Werk

Titel: Ein Algorithmus zur automatischen Lösung konstruktiver Problemstellungen

Autor: KLIX, W.D.

Jahr: 1977

PURL: https://resolver.sub.uni-goettingen.de/purl?301416052_0006|log11

Kontakt/Contact

Digizeitschriften e.V.
SUB Göttingen
Platz der Göttinger Sieben 1
37073 Göttingen

info@digizeitschriften.de

Beiträge zur Algebra und Geometrie
6 (1977), 55 – 69

Ein Algorithmus zur automatischen Lösung konstruktiver Problemstellungen

Susanne Harms und Wolf-Dieter Klix

Das Lösen konstruktiver Problemstellungen besteht in der Angabe einer Folge von Elementaroperationen, die schrittweise aus den bekannten Elementen das gewünschte Resultat zu konstruieren gestatten (Lösungsalgorithmus). Ein Verfahren, das für jede lösbare Aufgabe bestimmter Problemklassen einen solchen Lösungsalgorithmus liefert, wird als *Konstruktionsalgorithmus* bezeichnet.
Für Konstruktionsaufgaben einfacher Struktur werden Prinzip und geometrische Beispiele eines solchen Konstruktionsalgorithmus angegeben. Dabei wird vom Begriff der konstruktiven Teiltheorie (T, K, E) ausgegangen, wobei K bzw. E endliche Mengen von in der Theorie T formulierbaren konstruktiv ausführbaren Grundkonstruktionen bzw. -relationen bedeuten. Zur Beschreibung der konstruktiven Prozesse in (T, K, E) werden Elemente algorithmischer Sprachen benutzt.

1. Formale Beschreibung konstruktiver Problemstellungen

Die meisten Arbeiten zur Digitalgeometrie bzw. -graphik beziehen sich auf die Phase der Darstellung der Resultate, die bei der Lösung konstruktiver Aufgabenstellungen erhalten wurden. Dazu gehören u. a. die Entwicklung von Datenstrukturen zur Erfassung geometrischer Probleme auf dem Rechner, die Bereitstellung von geometrieorientierten Programmiersprachen (Fachsprachen) und Schaffung von Programmsystemen zur Zeichnungserstellung. Die eigentliche Lösung der geometrischen Problemstellung, die Ermittlung des konstruktiven Lösungsweges, bleibt dabei unverändert Aufgabe des Menschen; der Rechner erleichtert die Arbeit lediglich durch Übernahme von Routinearbeit (Koordinatenberechnungen, geometrische Manipulationen, Zeichenarbeit usw.).
Die vorliegende Arbeit versucht, einen Beitrag zur Problematik der automatischen Lösungsfindung konstruktiver Aufgabenstellungen zu liefern, indem die algorithmische Darstellbarkeit konstruktiver Prozesse untersucht wird. Dabei werden als Konstruktionsaufgaben (in vorläufiger Ausdrucksweise) Probleme der folgenden Art bezeichnet: Aus gegebenen Elementen eines festgelegten Bereiches sind durch zulässige Operationen bestimmte Elemente dieses Bereiches zu ermitteln. Derartige Aufgaben, deren historischer Ursprung sicher geometrischer Natur ist, treten in den verschiedenartigsten Gebieten inner- und außerhalb der Mathematik auf. Beispiele

hierfür sind das Beweisen von Sätzen in axiomatisch formulierten mathematischen Theorien, die Ermittlung (Konstruktion) von technischen Systemen aus in ihrer Wirkungsweise bekannten Bausteinen, die Konstruktion geometrischer Figuren aus einigen ihrer Bestimmungsstücke. — Die Lösung einer Konstruktionsaufgabe besteht in der Angabe einer Folge von Operationen, die schrittweise die gesuchten, durch ihre Eigenschaften bzw. ihre Beziehungen zu den gegebenen Elementen bestimmten Elemente aus den gegebenen zu konstruieren gestattet.
Obwohl sich die folgenden Ausführungen auf geometrische Konstruktionen beziehen, haben sie auch in anderen im obigen Sinne konstruktiv arbeitenden Disziplinen Gültigkeit. — Die klassischen geometrischen Konstruktionen, in der euklidischen Ebene aus Punkten, Geraden(-stücken) und Kreisen bestehende Figuren mittels Lineal und Zirkel aus gegebenen geometrischen Elementen zu konstruieren, werden in dem folgenden allgemeinen Ansatz mit erfaßt.

Es sei T eine (axiomatische) mathematische Theorie, und in T seien eine endliche Menge K konstruktiv-ausführbarer, als Grundkonstruktionen bezeichneter Operationen und eine endliche Menge E effektiv-entscheidbarer Relationen (Prädikate, Attribute) formuliert und ausgezeichnet.
Die Grundkonstruktionen sind bedingte eindeutige Existenzaussagen in T der Gestalt

$$\wedge x_1, \ldots, x_n\big(H_1(x_1, \ldots, x_n) \Rightarrow \vee\, !\, !\, y H_2(y, x_1, \ldots, x_n)\big),$$

die als n-stellige Konstruktionsoperationen geschrieben werden können:

$$y := \varkappa_i^{n_i}(x_1, \ldots, x_{n_i}), \quad \text{falls } H_1(x_1, \ldots, x_{n_i}).$$

Den Relationen aus E entsprechen n-stellige Test- oder Entscheidungsoperationen, die den in der jeweiligen Relation R stehenden Elemente-n-Tupeln den Wahrheitswert 1 oder 0 zuordnen (b ... Boolesche Variable):

$$b := \varepsilon_i^{n_i}(x_1, \ldots, x_{n_i})$$

mit

$$b = \begin{cases} 1 \Leftrightarrow (x_1, \ldots, x_{n_i}) \in R \Leftrightarrow R(x_1, \ldots, x_n), \\ 0 \Leftrightarrow (x_1, \ldots, x_{n_i}) \notin R \Leftrightarrow \neg R(x_1, \ldots, x_n). \end{cases}$$

Zur Vereinfachung der Schreib- und Ausdrucksweise wird die Gesamtheit der Entscheidungsoperationen ebenfalls mit E bezeichnet.
Der Index i gilt als Unterscheidungsindex für die k Konstruktionsoperationen $\varkappa_i \in K$ ($i = 1, \ldots, k$) bzw. e Entscheidungsoperationen $\varepsilon_i \in E$ ($i = 1, \ldots, e$).
Das Tripel (T, K, E) wird nach SCHREIBER [6, 7], wo diese Zusammenhänge ausführlicher dargestellt sind, als eine konstruktive Teiltheorie von T bezeichnet.

Definition 1. Die bedingte Existenzaussage $A \equiv \wedge \mathfrak{x}\big(H_1(\mathfrak{x}) \Rightarrow \vee\, \mathfrak{y} H_2(\mathfrak{x}, \mathfrak{y})\big)$ mit Tupeln $\mathfrak{x}$, $\mathfrak{y}$ aus eventuell verschiedenartigen Elementen von T heißt *Konstruktionsaufgabe in* (T, K, E): $\Leftrightarrow$
A ist unter ausschließlicher Verwendung von Elementen aus K und E (in endlich vielen Schritten) zu beweisen.

