andere Organisation im Rechenautomaten, da nach



a) passive Arbeitsweise



b) aktive Arbeitsweise

Abb. 1: Arbeitsweisen mit Rechenautomaten

Beenden eines Programms in der passiven Arbeitsweise die interne Beschreibung verlorengeht, aber in der aktiven Arbeitsweise erhalten bleiben muß. Die aktive Arbeitsweise wird durch Dialogsysteme (auch interaktive Systeme) realisiert. Es ist klar, daß ein Dialogsystem auch passive Arbeitsweise ermöglicht. Die Abb. 1b) angegebene Änderung bedeutet nicht, daß der Nutzer die interne Beschreibung kennen muß. Jedoch bewirken die Änderungen, die in ihrer Gesamtheit die Dialogsprache ausmachen, eine Veränderung der internen Beschreibung.

2.1. Einordnung in Programmiersprachen

Betrachtet man die bekannten graphischen Sprachen, so stellt man fest, daß sie meistens Erweiterungen der gängigen Programmiersprachen (z. B. FORTRAN, PL/1) sind. Dies hat mehrere Gründe:

1. Für die externe Beschreibung der geometrischen Gestalt eines Objektes sind oft viele numerische Angaben und Berechnungen notwendig (z. B. Bestimmung von Punktkoordinaten, Berechnung der Schrittgeraden zweier Ebenen oder des Durchstoßpunktes einer Geraden mit einer Fläche), die durch Sprachelemente bisheriger Programmiersprachen ausdrückbar sind.
2. Für die Übersetzung ist die Schaffung eines Vorkompilers ausreichend, der die Übersetzung der graphischen Sprachelemente auf das Niveau der zugrundegelegten Programmiersprache vornimmt.
3. Die Prinzipien für die Vorkompilation des vorliegenden Sachverhaltes sind einfach. Die Erstellung des Vorkompilers kann automatisch geschehen (s. DEPOT - System der TU Dresden) oder in der zugrundegelegten Programmiersprache erfolgen, so daß keine Spezialkenntnisse bzgl. der Programmierung notwendig sind.

Dadurch durchläuft ein in der graphischen Sprache geschriebenes Programm bis zur Erstellung in der Maschinensprache eine Sprachhierarchie, die in Abb. 2 skizziert ist. Um ein arbeitsfähiges Programm zu erhalten, müssen zum übersetzten Programm noch die erforderlichen Programme aus der jeweiligen Bibliothek (in Abb. 2 'graphische Bibliothek' genannt) dazugela-

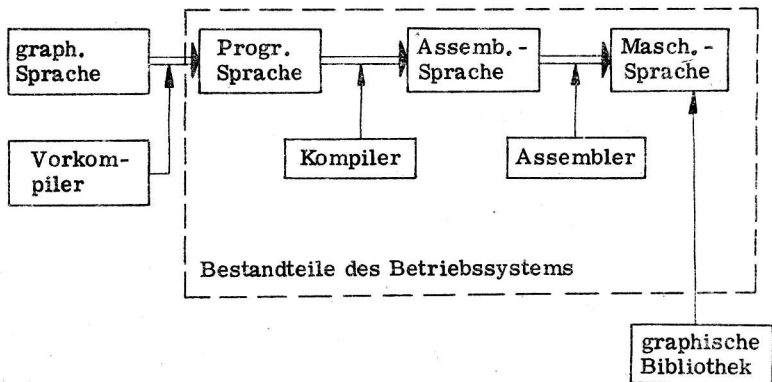


Abb. 2: Sprachhierarchie in der Übersetzungsphase

den werden. Diese Überlegungen geben einen Ansatzpunkt für eine klare Definition von Programmpaket und Programmsystem (s. dazu /3/).

2.2. Variablenprinzip und Werte

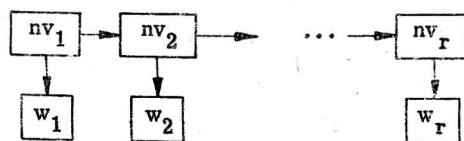
Als sehr brauchbar und tragfähig hat sich bei der Arbeit mit Programmiersprachen die Einführung von Variablen erwiesen. Eine Variable wird durch einen Namen, einen Typ und einen Wert charakterisiert. In den meisten Programmiersprachen ist eine feste Typmenge vorgegeben, aus der dann der Nutzer den jeweiligen Typ seiner Variablen auswählen kann. Der Wert ist entweder eine Zeichenkette (bei string-Variablen) oder ein logischer Wert (wahr oder falsch) oder ein Zahlenwert (in Variabel- oder Festkommadarstellung), der aus dem Zahlenbereich des entsprechenden Rechenautomaten stammt. Der Name ist eine alphanumerische Zeichenkette (beginnend mit einem Buchstaben), die vom Nutzer beliebig gebildet werden kann. Der Nutzer arbeitet nur mit den Namen (diese sind ihm vertraut), verknüpft sie mit vorgegebenen Operatoren und bildet zusammen mit Funktionen sehr komplizierte Ausdrücke. Für den Rechenautomaten bedeutet dies Verknüpfung der zugehörigen Werte und Berechnung eines Wertes nach der dem Ausdruck entsprechenden Vorschrift. Damit kann ein derartig ermittelter Wert wieder einer Variablen mit entsprechendem Typ zugewiesen werden (Ergibt-

zuweisung oder Ergibtanweisung). Darauf gründet sich die dynamische Arbeit eines Programms mit der Maßgabe:

Nur eine solche Variable darf in einem Ausdruck auftreten, die "vorher" in der dynamischen Abarbeitung einen Wert erhalten hat.

Bei graphischen Sprachen werden analog graphische Variable (numerische Variable sollen die oben erläuterten heißen) mit den gleichen Charakteristika eingeführt. Die Namen kann der Nutzer nach dem gleichen Prinzip selbst wählen, der Typ ist aus einer vorgegebenen graphischen Typmenge auswählbar, aber die graphischen Werte sind von bedeutend komplizierterer Struktur. So kann sich ein Graph (Streckenzug) aus einer unterschiedlichen Anzahl von Strecken zusammensetzen, die jeweils wiederum durch einen Anfangspunkt und einen Endpunkt definiert sind. Jeder Punkt ist selbst wieder durch zwei bzw. drei Zahlenwerte als Koordinaten definiert. Der Streckenzug könnte ebenfalls noch aus Schnitten von Ebenen entstanden sein. Daraus folgt, daß für die graphischen Werte andere Speicherprinzipien notwendig sind als für die Werte von numerischen Variablen.

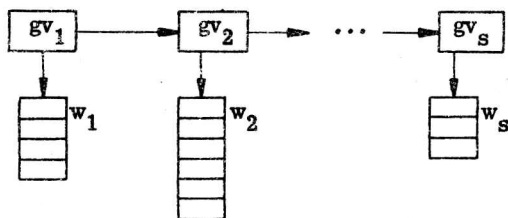
Aussagen über die Nutzung von Prinzipien für die Abspeicherung graphischer Werte liefert die Theorie der Datenstrukturen. Dazu werden einige Aussagen im 3. Abschnitt gemacht. In Abb. 3 werden die Unterschiede zwischen den Werten noch einmal aufgezeigt.



Konstanter Speicherbereich für Werte

nv_i Namen, w_i Werte für $i = 1(1)r$

a) Numerische Variable



Variabler Speicherbereich für Werte

gv_j Namen, w_j Werte für $j = 1(1)s$

b) Graphische Variable

Abb. 3: Werte von Variablen

Abb. 3 soll keine Aussage über die Abspeicherung der Werte im Speicher geben. Hier sind noch alle Möglichkeiten zugelassen, und sie hängt in starkem Maße von der Nutzung (z. B. Dialogbetrieb oder nicht) ab.

Die Abspeicherung der Werte graphischer Variablen auf externen Speichern kann im Prinzip in gleicher Weise wie bei numerischen Variablen erfolgen, wenn die Besonderheit der variablen Wertlänge beachtet wird.

Die Nutzung von graphischen Feldern (indizierte graphische Variable zum Ansprechen der Feldelemente) bereitet unter Beachtung der variablen Wertlänge keine prinzipiellen Schwierigkeiten. Hier sind in Abhängigkeit der Abspeicherungsform (d. h. welche Struktur wird verwendet) einige Spezifika zu berücksichtigen (z. B. wie variable Feldgrenzen behandeln, welcher Auf- und Abfragemechanismus für die einzelnen Elemente).

Damit ist es möglich geworden, mit den graphischen Größen in gleicher Weise umzugehen wie mit den numerischen Größen in den bekannten Programmiersprachen, wobei sogar das Nebeneinanderbestehen beider Arten von Größen gewährleistet ist.

2.3. Sprachelemente

Mit den graphischen Variablen und der Einführung von graphischen Operationen (anwendbar auf einzelne Typen von graphischen Variablen oder auf alle graphischen Variablen) können komplizierte graphische Ausdrücke gebildet werden, die Grundlage für die Bildung von graphischen Werten und für die Einführung der graphischen Ergibtanweisung sind. Damit kann die Laufanweisung, wie sie in mindestens einer Form in jeder Programmiersprache vorhanden ist, günstig zur Beschreibung geometrischer Sachverhalte verwendet werden.

Durch die Einführung graphischer Standardfunktionen und/bzw. Standardunterprogramme und spezieller Anweisungen graphischer Natur kann der Komfort in einer graphischen Sprache für den Nutzer bedeutend erhöht werden. Dieser Komfort ist in den einzelnen Sprachen unterschiedlich groß. So werden z. B. als graphische Standardfunktionen sehr oft spezielle Transformationen benutzt wie Drehung um die Koordinatenachsen, Verschiebung in Koordinatenachsenrichtung, Streckung bzw. Stauchung um einen Faktor u. a. m.; auch Zugriffs- und Änderungsprogramme zu Teilen eines graphischen Wertes sind oft vorhanden.

Ein weiteres, sehr brauchbares Hilfsmittel für die Anwendung ist die Unterprogrammtechnik, wenn sie in analoger Weise, wie sie aus der üblichen Programmiertechnik bekannt ist, auf graphische Variable übertragen wird. Umfangreicher interner Arbeitsaufwand entsteht dabei nur bei der Rückvermittlung graphischer Ergebnisparameter (s. dazu 3. Abschnitt).

Besondere Überlegungen sind bei der Ein- und Ausgabe graphischer Größen notwendig. Im folgenden Abschnitt erfolgt eine Beschränkung in der Darstellung dieser Problematik auf die Ausgabe graphischer Größen mit graphischen Ausgabegeräten (kurz graphische Ausgabe genannt), da dieser Komplex die meisten Besonderheiten gegenüber dem bisher Bekannten aufweist.

2.4. Graphische Ausgabe

Die Ein- und Ausgabe in üblichen Programmiersprachen erfolgt i. d. R. mittels zweier Anweisungen, der

Ein- bzw. Ausgabeanweisung und der

Zusatzanweisung

Durch die Ein- bzw. Ausgabeanweisung werden das ein- bzw. auszugebende Gerät und die Variablen, deren Werte ein- bzw. ausgegeben werden, angegeben. Die Zusatzanweisung enthält Angaben, wie die Ein- bzw. Ausgabe dieser Werte erfolgen soll. Die Zusatzanweisung ist immer eine markierte Anweisung. Eine Markenangabe in der Ein- bzw. Ausgabeanweisung stellt den Bezug zur entsprechenden Zusatzanweisung her.

Dieses Prinzip der beiden Anweisungen kann auch voll für die graphische Ausgabe übernommen werden. So werden in der graphischen Ausgabeanweisung das graphische Ausgabegerät und die graphischen Variablen, deren Werte die Zeichnung bzw. das Bild ergeben, angegeben. Die graphische Zusatzanweisung enthält Angaben zur Erstellung der Zeichnung bzw. des Bildes jedoch mit dem Unterschied, daß sich die Angaben auf die gesamte Zeichnung bzw. das gesamte Bild beziehen und nicht auf die einzelnen Variablen in der Liste der graphischen Ausgabeanweisung. Die Bindung zwischen beiden Anweisungen erfolgt auch hier durch eine Markenangabe und eine Markierung. Die Verwendung von zwei Anweisungen für die graphische Ausgabe hat auch Konsequenzen für den Dialog, auf die im 4. Abschnitt eingegangen wird.

Bemerkung: In einigen graphischen Sprachen werden für Angaben, die im obigen Sinne Teile der graphischen Zusatzanweisung sind, spezielle Anweisungen verwendet, z. B. in der graphischen Version der LOGO-Sprache (s. /2/) die SIZE-Anweisung als Fensterungsoperation (windowing), die dann solange ihre Gültigkeit besitzt, bis sie durch eine andere SIZE-Anweisung abgelöst wird.

3. Nutzung von Datenstrukturen

In 2.2. wurde schon festgestellt, daß die Werte graphischer Variablen durch Datenstrukturen dargestellt und daß für ihre Abspeicherung Speicherbereiche von unterschiedlicher Länge benötigt werden, die sich während der Abarbeitung eines Programms ständig ändert. Dieser Sachverhalt wird auch in Abb 3. b) skizziert.

Aus der Theorie der Datenstrukturen sind für die Speicherung derartiger Werte drei Grundmethoden bekannt: die sequentielle Methode, der Direktzugriff und die Listen. Wenn für eine Problematik eine Datenstruktur und damit auch eine Abspeicherungsmethode ausgewählt werden soll, müssen die folgenden Fragen beantwortet werden:

1. Wieviel Informationen müssen in einem derartigen Wert maximal untergebracht werden?
2. Wie gut muß der Zugriff zu diesen Informationen sein?
3. Welche Operationen sollen mit diesen Informationen und den Werten durchgeführt werden?

Für die Digitalgraphik ist die Art der Datenstrukturen und damit die Beantwortung der drei Fragen sehr wichtig. Die Datenstrukturen müssen dabei den folgenden Anforderungen genügen, wenn eine effektive Arbeit möglich sein soll:

- Zusammenfassung von Objekten mit gemeinsamen Eigenschaften, z. B. mit Hilfe von Feldern.
- Zusammenfassung von Objekten, die eine hierarchische Struktur bilden, z. B. Baumstrukturen.
- Besondere Eignung für dynamisches Wachsen und Ändern.
- Möglichkeit des Ansprechens eines beliebigen Objekts (z. B. einen mit dem Lichtstift ausgewählten Punkt) und Lieferung einer entsprechenden Information des Objektes (z. B. Transformationsangabe).

Soll nun der Nutzer die Möglichkeit der Manipulation mit den graphischen Größen erhalten (d. h. interaktive Arbeit), so muß für die innere Organisation das folgende Prinzip gewahrt sein:

Die Struktur der Objekte, die vom Nutzer in der externen Beschreibung angegeben wird, muß bis hin zur graphischen Ausgabe erhalten bleiben.

Dies ist die Voraussetzung für die weiteren Betrachtungen und hat viele Konsequenzen auf die Arbeitsweise und die Nutzung graphischer Systeme.

3.1. Abspeicherung graphischer Werte

Unter Beachtung der obigen Voraussetzung und der aufgestellten Anforderungen und Fragen kommen für die Abspeicherung graphischer Werte nur Listen in Frage. Dabei ist es gleichgültig, welche Art von Listen (lineare oder verbundene Listen, Kreislisten einfacher oder doppelter Verknüpfung u. a.) benutzt wird. Dies hängt dann von den vorhandenen Hard- und Softwaremöglichkeiten und den konkreten Problemstellungen ab.

Die hierarchische Struktur der Objekte (Objekt zerfällt in Teilobjekte und diese wieder in Teilobjekte usw.) und die Voraussetzung lassen mehrere Schlußfolgerungen zu:

1. Im Wert eines Objektes sind die Verweise auf Teilobjekte vorhanden.
2. Um die Manipulierbarkeit zu erhalten, müssen die Verweise auf die den Teilobjekten entsprechenden graphischen Variablen und nicht auf ihre Werte zeigen (s. Abb. 3b)).
3. Die Informationen eines graphischen Wertes sind im wesentlichen Zeiger auf andere graphische Variablen, d. h. sie sind durch Zeigerlisten realisierbar.

Hieraus folgt eine Aussage von sehr weitreichender Bedeutung, die auch für den Nutzer eine große Erleichterung bringt und seiner individuellen Denkweise gut angepaßt ist:

Es ist völlig gleichgültig, in welcher Reihenfolge der Nutzer seine Objekte und die zugehörigen Teilobjekte definiert.

Bemerkungen

- Damit wird die individuelle Denkweise des Nutzers gut unterstützt.
- Es ist eine einheitliche Basis für die aktive und passive Arbeit sowohl für den Nutzer als auch für die interne Arbeit im Rechenautomaten geschaffen.
- Die Prüfung der Vollständigkeit der Beschreibung eines Objektes ergibt sich automatisch bei den Prozessen, die alle Angaben über das Objekt benötigen, z. B. graphische Ausgabe, Konservierung (s. 3.2.). Fehlende Angaben können dann leicht über den Dialog ergänzt werden.
- Im Gegensatz zur Denkweise in den üblichen Programmiersprachen (soll algorithmische Denkweise genannt werden) kann hier von der strukturellen Denkweise gesprochen werden.
- Die Aussage gilt nicht nur für die Digitalgraphik sondern auch für andere strukturierte Prozesse.

Diese Art der Abspeicherung von graphischen Werten ermöglicht auch die automatische Nutzung von externen Speichern (automatisches Ein- und Ausladen, kurz: E/A). Das E/A arbeitet sehr praktikabel bei folgender Strategie:

1. Nur die Werte der graphischen Variablen werden dem E/A unterzogen.
2. Es sind so viele Werte auszuladen, damit eine Mindestmenge an Freispeicher für die Struktur gewonnen wird. Der Umfang dieser Menge ist eine systemabhängige Größe.
3. Beim Einladen des gewünschten Wertes werden u. U. noch weitere mit eingeladen, nämlich alle die, die im gleichen Block auf dem externen Speicher abgespeichert sind.

Es kann passieren, daß durch das Einladen das Ausladen angesprochen wird. Deshalb sollte durch einen Aufruf des Ausladens mehr Freispeicher geschaffen werden, als bei einem Aufruf des Einladens benötigt wird. Dadurch wird in starkem Maße das sogenannte "Flattern von Seiten" (page flutter) verhindert.

Andere Strategien des E/A (z. B. alle Informationen stehen auf dem externen Speicher und nur die Informationen sind im Arbeitsspeicher, die gerade benötigt werden) sind unter den gegebenen Bedingungen aufwendiger.

3.2. Konservierung von Strukturteilen

Die in 3.1. beschriebene Speicherung bedeutet auch, daß sich bei Änderung eines Wertes einer Variablen alle die Zeichnungen bzw. Bilder ändern, bei denen die zugehörigen Variablen der Ausgabeanweisung (s. 2.4.) einen direkten oder indirekten Bezug zur geänderten Variablen haben. Soll nun der Zustand einer Speicherstruktur bzw. die vollständige Beschreibung eines Objektes mit allen seinen Teilobjekten zu einem vorgegebenen Zeitpunkt festgehalten werden, so ist die Konservierung zu verwenden.

Bei der Konservierung werden alle Informationen eines Objektes aus der Speicherstruktur auf einen zusammenhängenden Speicherbereich gesammelt und als eine Konserve auf einem Speichermedium archiviert. Im gleichen oder einen anderen Programmjob kann dann die Konserve wieder eingelesen werden. Dabei entsteht das sogenannte Namenproblem: Es kann passieren, daß die in der Konserve verwendeten Variablennamen im vorhandenen Programm nicht mehr aktuell sind bzw. schon anderweitig vergeben sind. Durch Angabe von Umbenennungen können dann die gewünschten Namen eingesetzt werden.

Bei der Konservierung einer gesamten Struktur tritt das Namenproblem nicht auf. Hierbei kommt man schon mit den vorhandenen Mitteln der Aus- und Eingabe in den Programmiersprachen aus.

3.3. Unterprogrammtechnik mit graphischen Variablen

Die Unterprogrammtechnik mit graphischen Variablen erweist sich als sehr tragfähig bei der Schaffung problemorientierter Bausteine.

Beim Aufruf eines Unterprogramms mit graphischen Variablen (graphisches UP) liefert die Zuordnung von aktuellen zu formalen Parametern keine Schwierigkeiten; es ist die Variablenliste des graphischen UP aufzubauen, wobei die formalen Parameter den Vermerk EP erhalten und die in Abb. 4 strichlierten Verweise hergestellt werden. Variablen, die formale Parameter sind und im UP einen Wert zugewiesen bekommen, werden mit AP markiert. Diese beiden Markierungen sind beim Rücksprung aus dem UP für die Ver-

4. Graphischer Dialog

Wie schon eingangs erwähnt, fordert die graphische Ausgabe zum Dialog heraus. Hier sollen einige Möglichkeiten des Dialogs angegeben werden, die sich aus den bisherigen Ausführungen herleiten lassen. Daraus ergibt sich eine Einteilung des Dialogs in der Digitalgraphik. Es wird nicht auf die Struktur und Arbeitsweise der zugehörigen Programmsysteme eingegangen, da dies Gegenstand weiterer Vorträge der Tagung ist.

4.1. Dialog in der graphischen Ausgabe

Ausgangspunkt ist eine Zeichnung, die auf einem graphischen Gerät dargestellt wurde, das interaktive Arbeit ermöglicht (z. B. aktives graphisches Bildschirmgerät). Auf Grund dieser Zeichnung (i. w. Bild) will der Nutzer Veränderungen vornehmen. Die graphische Ausgabe wird durch zwei Anweisungen (s. 2.4.) angestoßen, für die hier eine formalisierte Schreibweise angegeben wird.

Ausgabeanweisung: DARST gerätemark. marke, ausgabeliste

z. B. für Bildschirm BS als 'gerätemark.', so daß folgt

DARST BS marke, ausgabeliste

Zusatzanweisung: marke ZUINFO (menge der zus.-inform.)

Beispiel: DIGRA 73 - System

- 12 Elemente in 'menge der zus.-inform.' für Darstellungskomplex Graphen.
- Insgesamt für alle Darstellungskomplexe 33 Elemente, die jedoch nicht alle zur gleichen Zeit benutzt werden können.

Hieraus lassen sich einzelne Etappen für den Dialog herleiten, wobei jede einzelne entsprechende Effekte erzielt.

1. Etappe: Änderungen der Elemente aus der Menge der Zusatzinformationen in der Zusatzanweisung

Hier beziehen sich die Dialogänderungen auf einen kleinen fest vorgegebenen Bereich. Es lassen sich damit gute Ergebnisse in der Aussagekraft der Bil-

der erreichen (z. B. 'Spaziergang' um einen Gebäudekomplex bei Änderungen des Projektionszentrums und der Projektionsebene). Die Arbeit läuft folgendermaßen ab:

1. Darstellung der Zusatzinformationen als Menü (bei geringer Anzahl) bzw. als Menübaum (bei größerer Anzahl).
2. Auswahl der entsprechenden Zusatzinformation und Angabe der gewünschten Änderung (Nutzeraktivität).
3. Eintragung der Änderung im vorgesehenen Speicherbereich für die Menge der Zusatzinformationen.
4. Wird eine neue graphische Ausgabe gewünscht?
 - 4.1. nein: gehe nach 2.
 - 4.2. ja : gehe nach 5.
5. Neue Ausgabe des Bildes unter Beachtung aller gegebenen Veränderungen.
6. gehe nach 1.

Bemerkung: Die wiederholte Ausgabe kann zeitlich gekürzt werden, wenn die Ausgabeliste in einer für die Ausgabe aufbereiteten Form vorliegt: Man erhält dadurch schnellere Reaktionszeiten.

2. Etappe: Änderungen in der Ausgabeliste der Ausgabeanweisung

Beschränken sich die Änderungen nur auf die Auswahl der Elemente in der vorhandenen Ausgabeliste, so kann es gegenüber der 1. Etappe noch Verbesserungen in den Reaktionszeiten geben, da weniger Bildinformationen zur Ausgabe kommen.

Haben die Änderungen die Aufnahme neuer Elemente in die Ausgabeliste zur Folge, so kann die wiederholte graphische Ausgabe nicht gekürzt werden, so daß größere Reaktionszeiten entstehen können.

3. Etappe: Einflußnahme auf einzelne Algorithmen im Prozeß der graphischen Ausgabe

So sind im Prozeß der Darstellung von Körpern Algorithmen notwendig (z. B. von Bestimmung Konturkurven oder der verdeckten Kanten), bei denen Angaben zum Auflösungsvermögen einer Zeichnung wesentlichen Ein-

fluß auf die Rechenzeit (aber nicht immer auf die Qualität des Bildes) haben. Diese Angaben über den Dialog mitgeteilt, können für konkrete Beispiele große Vorteile bringen.

Zur Erhöhung der Anschaulichkeit der Darstellung von Graphen kann der Aussparungsalgorithmus benutzt werden, in dem die Angabe der Aussparungsbreite ebenfalls über Dialog erfolgen kann.

Die einzelnen Etappen sollen hier immer als Erweiterungen der vorhergehenden angesehen werden, so daß der programmmäßige Umfang von Etappe zu Etappe größer wird, aber damit auch der Komfort des Dialoges steigt.

4.2. Dialog auf der Speicherstruktur

Im 3. Abschnitt wurde schon auf die Bedeutung der Datenstrukturen für die Digitalgraphik und ihre Auswirkungen (strukturelles Denken) hingewiesen. Für den Dialog mit graphischen Werten ergibt sich ebenfalls eine einfache Arbeitsweise, da bei der Abarbeitung einer Dialoganweisung nicht ihre Umgebung beachtet werden muß, d. h. die Änderung eines graphischen Wertes ist nicht abhängig von den vorhergehenden Aktivitäten mit anderen Werten.

Die Dialogform kann sehr gut mit der in der graphischen Ausgabe verknüpft werden, z. B. wird während der Ermittlung der Bildinformationen festgestellt, daß eine Variable in der Ausgabeliste keinen Wert besitzt. So kann dieser über den Dialog "nachgereicht" und die Ausgabe ordnungsgemäß fortgesetzt werden.

5. Schlußfolgerungen

Durch die Schaffung graphischer Sprachen (s. auch /1/) wurde ein rechen-technisches Hilfsmittel zur Beschreibung der geometrischen Gestalt eines Objektes entwickelt. Damit wird es möglich, aus einer Beschreibung beliebig viele unterschiedliche Bilder und Ansichten von einem Objekt zu erzeugen. Dies ist besonders im 3-Dimensionalen von großer Wichtigkeit und muß erst in den Denk- und Arbeitsweisen der Anwender seinen Niederschlag finden und voll aufgenommen und verarbeitet werden.

Das Variablenprinzip kann ebenfalls in die geometrische Betrachtungsweise mit allen Konsequenzen (graphische Ausdrücke, Ergibtanweisung, Unterprogrammtechnik) übertragen werden, auch wenn noch zusätzlich Dialogfähigkeit mit den definierten Größen gefordert wird.

Die Forderung der Dialogfähigkeit verlangt für die graphischen Werte eine Speicherstruktur von Listencharakter. Dies führt zur strukturellen Denkweise und einer leichten Ausführung des Dialogs auf diesen Strukturen, erfordert aber auch die Mittel der Konservierung.

In der Digitalgraphik ist der Aufwand zur Erstellung einer Zeichnung bzw. eines Bildes bedeutend höher (gültig besonders im 3-Dimensionalen) als der zur Ermittlung graphischer Werte.

Die Möglichkeiten der interaktiven Arbeit an einer konkreten Problemstellung sind gründlich zu analysieren, um richtige Entscheidungen für die Systemnutzung treffen zu können.

Die graphische Sprache und die Dialogsprache sollen eine Einheit bilden, da für den graphischen Dialog immer Vorarbeiten erforderlich sind, für die die graphische Sprache notwendig ist. Ein graphischer Dialog aus dem 'Nichts' heraus ist nicht vertretbar.