Die Aufgabe A heißt in (T, K, E) lösbar, wenn es einen derartigen Beweis gibt, und jeder solche Beweis heißt *Konstruktionsweg, Konstruktion, Lösung* oder *Lösungsalgorithmus* dieser Aufgabe und soll durch $\omega : \mathfrak{x} \to \mathfrak{y}$ bezeichnet werden. Dabei ist ω als Folge von Elementen aus $K \cup E$ aufzufassen, die $\mathfrak{x}$ schrittweise in $\mathfrak{y}$ überführt. Eine Veranschaulichung von ω durch ein Flußdiagramm, dessen Arbeits- bzw. Prüf-

knoten mit Elementen von K bzw. E belegt werden, rechtfertigt die Bezeichnung von ω als Lösungsalgorithmus für A.

Zur formalen Beschreibung dieser ihrer Natur nach algorithmischen Prozesse reichen die Ausdrucksmittel einer die Theorie T beschreibenden (formalen) Sprache im allgemeinen nicht aus, und es ist zweckmäßig bzw. für die folgenden Betrachtungen sogar erforderlich, die Lösungsalgorithmen in konstruktiven Teiltheorien durch algorithmische Sprachen zu beschreiben. Die algorithmischen Sprachen ermöglichen in adäquater Weise, den „dynamischen Charakter" algorithmischer Prozesse zu erfassen, während demgegenüber die Sprache von T dem „statischen Charakter" der Aussagen der Theorie T angepaßt ist.

Die Beschreibungen der Operationen $\varkappa_i \in K$ und $\varepsilon_i \in E$ sind bereits Elemente algorithmischer Sprachen, für die die Anweisungen grundlegende Bedeutung haben. Über allgemeine Eigenschaften algorithmischer Sprachen zur Erfassung algorithmischer Prozesse sei hier z. B. auf [8] verwiesen. — Im folgenden werden zur Vereinfachung der Beschreibung von Algorithmen für konstruktive Prozesse ohne Erklärung verständliche ALGOL-artige Sprachelemente verwendet.

Die eigentliche Lösung einer vorliegenden Konstruktionsaufgabe $A[\mathfrak{x}, \mathfrak{y}]$ in einer konstruktiven Teiltheorie (T, K, E) besteht darin, den Lösungsalgorithmus $\omega : \mathfrak{x} \to \mathfrak{y}$ zu finden, dessen Elemente $\omega_i \in K \cup E$ durch „Bausteine" der algorithmischen Sprache beschrieben werden. Für die Ermittlung von ω spielen sehr viele und unterschiedliche Faktoren wie Wissen, Erfahrung, Institution u. a. eine Rolle, die sich nicht bzw. nur teilweise algorithmisch erfassen lassen; hierin kommt der heuristische Charakter des Konstruktionsprozesses zum Ausdruck. In diesem Sinne läßt sich das Suchen nach einem Verfahren zur Ermittlung geeigneter Lösungsalgorithmen für Konstruktionsaufgaben auch in das Gebiet der kybernetischen Forschung, das unter der Bezeichnung Problemlösen bekannt ist und zu dem es bereits sehr viele Beiträge in der Literatur gibt, einordnen (vgl. [3, 4]).

Definition 2. $\mathfrak{K}$ sei eine Klasse von Konstruktionsaufgaben der konstruktiven Teiltheorie (T, K, E). α heißt ein *Konstruktionsalgorithmus für* $\mathfrak{K}$: $\Leftrightarrow$
Für jede Konstruktionsaufgabe $A \in \mathfrak{K}$ entscheidet α die Lösbarkeit, und für jede lösbare Aufgabe $A \in \mathfrak{K}$ liefert α einen Lösungsalgorithmus ω, d. h. $\alpha: A[\mathfrak{x}, \mathfrak{y}] \to \omega$ mit $\omega : \mathfrak{x} \to \mathfrak{y}$.

Die rechentechnische Realisierung eines Konstruktionsalgorithmus ermöglicht die automatische Lösung von Aufgaben der Klasse $\mathfrak{K}$ bzw. die Entscheidung, ob sie lösbar sind oder nicht. — Für relativ einfach zu überblickende Aufgabenklassen $\mathfrak{K}$ wird im folgenden Abschnitt ein mögliches Prinzip für Konstruktionsalgorithmen beschrieben.

Die heuristisch orientierte Denkweise bei der Ermittlung des Lösungsweges für kompliziertere Aufgabenklassen, die sich als eine unregelmäßige Folge heuristischer und algorithmischer Tätigkeiten beim Konstruieren widerspiegelt, wird rechentechnisch zweckmäßigerweise durch einen Dialogverkehr zwischen Mensch und Rechner realisiert. Der heuristisch gesteuerte Konstruktions„algorithmus" für derartige Aufgabenklassen wird auch als „heuristischer Algorithmus" (z. B. in [1]) bezeichnet.

2. Prinzip eines Konstruktionsalgorithmus

In der konstruktiven Teiltheorie (T, K, E) der Theorie T sei eine feste Aufgabenklasse $\mathfrak{K}$ wie folgt festgelegt: Für eine endliche Teilmenge M von Elementen der Theorie T seien sämtliche für die Aufgabenklasse wesentlichen Eigenschaften und

untereinander bestehenden Beziehungen in einem „*Speicher*" S erfaßt, so daß $\mathfrak{R}$ durch das Paar (M, S) vollständig repräsentiert wird. Insbesondere beinhaltet S, durch welche Konstruktionen aus K die Elemente von M auseinander hervorgehen und welche Relationen aus E für die Elemente von M zutreffen. Jede Aufgabe $A \in \mathfrak{R}$ besteht dann darin, bei Kenntnis von S aus den gegebenen Elementen $\mathfrak{x} \subseteq M$ durch Konstruktions- bzw. Entscheidungsoperationen $\varkappa_i \in K$ bzw. $\varepsilon_i \in E$ die gesuchten Elemente $\mathfrak{y} \subseteq M$ zu ermitteln.
Der hier vorzustellende Konstruktionsalgorithmus α für die Aufgabenklasse $\mathfrak{R}$ von (T, K, E) arbeitet nach dem folgenden Prinzip: Es sei $A[\mathfrak{x}, \mathfrak{y}]$ eine beliebige Aufgabe der durch M und S festgelegten Klasse. Auf die gegebenen Elemente der Ausgangsmenge $\mathfrak{x}$ werden die Konstruktionen $\varkappa_i \in K$ $(i = 1, \ldots, k)$ angewandt, und die dadurch entstehenden Elemente aus M werden zu $\mathfrak{x}$ hinzugefügt. Auf diese gegenüber $\mathfrak{x}$ im allgemeinen erweiterte Menge werden die Konstruktionen $\varkappa_i$ erneut angewandt usw. Dieses Verfahren wird so oft durchgeführt, bis entweder die gesuchten Elemente erhalten werden, dann ist die betreffende Aufgabe lösbar, oder bis die $\varkappa_i$ keine weiteren Elemente ergeben, dann ist die Aufgabe unlösbar.
Es bedeute $\varkappa_i^{n_i}[N]$ die Anwendung der n_i-stelligen (partiellen) Operation $\varkappa_i$ auf alle zulässigen n_i-Tupel aus Elementen von N, für die sich Elemente von $M \setminus N$ ergeben. Dabei sei zur Vereinfachung der Schreibweise angenommen, daß eventuell notwendige Testoperationen $\varepsilon_i \in E$ für die Tupel aus N durch $\varkappa_i[N]$ mit erfaßt sind; so ist beispielsweise der Schnitt zweier Geraden g_1, g_2 einer Ebene nur möglich, wenn sie nicht parallel sind, und dem Schneiden von g_1 und g_2 geht also der Test auf Parallelität voraus. — Damit läßt sich der erste Teil des Konstruktionsalgorithmus α für $\mathfrak{R} \equiv (M, S)$ beschreiben durch

Eing.: $A[\mathfrak{x}, \mathfrak{y}]$;

$N := \mathfrak{x}$; ANF: **for** $i := 1(1)\ k$ **do** $N_i := \varkappa_i^{n_i}[N]$; $N' := N \cup \bigcup_{i=1}^{k} N_i$;

if $\mathfrak{y} \subseteq N'$ **then goto** Fall L;

if $N' = N$ **then goto** Fall U **else** $N := N'$;

goto ANF.

Ausgang bei Fall L bedeutet Lösbarkeit, bei Fall U Unlösbarkeit von A.
Das Kernstück von α besteht in den Operationen $\varkappa_i[N]$, denn hieraus ergibt sich im Fall der Lösbarkeit von $A[\mathfrak{x}, \mathfrak{y}]$ der Lösungsalgorithmus $\omega : \mathfrak{x} \to \mathfrak{y}$, indem zunächst für jeden Schritt alle Test- und Konstruktionsoperationen „gemerkt" werden, durch die neue Elemente von M erhalten werden; anschließend ist aus dieser Gesamtheit Ω durch geeignete Verfahren eine nach noch festzulegenden Bewertungsmaßstäben möglichst günstige Folge $\omega = (\omega_1, \ldots, \omega_n)$ auszuwählen. Die Ermittlung der potentiell zum Lösungsweg ω beitragenden Konstruktionen erfolgt in jedem Schritt für jede der n_i-stelligen Operationen $\varkappa_i^{n_i}$ nach dem Algorithmus, wobei $\varphi_j^{n_i}(N)$ die Bildung der j-ten Kombination von n_i Elementen $(e_{j1}, \ldots, e_{jn_i}) =: \mathfrak{e}_j$ aus N bedeutet; I ist die Anzahl dieser möglichen Kombinationen:

Eing.: N, $\varkappa_i^{n_i}$;

$I := \binom{card\ N}{n_i}$; **for** $j := 1(1)\ I$ **do begin** $\mathfrak{e}_j := \varphi_j^{n_i}(N)$;

if ($e_j \in$ Definitionsbereich von $\varkappa_i^{n_i}$) $\wedge$ (e_j erfüllt die notwendigen Testbedingungen)

then begin $z_j := \varkappa_i^{n_i}(e_j)$;

if $z_j \notin N$ **then** Merke „$z_j := \varkappa_i^{n_i}(e_j)$“ **end end.**

Die Gesamtheit Ω aller durch α erzeugten Konstruktionsschritte ω_i der Gestalt $\omega \ldots z_j := \varkappa_i^{n_i}(e_1, \ldots, e_{n_i})$ ist mit einem Labyrinth vergleichbar, in dem im Fall der Lösbarkeit von $A \in \mathfrak{R}$ wenigstens ein Weg ω vom Eingang $\mathfrak{x}$ zum Ausgang $\mathfrak{y}$ existiert. Durch das folgende Verfahren werden zunächst alle Schritte ω_i aus Ω bestimmt, die einen Beitrag zur gesuchten Menge $\mathfrak{y}$ liefern:

$$\Omega' := \left\{\omega_i \in \Omega \mid \bigvee y_j \in \mathfrak{y}\left(y_j := \varkappa_i^{n_i}(e_j)\right)\right\}.$$

Anschließend werden die Elemente von M und die Konstruktionsschritte von Ω ermittelt, die die für die Bestimmung von Ω' erforderlichen Elemente $L = \bigcup_i e_j$ bereitstellen. Dies wird solange wiederholt, bis L nur gegebene Elemente enthält:

Eing.: $\mathfrak{x}$, $\mathfrak{y}$, Ω

$L := \mathfrak{y}$; $k := 1$; ANF: $\Omega_k := \left\{\omega_i \in \Omega \mid \bigvee y_j \in L\left(y_j := \varkappa_i^{n_i}(e_j)\right)\right\}$;

$L' := \bigcup_i e_j \setminus \mathfrak{x}$; **if** $L' = \emptyset$ **then goto** END **else** $L := L'$;

$k := k + 1$; **goto** ANF; END.

Da Ω nur solche Konstruktionsschritte enthält, die durch α aus $\mathfrak{x}$ erhalten wurden, bricht dieses Verfahren nach k Schritten ab. Ein Lösungsalgorithmus $\omega : \mathfrak{x} \to \mathfrak{y}$ ergibt sich nun durch Abarbeiten der Konstruktionsschritte ω_i aus den ermittelten Ω_j in der Reihenfolge $\Omega_k, \Omega_{k-1}, \ldots, \Omega_1$.
Weitere Überlegungen zur Ermittlung von ω aus Ω, die insbesondere möglichst effektive Lösungen ergeben, sollen hier nicht angestellt werden. Der oben erwähnte Zusammenhang zu den Labyrinth-Problemen verdeutlicht einerseits den Schwierigkeitsgrad und legt andererseits Ansätze zur Bearbeitung dieser Aufgabe nahe.

3. Realisierung mittels Matrizenoperationen

In diesem Abschnitt wird ein Konstruktionsalgorithmus angegeben, der das in Abschnitt 2 beschriebene Prinzip durch Matrizenoperationen realisiert. Dieser Algorithmus läßt sich auf Klassen solcher geometrischer Konstruktionsaufgaben anwenden, die sich auf die Inzidenzrelation sowie weitere binäre Relationen beziehen, z. B. Parallelität und Orthogonalität von Geraden. Aufgaben mit metrischem Charakter (Abstand, Winkel, ...), die also nicht mehr ausschließlich durch binäre Relationen erfaßt werden, können z. B. mit dem im nächsten Abschnitt beschriebenen Konstruktionsalgorithmus „Bestimmungslinien“ behandelt werden.
Eine Aufgabenklasse $\mathfrak{R} \equiv (M, S)$ wird dadurch erfaßt, daß die zwischen den Elementen von M bestehenden (binären) Relationen durch Matrizen beschrieben werden. Da M eine endliche Menge ist, läßt sich jede binäre Relation R aus M durch eine Relationsmatrix $\boldsymbol{R}$ beschreiben, für deren Elemente R_{ij} gilt:

$$R_{ij} \neq 0 \Leftrightarrow (x_i, x_j) \in R.$$

Die Gesamtheit dieser den Relationen aus M zugeordneten Matrizen entspricht dem „*Speicher*" S.
Auf der Menge M ist eine (Bewertungs-)Funktion β erklärt, die jedem Element $x \in M$ einen der Werte 0 oder 1 zuordnet:

$$\beta(x) = \begin{cases} 1, \text{ falls } x \in M \text{ bekannt ist,} \\ 0 \text{ sonst.} \end{cases}$$