Die gemachten Ausführungen deuten schon an, daß ein digitalgraphisches Programmsystem mit den eingangs formulierten Eigenschaften schnelle Rechner benötigt, deren Betriebssysteme folgende Anforderungen erfüllen sollen:

- Möglichkeiten des Multiprogrammings,
- Hilfsmittel zur Erstellung von Programmsegmenten,
- Hilfsmittel zur Errichtung von System- und Nutzerbibliotheken,
- günstige Speichernutzung.

Ausblick: Durch die Nutzung von Datenstrukturen wird die Entwicklung in Richtung der offenen Systeme gefördert. Schon in der in dieser Arbeit ange-deuteten Struktur ist die Typangabe nicht von Bedeutung, so daß auf der Struktur mit beliebigen Typen gearbeitet werden könnte. Jedoch sind hierzu noch weitere Untersuchungen auch im Hinblick auf die graphische Ausgabe und den graphischen Dialog notwendig.

Literatur

- /1/ **Nake, F. and Rosenfeld, A. (ed.):** Graphic Languages; Proceedings of the IFIP Working Conference on Graphic Languages, Vancouver, Canada, May 22-28, 1972,
North-Holland Publ. Comp., Amsterdam, London (1972).
- /2/ **Newman, W. M.:** A Prototype Low Cost Single-User Graphic System,
in: /1/, S. 291 - 297.
- /3/ **Kotzauer, A.:** Graphische Programmsysteme.
Rost. Math. Koll., H. 1, S. 61 - 78 (1976).

Eingegangen: 30. 11. 1976

Anschrift des Verfassers

Dr. rer. nat. Adolf Kotzauer
Wilhelm-Pieck-Universität Rostock
Sektion Mathematik
DDR - 25 Rostock, Universitätsplatz 1

Einheitliches Konzept der Realisierung der Kommunikation Mensch/EDVA in Dialogsystemen

1. Einleitung

Die dialogfähigen Datenverarbeitungssysteme (DDVS) kann man in zwei Gruppen einteilen: programmiersprachenorientierte und menüorientierte Systeme.

In programmiersprachenorientierten Systemen dominiert die Eingabe von Sprachelementen höherer Programmiersprachen in direkter Kommunikation mit dem DDVS. Wir sprechen von On-Line-Programmierung oder Dialogprogrammierung. Vertreter dieser Gruppe von DDVS sind u. a. CTSS und JOSS /11/, BASIC /6/, MULTICS /4/, DIALOG BESM-6 /5/ und MS /10/.

Menüorientierte Systeme sind aufs engste mit der graphischen Datenverarbeitung und dem Einsatz aktiver graphischer Bildschirmgeräte verbunden. Diese Systeme zeichnen sich dadurch aus, daß der Nutzer am Bildschirm Schlüsselwörter als Sprachelemente, als Elemente eines Menüs, angeboten bekommt, davon einige auswählt, durch zusätzliche Informationen ergänzt und vom System zur richtigen Benutzung geführt wird. Dadurch entfällt das Formulieren vollständiger Programmteile, die Fehlermöglichkeiten werden reduziert, und die Arbeit des Nutzers verläuft stärker problemorientiert. Die meisten digitalgraphischen Programmsysteme, so auch der Dialog des DIGRA 73-Systems, gehören zur Gruppe der menüorientierten Systeme. Eine Zusammenstellung und eine gewisse Analyse derartiger Systeme finden wir in /9/. Es fällt nicht schwer, die Gruppe der programmiersprachenorientierten Systeme als Spezialfall der menüorientierten aufzufassen. Gegenstand der nachfolgenden Betrachtungen sind menüorientierte Systeme. Für

die praktische Realisierung stehe ein aktiver Bildschirm mit einer alphanumerischen Tastatur, einem Lichtstift und einer Positioniereinrichtung für ein Lichtkreuz zur Verfügung.

2. Trennung von Verarbeitung und Kommunikation

Das Ziel der Arbeit des Menschen im Dialog ist die Steuerung der Bearbeitung von Daten bzw. Datenstrukturen bestimmter Objekte zur Lösung einer Aufgabe. Die Arbeit eines DDVS kann in zwei Teile zerlegt werden:

- Steuerung der Bearbeitung und Bearbeitung der Datenstrukturen sowie Ausgabe von Resultatinformationen in Abhängigkeit eines Satzes von Parametern (Eingabesatz).
- Steuerung des Aufbaus eines Eingabesatzes für die Bearbeitung durch das System infolge von Aktionen des Nutzers am Terminal.

Die Trennung beider Teile und ihre selbständige Realisierung und Behandlung stellt eine Grundlage eines einheitlichen Konzepts der Realisierung der Kommunikation Mensch/EDVA dar.

Während der Arbeit des ersten Teils ist der Nutzer im wesentlichen Betrachter. Er wartet auf Resultate und bewertet sie bezüglich der Lösung seiner Aufgabe. Im zweiten Teil ist der Nutzer aktiv und baut mit Unterstützung des DDVS und in Abhängigkeit des erhaltenen Resultats einen Eingabesatz für eine erneute Bearbeitung der Datenstrukturen auf. Somit unterscheiden wir Haupt- und Miniinteraktionspaar.

Das Hauptinteraktionspaar besteht aus den beiden Anteilen

- Formulierung eines Eingabesatzes
- Durchführung der Bearbeitung und Ausgabe eines Ausgabesatzes.

Das Miniinteraktionspaar besteht aus den Komponenten

- Elementaraktion des Nutzers
- Reaktion des Systems auf die Elementaraktion des Nutzers.

In einfachen Dialogsystemen können beide Interaktionspaare zusammenfallen. Bezeichnen wir die Ausführung von Hauptinteraktionspaaren als Hauptdialog, so beschränkt sich der Hauptdialog auf den Austausch von Ein- und Ausgabesätzen zwischen Nutzer und DDVS.

Die Ausführungen von Miniinteraktionspaaren bezeichnen wir als Minidialog. Die Führung des Minidialogs ist ein Hilfsprozeß für den Hauptdialog. Der Minidialog realisiert aber die eigentliche Kommunikation auf der Basis von Elementaraktionen, die der Nutzer ausführt, und den entsprechenden Reaktionen des Systems. Das Ziel des Minidialogs ist der Aufbau von Eingabesätzen und die richtige Initialisierung der Bearbeitung der Daten.

Das Prinzip der Trennung von Haupt- und Minidialog führt zum Aufbau und zur zweckmäßigen Realisierung von:

- P-Systemen (Parametergesteuerte Systeme) für die Steuerung der Organisation der Bearbeitung von Datenstrukturen und der Ausgabe eines Ausgabesatzes in Abhängigkeit eines Eingabesatzes
- K-Einrichtungen für die Realisierung der Systemunterstützung zur Führung des Minidialogs mit dem Ziel des Aufbaus eines Eingabesatzes für ein P-System.

Über P-Systeme wurde bereits mehrfach gesprochen (vgl. /7/ und /8/). Wesentlich ist die Tatsache, daß P-Systeme völlig unabhängig von der Art der Erzeugung der Eingabesätze arbeiten und damit von der Dialogführung und dem konkreten Terminal unabhängig sind. Es ist lediglich ein Speicherbereich für den Zugriff auf einen Ein- und Ausgabesatz (x und y) zu vereinbaren, über den der Austausch verläuft. Diesen Speicherbereich werden wir als Kontaktfläche (KF) bezeichnen. Zur Veranschaulichung der Zusammenarbeit von P-System und K-Einrichtung diene Abb. 1. Die K-Einrichtung liest den Ausgabesatz y von der Kontaktfläche, stellt diesen und das Menü auf dem Bildschirm dar und erzeugt im Minidialog einen Eingabesatz x auf der Kontaktfläche. Ist ein Eingabesatz x aufgebaut, wird das P-System aufgerufen.

3. Menü - ein wesentlicher Bestandteil der Kommunikationshilfsmittel

Der Minidialog wird mit Hilfe des auf dem Bildschirm dargestellten Menüs geführt. Damit ist das Menü ein wesentlicher Bestandteil der Kommunikationshilfsmittel.

Ein Menü ist ein geordnetes Tupel von Schlüsselwörtern, die als Licht- oder Funktionstasten verwendet werden. Jedes Schlüsselwort weist den Nutzer auf den Verwendungszweck hin. Wir werden hier der Einfachheit halber nicht zwischen Funktions- und Lichttasten unterscheiden, nur von Lichttasten sprechen und jedes Schlüsselwort (Element) des Menüs als Lichttaste bezeichnen. Jede Lichttaste besitzt ein Prädikat, das über die spezielle Funktion der Lichttaste entscheidet. Das Drücken einer Lichttaste (Aufsetzen des Lichtstiftes auf das Schlüsselwort oder Drücken der entsprechenden Funktionstaste) führt zur Interpretation der Lichttaste in Abhängigkeit von ihrem Prädikat und zur Einleitung neuer Elementaraktionen.

Die Lichttasten mit ihren Prädikaten und eine Indexfunktion, die der Zuordnung von Eingabesignalen zu den Komponenten des Eingabesatzes des P-Systems dient, legen die Gesamtheit der Kommunikationshilfsmittel einer K-Einrichtung fest. Man kann die K-Einrichtungen stets so aufbauen, daß nur die Kommunikationshilfsmittel als variabler Bestandteil der K-Einrichtung problemangepaßt gestaltet werden.

Alle anderen Bestandteile können fest definiert werden (problemunabhängig). Diese Bestandteile sind

- Kommunikationsbasis
- Kommunikationsprogramm.

Das heißt, für die Realisierung der Kommunikation sind einmalig die Kommunikationsbasis und das Kommunikationsprogramm festzulegen, und die Kommunikationshilfsmittel können problemangepaßt aufgebaut werden. In der Regel besitzt jedes DDVS mehrere K-Einrichtungen. Der Übergang von einer zur anderen wird ebenfalls über die Lichttasten realisiert. Jeder K-Einrichtung wird ein Tupel von nachfolgenden K-Einrichtungen zugeordnet. Jede K-Einrichtung besitzt ihr eigenes Menü. Alle K-Einrichtungen seien von Eins beginnend durchnummeriert. Eine K-Einrichtung kann in der vorher diskutierten Form mit einem P-System gekoppelt werden. In einem DDVS ist

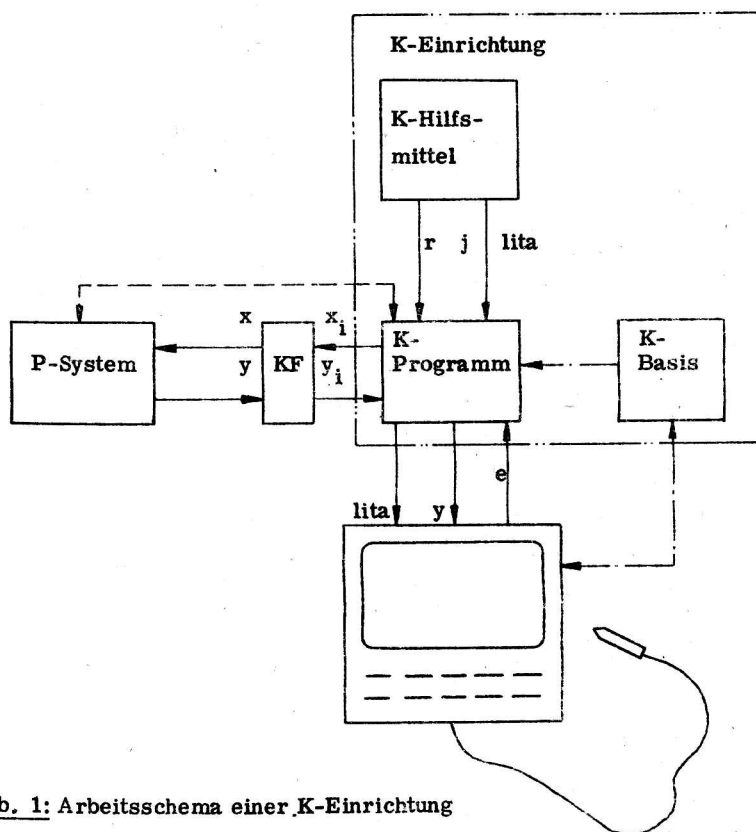


Abb. 1: Arbeitsschema einer K-Einrichtung

entweder eine K-Einrichtung oder ein P-System aktiv. Jeder K-Einrichtung und jedem P-System entspricht ein Zustand des DDVS. Das DDVS nimmt den Zustand an, wenn die entsprechende K-Einrichtung oder das P-System aktiv sind. Wir bezeichnen den Zustand des DDVS mit den Namen der K-Einrichtungen bzw. P-Systeme.

4. Kommunikationsbasis

Die Kommunikationsbasis (KB) legt die grundlegenden Komponenten für die Gestaltung der Kommunikationshilfsmittel und das Kommunikationsprogramm fest. Die KB wird wesentlich durch die Elementaraktionen, die der Nutzer am Bildschirm ausführen kann, bestimmt. KB besteht aus fünf endlichen Mengen:

- E_K Menge der möglichen Eingabesignale, Ergebnisse der möglichen Elementaraktionen
- PW_K Menge primärer Wirkungen der Eingabesignale
- R_K Menge spezieller Register
- Z_K Menge innerer Zustände, die eine K-Einrichtung annehmen kann
- P_K Menge von Prädikaten zur Bewertung der Lichttasten

Formal können wir notieren

$$KB = \langle E_K \quad PW_K \quad R_K \quad Z_K \quad P_K \rangle .$$

KB ist gegeben, wenn die 5 Mengen festgelegt wurden.

Zur Festlegung der einzelnen Mengen gehen wir davon aus, daß die folgenden Elementaraktionen ausführbar sind:

- Drücken einer Lichttaste
- Identifizieren eines Teilbildes mit dem Lichtstift
- Auswählen eines Namens eines Teilbildes aus einer dargestellten Namensliste (Teil eines Ausgabesatzes)
- Eingeben einer Zeichenfolge
- Eingeben eines numerischen Wertes
- Positionieren eines Lichtkreuzes auf eine beliebige Stelle des Bildschirms.

Jede Elementaraktion führt zu einem Eingabesignal und einer primären Wirkung.

Die Menge der Eingabesignale sei

$$E_K = \{ LT, IT, IN, ET, EN, PO \} .$$

Dabei bedeuten:

LT	Lichttaste wurde gedrückt
IT	Teilbild wurde mit dem Lichtstift identifiziert
IN	Name eines Teilbildes wurde gewählt
ET	Zeichenfolge wurde eingegeben
EN	numerischer Wert wurde eingegeben
PO	Lichtkreuz wurde neu positioniert.

Die Menge der primären Wirkungen der Eingabesignale wird festgelegt durch

$$PW_K = \{ \alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6 \} .$$

Jedes Eingabesignal hat genau eine Wirkung. Jede primäre Wirkung speichert in ein Register aus R_K einen Wert und führt zur Reaktion auf dem Bildschirm wie Blinken, Darstellen des Wertes u. ä. Die primäre Wirkung erfolgt sofort nach Ausführung der Elementaraktion bzw. nach Realisierung eines Eingabesignals aus E_K . Wir geben die primären Wirkungen in Tabelle 1 an.

$e \in E_K$	PW_K	Bedeutung	Kommentar
LT	α_1	NU: = j	j-te Lichttaste wurde betätigt
IT	α_2	TENI: = identifizator	Teilbild mit dem Identifikator "identifikator" wurde identifiziert
IN	α_3	TENI: = name	Name "name" eines Teilbildes wurde ausgewählt
ET	α_4	TENI: = zeichenfolge	Text "zeichenfolge" wurde eingegeben
EN	α_5	TENI: = convert (zahl)	Numerischer Wert "zahl" wurde eingegeben und konvertiert
PO	α_6	KOOR: = (kox, koy)	Ort des Lichtkreuzes wurde verändert und hat danach die Koordinaten (kox, koy)

Tabelle 1: Primäre Wirkungen der Eingabesignale

Die Menge der speziellen Register ist in unserem Fall

$$R_K = \{ \text{KOOR, NU, TENI} \} .$$

Dabei dienen

KOOR zur Aufnahme der Koordinaten des Lichtkreuzes,

NU zur Aufnahme der Nummer der Lichttaste,

TENI zur Aufnahme einer Zeichenfolge, eines numerischen Wertes, eines Teilbildnamens oder -identifikators.

Die Menge der inneren Zustände der K-Einrichtung ergibt sich zu

$$Z_K = \{ \text{lt, it, in, et, en, un, üg} \} .$$

Jedes Element ist ein Zustand, den eine K-Einrichtung annehmen kann. Der Zustand gibt an, welche Elementaraktion vom Nutzer am Bildschirm erwartet wird, oder in welchem Zustand der Nutzer die K-Einrichtung versetzt hat.

Dabei bedeuten:

- lt Warten auf Betätigen einer Lichttaste
- it Warten auf Identifizieren von Teilbildern
- in Warten auf Auswählen von Namen
- et Warten auf Eingeben einer Zeichenfolge
- en Warten auf Eingeben eines numerischen Wertes
- un Arbeit eines P-Systems wurde unterbrochen
- üg Vorbereitung des Übergangs auf eine beliebige K-Einrichtung.

Die Menge der möglichen Prädikate kann unterschiedlich festgelegt werden und ist den jeweils praktischen Gegebenheiten anzupassen.

Wir vereinbaren die Menge wie folgt:

$$P_K = \{r1, r2, \dots, r11\}$$

Die Bedeutung der Prädikate ist:

- r1 Übergang auf eine nachfolgende K-Einrichtung
- r2 Übergang zur Eingabe von numerischen Werten mittels alphanumerischer Tastatur
- r3 Eingabe von zwei numerischen Werten aus KOOR
- r4 Übergang zur Eingabe von Zeichenfolgen
- r5 Übergang zur Teilbildidentifizierung
- r6 Korrektur der Teilbildidentifizierung
- r7 Übergang zur Auswahl von Teilbildnamen
- r8 Korrektur der Auswahl von Teilbildnamen
- r9 Übergang auf eine beliebige K-Einrichtung des Systems
- r10 Aufruf des P-Systems
- r11 Unterbrechung der Arbeit des P-Systems.

5. Vereinbarung von Kommunikationshilfsmitteln

In Abschnitt 3 wurden bereits die Kommunikationshilfsmittel (KH) als ein Bestandteil einer K-Einrichtung beschrieben, der sich von K-Einrichtung zu K-Einrichtung verändert und über den die Kommunikation gesteuert wird. Bestandteile von KH sind

- eine Folge von Paaren $(lita, r)$, wobei $lita$ das Schlüsselwort einer Lichttaste (Zeichenfolge) und $r \in P_K$ das Prädikat der Lichttaste sind. Die gesamte Folge schreiben wir als Array $(lita, r) [1 : M]$ ($M \geq 1$) und das j -te Element in der Form $(lita, r) [j]$ bzw. die einzelnen Komponenten $lita [j]$ und $r [j]$.
- eine Indexfunktion $I(j, r [j], L)$, die jedem Paar $(lita, r) [j]$ in Abhängigkeit eines inneren Zählers L eindeutig einen Index zuordnet, sofern $r [j] \in \{r_2, r_3, r_4, r_5, r_6, r_7, r_8\}$ ist. Der berechnete Index gibt eine Zelle auf der Kontaktfläche an, in die ein Wert als Komponente des Eingabesatzes des P-Systems gespeichert wird.

Es empfiehlt sich, und wir vereinbaren für alle weiteren Ausführungen, die Paare $(lita, r) [j]$ ($j = 1(1)M$) so zu belegen, daß alle Paare mit $r [j] = r_1$, von $j = 1$ beginnend, fortlaufend stehen. Damit ist der ersten Lichttaste die erste nachfolgende, der zweiten die zweite usw. K-Einrichtung zugeordnet. Führt man weitere, von der Struktur des Eingabesatzes des P-Systems abhängige Festlegungen standardmäßig ein, kann die Indexfunktion einfach und für alle K-Einrichtungen in gleicher Weise festgelegt werden.

6. Kommunikationsprogramm

Das Kommunikationsprogramm (KP) realisiert die Steuerung der K-Einrichtung, d. h. die Steuerung der Kommunikation. In Abhängigkeit der konkreten KH kann die Arbeit des aktiven KP wie folgt dargestellt werden. Zur Veranschaulichung diene Abb. 1. Auf dem Bildschirm sind der Ausgabesatz des P-Systems und alle Lichttasten $(lita [j], j = 1(1)M)$ dargestellt, und KP wartet auf ein Eingabesignal. Wird eine Lichttaste betätigt, erhält das KP das Eingabesignal LT, und im Register NU befindet sich die Nummer j der Lichttaste. Aus KH (auf einem entsprechenden Speicherbereich abgespei-

chert) wird $r[j]$ gelesen und die Verarbeitung in Abhängigkeit von $r[j]$ vorgenommen. Ist die Verarbeitung abgeschlossen, wartet KP auf ein neues Eingabesignal. Zweckmäßiger Weise kann man ein KP durch drei Entscheidungstabellen (zur Verwendung von Entscheidungstabellen vgl. /3/), aufruf, nei und verarb beschrieben:

aufruf realisiert den Aufruf der KP

nei realisiert das Warten auf ein neues Eingabesignal und den nächsten Verarbeitungsschritt

verarb verarbeitet ein Eingabesignal LT in Abhängigkeit des Prädikats.

Die Einzelheiten sind den Tabellen 2, 3 und 4 zu entnehmen. Dabei werden außer den bereits eingeführten die folgenden Bezeichnungen verwendet:

Z_V Menge aller Zustände des Systems (K-Einrichtungen und P-Systeme)

K betrachtete K-Einrichtung ($K \in Z_V$)

P(K) P-System, das mit K verbunden ist ($P(K) \in Z_V$)

$K[j]$ j-te Nachfolger von K

K(NU) NU-te K-Einrichtung des DDVS

EX [1:N] Speicherbereich der Kontaktfläche für einen Eingabesatz.

Bei Beginn der Abarbeitung von nei wird die Ausführung der primären Wirkung des Eingabesatzes vorausgesetzt.

In den einzelnen Aktionen der Entscheidungstabellen treten Mitteilungen auf, die hier nicht weiter spezifiziert werden. Diese Mitteilungen erscheinen auf dem Bildschirm und geben dem Nutzer eine Information über die eingetreten-
de Situation.

tab aufruf

Nr	$z_V \in Z_V$	$z \in Z_K$	$z_V \in Z_V$	$z \in Z_K$	Aktion	GO TO
1	P(K)	\neq un	K	lt	EX löschen; Darstellung des Ausgabesatzes des P-Systems;	nei
2	P(K)	un	K	lt	Darstellung von Zwischenresultaten und eventueller Startmöglichkeiten;	nei
3	K	-	-	lt	Darstellung von lita [1:M] für K;	nei

Tabelle 2: Entscheidungstabelle aufruf von KPtab nei

Nr	$e \in E_K$	$z \in Z_K$	$z_V \in Z_V$	$z \in Z_K$	Aktion	GO TO
1	LT	-	-	-	-	verarb
2	IT	it	-	-	L := L + 1; J := I(NU, r [NU], L); EX [J] := TENI;	nei
3	IT	\neq it	-	-	Mitteilung;	nei
4	IN	in	-	-	wie bei 2.;	nei
5	IN	\neq in	-	-	Mitteilung;	nei
6	ET	et	-	-	wie bei 2.;	nei
7	ET	\neq et	-	-	Mitteilung;	nei
8	EN	en	-	-	wie bei 2.;	nei
9	EN	\neq en $\wedge \neq$ üg	-	-	Mitteilung;	nei
10	EN	üg	K(TENI)	lt	EX löschen;	aufruf
11	PO	-	-	-	-	nei

Tabelle 3: Entscheidungstabelle nei

Nr.	r [NU]	$z \in Z_K$	$z_V \in Z_V$	Aktion	GO TO
1	r1	lt	K [NU]	EX löschen;	aufruf
2	r2	en	-	L: = 0;	nei
3	r3	-	-	J: = I(NU, r3, 0); EX [J] := KOOR [1]; EX [J+1] := KOOR [2]; Darstellung der Werte;	nei
4	r4	et	-	L: = 0;	nei
5	r5	it	-	L: = 0;	nei
6	r6	it	-	L:=L-1, falls $L \geq 1$, sonst Fehlermeldung; J:=I(NU, r6, L+1); Löschen EX [J]; Dar- stellung der Korrektur;	nei
7	r7	in	-	L:=0; Darstellung (be- sondere) der Namenliste und Korrekturmöglich- keiten;	nei
8	r8	in	-	L:=L-1, falls $L \geq 1$, sonst Fehlermeldung; J:=I(NU, r8, L+1); Löschen EX [J]; Dar- stellung der Korrektur;	nei
9	r9	üg	-	Mitteilung über Eingabe der Nummer der K-Ein- richtung;	nei
10	r10	lt	P(K)	-	P(K) ¹⁾
11	r11	un	-	Mitteilung; Unter- brechung von P(K);	aufruf

Tabelle 4: Entscheidungstabelle verarb

¹⁾ Es erfolgt der Aufruf des Steuerprogramms des P-Systems.

7. Praktische Schlußfolgerungen

Aus den vorangegangenen Ausführungen ergibt sich ein einheitliches Konzept für die Gestaltung der Systemunterstützung bei der Kommunikation. Die Bestandteile für die Systemunterstützung sind (die einmalig festgelegten) Kommunikationsbasis und Kommunikationsprogramm und die problemangepaßte und damit variable Gesamtheit aller Kommunikationshilfsmittel. Für den Aufbau der Systemunterstützung ergibt sich das folgende Verfahrensschema:

- Definition der Kommunikationsbasis durch
 - - Festlegung der Menge aller möglichen Eingabesignale
 - - Festlegung der speziellen Register
 - - Festlegung und Realisierung der primären Wirkung für jedes Eingabesignal
 - - Festlegung der inneren Zustände
 - - Festlegung aller möglichen Prädikate für Lichttasten (analog für Funktionstasten)
- Ausarbeitung des Kommunikationsprogramms (das vorangegangene Beispiel mit den drei Entscheidungstabellen kann als Muster dienen)
- Bestimmung der Gesamtheit aller Kommunikationshilfsmittel (Lichttasten mit Prädikaten und Indexfunktion zur Abspeicherung der Eingabewerte, der Komponenten des Eingabesatzes).

Die Realisierung der ersten beiden Verfahrensschritte entspricht der Schaffung der Grundsoftware für die Führung des Dialogs. In der Software für die Kommunikation im System GIPS und DIGRA 73 finden wir wesentliche Grundzüge der vorgeschlagenen Prinzipien (/1/ und /2/). Für neue Probleme ist nur der dritte Verfahrensschritt zu wiederholen.

Es ist offensichtlich, daß mit der Verwirklichung der angegebenen Prinzipien eine einheitliche Beschreibungsform für die Dialogsoftware gegeben ist und Erweiterungen der Systemunterstützung für die Kommunikation klar definiert werden können.

Die Beschreibung des Systems für den Nutzer kann ebenfalls auf der Basis der Realisierung der Prinzipien einheitlich und eindeutig vorgenommen werden. Dieser Teil ist im wesentlichen eine Beschreibung der Kommunikationsmittel (aller K-Einrichtungen) mit ihren Wirkungen. Wir hatten alle K-Einrichtungen von Eins beginnend, fortlaufend durchnummeriert. Das überträgt sich naturgemäß in eindeutiger Weise auf die Kommunikationsmittel. Sind n K-Einrichtungen im DDVS vorhanden, so sind n Tabellen der nachfolgenden Form (Tabelle 5) anzugeben. Den Arbeitsbeginn des DDVS legen wir mit der ersten K-Einrichtung fest. Für das Ausfüllen der Tabellen lassen sich weitere Standards einführen.