Für jede Sorte von Elementen aus M wird ein (Boolescher) Elementevektor $\boldsymbol{X}$ gebildet, dessen Komponenten $X(i)$ die diesen Elementen durch β zugeordneten Werte sind:

$$\boldsymbol{X} = \big(\beta(x_1), \ldots, \beta(x_n)\big)^{\mathsf{T}}.$$

Für gegebene bzw. konstruierte Elemente x wird $\beta(x) = 1$ gesetzt. Das Ziel des Konstruktionsvorganges für die Aufgabe $A[\mathfrak{x}, \mathfrak{y}] \in \mathfrak{A}$ besteht darin, für jedes $y \in \mathfrak{y}$ den Wert $\beta(y) = 1$ zu erhalten:

$$A[\mathfrak{x}, \mathfrak{y}] \text{ gelöst } \Leftrightarrow \bigwedge y(y \in \mathfrak{y} \Rightarrow \beta(y) = 1).$$

Die in der konstruktiven Teiltheorie (T, K, E) zulässigen Konstruktions- und Testoperationen aus K und E werden durch arithmetische und logische Operationen zwischen den Relationsmatrizen und Elementevektoren realisiert.

3.1. Konstruktionen in der projektiven Inzidenzebene

In der konstruktiven Teiltheorie der projektiven Inzidenzgeometrie T mit den Konstruktionen

$\varkappa_1$ ... Verbinden zweier Punkte: $g := \varkappa_1(P_1, P_2) = P_1P_2$,

$\varkappa_2$ ... Schneiden zweier Geraden: $P := \varkappa_2(g_1, g_2) = g_1 \cap g_2$

und der Inzidenzrelation | zwischen den Punkten und Geraden gilt bekanntlich:
Die Gerade g ist genau dann konstruierbar, wenn es zwei bekannte oder konstruierbare Punkte P_i und P_j gibt, die mit g inzidieren:

$$\beta(g) = 1 \Leftrightarrow \bigvee P_i, P_j\big(\beta(P_i) = \beta(P_j) = 1 \wedge (P_i, g) \in | \wedge (P_j, g) \in | \wedge P_i \neq P_j\big).$$

Der Punkt P ist genau dann konstruierbar, wenn es zwei mit P inzidente bekannte oder konstruierbare Geraden g_i und g_j gibt:

$$\beta(P) = 1 \Leftrightarrow \bigvee g_i, g_j\big(\beta(g_i) = \beta(g_j) = 1 \wedge (P, g_i) \in | \wedge (P, g_j) \in | \wedge g_i \neq g_j\big).$$

Für die Aufgabenklasse $\mathfrak{A} \equiv (M, S)$ der konstruktiven Teiltheorie $(T, \{\varkappa_1, \varkappa_2\}, |)$ sei die Inzidenzmatrix $\boldsymbol{I} = (I_{jk})_{p,g}$ definiert durch $I_{jk} = 1 \Leftrightarrow (P_j, g_k) \in |$, und als Elementevektoren seien der Punktevektor $\boldsymbol{P} = \big(\beta(P_1), \ldots, \beta(P_p)\big)^{\mathsf{T}}$ und der Geradenvektor $\boldsymbol{G} = \big(\beta(g_1), \ldots, \beta(g_g)\big)^{\mathsf{T}}$ entsprechend den p Punkten und g Geraden von M gebildet. Damit läßt sich die Konstruierbarkeit der Punkte und Geraden von M berechnen:

$$\beta(g_k) = 1 \Leftrightarrow \boldsymbol{P}^{\mathsf{T}} \cdot (I_{1k}, \ldots, I_{pk}) \geqq 2,$$

$$\beta(P_l) = 1 \Leftrightarrow (I_{l1}, \ldots, I_{lg}) \cdot \boldsymbol{G} \geqq 2.$$

Die in Abschnitt 2 in der allgemeinen Darstellung des Konstruktionsalgorithmus α beschriebenen Schritte der Gestalt $\varkappa_i^n[N]$ werden hier im wesentlichen durch die Matrizenoperationen

$$\varkappa_1 \colon \boldsymbol{P}^{\mathsf{T}} \cdot \boldsymbol{I} =: \boldsymbol{G}^*,$$

$$\varkappa_2 \colon \boldsymbol{I} \cdot \boldsymbol{G} =: \boldsymbol{P}^*$$

realisiert. Aus $\boldsymbol{G}^*$ bzw. $\boldsymbol{P}^*$ ergeben sich durch die anschließenden Operationen

$$G^*(k) \geqq 2 \Rightarrow G(k) = \beta(g_k) = 1$$

bzw.

$$\boldsymbol{P}^*(l) \geqq 2 \Rightarrow P(l) = \beta(P_l) = 1$$

die „neuen" Geraden- bzw. Punktevektoren. Einem Durchlauf von α entspricht hier die Operationenfolge

$$\boldsymbol{P}^{\mathsf{T}} \cdot \boldsymbol{I} =: \boldsymbol{G}^{*\mathsf{T}}, \quad \boldsymbol{G}^* \to \boldsymbol{G}', \quad \boldsymbol{I} \cdot \boldsymbol{G}' =: \boldsymbol{P}^*, \quad \boldsymbol{P}^* \to \boldsymbol{P}'.$$

Falls damit alle Elemente von $\mathfrak{y}$ bekannt sind (Test 1), ist die Aufgabe $A[\mathfrak{x}, \mathfrak{y}] \in \mathfrak{A}$ gelöst; falls $\boldsymbol{G}' = \boldsymbol{G}$ und $\boldsymbol{P}' = \boldsymbol{P}$ ist (Test 2), ist A unlösbar; anderenfalls erfolgt ein weiterer Durchlauf mit den neuen Elementevektoren $\boldsymbol{P}'$ und $\boldsymbol{G}'$.
Das Konstruieren von $\mathfrak{y}$ aus $\mathfrak{x}$ besteht also darin, die Matrix $\boldsymbol{I}$ abwechselnd von links mit $\boldsymbol{P}^{\mathsf{T}}$ und von rechts mit $\boldsymbol{G}$ zu multiplizieren und $\beta(x_i) = 1$ zu setzen, falls die Multiplikation in der i-ten Komponente einen Wert $\geqq 2$ ergibt, sowie diesen Schritt für die Ermittlung von x_i zu merken. Bei der Multiplikation mit $\boldsymbol{I}$ brauchen jeweils nur die Spalten bzw. Zeilen von $\boldsymbol{I}$ berücksichtigt zu werden, für deren entsprechende Gerade bzw. Punkte $\beta = 0$ gilt.
Ein einfaches Beispiel hierzu liefert die durch die Desargues-Figur beschriebene Aufgabenklasse $\mathfrak{D}$: M besteht aus den zehn Punkten $P_1, \ldots, P_{10}$ und den zehn Geraden $g_1, \ldots, g_{10}$, deren Inzidenzbeziehungen nach Abb. 1 durch die Matrix

$$\boldsymbol{I} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

erfaßt werden. Jede Aufgabe $A \in \mathfrak{D}$ besteht nun darin, aus einer Teilmenge $\mathfrak{x} \subset M$ eine Teilmenge $\mathfrak{y} \subseteqq M$ unter Benutzung von $\boldsymbol{I}$ durch $\varkappa_1$ und $\varkappa_2$ zu ermitteln.

Beispiele

Aufgabe 1. Aus $P_1, P_2, P_3, P_4, P_5, P_7$ ist die Desargues-Figur zu ermitteln!

$\boldsymbol{P} = (1\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 0\ 0)^{\mathsf{T}}$;

$\boldsymbol{G} = (0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0)^{\mathsf{T}}$;

1. Durchlauf:

$G^* := P^T \cdot I = (3\ 2\ 2\ 2\ 1\ 2\ 2\ 2\ 1\ 1)^T \rightarrow G' = (1\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 0)^T;$

$P^* := I \cdot G' = (*\ *\ *\ *\ *\ 1\ *\ 1\ 2\ 2)^T \rightarrow P' = (1\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ )^T;$

Test 1: negativ; Test 2: negativ;

$P := P'$; $G := G'$;

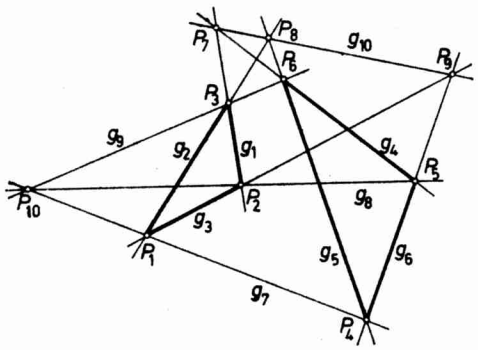

Abb. 1

2. Durchlauf:

$G^* := P^T \cdot I = (*\ *\ *\ *\ 1\ *\ *\ *\ 2\ 2)^T \rightarrow G' = (1\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 1\ 1)^T;$

$P^* := I \cdot G' = (*\ *\ *\ *\ *\ 2\ *\ 2\ *\ *)^T \rightarrow P' = (1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1)^T;$

Test 1: negativ; Test 2: negativ;

$P := P'$; $G := G'$;

3. Durchlauf:

$G^* := P^T \cdot I = (*\ *\ *\ *\ 3\ *\ *\ *\ *\ *)^T \rightarrow G' = (1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1)^T;$

Test 1: positiv; Aufgabe gelöst.

Aufgabe 2. Aus P_1, P_2, P_3, g_4, g_6, g_{10} ist P_{10} zu ermitteln!

$P = (1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0)^T;$

$G = (0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 1)^T;$

1. Durchlauf:

$G^* := P^T \cdot I \rightarrow G' = (1\ 1\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1)^T;$

$P^* := I \cdot G' \rightarrow P' = (1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 0)^T;$

Test 1: negativ; Test 2: negativ;

$P := P'$; $G := G'$;

2. Durchlauf:

$G^* := P^T \cdot I \rightarrow G' = (1\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 1)^T;$

$P := I \cdot G' \rightarrow P' = (1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 0);$

Test 1: negativ; Test 2: positiv; Aufgabe nicht lösbar.

3.2. Erweiterungen der konstruktiven Teiltheorie

Mit der gleichen Verfahrensweise läßt sich auch eine Inzidenzgeometrie für Punkte, Geraden und Kreise behandeln.
Es seien die Elemente der Mengen K und E
die Konstruktionen

$\varkappa_1$ . . . Verbinden zweier Punkte $g := \varkappa_1(P_1, P_2)$,
$\varkappa_2$ . . . Schneiden zweier Linien $P := \varkappa_2(b_1, b_2)$,
$\varkappa_3$ . . . Verbinden dreier Punkte $k := \varkappa_3(P_1, P_2, P_3)$,
$\varkappa_4$ . . . Kreis um Punkt P_1 $k := \varkappa_4(P_1, P_2)$,

die Relationen

| . . . Inzidenzrelation zwischen Punkten, Geraden und Kreisen,
|* . . . Mittelpunktsrelation zwischen Punkten und Kreisen,

wobei b_i Gerade oder Kreis sein kann. Dann gilt

$$\beta(g) = 1 \Leftrightarrow \vee P_i, P_j\big(\beta(P_i) = \beta(P_j) = 1 \wedge (P_i, g) \in | \wedge (P_j, g) \in | \wedge i \neq j\big),$$

$$\beta(k) = 1 \Leftrightarrow \vee P_i, P_j, P_l\big(\beta(P_i) = \beta(P_j) = \beta(P_l) = 1 \wedge (P_i, k) \in | \wedge (P_j, k) \in | \wedge (P_l, k) \in | \wedge i \neq j \neq l \neq i\big) \vee \vee P_i, P_j\big(\beta(P_i) = \beta(P_j) = 1 \wedge (P_i, k) \in |^* \wedge (P_j, k) \in |\big),$$

$$\beta(P) = 1 \Leftrightarrow \vee b_i, b_j\big(\beta(b_i) = \beta(b_j) = 1 \wedge (P, b_i) \in | \wedge (P, b_j) \in | \wedge i \neq j\big).$$

Der Speicher S der Aufgabenklasse $\mathfrak{A} = (M, S)$ von $(T, \{\varkappa_1, \ldots, \varkappa_4\}, \{|, |^*\})$ wird dargestellt durch

$$\boldsymbol{I} = (I_{ij})_{p,g+k} = (I^g_{ij})_{p,g}\,(I^k_{ij})_{p,k} \quad (\text{mit } I^g_{ij} = I_{ij},\quad I^k_{ij} = I_{i,j+g})$$

mit

$$I_{ij} = 1 \Leftrightarrow (P_i, g_i) \in | \text{ für } j \leqq g \vee (P_i, k_{j-g}) \in | \text{ für } j > g,$$
$$I_{ij} = 2 \Leftrightarrow (P_i, k_{j-g}) \in |^*.$$

Da nicht gleichzeitig $(P, k) \in |$ und $(P, k) \in |^*$ sein kann, können beide Relationen in einer Matrix erfaßt werden. Als Elementevektoren werden der Punktevektor $\boldsymbol{P} = \big(\beta(P_1), \ldots, \beta(P_p)\big)^{\mathsf{T}}$ und der Linienvektor $\boldsymbol{B} = \big(\beta(g_1), \ldots, \beta(g_g), \beta(k_1), \ldots, \beta(k_k)\big)^{\mathsf{T}}$ gebildet. Für die Konstruierbarkeit der Kreise (Punkte, Geraden wie in 3.1.) gilt dann:

$$\beta(k_j) = 1 \Leftrightarrow P^{\mathsf{T}}(I^k_{1,j} \cdots I^k_{p,g}) \geqq 3.$$

Der Aufbau der Indizenzmatrix I ermöglicht es, $\varkappa_3$ und $\varkappa_4$ durch eine Rechenoperation zu realisieren ($\varkappa_3^*$ . . . Kreis konstruieren):

$\varkappa_1$: $\boldsymbol{P}^{\mathsf{T}} \cdot \boldsymbol{I}^g =: \boldsymbol{G}^{*\mathsf{T}}$,

$\varkappa_2$: $\boldsymbol{I} \cdot \boldsymbol{B} =: \boldsymbol{P}^*$,

$\varkappa_3$: $\boldsymbol{P}^{\mathsf{T}} \cdot \boldsymbol{I}^k =: \boldsymbol{K}^{*\mathsf{T}}$.