Nr. K-Einrichtung:					
Nr. d. Lichttaste	Schlüsselwort Lichttaste	Prädikat	Übergang zu	Wirkung/Eingabewert	Kommen-tar
1					
2					
3					
4					
.					
.					
.					

Tabelle 5: Schema der Beschreibung der Kommunikation

Das Ausfüllen der einzelnen Felder dürfte klar sein. In den Spalten "Übergang zu" stehen Nr. der K-Einrichtungen, bzw. die Bezeichnung des P-Systems, wohin die Steuerung abgegeben wird, und in "Wirkung/Eingabewert" die Verwendung des Eingabewertes. Es sei lediglich vermerkt, daß die Kommunikationsmittel und mithin auch die ausgefüllten Tabellen 5 von der Struktur des Eingabesatzes des P-Systems abhängen.

Literatur

- /1/ Autorenkollektiv: Programmierhandbuch GIPS.
Institut für Schiffbau - Rechenzentrum, Rostock 1975.
- /2/ Autorenkollektiv: Bearbeitung graphischer Probleme der technischen Vorbereitung im Mensch-Maschine-Dialog mit der Bildschirmtechnik, Programmsysteme GIPS und DIGRA 73.
AUTEVO - Informationsreihe - 6, VEB Carl Zeiß Jena, FLZ AUTEVO, Jena 1975.
- /3/ Busch, H. -J.; Engeliien, M.; Stahn, H.: Entscheidungstabellentechnik - ein erweitertes Konzept mit Eignung zur Beschreibung von Prozessen von diskontinuierlich diskretem Typ - Teil I und II.
Techn. Univers. Dresden, Sektion Informationsverarbeitung, 08-3-75, 08-4-75 (1975).
- /4/ Corbato, F. J.; Vyssotsky, V. A.: Introduction and overview of the MULTICS system.
AFIPS Conference Proc., Vol. 27, p. 335, (1965).
- /5/ Филиппов, В.И.: Язык "Диалог" и его использование. Математическое обеспечение ЕЭСМ - 6, ВЦ АН СССР, Москва, 1975.
- /6/ Kurtz, T. E.; Lochner, K. M.: Supervisory systems for the Dartmouth time-sharing-system.
Computers and Automation, Vol. 14, No. 10, p. 25, (1965).
- /7/ Кучке, К.Х.: Математическая модель систем "Диалог" и язык для ее описания.
Журнал Программирование.

- /8/ Kutschke, K. -H.: Programmstrukturen in Dialogsystemen.
Vortrag auf der Thematischen Konferenz der KINWWT über Methodik
der Programmierung und Programmverifikation, Tagungsberichte,
Dresden 1976.
- /9/ Newman, W. M.; Sproull, R. F.: Principles of interactive computer
graphics
MC Graw - Hill, 1973.
- /10/ Polze, Ch.: Mehrfachzugriffssystem MS (Bericht 4)
- Einführung - ZfR - C - 76.02.
Akademie der Wissenschaften der DDR, Zentrum für Rechentechnik,
Berlin 1975.
- /11/ Wilkes, W. V.: Time-Sharing-Betrieb bei digitalen Rechenanlagen.
BSW B. G. Teubner Verlagsgesellschaft, Leipzig 1970.

Eingegangen: 30. 11. 1976

Anschrift des Verfassers:

Doz. Dr. Karl-Heinz Kutschke

Wilhelm-Pieck-Universität

Sektion Mathematik

DDR - 25 Rostock, Universitätsplatz 1

Ein Beitrag zur Realisierung eines ESER-orientierten interaktiven graphischen Terminals

Die Nutzung von Methoden und Verfahren der Konstruktionswissenschaften für die Praxis erfordert in immer stärkerem Maße den Einsatz geeigneter rechentechnischer Geräte und Programmsysteme. Damit können Voraussetzungen geschaffen werden, den Entwicklungsprozeß vom Entwurf bis hin zur Fertigung durchgehend auf Rechenanlagen durchzuführen, was letztendlich zu einer Verkürzung der Entwicklungszeiten führt.

Eine Analyse und Abstraktion der dabei auftretenden Probleme führt u. a. zu Fragen der 'Konstruktiven Geometrie' (darstellende und Differential-Geometrie), die nur dann befriedigend gelöst werden, falls dafür angepasste rechentechnische Anlagen genutzt werden können. Auf ihnen muß die Möglichkeit bestehen, Konstruktionselemente mit geeigneten Programmiersprachen zu beschreiben und sie zu verändern, wobei neben den notwendigen geometrischen Daten ebenfalls nichtgeometrische Daten - mit im allgemeinen großem Umfang - ständig zur Verfügung stehen sollten, um neben dem geometrischen Gestalten und Dimensionieren auch noch alle technischen und technologischen Vorgänge modellieren zu können. Der unmittelbare manuelle Eingriff in den Entwicklungsprozeß sollte über interaktive geometrie-orientierte periphere Geräte erfolgen.

Um einen Beitrag zur Lösung o. g. Problematik zu leisten, entstand an der TU Dresden, Sektion Mathematik, eine Arbeitsgruppe Digitalgeometrie, deren wesentlichste Zielstellung darin besteht, Methoden und Arbeitsmittel der konstruktiven Geometrie im Zusammenwirken mit EDV-Anlagen für die Bearbeitung natur- und ingenieurwissenschaftlicher Aufgaben zu entwickeln und für die Praxis bereitzustellen.

Erste Erfahrungen auf diesem Gebiet konnten bei der Untersuchung zu Schneckengetrieben (/1/, /2/) und bei der Konzeption und Implementierung einer Programmiersprache für ein 'Digitalgeometrie-orientiertes System' DIGOS (/3/, /4/) gewonnen werden. Die gerätetechnische Grundlage bildete jeweils eine Großrechenanlage mit entsprechenden alphanumerischer Peripherie in Verbindung mit einer (off-line gesteuerten) Zeichenmaschine.

Die nächste Arbeitsetappe beinhaltet den Aufbau einer geometrie-orientierten interaktiven Gerätetechnik auf der Grundlage eines ungarischen graphischen Bildschirmgerätes GD 71 und des Kleinrechnersystems KRS 4201 in Verbindung mit ESER.

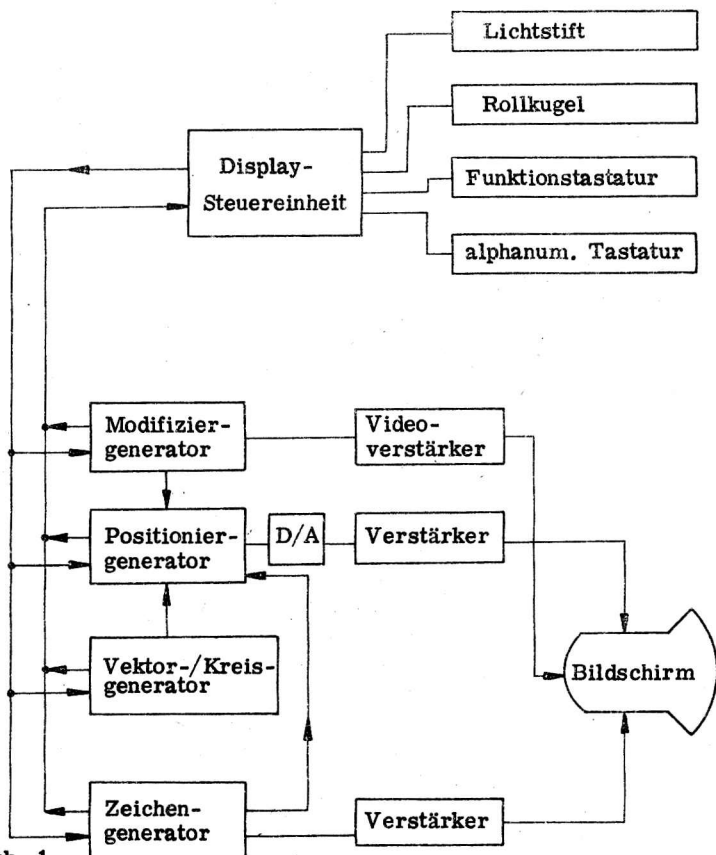


Abb. 1

1. Bildschirmgerät GD 71

Die prinzipielle gerätetechnische Konfiguration des GD 71 zeigt Abb. 1. Die vektororientierte Bildschirmeinheit besteht im wesentlichen aus der Anzeigeeinheit, den einzelnen Generatoren für die Erzeugung von Geraden(-stücken), Kreisbögen und alphanumerischen Zeichen, dem Positioniergenerator zur Positionierung des Lichtstrahles und den Generatoren zur Modifizierung der Strichart.

Der Vektor-/Kreisgenerator erzeugt Strecken/Kreisbögen ausgehend von einem Startwert, gesteuert durch den jeweiligen inkrementalen Zuwachs. Bei der Kreisbogenerzeugung wird außerdem die Umlaufrichtung des Kreises berücksichtigt.

Der Zeichengenerator erzeugt 64 alpha-numerische Zeichen und graphische Symbole in einer 5 x 7 Matrix im ASC II Code. Die Stelle des ersten Zeichens muß positioniert werden, die Positionierung der folgenden Zeichen erfolgt mittels Gerätetechnik. Die Zeichen besitzen eine einheitliche Größe von 5,6 mm x 4 mm. Der Bildschirm umfaßt maximal 32 Zeilen mit je 64 Zeichen.

Der Positioniergenerator ist in der Lage, den Lichtstrahl zu einem beliebigen Punkt des 1024 x 1024 Punktrasters des Bildschirmes zu leiten.

Der Modifiziergenerator steuert den Videoverstärker, wobei die unterschiedlichen Linienarten erzeugt werden.

Zum Bildschirmgerät gehören außerdem die Einrichtungen für den Dialog: eine aus 32 Tasten bestehende Funktionstastatur, eine alphanumerische Tastatur, ein Lichtstift und eine Rollkugel.

Das Display-Programm, das auf dem Bildschirm dargestellt werden soll, besteht aus Displaybefehlen, die sich im Speicher eines Display-Steuerrechners befinden. Das Lesen der Displayfiles erfolgt mittels der Display-Steuereinheit, die den Inhalt des Display-Puffers Wort für Wort über einen nichtprogrammierbaren Kanal den Bildgeneratoren zuführt. Ebenfalls erfolgt

die interne Steuerung der Dialogeinrichtungen mit Hilfe der Display-Steuer-
einheit. Die Display-Steuereinheit stellt somit ein wichtiges Bindeglied zwi-
schen Bildschirmgerät und dem Display-Steuerrechner dar. Ihr gerätetechni-
scher Aufbau muß sowohl die Eigenschaften des Bildschirmgerätes als auch
des Display-Steuerrechners berücksichtigen. Für die Kopplung des ungari-
schen Displaygerätes GD 71 an den DDR-Kleinrechner KRS 4201 bestand die
Aufgabe, eine entsprechende Display-Steuer-(und Kopplungs-)einheit zu ent-
wickeln. Zusätzlich machte sich der Aufbau einer Gerätesteuereinheit (GSE)
für den Anschluß des GD 71 an den KRS 4201 erforderlich. Die Konzeption für
diese GSE erfolgte derart, daß prinzipiell bis zu 4 Bildschirmseinheiten an den
KRS 4201 angeschlossen werden können.

2. Display-Steuereinheit zur Kopplung des GD 71 an KRS 4201

Die Entwicklung der Display-Steuereinheit erfolgte innerhalb der wissen-
schaftlichen Zusammenarbeit zwischen der Ungarischen Akademie der Wis-
sensschaften, Institut für Automatisierung und Rechentechnik und der TU
Dresden, Sektion Mathematik. Kurz zusammengefaßt bestehen die Aufgaben
der Display-Steuereinheit im folgenden:

- Sicherung des Direktzugriffs der Bildschirmseinheit über den nichtpro-
grammierbaren Speicherkanal zum Speicher des Steuerrechners für die
Abarbeitung des Displayfiles zum Zwecke der ständigen Bildwiederholung.
Dazu ist eine Decodierung der Displaybefehle (Steuer- und Bildbefehle), ein-
schließlich der Festlegung der internen Operationsfolge, der Ansteuerung
der Bildgeneratoren und der Steuerelektronik notwendig.
- Verarbeitung der Unterbrechungs- (Interrupt-)anforderungen der Dialog-
einrichtungen.
- Füllen und Lesen der Register vom Display-Steuerrechner aus, die in der
Display-Steuereinheit selbst enthalten sind, um die Dialogaktivitäten im
Steuerrechner verarbeiten zu können.

3. Konzeption der Kopplung des interaktiven Terminals

KRS 4201/GD 71 an ESER

Bei der Konzeption des interaktiven graphischen Terminals sahen wir eine wichtige Aufgabe darin, eine Gerätekonfiguration zu entwickeln, die an anderen Institutionen innerhalb der DDR in ähnlicher Weise eingesetzt werden kann. Unter diesem Gesichtspunkt erfolgt auch die Kopplung des Terminals an ESER.

Entsprechend der Möglichkeiten, die innerhalb der TU Dresden bestehen, erfolgt die ESER-Kopplung speziell an den Rechner ES-1022. Prinzipiell bieten sich die in Abb. 2 angegebenen Varianten für eine derartige Kopplung an.

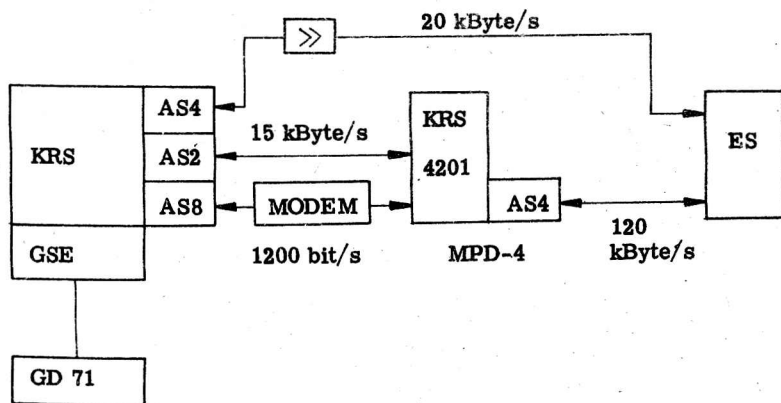


Abb. 2

Bei der Auswahl einer Variante gilt zu beachten: Realisierung einer hinreichend großen Übertragungsgeschwindigkeit der Informationen und die Überwindung einer vorgegebenen Entfernung zwischen Terminal und Großrechner. Eine überschlägige Schätzung ergibt eine Größe des BildwiederholSpeichers bei einer durchschnittlichen Auslastung des Bildschirms von etwa 4 K-Worten, d. h., es sind 8 kByte in möglichst kurzer Zeit zwischen Großrechner und Terminal zu übertragen, wenn von einer Datenverdichtung vorgegebener Datenstrukturen gegenüber dem Displayfile erst einmal abgesehen wird. Die zu

überwindende Entfernung richtet sich nach den räumlichen Gegebenheiten. In unserem Fall ist eine Entfernung von etwa 120 m zu überbrücken.

Um diesen Forderungen gerecht zu werden, entschieden wir uns für folgende Kopplungsvariante: Der Display-Steuerrechner KRS 4201 wird durch die Anschlußsteuereinheit AS4 erweitert und übernimmt somit gleichzeitig die Funktion des Multiplexers MPD 4. Zur Überbrückung der o. a. Entfernung ist es notwendig, eine Verstärkerstation zwischen MPD 4 und ES 1022 zu schalten, die bei Entfernungen zwischen Terminal und Großrechner kleiner als 20 m nicht benötigt wird. Diese Verstärkerstation entspricht dem ESER-Standard. Die reale Übertragungsgeschwindigkeit unter Berücksichtigung der Anforderungen, die das Betriebssystem ES/OS bzw. ES/DOS stellt, liegt in der Größenordnung von 20 kByte/s.

4. Erweiterung zu einem geometrie-orientierten Terminal

Parallel zu der Kopplung Terminal-ESER sind Arbeiten vorgesehen, die das Terminal zu einem echten geometrie-orientierten interaktiven Terminal erweitern. In erster Linie ist daran gedacht, Spezialrecheneinheiten zur Erweiterung der Funktionsfähigkeit des Kleinrechners einzuführen, so daß neben den üblichen arithmetischen und logischen Operationen auch spezifisch geometrische Operationen wie z. B. Verbinden, Schneiden, Transformationen, Rahmen- und Ausschnittsbildung gerätetechnisch realisiert werden können. Die Auslösung der Operationen erfolgt über entsprechende Befehlsmnemonics. Eine derartige Vorgehensweise ermöglicht eine schnellere Verarbeitung geometrischer Objekte auf dem Bildschirm gegenüber der Realisierung mit Hilfe von Unterprogrammen. Entsprechende Vorarbeiten in dieser Richtung erfolgen sowohl aus geometrischer (/5/, /6/, /7/) als auch schaltungstechnischer Sicht (/8/).

Die Ergebnisse derartiger Untersuchungen sollen zu einer qualitativen Erweiterung der Anwendungsmöglichkeiten interaktiver Terminals führen, die es erlaubt, schließlich auch bewegte Bilder darzustellen.

5. Grundsoftware

Obwohl für die Bildschirmleinheit GD 71 ein umfangreiches Grundsoftwarepaket in Zusammenhang mit dem Display-Steuerrechner TPA 70 existiert, steht vor uns die Aufgabe, weitestgehend eine eigene Grundsoftware für die Gerätekonfiguration GD 71 - KRS 4201 - ESER zu entwickeln.

Bei der Konzeption der Grundsoftware wird stets davon ausgegangen, daß das Terminal in Verbindung mit einem Großrechner arbeitet. In der Übergangsphase bis zur Fertigstellung der Kopplung wird es allerdings als eigenständiges System genutzt werden.

Als Anforderungen, die unsererseits an die Grundsoftware zu stellen sind, sollen genannt werden:

- Weitestgehende Nutzung der gerätetechnischen Gegebenheiten (Code der Bild- und Steuerbefehle, Aufbau der Bildhaupt- und Unterprogramme)
- Effektive Nutzung aller vorhandenen Dialogeinrichtungen
- Bereitstellung von Servicefunktionen
- Einbeziehung der Anschlußbedingungen Terminal - ESER
- Fehlerbehandlung
- Berücksichtigung der gewonnenen Erfahrungen hinsichtlich der konstruktiven Herangehensweise bei der Darstellung geometrischer Objekte.

Die Realisierung erfolgt in verschiedenen Ausbaustufen.

5.1. Graphischer Assembler

Während der Testphase des graphischen Terminals haben wir uns weitestgehend auf den von der Ungarischen Akademie der Wissenschaften entwickelten Assembler TAL, speziell auf den Teil GTAL zur Beschreibung graphischer Daten (/9/) gestützt. Ein von uns entwickelter Vorübersetzer erlaubt die Umsetzung beliebiger Befehlsmnemonics in vorgegebene Bitfolgen eines KRS-Wortes (/10/). Auf diese Weise haben wir die wichtigsten Bild- und Steuerbefehle übernommen und sie in die von der Gerätetechnik vorgeschriebenen Bit-

folgen umgewandelt, um weitestgehend die vorhandenen Testprogramme bei der Erprobung der Funktionsweise des Terminals zu nutzen.

Der nächste Schritt besteht nun darin, einen unseren Anforderungen angepaßten graphischen Assembler aufzubauen, der gegenüber dem vorhandenen einmal einen größeren Informationsgehalt pro Befehl besitzt und zum anderen sowohl die konstruktiven Belange bei der Beschreibung eines geometrischen Objektes als auch die Umsetzung von übergeordneten Datenstrukturen in Displayfiles bzw. in zeichenmaschinengerechte Steuerlochstreifen in stärkerem Maße berücksichtigt.

5.2. Bemerkungen zum Betriebssystem

Das Terminal KRS/GD 71 arbeitet z. Z. mit dem vom Hersteller des KRS gelieferten Betriebssystem STEP. Aufbauend auf diesem Betriebssystem wurde bereits eine Testvariante eines erweiterten Systems erarbeitet. Sie ermöglicht die Übersetzung eines im graphischen Assembler geschriebenen Programms in ein SYPS-Programm, den Aufbau der Displayfiles für den Bildwiederholpeicher und die Anwahl der Dialogeinrichtungen Lichtstift, Funktionstastatur, alphanumerische Tastatur und Rollkugel.

Bereits in der Testphase hat sich ergeben, daß die Variante für den späteren Einsatz nicht effektiv erscheint. Es ist deshalb vorgesehen, ein speziell für den Terminaleinsatz nutzbares Betriebssystem zu entwickeln, daß sowohl die Funktionsfähigkeit des KRS und seiner vorhandenen Peripherie garantiert als auch günstige Voraussetzungen für die Unterbrechungsbehandlung (einschließlich der Aktivierung der Dialogeinrichtungen), die Verarbeitung von im graphischen Assembler und auch von in übergeordneten Datenstrukturen beschriebenen geometrischen Objekten schafft.

In einer weiteren Ausbaustufe werden die Voraussetzungen für die Softwarekopplung des Terminals mit ESER-Anlagen entwickelt. Bisher ist vorgesehen, die Kopplung auf der Grundlage des von LfA des VVB Maschinelles Rechnen Berlin entwickelten Systems ROTAM vorzunehmen.

6. Datenstrukturkonzept

Sowohl für den Einsatz als autonomes Terminal als auch in Verbindung mit einem Hintergrundrechner sind entsprechende Datenstrukturen erforderlich, die den Erfordernissen einer Beschreibung (meist dreidimensionaler) geometrischer Objekte, deren Manipulation mittels der Dialogeinrichtungen bis hin zu ihrer Ausgabe auf zweidimensionalen Ausgabemedien entsprechen. Weiterhin sind noch nichtgeometrische Daten für geometrische Objekte zu beachten, die für eine echte Rechnerunterstützung bei der Modellierung technischer und technologischer Prozesse unbedingt notwendig sind.

Ausgangspunkt für ein zu erstellendes Datenstrukturkonzept sind zwei im Prinzip vorgegebene Teilstrukturen. Einmal handelt es sich um eine sogenannte Parameterdatenstruktur (PDS) wie sie im System DIGOS (/4/) schon Verwendung fand. Dabei werden für die im System vorkommenden geometrischen Elemente typische (möglichst wenige) Parameter gespeichert, die das geometrische Objekt eindeutig charakterisieren, die aber auch ausreichend sind, das Objekt zu manipulieren und es in einer Ebene darzustellen.

In diese Parameterdatenstruktur können nichtgeometrische Parameter (oder auch für den geometrischen Prozeß des Manipulierens und Darstellens nicht unbedingt notwendige geometrische Parameter) für geometrische Elemente aufgenommen werden.

Entsprechende Eingabeprogramme gewährleisten den Aufbau der PDS, wobei dem Nutzer genehme Parameter zur Beschreibung geometrischer Elemente in die internen Standardparameter umgewandelt werden. Ebenso ist es möglich, Ergebnisparameter intern ablaufender Programme in die PDS aufzunehmen.

Zum anderen ist der prinzipielle Aufbau der Displayfile-Struktur festgelegt, die die Grundlage für die gerätetechnisch gesteuerte Bildwiederholung bildet. Die Struktur baut das darzustellende Bild aus Teilbildern auf, die jeweils durch Identifikatoren charakterisiert sind. Die dynamische Verwandlung der Teilbilder erfolgt derart, daß dialogbedingte Bildänderungen sich unmittelbar in der Displayfile-Struktur widerspiegeln. Teilbilder selbst enthalten Instruk-

tionen für die Erzeugung von Punkten, Strecken, Kreisbögen und alphanumerische Zeichen mit entsprechender Intensität und in einer bestimmten Strichart.

Zwischen diesen beiden Strukturen ist nach bisherigen Erfahrungen mit dem System DIGOS eine weitere Struktur mit folgenden Eigenschaften erforderlich:

- Bei Aktivierung einer Ausgabeoperation für ein graphisches peripheres Gerät sollte der Namen und darstellungsfreundliche Parameter des auszugebenden geometrischen Objekts (und zugehöriger Teilobjekte) in diese Struktur aufgenommen werden.
- Bei Ausgabe- (oder Darstellungs-)operationen und auch solchen Operationen, die nur die Lage und die Größe des Objektes verändern, nicht aber dessen innere Struktur, sollen die Objektparameter möglichst nicht geändert, sondern durch einfache veränderbare Zusatzinformationen (z. B. in Form einer Matrix) berücksichtigt werden. Sich daraus ergebende Operationen sollten unmittelbar bei dem Aufbau der Displayfile-Struktur ausgeführt werden.
- Operationen wie Streichungen und Einfügen von Teilobjekten (am Bildschirm) müssen in geeigneter Weise in die Struktur eingehen.
- Die Überführung der Parameter geometrischer Objekte von einer Struktur in eine andere muß sowohl in Richtung PDS \rightarrow Struktur \rightarrow Displayfile als auch umgekehrt möglichst einfach durchführbar sein. Dabei dürfen keine Informationsverluste hinsichtlich geometrischer Daten auftreten.
- Überlegungen aus geometrischer Sicht über den Zusammenhang zwischen dem räumlichen Objekt und dessen Darstellung in Ebenen einschließlich der Rückgewinnung des räumlichen Objektes wurden bereits durchgeführt (/11/).
- Es muß die Möglichkeit bestehen, das Objekt gleichzeitig in mehreren Projektionsebenen unter Berücksichtigung der Sichtbarkeit abzubilden und die Parameter bereitzustellen.

- Es soll Berücksichtigung finden, daß ein Teil der geometrischen Operationen unmittelbar im Display-Steuerrechner ausgeführt werden soll (programm- oder gerätetechnisch) und daß dafür die notwendigen Parameter bereitstehen müssen.

Beim Aufbau der Struktur, in DIGOS Darstellungsdatenstruktur genannt, werden die mit DIGOS gewonnenen Erfahrungen weitgehend berücksichtigt. Dazu fließen noch Ergebnisse von Untersuchungen zu geometrischen Darstellungsverfahren, zum interaktiven geometrischen Konstruieren am Bildschirm und zur Minimisierung des Datenaustausches zwischen Terminal und Hintergrundrechner ein.

7. Graphische-/geometrie-orientierte Programmiersprache

Um den manuellen Aufwand für die Beschreibung graphisch- oder geometrie-orientierter Probleme möglichst gering zu halten, muß die Möglichkeit bestehen, sie in problemangepaßten Programmiersprachen zu beschreiben. Derartige Programmiersprachen müssen so aufgebaut sein, daß einmal für möglichst viele Einsatzgebiete die entsprechenden geometrischen Operationen und Operanden zur Beschreibung und Manipulation geometrischer Objekte bereitstehen und zum anderen der spezielle Anwender genau seine Operatoren und Operanden in ihr vorfindet.

Eine derartige Forderung wird durch eine allgemeine graphisch-/geometrie-orientierte Sprache kaum erfüllt werden können, vor allem wenn zusätzlich die spezifischen nichtgeometrischen Parameter für die Modellierung technischer und technologischer Prozesse Beachtung finden sollen.

Ein Ausweg aus dieser Situation besteht darin, daß eine allgemeine geometrie-orientierte Programmiersprache konzipiert und bausteinartig implementiert wird, die die grundlegenden geometrischen Operatoren und Operanden enthält. Spezifische geometrische Operatoren und Operanden sind Bestandteile problemorientierter Sprachen, die der Terminologie und den Eigenheiten des spezifischen Einsatzes dieser Sprachen weitestgehend entsprechen.

Derartige Fachsprachen werden neben dem geometrischen Aspekt i. a. auch mit dem Problem im Zusammenhang stehende nichtgeometrische Aufgabenstellungen berücksichtigen.

Die Umsetzung solcher Fachsprachen kann nun einmal erfolgen, indem aus den entsprechenden Bausteinen der allgemeinen (geometrie-orientierten) Programmiersprache die Fachsprache selbst aufgebaut wird oder indem durch ein Sprachtransformationssystem die Programme der Fachsprache auf Programme der allgemeinen Sprache zurückgeführt werden. In beiden Fällen können die in der allgemeinen Programmiersprache enthaltenen Algorithmen zur Verknüpfung und Darstellung geometrischer Elemente genutzt werden.