Ein Durchlauf von α wird durch folgende Rechen- und Testoperationen realisiert:

$$\begin{aligned} &\boldsymbol{P}^{\mathsf{T}} \cdot \boldsymbol{I}^{g} =: \boldsymbol{G}^{*\mathsf{T}}, && \boldsymbol{G}^{*} \to \boldsymbol{G}', \\ &\boldsymbol{P}^{\mathsf{T}} \cdot \boldsymbol{I}^{k} =: \boldsymbol{K}^{*\mathsf{T}}, && \boldsymbol{K}^{*} \to \boldsymbol{K}', \qquad \boldsymbol{B}' := \boldsymbol{G}'\boldsymbol{K}', \\ &\boldsymbol{I} \cdot \boldsymbol{B} =: \boldsymbol{P}^{*}, && \boldsymbol{P}^{*} \to \boldsymbol{P}'. \end{aligned}$$

In konstruktiven Teiltheorien mit der Parallelität Π von Geraden als einer Grundrelation bieten sich zwei Möglichkeiten für die Beschreibung von Π durch Matrizen an:
Die erste Möglichkeit geht von der Erweiterung der Inzidenzmatrix $\boldsymbol{I}$ aus, indem die Parallelität zweier Geraden durch ihren gemeinsamen uneigentlichen „Schnitt"-Punkt erfaßt wird. Das wird erreicht, indem M um die uneigentliche Gerade u mit $\beta(u) = 1$ und alle Richtungen (= „Schnitt"-Punkte) U_i paralleler Geraden erweitert wird. Zwei Geraden sind damit genau dann parallel, wenn es einen mit beiden inzidierenden uneigentlichen Punkt gibt:

$$(g_i, g_k) \in \Pi \Leftrightarrow \vee\, U_i\big((U_i, u) \in \mid, \quad (U_i, g_i) \in \mid, \quad (U_i, g_k) \in \mid\big).$$

Damit ist die Konstruktion $g_k := \varkappa_5(g_i, P) = P \parallel g_i$ einer zu g_i parallelen Geraden g_k durch P möglich.
Die zweite Möglichkeit ist gleichwertig mit der Beschreibung der Orthogonalitätsrelation Σ durch Matrizen:
Es wird außer $\boldsymbol{I}$ eine weitere Matrix $\boldsymbol{L} = (L_{ik})_{g,g}$ eingeführt, für deren Elemente gilt:

$$\begin{aligned} &L_{ik} = 1 \wedge i < k \Leftrightarrow (g_i, g_k) \in \Pi \Leftrightarrow g_i \parallel g_k, \\ &L_{ik} = 1 \wedge i > k \Leftrightarrow (g_i, g_k) \in \Sigma \Leftrightarrow g_i \perp g_k, \\ &L_{ik} = 0 \Leftarrow i = k. \end{aligned}$$

Oberhalb der Diagonalen von $\boldsymbol{L}$ ist Π, unterhalb Σ erfaßt. Damit ist außer $\varkappa_5$ die Konstruktion

$$g_k := \varkappa_6(g_i, P) = P \perp g_i$$

einer zu g_i senkrechten Geraden g_k durch P möglich. Der Konstruktionsalgorithmus α ist hierbei natürlich um die Operationen mit $\boldsymbol{L}$ in entsprechender Weise zu erweitern.

4. Realisierung mittels Listenmanipulationen

4.1. Allgemeines Vorgehen

Während die in Abschnitt 3 beschriebene Matrizenmethode nur für Teiltheorien spezieller Struktur anwendbar ist, wird sich ein Konstruktionsalgorithmus allgemeiner Art Mitteln der nichtnumerischen Datenverarbeitung bedienen. Das setzt die Bearbeitungsmöglichkeit alphanumerischer Zeichen voraus.
Der Speicher zur Erfassung der Beziehungen der Elemente der Aufgabenklasse stellt sich dar als Liste aller potentiell möglichen Konstruktionsschritte mit allen Elementen. Konstruiert wird, indem durch Vergleich der gegebenen Elemente $\mathfrak{x}$ aus M mit den Bestimmungsstücken der Konstruktionsschritte im Speicher ausführbare Konstruktionsschritte herausgefunden werden. Dieses Vorgehen soll für den speziellen Konstruktionsalgorithmus „Bestimmungslinien" für Aufgabenklassen der Teiltheorie

der Konstruktionen mit Zirkel und Lineal in der euklidischen Geometrie der Ebene spezifiziert werden.[1])

Elemente von M können zunächst Punkte, Geraden und Kreise sein. Zur eindeutigen bzw. rechentechnisch günstigen Darstellung der Konstruktionen im Rechner läßt man außerdem Halbgeraden, Kreisbögen, Geradenpaare bzw. gerichtete Strecken sowie Maßvariable für Längen und Winkel zu.

Jedes Element von M wird im Rechner durch n Zeichen dargestellt (hier: $n = 3$), wobei das erste Zeichen den Typ des Elements kennzeichnet.

Es sei

P ... Punkt, G ... Gerade,	K ... Kreis,
H ... Halbgerade,	B ... Kreisbogen,
D ... Geradenpaar,	S ... gerichtete Strecke,
R ... Abstand,	W ... Winkelgröße.

Aus der Wahl der Konstruktionsmittel ergeben sich als mögliche Elemente von K (X ... beliebiges alphanumerisches Zeichen):

$\varkappa_1$: Gerade := Verbinden (Punkt, Punkt)	*GER*	*PXX*	*PXX*
$\varkappa_2$: Kreis := Kreiszeichnen (Punkt, Abstand)	*KRS*	*PXX*	*RXX*
$\varkappa_3$: Punkt := Schneiden (Gerade oder Kreis, Gerade oder Kreis)[2])			
$\varkappa_4$: Gerade := Parallele 1 (Gerade, Punkt)	*PRL*	*GXX*	*PXX*
$\varkappa_5$: Geradenpaar := Parallele 2 (Gerade, Abstand)	*PAR*	*GXX*	*RXX*
$\varkappa_6$: Kreis := Thaleskreis (Punkt, Punkt)	*TKR*	*PXX*	*PXX*
$\varkappa_7$: Kreisbogen := Fußkreis (Strecke, Winkelgröße)	*FKR*	*SXX*	*WXX*
$\varkappa_8$: Halbgeraden := Winkel (Halbgerade, Winkelgröße)	*WIN*	*HXX*	*WXX*

Bemerkungen

1. $\varkappa_1$ bis $\varkappa_3$ sind Grundkonstruktionen, sie entprechen Axiomen der Theorie, die übrigen Konstruktionen, deren Wahl hier willkürlich ist, sind zusammengesetzte Konstruktionen, d. h. bewiesene Konstruktionsaufgaben.