Erste Erfahrungen hinsichtlich dieser Vorgehensweise konnten bei der Konzeption und Implementierung der Systemsprache zu DIGOS (/2/) und der Konzeption einer Fachsprache für die Konstruktion von Gestellen im Werkzeugmaschinenbau (/13/) gewonnen werden.

Es besteht nun die Aufgabe, die angegebenen Möglichkeiten für den Aufbau und die Implementierung geometrie-orientierter Sprachen für den Einsatz des oben beschriebenen ESER-Terminals weiter zu entwickeln, wobei die einzelnen Etappen darin bestehen:

- auf der Grundlage der Systemsprache zu DIGOS (realisiert auf BESM-6) eine Systemsprache für ESER-Anlagen zu konzipieren und zu implementieren, unter Beachtung der vielfältigen Formen des digitalgraphischen Dialogs
- Fachsprachen für spezielle Teilsysteme zu entwickeln, die nach oben beschriebenen Prinzipien auf der Systemsprache basieren.

Bei all diesen Arbeiten wird auch weiterhin der konstruktiv-geometrischen Herangehensweise bei der Beschreibung geometrischer Objekte und entsprechender Operatoren große Beachtung geschenkt.

8. Anwendungen der Digitalgeometrie

Wie bereits erwähnt, waren schon bei der Bildung unserer Arbeitsgruppe die Einsatzgebiete für die Anwendung der Ergebnisse weitestgehend vorgezeichnet. Schon vor Beginn unserer eigentlichen digitalgeometrischen Arbeiten existierte eine enge Zusammenarbeit mit dem TU-Wissenschaftsbereich und dem entsprechenden Industriebetrieb auf dem Gebiet der Getriebetechnik. Hierbei ging und geht es noch im wesentlichen um eine mathematisch-algorithmische Beschreibung von Schneckengetrieben einschließlich deren Herstellung.

Während unserer Arbeiten am System DIGOS und der Realisierung eines Systems für geometrische Grundoperationen für den Kleinrechner KRS 4201 (/7/) begann eine intensive Zusammenarbeit auf dem Gebiet der Konstruktion von Gestellen im Werkzeugmaschinenbau und der Simulation röntgenspektrographischer Untersuchungen von Werkstoffen mit Einrichtungen unserer Universität und den entsprechenden Industriebetrieben.

Ein weiteres Einsatzgebiet sind alle Fragen, die sich mit dem Problem von Lay out's in der Ebene und im Raum befassen (Einrichtung von Werkhallen, Anordnung der Bauteile bei elektrischen Antrieben u. a.).

Es zeichnet sich bereits ab, daß auch die zukünftigen Einsatzgebiete unserer Ergebnisse hauptsächlich in den Bereichen der Konstruktion und Fertigung vor allem im Maschinenbau liegen werden.

Bei dem Einsatz der Ergebnisse der Digitalgeometrie haben wir einige Erfahrungen gesammelt, die hier kurz zusammengefaßt aufgezählt werden sollen:

- Die Nutzer erwarten für ihr Einsatzgebiet speziell zugeschnittene Teilsysteme.
- Für die Modellierung technischer und technologischer Prozesse müssen diese Systeme geometrische und nichtgeometrische Komponenten besitzen.

- Sowohl kleinere mit geringem Aufwand zu realisierende als auch größere möglichst durchgängige Systeme von der Konstruktion bis zur Fertigung werden für die verschiedensten Anwendungen benötigt.
- Die Nutzer sind schon teilweise bereit, die Rechentechnik dahingehend zu unterstützen, indem sie z. B. stilisierte Baugruppen einführen und auch selbst Beiträge für die mathematische und gleichzeitig nutzerfreundliche Beschreibung der in ihrem Spezialgebiet vorkommenden geometrischen Objekte liefern.

Die Nutzung dieser Erfahrungen sind für die weitere Entwicklung der Digitalgraphik/-geometrie sowohl hinsichtlich ihrer theoretischen Grundlagen als auch hinsichtlich der Umsetzung der Ergebnisse für praktische Anwendungen unbedingt notwendig.

Literatur

- /1/ Bär, G.: Zur Verzahnungstheorie der Schneckengetriebe.
Diss. TU Dresden, Sektion Mathematik, 1972.
- /2/ Ortleb, R.: Zur Verzahnungs- und Fertigungsgeometrie allgemeiner
Zylinderschneckengetriebe.
Diss. TU Dresden, Sektion Mathematik, 1972.
- /3/ Ludwig, M.; Richter, Chr.; Weber, R.: DIGOS - ein digitalgeometrie-
orientiertes System für die Konstruktion und Darstellung geometri-
scher Objekte.
Rechentechnik/Datenverarbeitung 13, H. 1, S. 40 - 45 (1976).
- /4/ Ludwig, M.; Richter, Chr.; Weber, R.: Sprache und Datenstrukturen
des digitalgeometrie-orientierten Systems DIGOS.
Wiss. Z. d. TU Dresden 25, H. 3, S. 405 - 415 (1976).
- /5/ Ortleb, R.: Rechentechnisch-orientierte Darstellung einfacher geome-
trischer Grundaufgaben.
Rechentechnik/Datenverarbeitung 13, H. 1, S. 46 - 47 (1976).
- /6/ Geise, G.: Ein Kalkül für geometrische Aufgaben.
Tagung Digitalgrafik, Rostock/Warnemünde, Nov. 1976.
- /7/ Krause, H.: Anwendung geometrischer Grundoperationen in einem
Kleinrechner-Dialogsystem für KRS 4200.
Tagung Digitalgrafik, Rostock/Warnemünde, Nov. 1976.
- /8/ Monjau, D.: Zur gerätetechnischen Realisierung geometrischer
Operationen.
Tagung Digitalgrafik, Rostock/Warnemünde, Nov. 1976
- /9/ Gállai, I.; Forgacs, T.; Darvas, P.: The assembler language TAL
for the TPA 70/25 small computer.
MTA SZTAKI Department, Budapest, 1975.

- /10/ Ortlieb, R.: Programmierung in Assemblersprache KRS 4200/GD 71.
Tagung Digitalgrafik, Rostock/Warnemünde, Nov. 1976.
- /11/ Broschinski, D.: Rekonstruktion geometrischer Objekte aus ihrer Darstellung im Grund-Aufriß-Verfahren.
Tagung Digitalgrafik, Rostock/Warnemünde, Nov. 1976.
- /12/ Richter, Chr.: Die Sprachkomponente von DIGOS.
Tagung Digitalgrafik, Rostock/Warnemünde, Nov. 1976.
- /13/ Ludwig, M.: Eine Fachsprache für die Konstruktion von Werkzeugmaschinenengstellen.
Tagung Digitalgrafik, Rostock/Warnemünde; Nov. 1976.

Eingegangen: 30. 11. 1976

Anschrift des Verfassers:

Doz. Dr. Manfred Ludwig
Technische Universität Dresden
Sektion Mathematik
DDR - 8027 Dresden, Mommsenstr. 13.

Anwendungen des Programmsystems GIPS für ein aktives graphisches Bildschirmgerät und eine Zeichenmaschine

Am Rechenzentrum des Instituts für Schiffbau ist ein leistungsfähiges graphisches Gerätesystem installiert. Nach einer kurzen Beschreibung der Geräte wird auf das zur komplexen Nutzung dieser Geräte entwickelte "graphische interaktive Programmsystem GIPS" eingegangen, insbesondere auf die bei der Realisierung berücksichtigten Gesichtspunkte.

Die Anwendungsmöglichkeiten lassen sich in vier Aufgabengruppen gliedern:

- Ausgabe von Daten, insbesondere von graphischen Darstellungen,
- Eingabe von Programmen und Daten,
- Steuerung von Berechnungsabläufen in Abhängigkeit von Berechnungsergebnissen,
- Lösung von graphischen Aufgaben (Entwerfen, Anordnen, Darstellen).

Eine Reihe von Anwendungen wird ausführlicher dargestellt:

- Kontrolle von Steuerstreifen für die numerisch gesteuerte Propellerfräse des DMR,
- Prüfung von Daten zur numerischen Beschreibung der Schiffsgeometrie und Prüfung von interpolierten Bauspannten,
- Aufmessen von Diagrammen,
- Analyse des Schwingungssystems Motor- Welle- Propeller,
- Berechnung von Schiffskörperschwingungen,

- Aufbau von Plänen aus ebenen standardisierten Elementen,
- Konstruktion von Leiterplatten.

Die Einschätzung der erreichten Ergebnisse erlaubt Schlußfolgerungen für die weiteren Arbeiten auf diesem Gebiet.

Der Vortrag ist veröffentlicht im Sonderheft der Schiffbauforschung, 1977 (Symposium "Rationalisierung der technischen Produktionsvorbereitung im Schiffbau").

Eingegangen: 30. 11. 1976

Anschrift des Verfassers:

Math. Horst Rogmann
Institut für Schiffbau
Abteilung Rechenzentrum
DDR - 25 Rostock, Doberaner Str.

Anwendung des Bildschirmdialoges für Entwurfs- und Gestaltungsprozesse in der Architektur

1. Ansatzpunkte für die Rationalisierung der Entwurfs- und Gestaltungsprozesse in der Architekturpraxis

In der sozialistischen Gesellschaft ist die ästhetische Gestaltung der den Menschen umgebenden, räumlich baulichen Umwelt ein wichtiger Aspekt der Entwurfs- und Gestaltungspraxis in der Architektur. Ihre Bedeutung ergibt sich aus dem Einfluß der bebauten Umwelt infolge ideologischer, physischer und psychischer Wirkungen auf Leistungsvermögen, Kreativität und Vitalität des Menschen. Aus diesen Zusammenhängen ergeben sich hohe Anforderungen an die Entwurfs- und Gestaltungsprozesse.

Zur Zeit ist die Tätigkeit des Architekten jedoch noch durch eine Fülle mechanischer Zeichenarbeit gekennzeichnet. Diese erstreckt sich von der Anfertigung der ersten Entwürfe über alle Stufen der Gestaltung bis hin zur Publikation der fertigen Bebauungskonzeption. Es gilt, diesen manuellen Anteil zu automatisieren, um der schöpferischen Arbeit mehr Raum zu geben.

Eine weitere Mangelerscheinung besteht in der Diskrepanz zwischen den Arbeitsmethoden in den Architektenbüros und dem hohen Reifegrad der Typisierung der gegenwärtig zur Bauausführung gelangenden Baukörper. Das Sortiment an Hochbauerzeugnissen, das für die Gestaltung von Bebauungskonzeptionen zur Verfügung steht, ist so weit gediehen, daß in der Regel in Neubaugebieten keine Sonderbauten mehr vertreten sind. Diesen veränderten Bedingungen wurde bisher noch nicht Rechnung getragen.

Ferner ist zu beachten, daß die Anfertigung der Entwürfe an eine Vielzahl von Randbedingungen geknüpft ist, die durch Verkehrsplanung, Geologie, Grünplanung oder durch den Abwasser- und Versorgungsleitungsbau gegeben sind. Damit sind wichtige Voraussetzungen für den Einsatz der Digitalgraphik erfüllt: Schematisierbare Zeichenaufgaben, Vorhandensein geometrischer Objekte mit Wiederverwendungscharakter, Anknüpfungsbedingungen geometrischer Natur und Darstellungsprobleme.

Eine erste Stufe der Rationalisierung ist die automatische Erstellung der perspektivischen Ansichten, die für Publikationen oder als Entscheidungsgrundlagen Verwendung finden. Die notwendige technische Basis hierfür besteht aus EDVA und Zeichenautomat. Für die nächsten Schritte kommt zu dieser Ausrüstung noch das Bildschirmgerät hinzu. Dieses gestattet einen unmittelbaren Dialog mit der EDVA. Damit läßt sich die Ausgabe der perspektivischen Ansichten rationell vorbereiten. Des weiteren ist es möglich, den Bildschirmdialog bereits in dem frühen Stadium des Entwurfs zu nutzen, in dem der Architekt seinen Arbeitsplatz wenigstens zeitweise an den Bildschirm verlagert. Diese letzte Version läßt sich in der Zukunft durch eine Lichtstifteingabe geometrischer Figuren erweitern, so daß über die oben erwähnte Arbeit mit Typenprojekten auch Sonderbauten beliebiger Gestalt erfaßt werden können. Damit wird auch die Rekonstruktion von Wohngebieten der Rationalisierung durch die Digitalgraphik erschlossen.

2. Die Zielstellung für die Darstellungs- und Dialogtätigkeit

Die zur Zeit formulierbare Zielstellung geht, was den Bildschirmdialog anbelangt, über die momentanen Möglichkeiten der Verfügbarkeit von BS-Geräten hinaus. Ein zeitlicher Vorlauf ist jedoch von Nutzen, da es schwierig ist, umfangreiche Dialogsysteme aufzubauen.

Die Probleme liegen nicht so sehr bei der Programmentwicklung, denn hierbei können diese durch gezieltes Nachnutzen bereits vorliegender Arbeiten verringert werden, als vielmehr in der Anpassung an die Arbeitsmethodik des Anwenders.

Der Architekt, der einen Entwurf erarbeitet, bringt seine Vorstellungen mittels einfacher geometrischer Informationen zu Papier. Diese Arbeit hat experimentierenden Charakter. Es entstehen mehrere Entwürfe oder mehrere Varianten eines Entwurfes. Einige Entwürfe sollen weiterbearbeitet werden, indem sie zeichnerisch näher ausgeführt und in bestimmten Details genauer untersucht werden.

Diese Arbeitsprozesse muß der Dialog gestatten, ohne den Nutzer mit den programminternen Problemen zu behelligen, z. B. der Verwaltung der großen Datenmengen, die sich hinter graphischen Problemen rechen-technisch verbergen, oder dem Aktivieren der entsprechenden Dialogprogramme. Der Anwender muß in der Lage sein, jeden Entwurf uneingeschränkt realisieren zu können, und zwar mit minimalem Eingabeaufwand. Das Dialogprogramm hat dem Nutzer weitgehend entgegenzukommen, indem es ein sinnvoll, aber sparsam gewähltes Menü von Aktionen am Bildschirm bereithält.

Die Dialogarbeit hat die Aufgabe, die Fähigkeiten des Anwenders zu unterstützen und seine Arbeitsmittel zur besseren Durchdringung seiner Aufgaben zu ergänzen. Keinesfalls dürfen seiner schöpferischen Wirkungsbreite durch die Anwendung der Digitalgraphik Beschränkungen auferlegt werden.

Die Konkretisierung der Aufgaben des Dialoges ergibt im Falle der Anwendung für den Architekten exakt formulierbare Anforderungen. Zunächst müssen alle typisierten Baukörper über Namensaufruf erreichbar und auf dem Bildschirm darzustellen sein.

Die Zusammenstellung zu Baukörperkomplexen erfordert das Erzeugen von Kopien der im Speicher vorhandenen Gebilde. Zum Experimentieren sind Translation, Drehung, Erzeugen und Tilgen notwendig und zur entsprechenden Darstellung auf dem Bildschirm die Wahl der Projektionsart und des Projektionszentrums mit der Blickrichtung. Das Festhalten der Ergebnisse des Dialogs mittels eines Zeichenautomaten ist wahlweise mit einem im Vergleich mit dem auf dem Bildschirm erzeugten Bild höheren Komfort bzw.

Informationsgehalt zu ermöglichen. Von Vorteil ist es, wenn der Nutzer durch einfache Informationen zu einem der vorher erreichten Zustände des Dialogs zurückkehren kann.

Hinter diesen Vorgängen verbergen sich rechnerintern schwierige Probleme der Daten- und Speicherverwaltung. Es sind geeignete Datenstrukturen und Programmhierarchien zu verwenden. Die Organisation des Dialogs bleibt der Menütechnik, die eine klare und schnelle Kommunikation zuläßt, vorbehalten. Das eigentliche Problemprogramm selbst sollte alle Prozesse, die ohne das Eingreifen des Nutzers ablaufen können, enthalten. Dieses betrifft die Definition von Projektionsebenen, die Zusammenstellung von Polygonen, die Berechnung der Winkel für die Blickrichtung und der Maßstabsfaktoren, aber auch Probleme der Datenstrukturen wie Schaffung eines dem Algorithmus genehmen Aufbaus oder die Segmentierung bei umfangreichen Gebilden.

3. Das Dialogprogramm für Baukörperdarstellung

3.1. Grundzüge

Bei der Entwicklung des Programms für Baukörperdarstellung wurde in der Bauakademie der DDR von den realen Bedürfnissen der Nutzer ausgegangen.

Dabei war der Spezifik der Rationalisierung durch die Datenverarbeitung Rechnung zu tragen, indem gewisse Einschränkungen zu berücksichtigen, aber auch Vorzüge zu nutzen waren, die neben der Rationalisierung eine Qualitätssteigerung verursachten. Auf Grund der jeweils vorhandenen Programme für perspektivische Darstellungen, konnte durch Auswertung der praktischen Nutzung die Richtung der Weiterentwicklung festgelegt werden. Ein weiterer Anhaltspunkt für die Festlegung der Aufgabenstellung für das Dialogprogramm war das Ziel, die Relevanz des Bildschirmdialogs und insbesondere einiger spezieller Funktionen für die Arbeit des Architekten zu prüfen.

Zu diesem Zwecke wurde eine Auswahl von Aktionen am Bildschirm getroffen, die geeignet scheint, den Anforderungen des Entwerfens gerecht zu werden, wobei bereits einem großen Teil der schon genannten Vorstellungen entsprochen wird. Die Darstellung der Baukörper geschieht zunächst in der Form 3-dimensionaler Kantengraphen. Die Methode der Digitalisierung erfaßt jedoch, auch den Innenraum der Baukörper, um die Anwendung eines Sichtbarkeitstests für die weitere Entwicklung des Dialogprogramms vorzubereiten.

Das Digitalprogramm besteht aus einem Programmteil für perspektivische Darstellungen und einem Dialogteil. Die Korrespondenz zwischen diesen beiden Teilen wird durch ein Steuerprogramm organisiert. Der Dialogteil umfaßt den DIGRA-Dialog entsprechend /1/. Die in diesem enthaltene Hauptmenüfunktion DARSTELL wurde unter Beibehaltung ihres strukturellen Aufbaus mit geänderten Wirkungen der Endstellen des Menübaumes und neuem Menütext verwendet. Wirkungen und Text entsprechen der von den Nutzern des Perspektiveprogramms geübten Arbeitsweise.

3.2. Der Gegenstand der Dialogarbeit

Zur Bearbeitung im Dialog gelangen Bebauungskonzeptionen des Städtebaus. Dabei besteht sowohl der Wunsch neue Konzeptionen aufzubauen, als auch vorhandene zu modifizieren. Als untergeordnete Einheit, die sinnvollerweise ansprechbar sein muß, ist lediglich die Einheit "Baukörper" anzusehen. Beim Digitalisieren des Lageplanes einer Bebauungskonzeption wird die Einheit "Baukörper" durch einen Festpunkt und einen Winkel erfaßt. Die Baukörper werden den Nutzern für die Arbeit in digitalisierter Form bereitgestellt. Das Digitalisieren der Baukörper, das der Nutzer nur für Sonderbauten selbst durchzuführen hat, basiert auf eine Auffassung der Baukörper als einer Einheit aus einem Körper (Punktmenge) und einem Graphen. Um den verschiedensten Anwendungsformen gerecht werden zu können, wird der Graph in drei Teilgraphen zerlegt. Der Teilgraph I stellt die Umrisse (Kanten) der Baukörper dar. Der Teilgraph II gibt gewisse Flächenaufteilungen wieder. Im wesentlichen enthält er Geschoßhöhen. Der Teilgraph III umfaßt

Details wie Fenster, Türen und Fassadenelemente. Abb. 1 verdeutlicht diesen Sachverhalt.

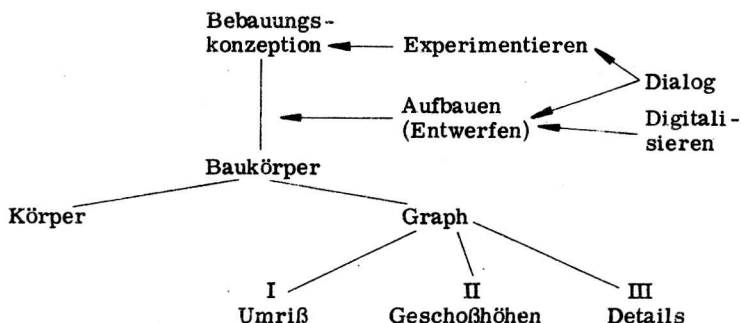


Abb. 1

3.3. Nutzerfreundliche Digitalisierungsmethoden

3.3.1. Graph

Die Methode der Erfassung des Graphen erfüllt die Forderung nach leichter Erlernbarkeit und geringer Fehleranfälligkeit. Zunächst steht der Nutzer vor der Aufgabe, einen Kantengraphen zu entwerfen, der den Baukörper möglichst gut charakterisiert. Danach sind die Stützpunkte desselben zu benennen (Zahlen). Das Digitalisieren erfolgt nun durch das Notieren der Koordinaten x , y und z der Stützpunkte, sowie der unmittelbaren Nachbarnpunkte, die durch eine darzustellende Kante erreichbar sind (Abb. 2).

Punkt	Koordinaten			Nachbarnpunkte				
Nr.	x	y	z	1	2	3	4	5

Abb. 2

Innerhalb dieser Anordnung ist es unerheblich, ob einzelne Verbindungen doppelt angegeben oder in welcher Reihenfolge sie aufgeführt sind. Desweiteren ist die Art der Numerierung der Stützpunkte ohne Belang, ob sie

aufsteigend oder lückenlos ist. In der Praxis hat sich diese Art der Definition eines Graphen im Vergleich zu der strengeren Form durch Angabe einer Stützpunktmenge und einer exakten Strukturinformation gut bewährt.

3.3.2. Körper

Die Angabe der Körperinformation ist nicht ganz so problemlos. Hier muß eine genaue Strukturinformation in Zusammenhang mit einer korrekten Punktzuordnung gegeben werden.

Der Sichtbarkeitstest wird nach /2/ durchgeführt. Der Körper ist gegeben durch eine Punktmenge, die durch einen logischen Ausdruck in linearen Relationen beschrieben wird. Der logische Ausdruck enthält Konjunktion, Disjunktion und Klammern.

Im gegebenen Zusammenhang interessieren nur ebenflächig begrenzte Körper; demgemäß treten im definierenden logischen Ausdruck auch nur lineare Relationen in x , y , z auf, wobei jede Relation einen Halbraum definiert. Die Vereinfachung des Verfahrens besteht nun darin, daß der Nutzer nicht die Koeffizienten der Ebenengleichungen benennt, sondern lediglich 3 Punkte, die die Ebene aufspannen. Die Punkte sind sehr leicht in geeigneter Weise aus der Stützpunktmenge des Graphen zu entnehmen. Dasselbe gilt für einen beliebigen Punkt im Inneren des Halbraumes, der die Relation festlegt. Die Berechnung der Koeffizienten übernimmt das Programm.

Es verbleibt die Angabe der Strukturinformation. Diese ist durch Ja- bzw. Neinreferenzen ausdrückbar. Dabei muß die Anordnung der Ebenen (Punkte, Referenzen) in ihrer Reihenfolge (Feld) dem Körperausdruck entsprechen. Die Referenzen beziehen sich auf den Zeilenindex innerhalb dieses Feldes (Abb. 3).

Ebenenpunkte	Relat. -	Ja -	Nein -
1. 2. 3.	Punkt	Referenz	Referenz

Abb. 3

Die Referenzen dienen der Behandlung des Körperausdruckes bezüglich der Prüfung der Zugehörigkeit eines Punktes zum Körper als der Grundlage für den Sichtbarkeitstest.

3.4. Die Hauptmenüfunktion ENTWURF

Die schon erwähnte Hauptmenüfunktion DARSTELL des DIGRA 73-Systems wurde entsprechend Abb. 4 verwendet. Die Grundmenüs gestatten folgende Aktionen des Nutzers:

A/E (LK)	Lochkarten-Ein- und Ausgabeoperationen
BAUKOERPER	Ausgabe des aktuellen Gebildes
PROJ-DATEN	Ausgabe der Projektionsangaben
EINGABE	Einlesen des nächsten im Eingabestrom vorgesehenen Baukörperkomplexes
STANDPKT	Alphanumerische Eingabe eines neuen Projektionszentrums und eines neuen Öffnungswinkels des Sehfensters
ZIELPKT	Alphanumerische Eingabe eines neuen Zielpunktes
AENDERN	Verschieben, Drehen oder Tilgen von Baukörpern
BILDAUSW	Rückkehr zu einem vorher gespeicherten Projektionszentrum bzw. Isometrie

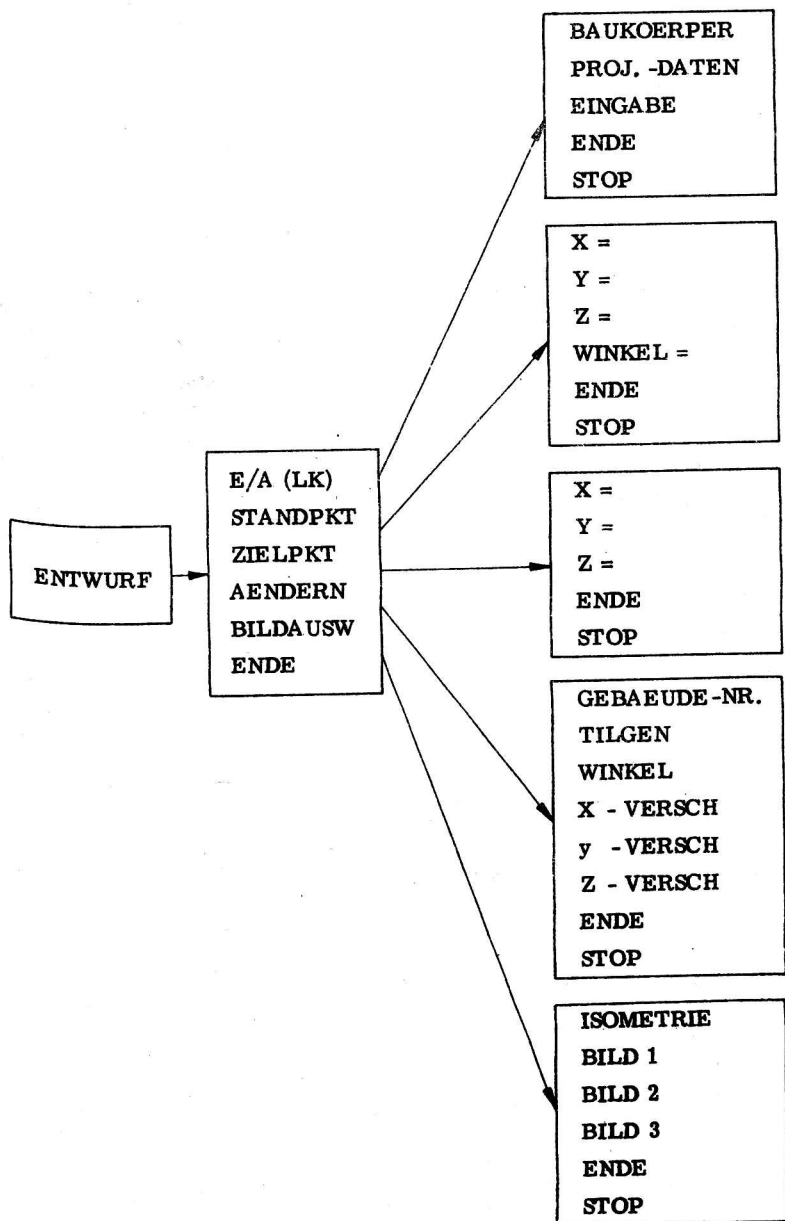


Abb. 4

Innerhalb eines Dialogschrittes ist es dem Anwender möglich, eine Reihe von Aktionen zu veranlassen. Nach Beendigung eines Dialogschrittes führt das Perspektiveprogramm alle Aktionen aus. Desweiteren stehen alle übrigen Hauptmenüfunktionen des DIGRA 73-Systems, sofern sie auf die zweidimensionale Datenbasis wirken, zur Verfügung. Unter anderem können dadurch Beschriftungen angebracht und Handkopien über die Zeichenmaschine gezogen werden.

Literatur

- /1/ Berndt, E.; Kutschke, K. -H.: Dialog für die automatische Erzeugung von Zeichnungen mit dem DIGRA -73-System.
Rechentechnik Datenverarbeitung, 12, Beiheft 4, 27-31 (1975).
- /2/ Thielcke, H.: Verarbeitung von logischen Ausdrücken in der Digitalgraphik.
EIK 9, 569 - 578 (1978).

Eingegangen: 30. 11. 1976

Anschrift des Verfassers:

Dipl. -Ing. Walter Scholze
Bauakademie der DDR
Institut für Projektierung
DDR - 108 Berlin, Unter den Linden 19

Datenbanksysteme und Digitalgraphik

1. Zur Entwicklung der Datenbanktechnologie

Datenbanksysteme nehmen heute bereits in vielen automatisierten Informationssystemen (AIS) einen wichtigen Platz ein; ihr Einsatzbereich ist in ständiger Ausdehnung begriffen. Die Einführung eines Datenbanksystems wirkt sich nicht nur auf die eigentliche Datenverwaltung aus, sondern hat wesentlichen Einfluß auf Methodik und Technologie der gesamten EDV-Anwendung.

Der Übergang zur Datenbanktechnologie ist Ausdruck und notwendige Folge eines bestimmten Reifegrades der Rechnernutzung. Dieser Reifegrad ist dadurch gekennzeichnet, daß der Rechner in einem bestimmten Anwendungsbereich ein vielfach genutztes Arbeitsinstrument zur Bewältigung von Informationsprozessen geworden ist. Bezüglich der durch den Rechner bearbeiteten Informationen drückt er sich darin aus, daß

- von zahlreichen verschiedenen Informationsprozessen auf ein und dieselben Informationsbestände Bezug genommen wird und/oder
- Informationsbestände von bestimmten Informationsprozessen, in der Regel nach Aktualisierung, mehrfach wiederverwendet werden.

Unter diesen Bedingungen wird es notwendig, von der im Anfangsstadium der Rechneranwendung üblichen prozeßorientierten Datenbehandlung zu einer prozeßunabhängigen und damit funktionell und programmtechnisch verselbständigten Datenorganisation überzugehen. Diese Organisationsform wird durch Datenbanksysteme (DBS) verwirklicht.

Ein DBS besteht aus der Datenbank (DB) und dem Datenbankbetriebssystem (DBBS). Ein DBBS ist eine Softwarekomponente, bestehend aus Programmen zur Verwaltung (d. h. zum Speichern, Aktualisieren, Bereitstellen, Sichern usw.) von Daten. Diese Programme nutzen die Datenverwaltungsfunktionen (Speicherungs- und Zugriffsfunktionen) des gegebenen allgemeinen Betriebssystems; sie können als datenbankspezifische Erweiterung dieses Betriebssystems aufgefaßt werden. Die zur Steuerung der Funktionen eines DBBS verfügbaren sprachlichen Mittel heißen Datenbanksprachen. Die Datenbank ist eine Gesamtheit von Daten, die von einem gegebenen DBBS verwaltet werden. Auf Grund der oben charakterisierten Rolle eines DBS in einem AIS verkörpert eine Datenbank in der Regel ein Abbild des für eine Gesamtheit automatisierter Informationsprozesse relevanten Teils der Realität.

Wir geben zunächst einen kurzen Abriss der Entwicklung, die zu den in der heutigen Praxis existierenden beiden Grundtypen von DBS, den operationellen und den Rechensystemen geführt hat.

Die erste Linie (siehe z. B. / 6 /) ist mit der sogenannten ökonomischen Datenverarbeitung verbunden, für die die (in der Regel sequentielle) Verarbeitung von Massendaten typisch ist. Sie hat ihren Ursprung in der Lochkartenmaschinellen Datenverarbeitung. Die mit Magnetbandspeichern ausgerüsteten Rechner der zweiten Generation, das heißt die ersten eigentlichen EDVA, ermöglichten zunächst quantitative Verbesserungen (größere Datenmengen, kleinere Verarbeitungszeiten, kompliziertere Algorithmen). Die wachsende "Anwendungsichte" der EDV führte zur Einführung des Stammbandprinzips der Datenverwaltung. Dieses Prinzip ist durch die einmalige Speicherung mehrfach verwendeter, objekt- bzw. sachgebietsorientiert gebildeter Informationsbestände in jeweils einer sequentiellen "Stammdatei" charakterisiert. Verglichen mit der prozeß- bzw. programmorientierten Datenorganisation werden damit die Datenerfassung und der Änderungsdienst wesentlich rationalisiert. Die Anforderungen der verschiedenen Anwendungsprogramme können mit Hilfe einer Stammdatei mit gleichermaßen aktuellen Informationen befriedigt werden. Der wesentliche Nachteil

dieser Organisationsform besteht in der Notwendigkeit von Sortierläufen für alle diejenigen Programme, die den Inhalt einer Stammdatei mit einer anderen Satzfolge als der in der Stammdatei festgelegten physischen Folge benötigen. Die entscheidende technische Voraussetzung zur Überwindung dieses Nachteils wurde mit der Bereitstellung von Massenspeichern mit Direktzugriffsmöglichkeit, insbesondere von Magnetplattenspeichern, als Komponenten von Rechnern der dritten Generation geschaffen. Diese Speicher ermöglichen es, durch Adreßverkettung von Sätzen in ein und derselben physischen Datei die Menge aller Sätze oder Teile dieser Menge in verschiedenen Sortierfolgen zu erzeugen, wenn nur die Adresse des jeweils ersten Satzes (des "Ankersatzes") bekannt ist.

Auf dieser Grundlage entstanden die sogenannten operationellen Datenbanksysteme. Die Funktion dieser Systeme besteht neben der Durchführung solcher Operationen wie Laden und Aktualisieren in der Bereitstellung einer im voraus festgelegten Gesamtheit von Satzfolgen aus einer Datei oder einem System von Dateien. Datenbanksysteme dieses Typs dienen deshalb in erster Linie der informationellen Sicherstellung einer gegebenen Menge von (in der Regel häufig abzuarbeitenden) Anwendungsprogrammen. In manchen operationellen Systemen wird die Erzeugung von Satzfolgen mit bestimmten Verarbeitungsprozessen verbunden (z. B. ermöglicht das System BASTEI bei Anwendung in der technischen Produktionsvorbereitung die Berechnung totaler Stücklisten).

Eine zweite Entwicklungslinie der Datenbanktechnologie ist mit der Schaffung automatisierter Informationsrecherchesysteme verbunden. Gegenstand der Speicherung ist hier eine Menge von Beschreibungen realer Objekte, z. B. wissenschaftlicher Arbeiten, durch Werte vorgegebener Merkmale (z. B. Autor, Titel, Zeitschrift) sowie Deskriptoren eines Thesaurus. Die geforderte Funktion eines solchen Systems besteht darin, diejenige Teilmenge der Objektbeschreibungen bereitzustellen, für die ein vorgegebener logischer Ausdruck auf den Mengen der Merkmalswerte und Deskriptoren den Wert "wahr" hat. Aus der Zeit der Lochkartentechnik sind mechanisierte Lösungen für relativ kleine Informationsbestände

mit Hilfe von Sortiermaschinen bekannt (z. B. Kriminaltechnik). Die Magnetbandtechnik ermöglichte die Beherrschung größerer Informationsbestände (z. B. R 300-Typenprojekt "Information und Dokumentation"). Die Notwendigkeit des sequentiellen Durchsuchens der auf Magnetband gespeicherten Objektbeschreibungen zwang zur Beschränkung auf den Stapelbetrieb, bei dem mehrere Anfragen bei einem Banddurchlauf bearbeitet werden können. Auch hier brachten die Massenspeicher mit Direktzugriffsmöglichkeit eine qualitative Veränderung der Technologie mit sich. Diese ist durch den Aufbau von Adreßlisten (auch Indexlisten, inverse Dateien) gekennzeichnet, die für jeweils einen Merkmalswert oder Deskriptor die Adressen derjenigen Objektbeschreibungen enthalten, in denen dieser Wert bzw. Deskriptor auftritt. Anhand dieser Listen und ohne Bewegung der Objektbeschreibungen selbst werden die Adressen der bezüglich einer Anfrage relevanten Beschreibungen bestimmt. Diese werden dann mittels Direktzugriff bereitgestellt.

Die Adreßlistentechnologie bildet die Grundlage eines zweiten Typs von Datenbanksystemen, die als Recherchesysteme bezeichnet werden. Als Beispiel ist hier das für die Betriebssysteme DOS/ES und OS/ES entwickelte AIDOS zu nennen.

Es sei darauf hingewiesen, daß die Implementierungen von operationellen Systemen mittels Adreßverkettung und von Recherchesystemen mittels Adreßlisten sicher die in vielen Fällen optimalen, aber nicht die einzigen Lösungsvarianten bilden. Beispielsweise ist es bei operationellen Systemen auch möglich, die Adreßverweise getrennt von den Datensätzen zu speichern und damit gewissermaßen für jede benötigte Satzfolge eine Adreßliste zu führen. Eine "echte" Adreßlistenvariante mit merkmalswert- bzw. deskriptorbezogenen Adreßlisten würde jedoch eine zusätzliche Sortierung der gefundenen Satzadressen notwendig machen. Andererseits ist es denkbar, die Adreßverkettung, allerdings nur unter speziellen Bedingungen, bei Recherchesystemen anzuwenden, beispielsweise in Form der Vielfachverkettung / 3 /.

Neben den dargestellten beiden Entwicklungslinien gibt es noch weitere, denen unterschiedliche Anwendungsbedürfnisse und/oder programmtechnische Prinzipien zugrunde liegen. So ist es nicht verwunderlich, daß in der heutigen Praxis eine ganze Reihe von Datenbanksystemen mit unterschiedlichen Funktions- und Leistungscharakteristiken im Einsatz ist und daß als Softwarebestandteile einer EDVA die entsprechenden unterschiedlichen Datenbankbetriebssysteme angeboten werden. In der Praxis werden heute Datenbanksysteme in der Regel mit Hilfe solcher vorgefertigter DBBS aufgebaut und verwaltet; auf Grund der Kompliziertheit dieser Systeme wäre eine Eigenprojektierung mit sehr hohem Aufwand verbunden. Bei der Einführung der Datenbanktechnologie in einem neuen Anwendungsbereich wie dem der Digitalgraphik entsteht folglich die Frage nach der Eignung verfügbarer DBBS und einer vergleichenden Bewertung dieser Systeme. Im folgenden wird ein Klassifizierungsschema für Datenbanksysteme als Hilfsmittel für die Beantwortung dieser Frage angegeben. Zur Veranschaulichung werden in dieses Schema einige der in der DDR für EDVA des ESER verfügbaren Datenbankbetriebssysteme eingeordnet.

2. Charakteristiken von Datenbankbetriebssystemen

Im folgenden wird nur eine Auswahl von DBBS-Merkmalen betrachtet, die in dem gegebenen Zusammenhang besonders wichtig sind. Ein sehr umfangreiches Schema zur Beschreibung und Klassifizierung von DBBS findet man beispielsweise in / 2 /.

Eine erste für die Beurteilung der Anwendungsmöglichkeiten eines DBBS wichtige Frage betrifft das Spektrum der Informationen, die mit diesem System verwaltet werden können. Informationen existieren in einem AIS in Form von Ausdrücken in bestimmten Zeichensystemen. Die gestellte Frage bezieht sich auf Darstellungen in einem den Nutzern des AIS geläufigen Zeichensystem, auf die sogenannte externe Informationsdarstellung. Im Interesse einer rationellen Arbeitsweise werden hierzu in der Praxis sogenannte Zwischensprachen verwendet, die sowohl dem Nutzer verständlich als auch für die Eingabe in die EDVA geeignet sind. Ein

Beispiel bildet eine Sprache, deren elementare Ausdrücke Informationseinheiten IE, das heißt n-Tupel von Merkmalsnamen mit zugehörigen Merkmalswerten sind:

$$IE = \{ (M_j, v_j) \mid j = 1, 2, \dots, n \}$$

mit M_j - Merkmalsname, v_j - Wert des Merkmals M_j

oder

$$IE = (v_1, v_2, \dots, v_n).$$

Die letztere Darstellung setzt voraus, daß bei der Verwendung der Informationseinheit die Folge der Merkmalsnamen bekannt ist.

Die am weitesten verbreiteten und von vielen DBBS zugelassenen Formen von Informationseinheiten sind solche mit konstanter und mit variabler Anzahl von (Name, Wert)-Tupeln sowie als Spezialfall variabler Anzahl solche mit sogenannten Wiederholungsgruppen (wiederholt auftretende Gruppen von ≥ 1 Merkmalen). Verschiedene DBBS gestatten darüber hinaus die Behandlung unstrukturierter Zeichenketten.

Vor dem Aufbau eines Datenbanksystems ist zu entscheiden:

- Was ist die elementare Einheit der Information, die als Ganzes zwischen externem Speicher und Arbeitsbereichen des Hauptspeichers transportiert wird, und auf welche Bestandteile einer solchen Einheit sollen Suchoperationen Bezug nehmen können?
- Welche Strukturierung der gegebenen Informationsbestände (abbild- oder programmorientierte Struktur, Zuordnung zu Dateien) ist aus der Sicht der Nutzer zweckmäßig?

Aus Abb. 1 ist die Rolle eines Datenbanksystems in einem (programmtechnisch realisierten) AIS ersichtlich. Dieses Bild spiegelt die für die dritte Rechnergeneration auf dem Niveau des Betriebssystems DOS/ES typischen Arbeitsweisen bei der Lösung von Anwendungsaufgaben wider.

Danach können Aufgaben sowohl allein mit Hilfe des gegebenen Betriebssystems als auch mit Hilfe der das allgemeine Betriebssystem nutzenden Erweiterungen durch das Datenbasis- bzw. Programmbasisbetriebssystem bearbeitet werden. Der Begriff Datenbasis wird allgemeiner als der der Datenbank aufgefaßt. Unter der Datenbasis wird die Gesamtheit aller Datenbestände eines AIS verstanden. Gegenwärtig existieren in einer Datenbasis oft mehrere von jeweils einem DBBS verwaltete Datenbanken. In Analogie zum Datenbasisbetriebssystem werden zum Programmbasisbetriebssystem solche Programme gerechnet, mit denen Funktionen der Programmerzeugung realisiert werden, die nicht im allgemeinen Betriebssystem enthalten sind. Beziehungen zwischen Datenbasis- und Programmbasisbetriebssystem im Sinne der wechselseitigen Nutzung bestimmter Funktionen bestehen bei den DOS/ES-Systemunterlagen nicht; sie sind jedoch in den OS/ES-Systemunterlagen realisiert.

Es kann eingeschätzt werden, daß in Zukunft, beginnend bei Reihe II des ESER, die gegenwärtig vorhandene Heterogenität des Betriebssystems und der zu dessen Steuerung eingesetzten Sprachen schrittweise überwunden werden wird. Dies wird sich u. a. in der einheitlichen Behandlung von Daten- und Programmbasis und der Verwirklichung des Bankprinzips auch für Programme, in der Integration der gegenwärtig existierenden Betriebssystemerweiterungen in ein einheitliches Betriebssystem sowie in der Einführung eines abgestimmten, auf einer einheitlichen Konzeption basierenden Systems von Sprachen äußern.

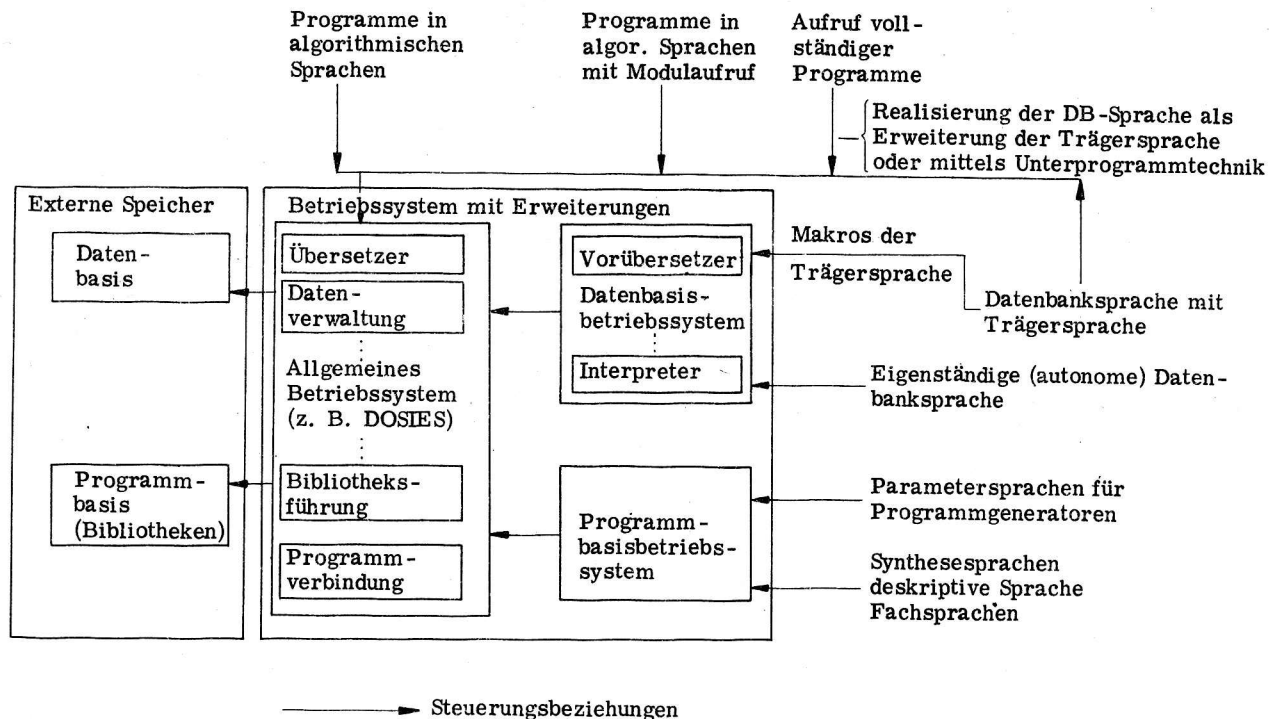


Abb. 1: Die Rolle eines Datenbanksystems in einem AIS

In Tabelle 1 sind die für die Klassifizierung von DBBS ausgewählten Merkmale zusammen mit häufig auftretenden Merkmalswerten zusammengestellt.

Die Merkmale "Nutzerklassen", "Datenmodelle" und "Datenunabhängigkeit" stehen in engem Zusammenhang. Sie charakterisieren die "Nutzerfreundlichkeit" eines DBBS. Als Nutzerklassen werden gewöhnlich neben denen der Systemprogrammierer und des "Datenbankadministrators" die Klassen der Anwendungsprogrammierer, der Fachsprachennutzer und der parametrischen Nutzer (d. h. die Klassen der eigentlichen Nutzer) unterschieden. Im Idealfall wird jeder Nutzerklasse von einem DBBS eine eigene, ihrem Aufgabenprofil und Qualifikationsniveau adäquate Datenbanksprache zur Verfügung gestellt. Die heutigen Systeme sind jedoch meist auf einen Teil der genannten Klassen spezialisiert. Die Systemprogrammierer verwenden in der Regel diejenige Sprache, in der das DBBS programmiert ist. Von einer Unterstützung der Klasse des Datenbankadministrators, der für die Festlegung der internen Datenstrukturen zuständig ist, kann gesprochen werden, wenn mit dem DBBS eine spezielle Datendefinitionssprache (DDS) gegeben ist (s. /1/, /4/, /5/).

Als Datenmodell wird diejenige (gegenüber der DBS-internen Darstellung mehr oder weniger vereinfachte) Modellvorstellung von den in der Datenbank enthaltenen Daten bezeichnet, von der ein Nutzer bei der Kommunikation mit der Datenbank ausgeht und über die er demzufolge verfügen muß. Je detaillierter und damit komplizierter ein Datenmodell ist, desto größer sind die Anforderungen an die rechentechnische Qualifikation eines Nutzers, aber auch die Möglichkeiten der gezielten Ausnutzung der DBBS-Funktionen zur Erhöhung der Programmeffektivität. Zu den komplizierten Modellen sind insbesondere die Baum- und die Netzwerkmodelle zu rechnen; sie werden deshalb vorwiegend von der Klasse der Anwendungsprogrammierer verwendet. Die heutigen DBBS unterstützen in der Regel nur eines der genannten Datenmodelle (wobei das Netzwerkmodell als Verallgemeinerung des Baummodells zu verstehen ist). Dies hat zur Folge, daß oft Informationsbestände in unnatürliche Schemata gepreßt werden müssen. Schwerwiegender ist jedoch die Tatsache, daß deshalb in ein und demselben

Anwendungsbereich meist mehrere verschiedene Datenbanksysteme betrieben werden müssen, um allen Anforderungen gerecht werden zu können, und daß in diesen Systemen auch gleiche bzw. abgeleitete Informationen verwaltet werden.

Tabelle 1: Merkmale von Datenbankbetriebssystemen

M 1: Variabilität der elementaren Informationseinheit

- 1 - konstante Anzahl von (Name, Wert)-Tupeln
- 2 - variable Anzahl von (Name, Wert)-Tupeln
- 3 - Informationseinheit mit Wiederholungsgruppen
- 4 - unstrukturierte Zeichenketten
- 5 - Kombinationen von Segmenten verschiedenen Typs

M 2: Anwendungscharakteristik

- 1 - operationelles System
- 2 - Recherchesystem

M 3: Nutzerklassen

- 1 - Systemprogrammierer
- 2 - Datenbankadministrator
- 3 - Anwendungsprogrammierer
- 4 - Fachsprachennutzer
- 5 - parametrischer Nutzer

M 4: Datenmodelle

- 1 - unstrukturierte Menge von Datensätzen
- 2 - hierarchisches bzw. Baummodell
- 3 - Netzwerkmodell
- 4 - invers strukturiertes Modell
- 5 - Matrixmodell
- 6 - Menge von Relationen

M 5: Datenunabhängigkeit

- 1 - Unabhängigkeit vom Typ der Datenelemente
- 2 - Unabhängigkeit von der Satzstruktur
- 3 - Unabhängigkeit vom Zugriffsweg

M 6: Programmtechnische Realisierung der Datenbanksprache

- 1 - eigenständige oder autonome Datenbanksprache
- 2 - Datenbanksprache mit Trägersprache
 - a) echte Spracherweiterung/Kompiliererweiterung
 - b) Datenbankoperationen als Makros der Trägersprache/
Vorübersetzer, Interpreter
 - c) Datenbankoperationen als Unterprogramme der Trägersprache

Unter Datenunabhängigkeit versteht man die Unabhängigkeit der in einer Datenbanksprache formulierten Informationsforderungen von der konkreten Darstellung der Informationen als Daten in der Datenbank. Die programmtechnische Realisierung der Datenunabhängigkeit beruht auf der Speicherung eines mehr oder weniger umfangreichen Teiles der Datenbeschreibung zusammen mit den eigentlichen Daten in der Datenbank. Der erreichte Grad der Datenunabhängigkeit (s. auch /4/, /5/) äußert sich u. a. in der Größe des Detailliertheitsunterschiedes zwischen dem Datenmodell und der vollständigen Beschreibung der internen Datendarstellung (in der CODASYL-Terminologie als "Schema" bezeichnet).

Das "Nutzerinterface" eines DBBS wird auch wesentlich durch die Art der programmtechnischen Realisierung der Datenbanksprache bzw. von dem gebotenen Sprachspektrum beeinflusst. Eigenständige oder autonome (engl. self-contained) Datenbanksprachen eignen sich besonders für die Klassen der Fachsprachen- und parametrischen Nutzer. Datenbanksprachen mit einer (algorithmischen) Trägersprache (engl. host-language) entsprechen dagegen mehr den Bedingungen der Klasse der Anwendungsprogrammierer. Für die letzteren Sprachen gibt es verschiedene Möglichkeiten der Implementierung (siehe Abb. 1).

Weitere für die Beurteilung eines DBBS wichtige Merkmale beziehen sich auf Datenschutz und Datensicherheit in der Datenbank, auf das verfügbare Operationsspektrum der Datenbanksprachen, auf Möglichkeiten der interaktiven Arbeit und der Fernverarbeitung sowie die Variabilität und Anpassungsfähigkeit der internen Datenstrukturen. Auf diese Merkmale kann hier nicht näher eingegangen werden (s. /1/, /4/, /5/).

Tabelle 2 gibt anhand der oben angeführten sechs Merkmale eine Übersicht über einige der in der DDR für EDVA des ESER verfügbaren DBBS. Das Ziel ist dabei nicht eine redundanzfreie oder "orthogonale" Beschreibung; zwischen den Werten verschiedener Merkmale bestehen offensichtlich Korrelationen. Die Merkmalswerte wurden so gewählt, daß sich eine hohe Funktionsbreite im Auftreten mehrerer Werte von Merkmalen für ein und

dasselbe DBBS äußert. Aus Tabelle 2 ist ersichtlich, daß insbesondere das unter dem Betriebssystem OS/ES anwendbare Datenbankbetriebssystem/ Robotron wesentliche Fortschritte in Richtung auf eine funktionelle Integration ermöglicht.

Name des DBBS bzw. DBS, Be- triebssystem(e)	M 1	M 2	M 3	M 4	M 5	M 6
	Variabilität der elementa- ren IE	Anwendungs- charakte- ristik	Nutzer- klassen	Daten- modelle	Datenunab- hängigkeit	Realisierung der DB-Sprache
SAWI DOS/ES	1	2	1, 2, 3, 4	1	1, 2	1
BASTEI DOS/ES	1	1	1, 3, 5	3	1	2b
AIDOS DOS/ES und OS/ES	2, 3 (Deskriptoren als Wiederho- lungselemente)	2	1, 2, 4, 5	4	1, 2, 3	1
DASY DOS/ES	1 (innerhalb einer Matrix)	1	1, 2, 3	5	1	1, (2c)
DBS/R (1. Version) OS/ES	1, 2, 3, 5 (4 als Segment)	1, (2)	1, 2, 3, 4	1, 3, (4)	1, 2	1, 2

Tabelle 2: Beschreibung von ESER-Datenbankbetriebssystemen

(Codierung der Merkmalswerte siehe Tabelle 1; Werte in Klammern bedeuten Gültigkeit mit Einschränkungen)

3. Einsatzmöglichkeiten der Datenbanktechnologie in der Digitalgraphik

Bei der Beurteilung der Einsatzmöglichkeiten von Datenbanksystemen in der Digitalgraphik ist es wichtig zu beachten, daß die von einem DBBS unterstützten Datenstrukturen im allgemeinen nicht mit den in der Digitalgraphik verwendeten Datenstrukturen gleichgesetzt werden dürfen. Übereinstimmung besteht in den (abstrakten) Datenmodellen; in der Digitalgraphik spielen Listen- und spezielle Netzwerkstrukturen (Bäume, Ringe) eine große Rolle. Die elementaren (als Ganzes zu transportierenden) Informationseinheiten sind jedoch in einer Datenbank im allgemeinen aus Effektivitätsgründen wesentlich umfangreicher als die "Items" einer mit Programmen der Digitalgraphik zu manipulierenden, gewissermaßen mikroskopischen Datenstruktur. Beim Einsatz von Datenbanksystemen in der Digitalgraphik entsteht folglich das Problem einer zusätzlichen makroskopischen Strukturierung von Beschreibungen graphischer Gebilde zum Zwecke der Behandlung durch ein DBBS.

Prinzipiell können die folgenden Arbeitsweisen mit externen Direktzugriffsspeichern unterschieden werden:

1. Verwendung des externen Speichers als programmgesteuerte Hauptspeichererweiterung; hierzu ist kein DBBS erforderlich. Eine Vereinfachung für den Nutzer wird hier möglicherweise durch Einführung der virtuellen Speichertechnik eintreten.
2. Speicherung, Verwaltung, Wiederfindung und Bereitstellung von Beschreibungen graphischer Gebilde als Ganzheiten, etwa im Sinne eines Archivspeichers, mit folgenden beiden Bedienungsfunktionen:
 - Zugriff über jeweils ein Haupt- oder Nebenordnungskriterium;
 - Kriterienrecherche (siehe Recherchesystem).
3. Speicherung, Verwaltung, Wiederfindung und Bereitstellung von Beschreibungen bzw. Modellen elementarer graphischer Gebilde, etwa als Bausteine für die Synthese neuer Objekte. Hier liegen gleichzeitig die Anwendungscharakteristiken eines operationellen Systems (Darstellung der verschiedenen Kopplungsmöglichkeiten von Bausteinen durch

Adreßverkettung) und eines Recherchesystems (Kriterienrechercher nach Bausteinen mit vorgegebenen Eigenschaften) vor.

4. Komplexe Anwendungen, die durch Verbindung von Aufgaben der Digitalgraphik z. B. mit Konstruktionsberechnungen oder Aufgaben der technischen Produktionsvorbereitung und durch entsprechende Beziehungen zwischen den beteiligten Datenbeständen gekennzeichnet sind. Hier werden wieder die Funktionen von operationellen und Recherchesystemen benötigt, außerdem flexible Möglichkeiten der Verbindung zwischen Programmen in der Datenbanksprache und in algorithmischen Sprachen.

Es kann festgestellt werden, daß von den in Tabelle 2 beschriebenen DBBS das System DBS/R den Anforderungen der Digitalgraphik am besten gerecht wird. Mit DBS/R sind beispielsweise sowohl der Zugriff zu Dateneinheiten über ein Haupt- oder Nebenordnungskriterium, der Zugriff zu adreßverketteten Satzfolgen als auch die Kriterienrecherche möglich. Die letztere ist bei der ersten Version von DBS/R nur bei nicht zu großen Datenbeständen effektiv, da sie wie bei SAWI durch sequentielles Suchen realisiert wird. DBS/R besitzt weiterhin eine verglichen etwa mit BASTEI wesentlich höhere Variabilität der möglichen Netzwerkdatenmodelle. Die Datenmanipulationssprache von DBS/R ist eine eigenständige Sprache. Von Programmen in dieser Sprache kann an definierten Austrittspunkten die Steuerung an katalogisierte Assembler- oder PL/1-Programme übergeben werden. Auch die umgekehrte Arbeitsweise, d. h. mit einem in einer algorithmischen Sprache geschriebenen Hauptprogramm und Übergabe der Steuerung an DBS/R-Programmeile, ist möglich. Die DBS/R-Datenmanipulationssprache kann deshalb auch als Sprache mit Trägersprache aufgefaßt und verwendet werden. Schließlich wird DBS/R durch ein Exekutivsystem ergänzt, das einer Vielzahl von Nutzern mittels Datenfernverarbeitung den Zugriff zu zentral gespeicherten Daten und Programmen, also den Teilhaberbetrieb ermöglicht /8/.

Abschließend sei noch auf ein in der Sowjetunion für ESER-Anlagen und das Betriebssystem DOS/ES entwickeltes DBBS mit der Bezeichnung NABOB hingewiesen (s. /1/, /7/). Dieses System erscheint ebenfalls für die

Nutzung im Anwendungsbereich der Digitalgraphik geeignet. Es läßt Verkettungen zwischen den gespeicherten Sätzen der Typen NEXT, PRIOR und OWNER zu (s. Abb. 2). Seine auf der Trägersprache PL/1 basierende

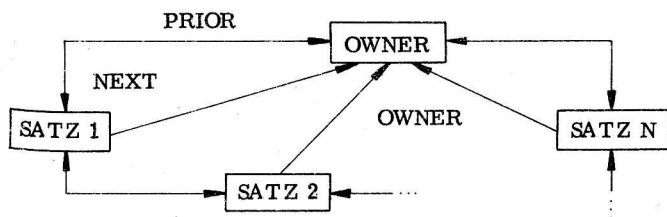


Abb. 2: Datenmodell des DBBS NABOB

Datenmanipulationssprache gestattet die Verwendung der für die Listenverarbeitung wichtigen Operatoren FINDFIRST, FINDLAST, FINDNEXT, FINDPRIOR, FINDOWNER, FINDi (finde den i-ten Satz relativ zum OWNER - Satz).

Literatur

- /1/ Fridlender, F. L.; Mamaev, E. A.: Osnovnye principy postroenja SUBD NABOB. Algoritmy i organizacija rešenija ekonomičeskich zadač, Vypusk 5, 66 - 88
- /2/ Hugenberg, D.; Kaiser, E. -V.: Beurteilungskriterien für den Einsatz von Datenbank-Management-Systemen (DBMS) in entscheidungsorientierten Informationssystemen. BIFOA-Arbeitsbericht 72/11. Wison-Verlag, Köln, 1972
- /3/ Maurer, H.: Datenstrukturen und Programmierverfahren. B. G. Teubner, Stuttgart, 1974
- /4/ Olle, T. W.: Current and Future Trends in Data Base Management Systems. Information Precessing 1974, 009 - 1006. North-Holland Publ. Comp., Amsterdam, 1974
- /5/ Schmid, H. A.: Architektur und Implementierung von Datenbank-systemen. GMD-Spiegel 3/76, 78 - 122 (1976)
- /6/ Senko, M. E.; Altmann, E. B.; Astrahan, M. M.; Fehder, P. L.: Data structures and accessing in data-base systems I - Evolution of information systems. IBM Systems Journal 30 - 44 (1973)
- /7/ Shadan, N. W.: Jazyk manipulirovanija dannymi SUBD NABOB. Algoritmy i organizacija rešenija ekonomičeskich zadač, Vypusk 5, 88 - 100
- /8/ VEB Kombinat Robotron: Datenbankbetriebssystem Robotron DBS/R. Einführungsschrift (Stand 31. 8. 1974)

Eingegangen: 31. 12. 1976

Anschrift des Verfassers:

Prof. Dr. D. Schubert

Technische Universität Dresden
Sektion Informationsverarbeitung

DDR - 8027 Dresden, Mommsenstr. 13

Industrielle Anwendungsprogramme für die Digitalgraphik

1. Einleitung

In unserem Institut legen wir viel Wert darauf, daß wir unsere theoretischen Ergebnisse in der Praxis anwenden und in den Dienst der Produktion stellen. Diese Bestrebung hat auch Rückwirkungen auf unsere wissenschaftliche Arbeit.

Das in unserem Institut entwickelte graphische Bildschirmgerät GD 71 sowie die dazugehörige Basis-Software und graphische Software sind die Merkmale unserer vorhergenannten Bestrebungen. Wir waren bemüht, die Kennzeichen der Hardware und Software auf wissenschaftlichem Niveau verallgemeinernd zu konzipieren. Bei der Ausführung der Arbeit wurde immer die Möglichkeit der konkreten Anwendung und die daraus folgenden Bedingungen und Ansprüche in Betracht gezogen.

Dieser Gesichtspunkt ermöglichte, daß wir vielfältige industriell verwendbare Programme entwickelten, die einerseits die Planungs- und Durchlaufzeiten vermindern, andererseits die Produktionssicherheit erheblich verbessern. Die zu erwartenden Ansprüche der industriellen Verwendung wurden auf mehreren Gebieten untersucht. Wir beschäftigen uns mit kartographischen Problemen und mit Luftfahrtskontrollproblemen. Die Traditionen unseres Institutes wie auch die Bedürfnisse der Industrie bestärkten uns, unsere Anwendungsprogramme für die elektronische Industrie und die Maschinenbauindustrie herzustellen.

2. Anwendungsprogramme für die Elektronik

Unsere ersten, sich mit der Elektronik befassenden Programme waren versuchsbedingt. Diese Versuchsbedingtheit ergab sich in erster Linie aus der Tatsache, daß wir ohne Plattenspeicher, gemessen an der Wirklichkeit, mit beschränkten Bestandteilmengen arbeiten mußten.

Mit Hilfe des Programms für logische Schaltungen wollten wir die Betriebsdokumentation erleichtern. Das Stromkreisbeschreibungsprogramm arbeitete ebenfalls mit beschränkten Bestandteilmengen, obwohl die Bestandteillarten komplett waren. Die Ausgabedaten dieses Programms sind für Verarbeitungen auf Großrechnern geeignet. Es ist anschließbar an analytische Programme. In diesem Fall sind die Ergebnisse der Analyse wie Spannungsablauf, Phasenverschiebung, Verstärkung, Zeitdiagramm auf dem Bildschirm des GD 71 projizierbar. Ferner kann es mit dem planenden Programm für gedruckte Schaltungen gekoppelt werden. Die sich daraus ergebenden Daten sind ebenfalls auf den Bildschirm projizierbar.

Nach dem Abschluss der Versuche gingen wir zur Planung und Verwirklichung von betriebsmäßig benutzbaren Programmen für die elektronische Industrie über. Das endgültig realisierte Großrechnerprogramm für Planung gedruckter Schaltungen, das schon längere Zeit in der Industrie verwendet wird, behandelt nur integrierte Schaltungen. Im Falle von multi-komplizierten Schaltungen nimmt die geplante Platte einen größeren Platz als erlaubt ein. Wegen dieser Eigenschaften wurde es erforderlich, daß wir zum Planungsprogramm noch ein interaktives Kontroll- und Korrekturprogramm erstellten. Dieses Programm dient zur Veranschaulichung von auf dem Großrechner geplanten gedruckten Schaltungsplatten und zur Abschreibung von Streichungen (im weiteren Wischungen) und Einfügungen (im weiteren Einstichen). Es visualisiert weiterhin handlich geplante und geschriebene Schaltungen. Von ihm werden auch die im Laufe der Messungen benötigten Korrekturen angesprochen und ausgeführt. Auf Grund der Ausgabedaten des Programms wird die gedruckte Schaltungsplatte erzeugt.

Im Interesse einer wirksamen und fehlerfreien Arbeit kontrolliert das Programm ferner, ob beim Wischen kein Potential abgerissen ist und beim Einstich, die neuplatzierten Leitungen und Bohrungen keinen Kurzschluß verursachen. In beiden Fällen werden Fehlermeldungen an den Operator weitergegeben. Wir sind überzeugt, daß diese Kontrolle eine wichtige und fundamentale Neuerung des Programms ist.

3. Das Korrekturprogramm

Die Eingabeinformation des Programms ist die geometrisch-technologische Beschreibung des Druckbildes. Sie enthält die Liniendicke, die Seitenbezeichnung, die absoluten Koordinaten der einzelnen Punkte, die in einigen Punkten zu entwickelnden geometrischen Figuren und die technologischen Parameter des Elementes.

Die Eingabedaten werden vom Programm gemäß seiner spezifischen inneren Datei umgewandelt, so daß das neue Modell des Druckbildes die vorhergehenden Informationen vollkommen enthält und ein schnellerer und wirksamerer Ablauf gesichert ist.

Hier möchten wir bemerken, daß das System 16 unterschiedliche technologische Parameter behandeln kann. Diese Parameter können den Ansprüchen des Nutzers entsprechend mit einem anderen Programm umgeschrieben werden.

Das Programm besteht aus dem Steuermodul und aus 6 selbständigen Modulen, die auch die Segmente des Overlays sind. Wie aus Abb. 1 ersichtlich, sind es die Segmente: Eingabe, Optimierung, Potenzierung, Modifizierung, Verschiebung, Ausgabe. Sämtliche Segmente können einander aktivieren. Der Aufruf erfolgt durch den Steuermodul (in Abb. 1 Steuerprogramm).

Der Übersichtlichkeit halber zeigen wir in Abb. 1 nur die möglichen Verbindungen des Modifizierungssegments. Die Segmente holen ihre Daten von der Magnetplatte, die Ergebnisse werden dort abgespeichert. Dadurch war es möglich, das Maß und die Kompliziertheit der gespeicherten Schaltungen praktisch unbegrenzt zu machen.

Wir müssen unter den Segmenten das Modifizierungssegment besonders hervorheben, da die übrigen Segmente nur in der Speicherung des Moduls Veränderungen hervorrufen. Das Modifizierungssegment kann auf dem Modell Veränderungen vornehmen, das Modell veranschaulichen, und man kann Druckelemente wischen und einstechen.

Das Eingabesegment bekommt die Eingabedaten entweder von einem Lochstreifen oder von einer Magnetplatte. Es kontrolliert sie auf syntaktische Richtigkeit und erzeugt das zur Weiterarbeit benötigte Modell.

Danach speichert es, wenn die Daten fehlerfrei sind, das Resultat auf der Magnetplatte.

Das Optimierungssegment speichert die einzelnen Linien so um, damit der Elektronenstrahl des Bildschirms während der Veranschaulichung den minimalen Weg zurücklegt. Auf diese Weise wird auch die nötige Produktionszeit der gedruckten Schaltungsplatte minimalisiert und dem Flackern des Displaybildes kann vorgebeugt werden.

Das Potenzierungssegment bestimmt die Teilmengen der galvanisch zusammenhängenden Punkte (jede Teilmenge wird mit einer Nummer versehen). Auf diese Weise wird es im Laufe der Modifizierung möglich, Fehlermeldungen (Kurzschluss, Potenzialabbruch) auszugeben.

Die im ausgewählten Rechteck befindlichen Punkte des Druckbildes werden vom Verschiebungssegment entsprechend dem gewünschten Wert verschoben.

Das Ausgabesegment produziert die Liste und den Lochstreifen des geometrisch-technologisch beschriebenen fertigen Bildes und speichert das Ergebnis auf die Magnetplatte. Die gespeicherten Daten ergeben die Eingabedaten für den Postprozessor, der dann den Lochstreifen für die gedruckte Schaltungsplatte herstellende NC-Maschine erzeugt.

Der operative Speicher wird gemäß Abb. 2 ausgenutzt. Hier sind die mit Stern bezeichneten Gebiete die, von welchen ein oder mehrere zugunsten einzelner Programmsegmente befreibar sind. Dadurch wurde es möglich, die Zahl der Diskübertragungen bei einzelnen Programmen erheblich zu reduzieren. Wie es zu erwarten war, war das Speichergebiet für die Subsegmente des Modifizierungssegments am engsten. Das bedeutet, daß das

laufende Programm, das graphische Programmsystem, das Modell des veranschaulichten Bildteiles und die Bildbeschreibung gleichzeitig im operativen Speicher Platz finden mußten. Dies ist der Grund dafür, daß die Korrektur nur auf dem 60 mal 60 Rasterteil des Druckbildes vorgenommen werden kann. Und trotzdem mußten wir das Modifizierungsprogramm in mehrere Subsegmente zerlegen.

Die Subsegmente des Modifizierungssegments sind: Veranschaulichung, Vergrößerung, Wischung und Einstechung. Zur Erleichterung der Korrektur dienen die Subsegmente: Anordnung des IC Gehäuses, Aushebung und Auslöschung, Modifikation der Liniendicke und Abfragen.

Zu den Wischungs- und Einstechungssegmenten müssen wir bemerken, daß mehrere Elemente nacheinander wischbar bzw. einstechbar sind. Die Annahme der Wischung oder der Einstechung muß besonders angezeigt werden. Angenommene Korrekturen werden auch durch den Zeilendrucker ausgeschrieben. Die Korrektur geschieht immer auf dem Modell und wird in die Bildbeschreibung aufgenommen. Korrekturen und Wischungen werden an das Modell angekoppelt. Der Einbau und die Einordnung in das gesamte Modell geschehen durch Aufruf des Optimierungs- und Potenzierungssegments.

Und nun einige Worte über das interne Modell der Druckschaltungsplatte. Jedes Element ist in drei Wörtern zu je 2 Byte gespeichert (s. Abb. 3).

Das graphische Modell enthält die Adressen und die Typen der Elemente entsprechend der Vorschrift des GTU-Systems.

4. Maschinenbauindustrie

Die Anwendung der Digitalgraphik in der Maschinenbauindustrie ist stark mit unserer jahrelangen Modernisierungs- und Automatisierungsarbeit in der Maschinenbauindustrie verbunden. Diese Arbeit wurde immer Hand in Hand mit den Kollegen der Csepe-Werkzeugmaschinenfabrik verrichtet.

Die Einführung der numerischen Methoden in der Csepel-Werkzeugmaschinenfabrik fing mit der Implementierung APT-artiger Programmiersprachen und mit dem Schreiben entsprechender Postprozessoren an. Infolge dieser unserer Tätigkeit waren die Grundlagen der rechnerprogrammierten Metallbearbeitung festgelegt.

Gleichzeitig beschäftigen wir uns mit Problemen, wie man die Benutzung der APT-artigen Programmiersprachen erleichtern könnte. Zwei Spezialprogramme wurden zu diesem Zweck entwickelt.

Eines zur Erleichterung des Teileprogrammschreibens und das andere zur Kontrolle der CLDATA. Das Kommunikationsmittel zwischen dem Programmierer und der Programme ist in beiden Fällen das Bildschirmgerät GD 71. Mit Hilfe des Programms zur Erleichterung des Teileprogrammierens kann man im Dialog APT-artige Teileprogramme syntaktisch fehlerfrei schaffen. Hier wurde aber das Bildschirmgerät als alphanumerisches Mittel benutzt, deshalb wird dieses Programm nur als einer unserer ersten Schritte auf dem Gebiet der Maschinenbauindustrie erwähnt.

4.1. Digitalgraphische Kontrolle

Unser für die Maschinenbauindustrie geschaffenes erstes digitalgraphisches Programm, der Display-Postprozessor, war für die Kontrolle der prozessierten Teileprogramme ausgearbeitet. Diese schnelle qualitative Kontrolle ist besonders wichtig, da hierdurch stundelange Probearbeitungen an kostspieligen Werkzeugmaschinen erspart werden können. Folglich ergibt es sich, daß dieses Programm hauptsächlich in der Produktionsvorbereitung angewendet wird; es kommt aber auch als ein Baustein der rechnergestützten Konstruktion (CAD) in Betracht.

Die Hauptzielsetzung dieses Programmes war die Veranschaulichung der Werkzeugbewegung, deren Angaben in den CLDATA (Übergabegrößen des Programms) festgelegt sind. Die dynamische Visualisierung der Reihenfolge der Bearbeitung, eine nützliche und charakteristische Information, war eine unserer Zielsetzungen. Die Probleme ergaben sich aus der weiteren Forderung, daß wir diese Daten in einfache, gut übersichtliche Figuren abbilden wollten. Die Frage der Übersichtlichkeit erhob sich auch

aus der 2 1/2-dimensionalen Bearbeitung, die auf dem zweidimensionalen Bildschirm dargestellt sein sollte. Dieser Grundwiderspruch beinhaltet aber auch die Lösung der Probleme. Die 2 1/2-dimensionale Bearbeitung ist typisch für Terrassenbearbeitungen geeignet. Veranschaulicht man die Terrassen in einzelnen Stufen, so erreicht man die zweidimensionale Abbildung.

Von diesem Gedanken ausgehend werden die zur Bearbeitung eines Teiles benötigten Werkzeugstrecken in mehreren Bildern veranschaulicht. Im Falle der Bohr- und Fräsebearbeitungen weicht die Organisation der Bilder ein wenig voneinander ab.

Im Falle der Bohrbearbeitungen werden die Bewegungen der Werkzeuge beziehungsweise die Strecken des Werkzeugmittelpunktes, deren z-Koordinaten gleich sind, in einem Bild abgebildet.

Werkzeugheraushebung aus der aktuellen (X, Y)-Ebene (Terrasse) bildet eine Ausnahme. Die Punkte der Aushebung werden am Bilde markiert, und die Z-Koordinaten der Aushebung erscheinen an der linken Seite des Bildschirms. In den Bearbeitungspunkten erscheinen auch die Werkzeugumkreise mit punktierter Linie.

Die Organisation der Bilder im Falle der Fräsbearbeitungen ist nicht nur bewegungsabhängig, sondern auch werkzeugdiameternabhängig. Das bedeutet, daß die Bewegung, deren Z-Koordinaten gleich sind und welche mit demselben Werkzeug (d. h. Werkzeugdiameter) durchgeführt sind, in eine Figur abgebildet werden. Werkzeugheraushebung wird in diesem Falle ebenso behandelt wie bei den Bohrbearbeitungen.

Wenn die Werkzeugmaschine mit Streckensteuerung versehen ist, wird die Zahl der Approximationsgeraden der einzelnen Kurven am Bildschirm auch erscheinen. Wenn die Werkzeugmaschine mit Bahnsteuerung ausgerüstet ist, finden wir natürlich keine Approximationsgeraden. Die Bilder selbst enthalten in beiden Fällen die analytischen Kurven.

Berücksichtigend, daß die einzelnen Bilder sich zu einzelnen Werkzeugen anschließen, könnte man den Werkzeugumkreis auch abbilden.

Die Figuren enthalten auch Ergänzungsinformationen wie z. B. den Maßstab der Abbildung und den Z-Wert der veranschaulichten Ebene.

Bisher besprachen wir, wie die Bildorganisation auf Grund der im CLDATA gespeicherten Werkzeugmittelpunktwegangaben verwirklicht wird. Das Softwaresystem des Displaygerätes GD 71 ermöglicht auch die dynamische Behandlung der Bilder. Es werden die Angaben der Bilder mit Hilfe der Grundsoftware des GD 71 im Speicher des Gerätes eingelagert. Die gespeicherten Daten werden dann in die Rezirkulation ein- oder ausgeschaltet. So kann man einige Bilder noch einmal auf den Bildschirm rufen oder sogar mehrere Bilder gleichzeitig erscheinen lassen.

Der Wiederaufruf einer oder mehrerer Bilder wird durch einen Mensch-Rechner-Dialog verwirklicht. Diese Eigenschaft des Systems ist besonders in der Kontrolle der Terrassentechnik vorteilhaft.

In ähnlicher Weise kann auch der Bearbeitungsablauf gut kontrolliert werden, wenn man die Bilder verschwinden und die Werkzeugmittelpunktbewegung Strecke für Strecke wieder erscheinen läßt. Diese Funktion kann mit den Funktionstasten ausgelöst werden, aber für den schnelleren Aufruf der Strecken kann auch der Lichtstift benutzt werden.

Der oben beschriebene Display-Postprozessor ist im täglichen Gebrauch zur Kontrolle der Teileprogramme in der Numerischen Werkstatt der Csepel-Werkzeugmaschinenfabrik und benutzt das graphische Programmsystem GTU für die Visualisierung.

Dieses System ermöglichte uns, daß wir mit Hilfe der Bildteilvergrößerung von der quantitativen Kontrolle zur qualitativen weiterkommen konnten. Nachdem wir APT-artige Sprachen in Nutzung nahmen, planten wir die Realisierung eines integrierten Material- und Datenverarbeitungssystems. Als ersten Schritt bauten wir ein 8 Maschinen steuerndes DNC-System aus, dessen Rechenzentrum aus zwei Kleinrechnern besteht. Die eine EDV-Anlage dient technischen Produktionsplanungszwecken. Die letztere steht zeitweilig mit der Rechnerzentrale des Betriebes in Verbindung. Dementsprechend verfügt die Rechnerzentrale der Werkstatt über einen ziemlich großen und mannigfaltigen Softwareapparat. Hier möchten wir zwei graphische Systeme

besprechen. Das Programmsystem für Drehmaschinen dient für die interaktive Beschreibung der Geometrie und Technologie der Teile mit dem Ziel, die Datei der Bearbeitungssteuerung zustandezubringen.

Das Preßwerkzeugplanungssystem ist ein CAD-System, das ausgehend von der Beschreibung des zu pressenden Teiles das Preßwerkzeug interaktiv plant und den werkzeugbearbeitungssteuernden Lochstreifen anfertigt.

Die mit diesen Systemen hergestellte Bearbeitungsdatei kommt hiernach in die Datenbasis des zweiten Rechners des integrierten Systems, wenn es das tägliche Produktionssteuerungsprogramm vorschreibt.

4.2. Digitalgraphisches Programmieren von Drehmaschinen

Das digitalgraphische Drehmaschinenprogramm ist ein Teileprogrammiersystem. Seine Eingabedaten sind die interaktiv beschriebenen Daten des Teiles. Die Ausgabedatei ist der international standardisierte CLFILE.

Die Benutzung des Systems fängt mit der Angabe der Initialdaten an wie der Hauptmaße des Teiles, des Spannzeuges, der Einspannung und der Bearbeitungswerkzeuge. Dann können wir die Geometrie des Rohrstückes und des Fertigstückes interaktiv beschreiben. Die Technologie der Bearbeitung besteht teils aus der Angabe von Daten, teils aus der Angabe der Geometrie der Schnittweise des zu erzeugenden Halbfertigstückes.

Aus dieser Geometrie errechnet das System die Bewegungsstrecke der theoretischen Spitze des Werkzeuges. Die Veranschaulichung der beschriebenen Geometrie sichert dem Nutzer kontinuierlich die Selbstkontrollmöglichkeit. Die vom Nutzer angenommenen Daten werden vom System gespeichert und ein selbständiges Programmteil sorgt für die Ausgabe des CLFILE. Diese Datei wird von traditionellen Postprozessoren verarbeitet.

Das ganze System besteht also aus drei seriell gekoppelten Programmteilen: dem interaktiven Teil, dem Ausgabeteil und dem Postprozessor.

Mit dem Koppeln eines Transformierprogramms können auch andere CL-Dateien auf die Form der inneren Datei des Systems gebracht, veranschaulicht und verbessert werden. Dadurch schalteten wir die in den Rechnern des Betriebes implementierten APT-Sprachen in das integrierte Produk-

tionssystem ein, wobei die im übrigen starre Anwendung flexibler wurde. Zum vollkommenen Ausbau des Systems gehört das Werkzeugbeschreibungs- und Suchprogramm, das die in der Datenbank des integrierten Produktionssystems gespeicherten Werkzeugdaten zugänglich macht und den Programmierern in der Auswahl des entsprechenden Werkzeuges hilft bzw. neue Werkzeuge definiert und in die Datenbank speichert (s. Abb. 4).

Bei der Planung des interaktiven Programmoduls verwendeten wir besondere Aufmerksamkeit auf die rationelle Arbeitsteilung zwischen Mensch und Maschine.

Zur Erleichterung der menschlichen Arbeit diente die Möglichkeit, daß typische Drehoperationen wie Abrundung, Eindrehen, Nutdrehen einfach beschreibbar wurden. Der Maschine wird damit geholfen, daß der Nutzer das Werkzeug auf der Werkzeugbahn verschieben kann und mit einem Blick die Kollisionssituation erkennt. Diese Prozedur würde bei maschineller Bearbeitung langwierige und komplizierte Berechnungen benötigen. Der interaktive Modul ist in traditioneller Baumstruktur aufgebaut. Diese Struktur weist u. a. auch auf das Menüsystem des Moduls hin (s. Abb. 5). Der Programmierer wählt die Menüelemente mit dem Lichtstift aus und übergibt danach die nötigen Daten. Geometrische Daten können inkremental oder absolut angegeben werden. Man kann sie auch mittels Konstruktion oder durch Auswahl typischer geometrischer technologischer Konfigurationen aus dem Menü angeben.

Nach der Veranschaulichung des Elementes kann der Nutzer entscheiden, ob er die veranschaulichten Elemente annimmt. Die Daten der Elemente gehen erst dann vom Arbeitsfile in die Datei des Systems über. Diese Datei speichert die Daten und ihre Typen in der Reihenfolge ihrer Angabe. Aus diesem File baut sich auch das veranschaulichende Bild auf. Zur Veranschaulichung und Behandlung der Bilder benutzen wir dasselbe GTU graphische Programmpaket, das auch vom gedruckten-schaltungsplatte-korrigierenden Programm verwendet wird.

Die Angabe der Technologie geschieht durch die Auswahl von Menüelementen, und wo es notwendig ist, gibt man auch die Daten ein. Die Bewegung des Werkzeuges kann mit Relativkoordinaten schrittweise beschrieben

werden. Die Automatisierung des Beschreibens wird in Zukunft stufenweise geschehen. Die technologischen Daten werden von der Datei des Systems in der Reihenfolge der Angabe gespeichert. Die Werkzeugstrecken werden auch in der Reihenfolge der Angabe gespeichert, aber die Daten kommen umgerechnet auf die theoretische Werkzeugspitze in die Datei.

4.3. Digitalgraphische Planung

Ein anderes digitalgraphisches Programm des integrierten Produktionssystems ist das mit Rechner gestützte Planungsprogramm für Preßwerkzeuge. Mit diesem Programm kann man die zu stanzende Form beschreiben und visualisieren. Danach kann die Bandplanung beginnen, die auf maximale Materialausnutzung bestrebt ist.

Bei der Planung des Stanzers nimmt das System die Routinen der mechanischen Rechnung in Anspruch. Zur endgültigen Ausführung des Werkzeuges werden von den standardisierten Stanzelementen diejenigen ausgewählt, die dem angegebenen Sortiment zur Verfügung stehen. Zur Bearbeitung der fehlenden Werkzeugelemente wird der Lochstreifen hergestellt.

Die Struktur und Benutzung des Programms für Abschreiben und Visualisieren der Stanzform gleicht in vielen Gesichtspunkten den vorher beschriebenen Programmen. Die Veranschaulichung und Behandlung der Bilder vollzieht auch hier das Programmpaket GTU. Die Art der Abschreibung der Geometrie ist hier breiter, aber die Ordnung der Angabe ist gebundener.

Nur nach der Abschreibung der Stanzform kann mit der Kenntnis der Materialbeiwerte die Auswahl der Stanzoperationen und Bandplanung erfolgen.

Während die auf dem Band plazierte Konturen verschoben und verdreht werden, wird von der Routine der Bandplanung die Bandausnutzung realtime errechnet und angezeigt. Danach fängt die Errechnung der mechanischen Eigenschaften an. Im ersten Schritt bezeichnet der Nutzer mit dem Lichtstift die kritischen Segmente des Bandes. Im zweiten Schritt können die Stufen des Stanzens geplant werden. Danach wird das Stanzwerkzeug mit selbständigem Programmteil geplant.

5. Zusammenfassung

Die bisher beschriebenen in der Industrie verwendeten Programme wie

- das Programm für die Korrektur von Druckschaltungsplatten,
- das Programm für die Kontrolle des CLFILE,
- das Programm für Drehteile und
- das Programm für Preßwerkzeugplanung

wurden in verhältnismäßig kurzer Zeit durch ein kleines Forscherkollektiv realisiert. Das war nur dadurch möglich, daß das Bildschirmgerät GD 71 und sein Softwaresystem auf Grund sorgfältiger Planung allen diesen Aufgaben gerecht werden konnte.

Die Lösung dieser Aufgaben wurde auch durch die Peripherie des Displays wie der Funktionstastatur, der alphanumerischen Tastatur, dem Lichtstift und dem Positionierball unterstützt. Das Display ist an den in dem Zentralen Forschungsinstitut für Physik entwickelten TPA-70 Kleinrechner gekoppelt.

Die Speicherkapazität des Rechners beträgt maximal 32 KByte; externes Speichermedium ist eine Magnetplatte mit 5 MByte. Die Peripherie besteht aus der Konsole, dem Lochstreifenstanzer und -leser, dem Zeilendrucker und dem Plotter.

Aus dem Softwaresystem des GD 71 möchten wir das GTU-graphische Programmpaket hervorheben, dessen Eigenschaften dem Planer erheblich helfen. Dieses Paket ermöglicht das Definieren von Bildausschnitten, auch Verkleinerungen, Vergrößerungen und Transformationen in der Ebene können vorgeschrieben werden. Das System speichert die Bildelemente in dem ihm angewiesenen Bildpuffer. In diesem sind für die freie Behandlung und Identifikation mit Lichtstift Bildsegmente definierbar. Die im Segmentkopf gespeicherte Information sorgt für die Kopplung der Segmente, wodurch die Speicherung und Veranschaulichung neuerer Elemente verschnellert wird.

In diesem Referat wollten wir darüber berichten, wie die im Institut für Rechentechnik und Automatisierung der Ungarischen Akademie der Wissenschaften betriebene Forschungsarbeit (umfassend Rechentechnik,

wissenschaftliche Aspekte der Stromkreisplanung, einige Aspekte der Automatisierung der Maschinenbauindustrie) immer in den Dienst der Produktion gestellt wird. Ferner wollten wir auch veranschaulichen, daß Teilaufgaben mit entsprechender Tempierung als Subsysteme gut geplanter Systeme schnell erstellt und eingesetzt werden können, einerseits als unmittelbare Produktionshilfe und andererseits dafür, daß die Fachleute der Industrie die neuen Methoden schneller in Besitz nehmen.

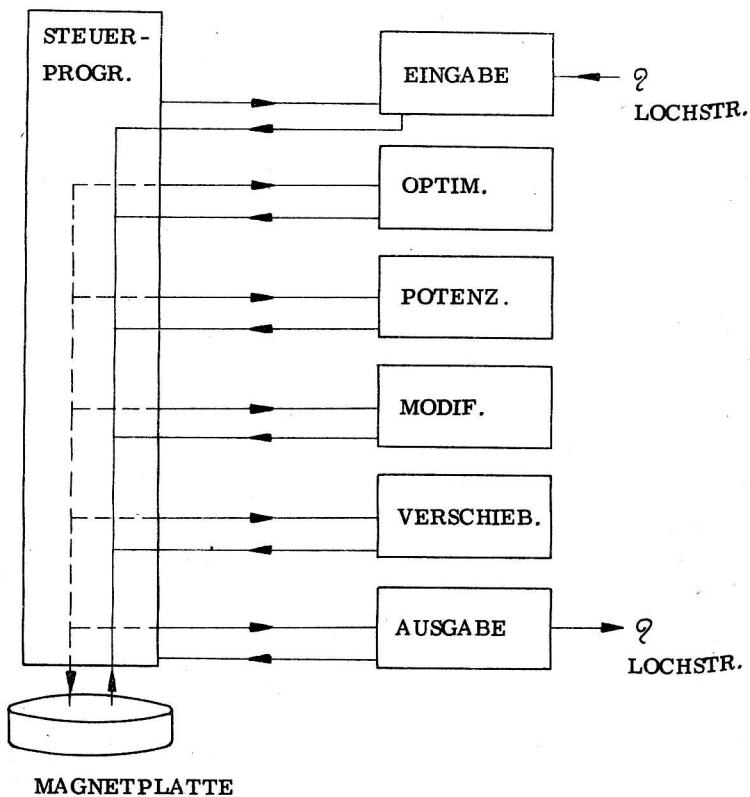
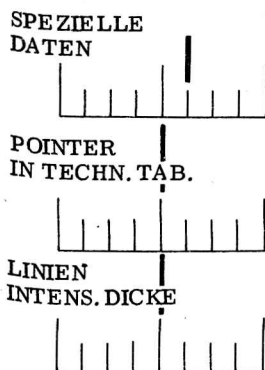


Abb. 1

BILD- SPEICHER	x	8,6 Kbyte
WIRKEND. SEGMENT		2,1 Kbyte
STEUERPROGRAMM		
MODELL SPEICHER	x	7,2 Kbyte
GTU - ROUTINE	x	2,4 Kbyte
POPERAT. SYST.		4 Kbyte

Abb. 2



POTENZIALIDENTIFIKATION



X KOORDINATE IN 0,1 RASTER



Y KOORDINATE IN 0.1 RASTER

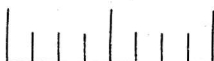


Abb. 3

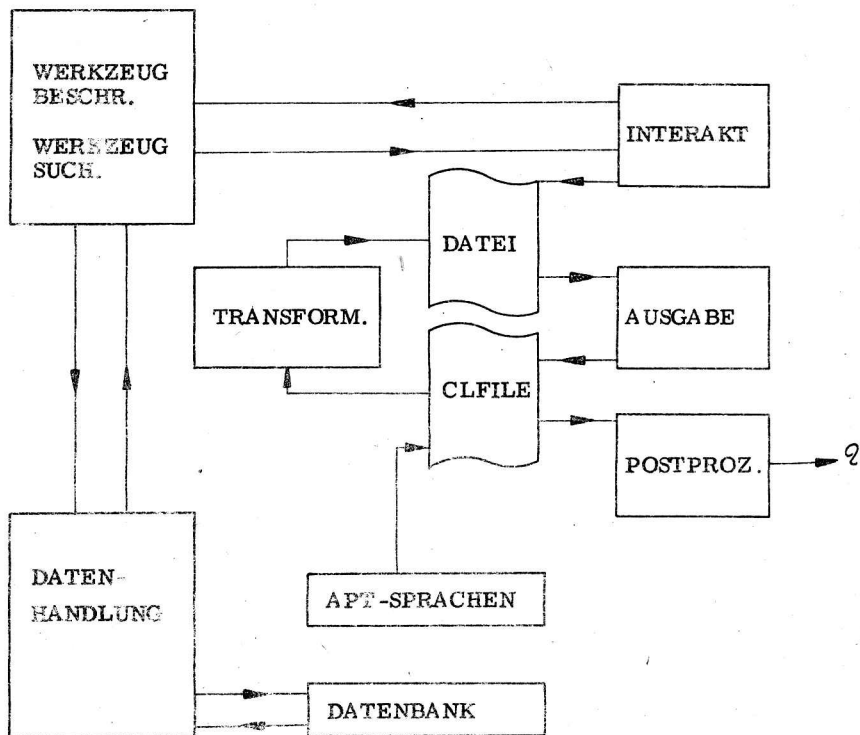


Abb. 4

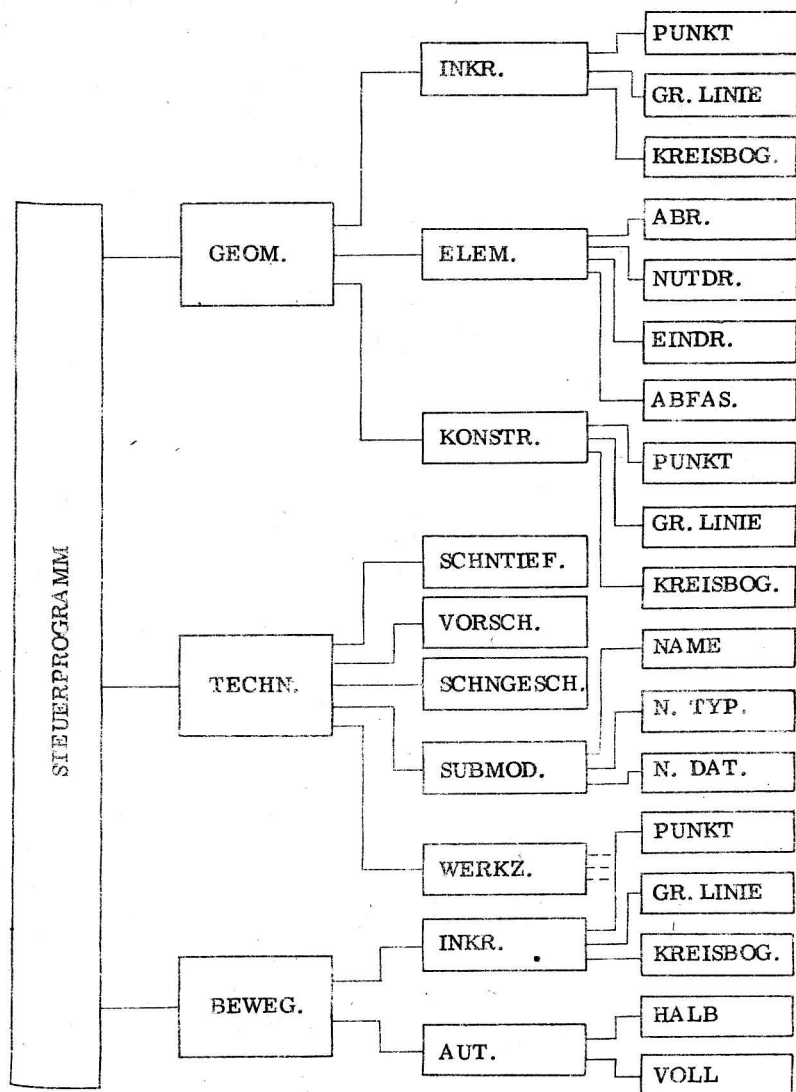


Abb. 5

Literatur

- /1/ Kovacs, M.; Vincze, A.: Computer Aided Design of Printed Circuit Boards. Papers of a Joint Colloquium on Computer Aided Design, Budapest, 1973, 91 - 103
- /2/ Kerestely, D.; Tolnay-Knefely, T.: Interaktiv grafikus program nyomatott áramköri kártyák rajzolatának módosítására (Inter-aktives graphisches Programm zum Korrigieren und Modifizieren des Druckbildes gedruckter Schaltungsplatten). Javításra Meres és Automatika, 1976
- /3/ Simon, V.: Anwendung von Bildschirmgeräten in der automatisierten Fertigungsvorbereitung im Maschinenbau. Tagung INFERT 74, Dresden, 1974
- /4/ Nemes, L.: Progress in the Development of an Integrated Data and Material Processing System in Hungary. Seminar on Automated Integrated Production Systems in Mechanical Engineering, Prague, 1976
- /5/ Pikler, G.: Interactive Design System for Blanking Die. Papers of a Joint Colloquium on Computer Aided Design, Budapest, 1973, 123 - 137
- /6/ Labadi, A.: The GDIO Graphical Input-Output System. Programming Manual. Budapest, 1974
- /7/ Krammer, G.: The Assembler Language TAL for the TPA -7025 Small Computer. Programming Manual, Budapest, 1975
- /8/ Gallai, I.: DOST - Disk Operating System for the TPA -7025 Computer. Budapest, 1975
- /9/ Darvas, P.; Hosszu, P.: GTU grafikus run-time rutinok. (GTU-graphische run-time Routinen). Budapest, 1975

Eingegangen: 5.11.1976

Anschrift des Verfassers:

Dipl.-Ing. Vera Simon
Ungarische Akademie der Wissenschaften
Institut für Rechentchnik und Automatisierung

H-1502 Budapest XI

Kende u. 13 - 17

Zum Anwendungsbereich des algorithmischen Systems Entscheidungstabellentechnik

Übersicht

In zahlreichen Anwendungsfällen und -bereichen wurde die Entscheidungstabellentechnik zu einem leistungsfähigen algorithmischen System entwickelt.

Ausgehend von der algorithmischen Problemstruktur im Anwendungsbereich wird die Struktur des Entscheidungstabellensystems entwickelt und werden die wesentlichen Merkmale der Entscheidungstabellentechnik charakterisiert.

Nach einer Übersicht über Hauptanwendungsbereiche wird ein Konzept zur Beschreibung von musterverarbeitenden, entscheidungstabellengesteuerten TURING-Automaten dargelegt.

1. Zur problemorientierten Interpretation von Abbildungen

Es sei

$$G : B \dashrightarrow A$$

eine Abbildung (Zuordnungsvorschrift) mit mehrdimensionalem Definitionsbereich und mehrdimensionalem Wertebereich. In dieser Notation ist G der Repräsentant für die konkrete Angabe dieser Zuordnungsvorschrift u. a. auch als System von Entscheidungstabellen (ETS).

Wir können als hinreichend allgemeine Notation annehmen

$$\text{Definitionsbereich von } G = D(G) = B \subseteq \prod_{i=1}^{n1} P(M_i)$$

$$\text{Wertebereich von } G = W(G) = A \subseteq \prod_{j=1}^{n2} P(M_j)$$

mit $P(M_i)$ - Potenzmenge der Menge M_i .

Folgende spezielle Fälle sind zu diskutieren:

a) Zuordnungsvorschriften im Erweiterten Formalen System

Wenn vom Standpunkt der Anwendung die Möglichkeit der Verwendung von Teilmengen eines i -ten Merkmales M_i bezüglich eines Objektes betont werden soll, wird das Formale System (s. /9/) auch als Erweitertes Formales System (EFS) bezeichnet (s. /15/). Es beschreibt Substitutionsalgorithmen, z. B. Markovsche Normalalgorithmen.

Es gilt

$$ETS = EFS : \bigcap_{i=1}^{n1} P(M_i) \dashrightarrow \bigcap_{i=n1+1}^{n1+n2} P(M_i)$$

bzw.

$$ETS = EFS \subseteq \bigcap_{i=1}^{n1+n2} P(M_i)$$

Anwendungsbereiche sind Substitutionsalgorithmen, Bedienalgorithmen für Geräte und Anlagen (s. /11/).

b) INPUT-OUTPUT-Verhaltensweise

Einer speziellen Zuordnungsvorschrift kann die inhaltliche Bedeutung einer Eingang-Ausgang-Verhaltensweise eines Operators (abstrakter Automat, Informationskomplex) zugeordnet sein. Es gilt dann für die Ergebnisrelation R

$$ETS = R : X \dashrightarrow Y \quad - \text{Ergebnisrelation}$$

Die Input-Output-Verhaltensweise ist ein Spezialfall der nachfolgenden.

c) INPUT-OUTPUT-STATE-Verhaltensweise

Es gilt entsprechend

$$ETS : X \times Z \dashrightarrow Z \times Y$$

bzw. $R : X \times Z \dashrightarrow Y$ - Ergebnisrelation

$$S : X \times Z \dashrightarrow Z \quad - \text{Folgerelation}$$

Anwendungsbereiche sind: determinierter abstrakter Automat, nichtdeterminierter abstrakter Automat, linearer abstrakter Automat, Verhaltensweise von Informationskomplexen (s. /16/) und Bedienalgorithmen.

d) Programmierbare INPUT-OUTPUT-STATE-Verhaltensweise

Es gilt

$$\text{ETS} : X \times Z \dashrightarrow Z \times Y \times W$$

bzw.

$$R : X \times Z \longrightarrow Y \quad - \text{Ergebnisrelation}$$

$$S : X \times Z \longrightarrow Z \quad - \text{Folgerelation}$$

$$Q : X \times Z \longrightarrow W \quad - \text{Verschieberelation}$$

Anwendungsbereiche sind: Einband-TURING-Automat, musterverarbeitender TURING-Automat.

Wir fassen die Aussagen wie folgt zusammen:

Mindestens eindeutige Zuordnungsvorschriften (und prinzipiell auch nicht-eindeutige) lassen sich je nach Beschreibungsinhalt als Substitutionsvorschriften, abstrakte Automaten, TURING-Automaten o. ä. interpretieren. Es soll festgestellt werden, daß als effektives Darstellungsmittel solcher abstrakter Objekte die Entscheidungstabellentechnik verwendet werden kann.

2. Entscheidungstabellentechnik als algorithmisches System

2.1. Problemorientierte Prozeßstruktur

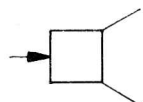
Ein Entscheidungstabellensystem ETS ist im Sinne der Algorithmentheorie charakterisierbar durch die Endlichkeit der Darstellung, die Determiniertheit, die Anwendbarkeit und die Allgemeinheit.

Ein ETS sei anwendbar auf eine Folge von n-Tupeln von Werten oder von Teilmengen von Werten (Unterschied siehe Formales System - Erweitertes Formales System). Dann sind die n-Tupel die Operanden, auf die ETS anwendbar ist. Im Rahmen der Entscheidungstabellentechnik wird ein n-Tupel als "Situation" bzw. "Situationstupel" σ bezeichnet. Prinzipiell umfaßt das



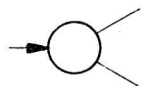
bedingte/unbedingte Operation

(die bedingte Operation wird dann und nur dann ausgeführt, wenn die Bedingung erfüllt ist. Der Prozeß "wartet" an dieser Stelle auf die "externe" Erfüllung der Bedingung)



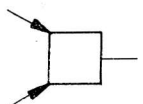
bedingte Verzweigungsoperation

(der Prozeß verläuft alternativ auf einem der Ausgangszweige)



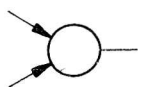
bedingte/unbedingte Parallelzweigoperation

(der Prozeß verläuft parallel auf allen Ausgangszweigen)



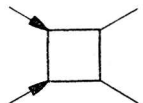
unbedingte Abschlußoperation (ODER)

(der Prozeß wird auf dem Ausgangszweig fortgesetzt, wenn einer der Eingangsteilprozesse abgeschlossen ist)

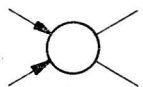


bedingte Abschlußoperation (UND)

(der Prozeß wird nur dann fortgesetzt, wenn die Teilprozesse auf allen Eingangszweigen abgeschlossen sind)



bedingte Abschluß-Verzweigungsoperation



bedingte Abschluß-Parallelzweigoperation



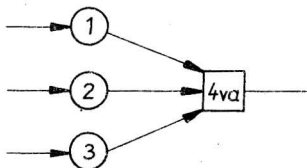
Übergang Operation - Operation

Abb. 1.: Zur Definition des Graphen eines Teilprozesses

Situationstupel eine Menge von Werten oder Teilmengen von Werten. Ohne Beschränkung der Allgemeinheit können wir im Weiteren das Situations-tupel als Formatiertes Tupel von Werten voraussetzen. Dieses Situations-tupel ändert sich zyklisch, bedienungs- oder bedingungsabhängig.

ETS wird auf das Situationstupel \mathcal{G} angewendet:

- nach dem Taktprinzip (beat principle) oder



Operation	Erläuterung	
Aktion 1		a1
Aktion 2		a2
Aktion 3		a3
Aktion 4va	wenn (a1 vollzogen ^ a2 vollzogen ^ a3 vollzogen)	so goto nächste Operation

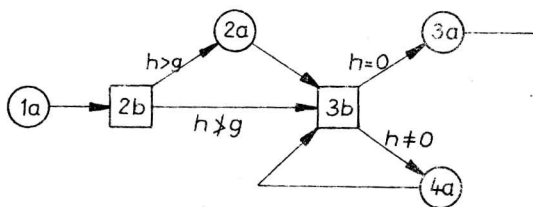
Die Aktionen a1, a2, a3 sind bezüglich ihrer Abarbeitungsreihenfolge äquivalent.

Abb. 2: Halbordnung von drei Aktionen

- nach dem Ereignisprinzip (principle of next event) oder
- nach dem gemischten Funktionsprinzip (s. /13/).

Ein Entscheidungstabellensystem ETS besteht u. a. aus einer Menge $ET = \{ et \}$ von Entscheidungstabellen. Wir betrachten zuerst eine Entscheidungstabelle.

Eine Entscheidungstabelle beschreibt die Verhaltensweise eines Teilprozesses (sowohl eines Basisprozesses oder auch eines Steuerungs-, Lenkungs- oder Leitungsprozesses). Elemente eines Teilprozesses sind die Operationen, Operationsvorschriften und Operationsziele (s. /15/). Die Beschreibung kann (wie schon ausgeführt) nach dem Taktprinzip oder/und dem Ereignisprinzip dargestellt werden. Wir betrachten vier Beispiele und wollen dazu



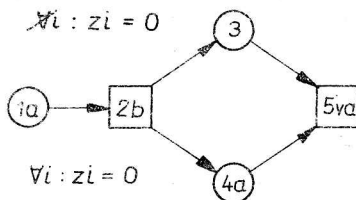
gesucht $\text{ggT}(g, h)$ mit $g, h \in \Gamma^+$

Operation	Erläuterung	
Aktion 1a		Belege g und h mit den gegebenen zwei Zahlen
Bedingung 2b prüfen	$h > g$	
Aktion 2a		Vertausche die Belegungen von g und h
Bedingung 3b prüfen	$h = 0$	
Aktion 3a		Beende das Verfahren: der Wert von g ist der gesuchte $\text{ggT}(g, h)$
Aktion 4a		Bestimme den Rest, der sich bei Teilung von g durch h ergibt. Belege g mit dem Wert von h und h mit dem erhaltenen Rest

Abb. 3: Euklidischer Algorithmus

eine spezielle Notation vereinbaren, mit der alternative und kollaterale Operationen gemeinsam beschrieben werden können (s. Abb. 1).

Abb. 2 - 5 stellen die Beispiele dar.



Operation		Erläuterung
Aktion 1a		$i=1(1)n$: Bestimme z_i' aus gegebener Zuordnungsvorschr. $s1: (x_i, z_i) \mapsto z_i'$
Bedingung 2b prüfen	$\forall i: z_i = 0$	
Aktion 3	$i=1(1)n$: $z_i \neq 0$	$((\text{setze } z_i=0 \wedge y=i) \wedge \text{goto } 5va)$
Aktion 4a		setze $y=0$
Aktion 5va		Verbundaktion: goto nächster Takt1)

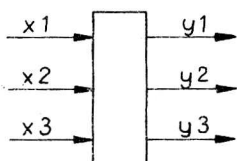
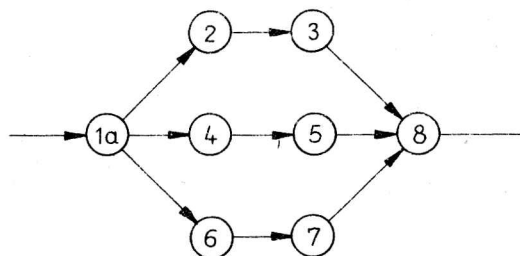
1) Vereinfachend wurde das Taktprinzip vorausgesetzt.

Bei Taktprinzip: goto nächster Takt (nt)

Bei Ereignisprinzip: goto Monitorprogramm (mp)

Quelle: STAHN (1976), s. /17/.

Abb. 4: Präferenzautomat



Operation	Erläuterung	
Aktion 1a		op1, op2, op3 auslösen
Aktion 2	Hat op1 Endstellung erreicht	op1 umschalten
Aktion 3	Hat op1 Ausgangsstellung erreicht	op1 ausschalten
Aktion 4	Hat op2 Endstellung erreicht	...
...		
Aktion 7	...	op3 ausschalten
Aktion 8	op1 ausgeschaltet \wedge op2 ausgeschaltet \wedge op3 ausgeschaltet	goto nächste Operation

Quelle: STAHN (1974), s. /14/.

Abb. 5: Ausschnitt Steuerungssystem Taktstraße

Werden diese Teilprozesse in Entscheidungstabellen umgesetzt, so sind dabei die Festlegungen des algorithmischen Systems "Entscheidungstabellentechnik" zu berücksichtigen.

2.2. Zur Entscheidungstabellentechnik

1. Eine Entscheidungstabelle (et) besteht aus einer Menge von Regeln. Die Menge der Regeln ist geordnet. Zwei Klassen von Entscheidungstabelle-konzepten unterscheiden sich hinsichtlich des Merkmals "Relevanz der Ordnung der Regeln (einer Tabelle)": Wird Beispiel 1 (Abb. 2) in eine et umgesetzt, so ist die Ordnung der Regeln ohne algorithmische Bedeutung (nichtrelevant). In den anderen Beispielen (Abb. 3 - 5) ist die algorithmische Relevanz der Ordnung der Regeln im Interesse der Effektivität zweckmäßig. Daraus ziehen wir die nachstehende Schlussfolgerung.
2. Die Ordnung der Menge der Regeln einer Entscheidungstabelle ist algorithmisch relevant. Jede Regel kann markiert sein.
3. Der Kern einer Regel rr ist eine Implikation

$$b \dashrightarrow a.$$

Die linke Seite (Bedingungen, Prämisse) ist ein mehrstelliges Attribut (Aussagenformverbindung), das durch Anwendung auf das Situationstupel den Wahrheitswert wahr oder falsch annimmt. Mehrwertige Attribute sind prinzipiell nicht ausgeschlossen, werden jedoch in der Entscheidungstabellentechnik noch nicht verwendet. Die rechte Seite (Aktionen, Conclusio, Konsequenzen, Aussagen, Bewertungen) sind partielle Funktionen. Sie können insbesondere auch Verbund- bzw. Sprungaktionen einschließen. Diese Sprungaktionen realisieren die Übergangsrelation zwischen den Regeln einer Tabelle (bzw. zwischen den Tabellen). Jede Regel kann ferner durch einen Kommentar ergänzt werden. In Übereinstimmung mit der Algorithmentheorie wird nun vereinbart: Eine Regel rr ist auf das Situationstupel \mathcal{G} anwendbar, wenn b aus rr angewendet auf \mathcal{G} eine wahre Aussage gibt. Ist eine Regel anwendbar, so werden die Aktionen a kollateral ausgeführt. Ist eine Regel nicht anwendbar, so wird zur nächsten Regel

übergangen. Ist keine Regel mehr anwendbar oder ist die anwendbare Regel eine Schlußregel, so bricht der Algorithmus ab.

4. Es wird vereinbart, daß

- Aussageformen als Bestandteil der Bedingung einer Regel nur dann angegeben werden, wenn sie für die betreffende Regel relevant sind,
- Werte in Aktionen nur dann angegeben werden, wenn die Werte des betreffenden Ausdrucks geändert werden sollen.

5. In einem Entscheidungstabellensystem ETS wird vereinbart, daß nicht in jeder Tabelle $et \in ET$ der volle Definitionsbereich des ETS angegeben wird, sondern nur der für den betreffenden Teil des Algorithmus relevante:

$$et : D(et) \dashrightarrow W(et)$$

mit

$$D(et) \subseteq D(ETS)$$

$$W(et) \subseteq W(ETS)$$

2.3. Zuordnung problemorientierte Prozeßstruktur - Entscheidungstabellenstruktur

Aus dem Dargelegten wollen wir Grundaussagen der Entscheidungstabellentechnik ableiten:

1. Die inhaltliche Prozeßstruktur muß in die Struktur des Entscheidungstabellensystems und damit in eine Ordnung von Regeln innerhalb einer Entscheidungstabelle überführt werden.
2. Eine betrachtete Tabelle werde im Rahmen des Taktprinzips genutzt. Dann kann in Übereinstimmung mit der Automatentheorie vorausgesetzt werden, daß eine Tabelle zu einem Taktzeitpunkt nur einmal abgearbeitet wird.

Wird eine betrachtete Tabelle im Ereignisprinzip genutzt, so ist es möglich, daß auf das Eintreten tabellenexterner Ereignisse (Situationen) gewartet werden muß. Am zweckmäßigsten wird dieses "Warten" durch einen unbedingten Rücksprung (im Sinne einer Warteschleife) realisiert.

3. Im Interesse der rechentechnischen Implementation von Entscheidungstabellen wird vorausgesetzt, daß die Abarbeitung einer Tabelle immer nur über einen Eingang, d. h. in der 1. Regel beginnt. Angestrebt wird, die Tabelle auch nur über einen Ausgang zu verlassen.

Für die angegebenen Beispiele ergibt sich dann nachstehende Zuordnung Prozeßstruktur - Tabellenstruktur (Abb. 6).

2.4. Merkmale von Entscheidungstabellenkonzepten

Merkmale mit dem zugehörigen Wertevorrat zur Beschreibung und komparativen Bewertung von Entscheidungstabellenkonzepten sind folgende (s. /6/):

- Algorithmische Relevanz der Ordnung der Regeln einer Entscheidungstabelle

Dieses Merkmal war schon vorstehend erläutert worden.

Wir bezeichnen die beiden Varianten mit

- Ordnung algorithmisch relevant,
- Ordnung algorithmisch nicht relevant.

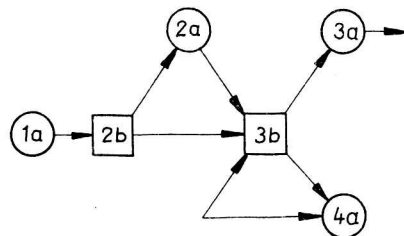
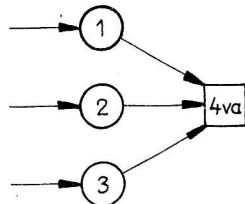
- Form einer Entscheidungstabelle

- normal: jede Regel bildet eine Zeile in einer et
- invers: jede Regel bildet eine Spalte in einer et

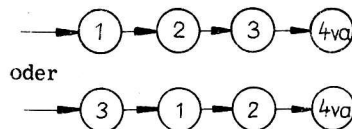
Die Vorteile der normalen Form sind durch die Anlehnung an andere algorithmische Systeme (Markowscher Normalalgorithmus, TURING-Programm, problemorientierte Sprachen, Allgemeiner Algorithmus (s. /17/) gegeben. Sie entspricht der modernen internationalen Entwicklung.

- Sprünge (Verbundaktionen) bei der Ausführung einer Regel
 - a: Sprung zu einer Regel derselben Tabelle oder zur ersten Regel einer anderen Tabelle ist erlaubt.
 - b: Sprung zur ersten Regel derselben oder einer anderen Tabelle ist erlaubt.
 - c: Im Rahmen eines speziellen Konzeptes (s. /8/) werden auch die Elemente der Bedingungen geordnet und sind markierbar. Dadurch

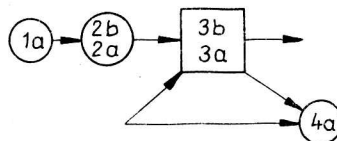
Prozeßstruktur



et-Struktur

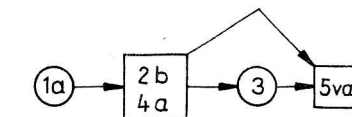
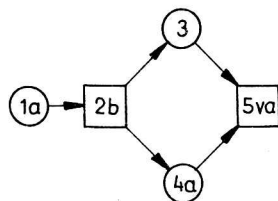


...

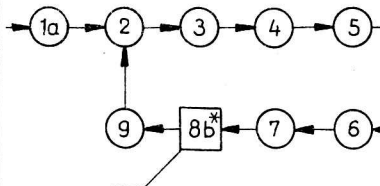
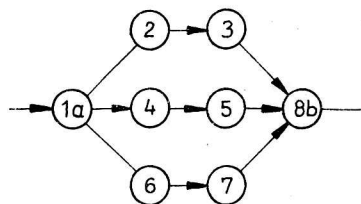
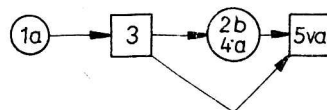


Die Zuordnung einer et-Struktur zur Prozeßstruktur ist im allgemeinen nicht eindeutig

In der et-Struktur wird eine Regel übergangen, wenn die angegebene Bedingung durch Einsetzen des Situationstupels zu einer falschen Aussage wird



oder



Der Übergang $8b^* - 9 - 2$ muß als Zykelselement eingefügt werden, weil die et nach dem Ereignisprinzip auf die Erfüllung aller et-externen Bedingungen "warten" muß.

Abb. 6: Zuordnung Prozeßstruktur \longrightarrow et-Struktur (Beispiele)

wird der Sprung zu einer Bedingung einer Regel möglich.

d: Kein Sprung

Das Konzept a ist das derzeitig günstigste hinsichtlich Optimum an Nutzen zu Aufwand.

- Matrixbewertungsform

- et mit erweiterten Eingängen
- et mit beschränkten Eingängen

Bei et mit beschränkten Eingängen ergibt sich durch Einsetzen der Werte des Situationstupel \mathcal{G} in die Aussageformverbindung im Regelkopf eine wahre oder falsche Aussage. Dieser Wahrheitswert muß mit dem geforderten Wert (wahr oder falsch) im Regelrumpf übereinstimmen.

Bei den et mit erweiterten Eingängen wird die Aussagenformverbindung aus dem Regelkopf (< Ausdruck >) zusammen mit dem Regelrumpf (< Vergleichsoperator > < Ausdruck >) gebildet. Die Regel ist dann anwendbar, wenn das Einsetzen der Werte des Situationstupels in die Aussagenformverbindung einen wahren Wert ergibt. In diesem Zusammenhang muß betont werden, daß alle bekannten Vergleichsoperatoren und Ausdrücke (z. B. auch mengentheoretische: $x \in X$; $M1 \subseteq M$) zulässig sind, wenn auch bestimmte Implementationen von Vorübersetzern nicht alle zulassen.

Bei et mit erweiterten Eingängen ist eine größere Allgemeinheit und damit größere Anwendungsbreite bei Gewährleistung der Übersichtlichkeit realisierbar.


2.5. Niveaus von Entscheidungstabellen

Es werden verschiedene Niveaus der algorithmischen Darstellung einer Entscheidungstabelle angegeben (s. /6/ und Abb. 7).

et-Niveau	Merkmale
1	vollständige Aufzählung (Enumeration) aller Werte des Definitionsbereiches und der Verhaltensweise
2	Zerlegung des Wertebereiches in Äquivalenzklassen der Verhaltensweise und Beschreibung der Zuordnungsvorschrift für jede Klasse der Zerlegung
3	Trennung des Wertebereiches W von ETS mit $\dim(W) \geq 2$ in Faktoren und Beschreibung der Teilrelationen. Dabei impliziert diese Auftrennung das Verhaltensmodell eines abstrakten Automaten (siehe Abschnitt 1).

Abb. 7: Niveaus von Entscheidungstabellen

Vereinbarung der Notation

Automatenband eines Mealy-Automaten (X, Y, Z, r, s)	Schematische Notation																
<table><tr><td>x</td><td></td><td>ξ</td><td></td></tr><tr><td>z</td><td></td><td>ξ</td><td>ξ'</td></tr><tr><td>y</td><td></td><td>η</td><td></td></tr><tr><td>t</td><td>$\tau-1$</td><td>τ</td><td>$\tau+1$</td></tr></table>	x		ξ		z		ξ	ξ'	y		η		t	$\tau-1$	τ	$\tau+1$	 · τ ·
x		ξ															
z		ξ	ξ'														
y		η															
t	$\tau-1$	τ	$\tau+1$														

Verhaltensmodelle

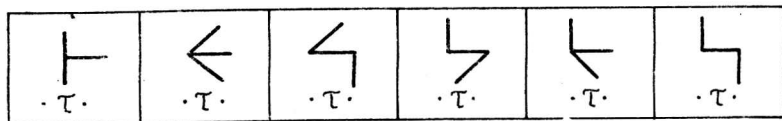
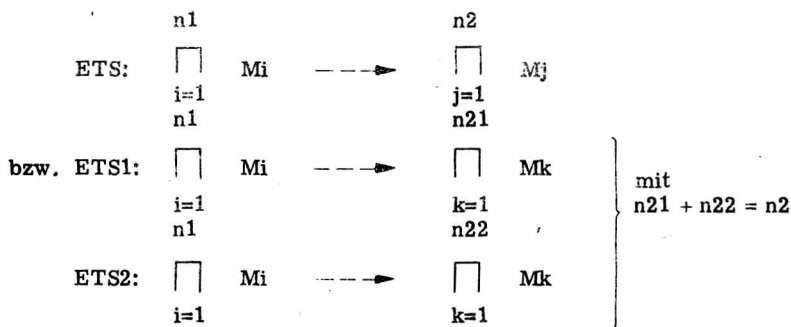


Abb. 8: Verhaltensmodelle abstrakter Automaten

Das am meisten angewandte Niveau ist das Niveau 2.

Entscheidungstabellen normaler Form des Niveaus 1 zur Beschreibung der Verhaltensweise von TURING-Automaten sind unter der Bezeichnung "TURING-Tafel" bekannt.

Die verschiedenen Verhaltensmodelle abstrakter Automaten (Abb. 8) können für die zweckmäßige algorithmische Beschreibung von Prozessen von wesentlicher Bedeutung sein. Dabei wird die Relation ETS in folgender Weise innerhalb des Niveaus 3 aufgetrennt:



2.6. Struktur von Entscheidungstabellensystemen

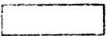


Ein Entscheidungstabellensystem ETS besteht aus einer Menge $ET = \{ et \}$ von Entscheidungstabellen, aus einer nichtleeren Menge $K1$ von Verbunden und einer nichtleeren Menge $K2$ von Wertetransfers sowie zweier mindestens eindeutiger Inzidenzabbildungen $\varphi 1$ und $\varphi 2$:

$$\varphi 1: K1 \longrightarrow ET \times ET$$

$$\varphi 2: K2 \longrightarrow ET \times ET$$

Einem ETS kann ein Graph zugeordnet werden (Abb. 9). Dann ist

$$ETS = (ET, K1, \varphi 1, K2, \varphi 2)$$

ETS	Graph
$et \in ET$	Knoten 
$k1 \in K1$	Kante 1 
$k2 \in K2$	Kante 2 
Bezeichnung der et (Quelladresse, Zielaadresse)	Knotenbewertung
Bezeichnung der Werte	Kantenbewertung zu k1
	Kantenbewertung zu k2

Beispiele

mit $ETS = (ET, K1, \varphi_1, K2, \varphi_2)$

ET - Menge von Entscheidungstabellen

K1 - Menge von Verbunden

K2 - Menge von Wertetransfers

φ_1 : - $K1 \longrightarrow ET \times ET$

φ_2 : - $K2 \longrightarrow ET \times ET$

Abb. 9: Zur Definition des Graphen eines Entscheidungstabellensystems ETS

Unter der Voraussetzung, daß φ_1 und φ_2 existieren, kann man dafür auch schreiben

$$ETS = (ET, K1 \subseteq ET \times ET, K2 \subseteq ET \times ET)$$

In zahlreichen Fällen wird ETS in Anlehnung an die Programmierungstechnik vereinfachend durch

$$ETS = (ET, K1 \subseteq ET \times ET)$$

dargestellt (z. B. /5/).

3. Anwendungsbereiche der Entscheidungstabellentechnik

Anwendungsfälle von Entscheidungstabellen sind aus vielen Bereichen bekannt. Eine Bibliographie des VEB Carl Zeiss Jena (s. /2/) gibt eine Übersicht über bekannte Anwendungsfälle.

Evident ist, daß Entscheidungstabellen nur in determinierten oder stochastischen Entscheidungsprozessen zweckmäßig verwendbar sind, d. h. Ablaufprozesse, deren Graph eine Ordnung ist, werden durch Entscheidungstabellen nicht effektiver beschrieben. Entscheidungstabellen sind ein effektives Arbeitsmittel zur Beschreibung von determinierten Prozessen, die Alternativentscheidungen bzw. parallele Fortschrittentscheidungen enthalten oder die halbgeordnet sind (siehe Abschnitt 2). Sie sind sowohl als ein aufzählendes, als auch als ein beschreibendes Mittel zu verwenden.

Markovsche Normalalgorithmen, TURING-Programme und Ljapunovsche Operatorenschreibweise und natürlich auch die graphische Methode der Algorithmenablaufplanung stehen in direkter Relation zur Entscheidungstabellentechnik.

Hauptanwendungsbereiche sind:

- Konzipierung, Projektierung und Dokumentation von Programmen und Programmsystemen einschließlich Übersetzerprogramme und Betriebssysteme (s. /4/, /10/, /8/).
- Analyse und Synthese von rechnerunterstützten Informationsprozessen in der Leitung und Planung von Betrieben und Kombinat (ASUTP), in der Lenkung und Kontrolle von technologischen Prozessen (ASUTP) einschließlich der Modellierung stofflich-energetischer Basisprozesse (s. /14/, /5/, /7/, /1/).
- Modellierung logisch-funktioneller Strukturen in Einheit von Geräte- und Algorithmentechnik (s. /4/, /3/, /12/, /18/).
- Darstellung von komplizierten Bedienalgorithmen (s. /17/, /11/).

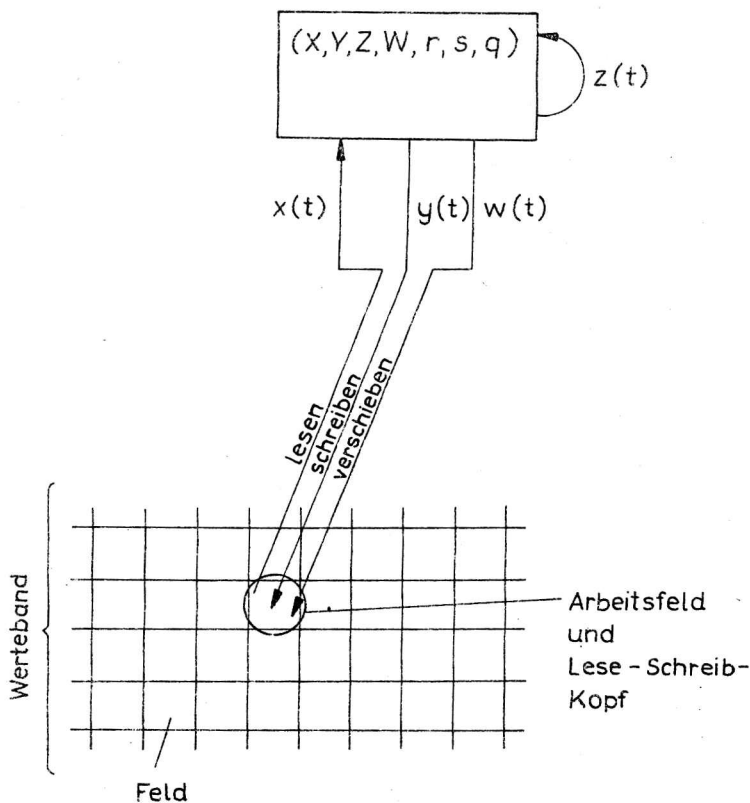


Abb. 10: Schema des TURING-Musterautomat mit
zweidimensionalem Werteband

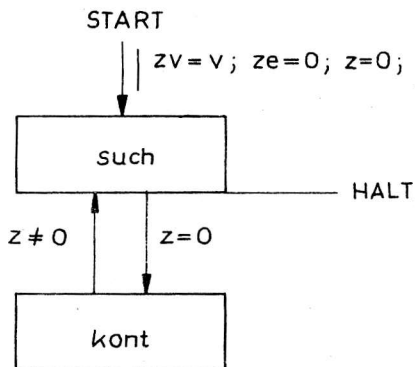


Abb. 11: Entscheidungstabellensystem eines TURING-Musterautomat

such	x	zv	ze	z	zv	ze	z	y	w	goto
1		r		≠0			0		1,0	
2		l		≠0			0		-1,0	
3	1					0				kont
4	0	r				0			1,0	
5	0	l				0			-1,0	
6	E	r	0		l	E			-1,-1	
7	E	l	0		r	E			1,-1	
8	E		E							
										HALT

kont	x	z	z	y	w	goto
1		0	1		1,0	
2	≠1	1			-1,0	10
3	1	1	2		-1,-1	
4	≠1	2			0,1	10
5	1	2	3		-1,1	
6	≠1	3			1,0	10
7	1	3	4		1,1	
8	≠1	4			0,-1	10
9	1	4			0,-1	such
10			2			such

Abb. 12: Entscheidungstabellen eines TMA (Beispiel)

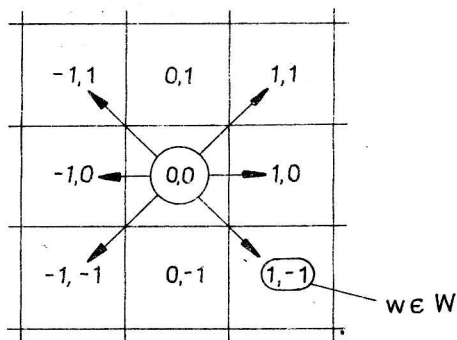


Abb. 13: Wertevorrat der Verschiebung des Lese-Schreib-Kopfes

EEEEEEEEEEEEEEEE		EEEEEEEEEEEEEEEE
E		E
E		E
E	111	E
E	111	E
E	1111	E
E	11111 1	E
E	11111111	E
E	111111111	E
E	1111 1111	E
E	1111	E
E	1111	E
E	1111	E
EEEEEEEEEEEEEEEE		EEEEEEEEEEEEEEEE

EEEEEEEEEEEEEEEE		EEEEEEEEEEEEEEEE
E		E
E		E
E	222	E
E	212	E
E	2122	E
E	22212 2	E
E	21111122	E
E	211211112	E
E	2112 2222	E
E	2112	E
E	2112	E
E	2222	E
EEEEEEEEEEEEEEEE		EEEEEEEEEEEEEEEE

Im Beispiel wurde der Wert $0 \in X$ durch das Leerzeichen ersetzt.

Abb. 14: Beispiel für die Konturmarkierung eines Musters

4. Entscheidungstabellengesteuerter TURING-Musterautomat mit zweidimensionalem Werteband

Ein TURING-Musterautomat TMA wird durch die Zusammenfassung

$$TMA = (X, Y, Z, W, r, s, q)$$

und im speziellen für ein zweidimensionales Werteband durch das Schema nach Abb. 10 bestimmt.

Es gilt dabei:

$x \in X$ - Eingangsmenge ($\dim(X) = 1$)

$y \in Y$ - Ausgangsmenge ($\dim(Y) = 1$)

$z \in Z$ - Zustandsmenge ($\dim(Z) \geq 1$)

$w \in W$ - Verschiebemenge ($\dim(W) \geq 1$)

$t \in T$ - Menge der Taktzeitpunkte

$y(t) = r(x(t), z(t))$ - Ergebnisfunktion

$z(t+1) = s(x(t), z(t))$ - Folgeabbildung

$w(t) = q(x(t), z(t))$ - Verschiebeabbildung

Die Funktionsweise ist analog der des TURING-Automaten mit eindimensionalem Werteband. In unseren Untersuchungen wurde festgestellt, daß die Anwendung dieses abstrakten Automatenmodells für Algorithmen der Vorverarbeitung von Mustern (Störunterdrückung, Glättung, Konturermittlung usw.) gut geeignet ist.

TMA Konturmarkierung eines zweidimensionalen Schwarz-Weiß-Musters

Ein zweidimensionales Schwarz-Weiß-Muster ist die Abbildung MU

$$MU : U \times V \rightarrow M$$

mit U, V - Ortskoordinaten

$$M = M^* \times \{E\}$$

wobei E - Randwert

$$M = \{0, 1\}$$

wobei 0 - Weißwert

1 - Schwarzwert

Der spezielle TMA ist ein einfaches Beispiel:

$$TMA = (X, Y, Z, W, r, q)$$

$$X = M \cup \{2\}$$

wobei 2 - Konturwert

$$Y \subseteq X$$

Das Entscheidungstabellensystem ist in Abb. 11 und die speziellen Tabellen sind in Abb. 12 dargestellt. Für die Werte $w \in W$ der Verschiebung gilt Abb. 13. Abb. 14 gibt ein einfaches Beispiel für die Konturmarkierung eines Musters.

Zusammenfassung

Entscheidungstabellen haben sich als algorithmisches System international durchgesetzt. Der praktische Anwendungsbereich und die theoretische Fundierung werden noch intensiv weiterentwickelt.

Über Entscheidungstabellenvorübersetzer wurde trotz ihrer großen Bedeutung hier nicht berichtet.

Literatur

- / 1/ Belke, W.; Jäkel, I.; Oschmann, A.: ETVOP - ein Entscheidungstabellenvorübersetzer für informationelle Prozesse der Leitung. Rechentechnik/Datenverarbeitung (demnächst).
- / 2/ Bibliographie Entscheidungstabellen. II. Auflage. VEB Carl Zeiss Jena, Jena, 1975.
- / 3/ Busch, H.-J.; Engeliën, M.; Stahn, H.: Entscheidungstabellentechnik - ein erweitertes Konzept mit Eignung zur Beschreibung von Prozessen vom diskontinuierlich diskreten Typ. Wiss. Z. Technische Universität Dresden 25, H. 1/2, 179 - 187 (1976).
- / 4/ Eichler, G.: Beiträge zur Übersetzung und Abarbeitung von Ein- und Ausgabeanweisungen problemorientierter Programmiersprachen. Dissertation, TU Dresden, 1975.
- / 5/ Engeliën, M.; Stahn, H.: Erweitertes Beispiel der Anwendung von Entscheidungstabellen zur Steuerung und Überwachung einer Brikettfabrik (Ausschnitt). Neue Bergbautechnik 6, H. 3, 234 - 237 (1976).
- / 6/ Engeliën, M.: Entscheidungstabellentechnik - Stand und Anwendungen bei der Analyse diskreter Prozesse. Die Technik 31, H. 6, 378 - 379 (1976).
- / 7/ Freitag, G. et al.: Einführung in die Entscheidungstabellentechnik. VEB Verlag Technik, Berlin, 1976.
- / 8/ Kjossew, G.; Werner, D.: Einsatz von Entscheidungstabellen zur Beschreibung von Programmsystemen. Rechentechnik/Datenverarbeitung (demnächst).

- / 9/ Malcev, A. I.: Algorithmen und rekursive Funktionen. Akademie-Verlag, Berlin, 1974.
- /10/ Petkoff, B.: Aspekte bei der Entwicklung von problemorientierten automatisierten Systemen der Informationsverarbeitung. Dissertation, TU Dresden, 1976.
- /11/ Schmidt, E. et al.: Praktikum zur Fertigungsüberwachung und Fertigungslenkung mit Geräten des Systems daro 1600/1602. Wiss. Z. TH Magdeburg (demnächst).
- /12/ Seifert, Ch.-B.: Entscheidungstabellen als Darstellungsmittel der Verhaltensweise von Turingautomaten mit zweidimensionalem Werteband im Hinblick auf die Darstellung von Algorithmen zur Vorverarbeitung zweidimensionaler, diskontinuierlicher Muster. Diplomarbeit, TU Dresden, Sektion Informationsverarbeitung, 1976.
- /13/ Stahn, H.: Zur digitalen Simulation diskreter Strukturen. Wiss. Z. Technische Universität Dresden 22, H. 6, 1033 - 1036 (1973).
- /14/ Stahn, H.: Entscheidungstabellen als Darstellungsmittel abstrakter Automaten. Wiss. Z. Technische Universität Dresden 23, H. 1, 159 - 162 (1974).
- /15/ Stahn, H.: Stand und Entwicklung automatisierter Systeme technologischer Prozesse (ASUTP). Die Technik 30, H. 10, (1975).
- /16/ Stahn, H. et al.: Zustandsgrößen und Kopplungsrelationen - eine Erweiterung der ESER-Rahmenmethodik. Rechentchnik/Datenverarbeitung 11, H. 11 (1974).
- /17/ Stahn, H.: Algorithmen; in: Baldeweg et al.: Grundlagen der Kybernetik II. Zentralinstitut für Kernforschung Rossendorf, Manuskriptdruck Nr. 308, März 1976.
- /18/ Winter, H.: TURING-Musterautomat mit zweidimensionalem Werteband. TU Dresden, Sektion Informationsverarbeitung, unveröffentlichtes Manuskript.

Eingegangen: 30. 11. 1976

Anschrift des Verfassers:

Prof. Dr. H. Stahn

Technische Universität Dresden

Sektion Informationsverarbeitung

DDR - 8027 Dresden

Mommensenstraße 13