2. Aus Gründen der Rechenzeit werden hier pro Konstruktion zwei Bestimmungsstücke erzwungen, damit die Anzahl der möglichen Kombinationen (vgl. Abschnitt 2) nicht zu groß wird.

Der Konstruktionsalgorithmus Bestimmungslinien realisiert die Konstruktionen außer Schneiden über Namen; das Schneiden erfolgt, indem zwei nichtidentische Bestimmungslinien angegeben werden, die mit den zu konstruierenden Punkten inzidieren.

Der Speicher enthält alle möglichen Konstruktionsschritte in Form von Bestimmungslinien für Punkte (z. B. Punkt A inzidiert mit Kreis um Punkt B mit Radius R), er wird mit Liste LB bezeichnet.

Alle gegebenen und konstruierten Elemente von M werden in eine Liste (L) eingetragen. Für die Paare aus L wird geprüft, ob sie Bestimmungsstücke einer Bestimmungslinie aus LB sind. Ist eine konstruierbare Bestimmungslinie erste Bestimmungslinie für einen Punkt P, so wird sie in eine Arbeitsliste (AL) eingetragen, anderenfalls

[1]) Dieser Algorithmus wurde von der Verfasserin in der Diplomarbeit [5] erarbeitet.

[2]) Zur Problematik des Ergebnisses beim Schneiden von Kreisen mit Kreisen oder Geraden siehe auch [6].

enthält AL eine weitere Bestimmungslinie für P, P ist somit konstruiert, und L kann um P erweitert werden. Bei der Realisierung ergeben sich folgende Probleme:

— Sind alle Elemente von $\mathfrak{x}$ Maßvariable, so sind stets zwei Punkte zusätzlich als gegeben zu betrachten, deren Abstand eine dieser Längen ist. Der Konstruktionsalgorithmus führt diesen „Konstruktionsbeginn" mittels einer Liste (HL 1) aus, die den Strecken die Endpunkte zuordnet.

— Über LB werden nur Punkte konstruiert; wegen der Beschränkung auf zwei Bestimmungsstücke sind aber nicht alle Konstruktionen auf Punkte bezogen. Es ist nötig, mit zwei konstruierten Punkten z. B. die Gerade durch diese Punkte (falls sie zu M gehört) als konstruiert zu betrachten. Zuordnungen dieser Art werden über Hilfsliste 2 (HL 2) vorgenommen, die so konstruierten Elemente in L eingetragen.

— Zwei Bestimmungslinien für Punkte können identisch sein, obwohl sie unterschiedlichen Konstruktionsschritten entsprechen. In Hilfsliste 3 (HL 3) werden alle Gruppen identischer Bestimmungslinien erfaßt.

4.2. Ein Konstruktionsalgorithmus für Dreieckskonstruktionen

Der in 4.1. beschriebene Algorithmus soll auf die Aufgabenklasse spezieller Dreieckskonstruktionen angewandt werden (vgl. auch [2]).
Es ist die Konstruierbarkeit eines Dreiecks aus drei Stücken zu entscheiden und der Lösungsalgorithmus anzugeben. Als Elemente von $\mathfrak{x}$ sind die Längen der Dreiecksseiten, der Seitenhalbierenden, der Höhen und die Größen der Winkel zugelassen. Das Dreieck gilt als konstruiert, wenn die Eckpunkte A, B, C $(= \mathfrak{y})$ konstruiert wurden. Bei Abarbeitung des Konstruktionsalgorithmus werden willkürlich die zwei Punkte, die Endpunkte der zuerst eingegebenen Strecke sind, als der Lage nach gegeben gewählt (vgl. 4.1.). Längen von Winkelhalbierenden sind als Elemente von M nicht zugelassen, es werden nur Lösungsalgorithmen angegeben, die ohne Anwendung von Ähnlichkeit gefunden werden. Die Menge M enthält folgende Elemente:

$\mathfrak{x}$	Länge der Seiten	RA	RB	RC
	Länge der Seitenhalbierenden	RSA	RSB	RSC
	Länge der Höhen	RHA	RHB	RHC
	Winkelgrößen	WA	WB	WC
$\mathfrak{y}$	Eckpunkte des Dreiecks	PA	PB	PC
Hilfs-	Mittelpunkte der Seiten	PSA	PSB	PSC
elemente	Höhenfußpunkte	PHA	PHB	PHC
	Schnittpunkt der Seitenhalbierenden	PS		
	weitere Hilfspunkte	$PD1$	$PD2$	$PD3$
		$PG1$	$PG2 \ldots$	$PG6$
		$PT1$	$PT2$	$PT3$
	Verlängerung der Seiten	GA	GB	GC
	Verlängerung der Höhen	GHA	GHB	GHC
	orientierte Strecken	SA	SB	SC
		SSA	SSB	SSC
		SHA	SHB	SHC
		$SD1$	$SD2$	$SD3$
		$SS1$	$SS2$	$SS3$

Bemerkung. Über den Aufbau von *LB* werden mittels Kennzeichen 1/2 Seitenlänge, 1/3 und 2/3 Länge der Seitenhalbierenden, Ergänzungswinkel, negative Winkel und Richtungsangabe der orientierten Strecken realisiert.

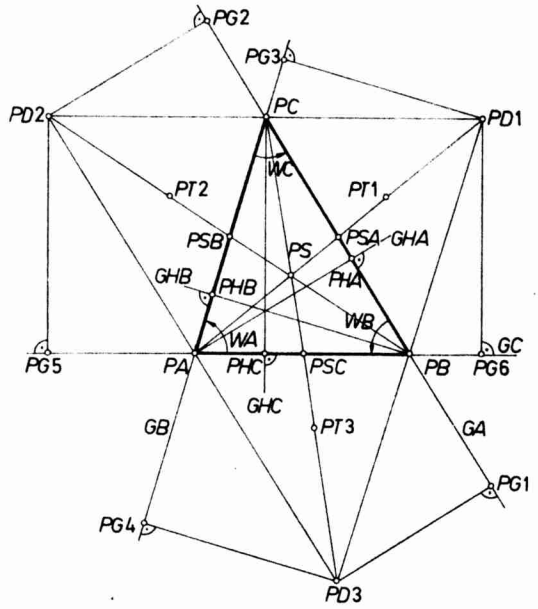

Abb. 2

Die Verhältnisse der Elemente von *M* zeigt Abb. 2, die „Zerlegung" dieser Analysefigur ergibt die Liste *LB*. Die Bezeichnung der Konstruktionen entspricht der in 4.1. angegebenen Variante mit Ausnahme von $\varkappa_8$. Auf Grund der speziellen Struktur der Aufgabenklasse kann die Konstruktion mit „*WIN GXX WXX*" beschrieben werden. Die verwendeten Listen sind Zeichenkettenfelder folgender Struktur:

LB ... | *PXX* | *KON* | *KZ* | *BS*1 | *KZ* | *BS*2 |

KON ... Name der Konstruktion
KZ ... Kennzeichen
BS ... Bestimmungsstück

Beispiel. *PSC GER_ PS__ PT*3
Punkt *SC* liegt auf der Geraden durch Punkt *S* und Punkt *T*3
PS_ KRS 4 *RSA_ PA_*
Punkt *S* liegt auf dem Kreis um Punkt *A* mit dem Radius 2/3 · Länge der Seitenhalbierenden s_a

*HL*1 ... | *RXX* | *PXX* | *PXX* |

Beispiel. *RSA PA_ PSA*
Endpunkte der Seitenhalbierenden s_a sind die Punkte *A* und *SA*

*HL*2 ... | *PXX* | *PXX* | { *GXX* / *RXX* / *SXX* |

Beispiel. *PA_ PB_ GC_*
Gerade g_c ist die Verbindungsgerade der Punkte A und B

HL3, HL3I

Z	KON	KZ	BS1	KZ	BS2

Z ... zugeordnete Zahl der Liste *HL3I*

Beispiel. 1 *GER_ PA_ _ PB_*
Gerade durch Punkt A und Punkt B ist mit allen mit 1 gekennzeichneten Bestimmungslinien identisch

Das Prinzip des Konstruktionsalgorithmus zeigt Abb. 3.

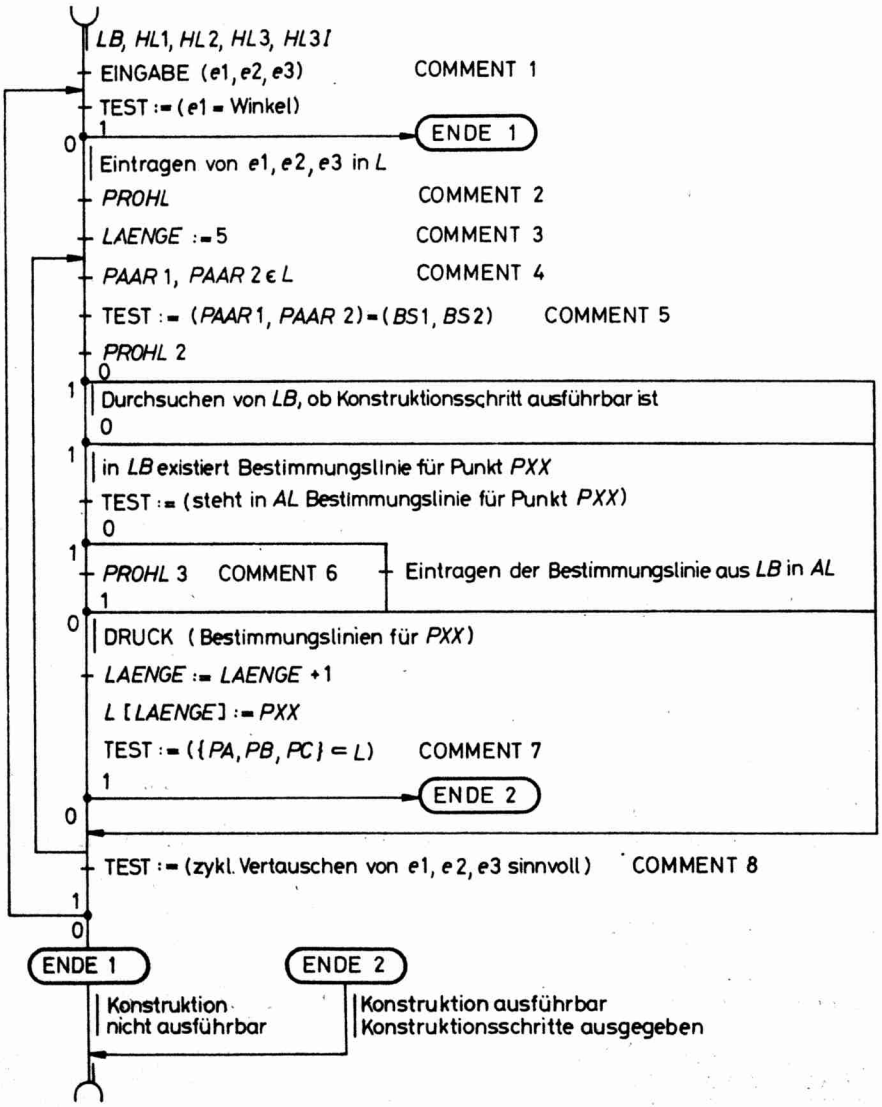

Abb. 3

COMMENT 1: Eingabe in alphabetischer Reihenfolge.
2: Konstruktionsbeginn, Zuordnung der Endpunkte zu *e*1.
3: *LAENGE* ... Elementeanzahl von *L*.
4: Kombinationen von zwei Elementen aus *L*
5: Test, ob (*PAAR*1, *PAAR*2) als Bestimmungsstücke für Konstruktionen in Frage kommen.
6: Test, ob beide Bestimmungslinien identisch sind
1 ... identisch, 0 ... nicht identisch
7: Dieser Test wird durch Zählen der konstruierten Elemente von $\mathfrak{y} = \{PA, PB, PC\}$ ausgeführt.
8: Wahl eines neuen Konstruktionsbeginns durch zyklisches Vertauschen von *e*1, *e*2, *e*3.

Bemerkung 1. Prozeduren, die sich der Hilfsliste *HLX* bedienen, werden mit *PROHLX* bezeichnet.

2. In das Flußbild wurde das Sperren der Bestimmungslinien für konstruierte Punkte (vgl. 4.1.) nicht aufgenommen.

LITERATUR

[1] Baatz, U.: Bildschirmunterstütztes Konstruieren. VDI-Verlag, Düsseldorf 1973.
[2] Ehlich, H., und E. Pittnauer: Mechanisierung von Dreieckskonstruktionen. Computing *5* (1970), 128–135.
[3] Klix, F., u. a. (Hrsg.): Analyse und Synthese von Problemlösungsprozessen. Akademie-Verlag, Berlin 1972.
[4] Krause, W., und B. Krause: Suchraumeinschränkung und -erweiterung beim Finden konstruktiver Lösungen mittels Automaten. Wiss. Z. TH Ilmenau (XIV. Int. Wiss. Koll.) (1969).
[5] Rudolf, S.: Algorithmen für geometrische Konstruktionen in der Ebene. Diplomarbeit TU Dresden, Sektion Mathematik, 1973.
[6] Schreiber, P.: Konstruktive Teiltheorien. Z. math. Logik Grundl. Math. *17* (1971), 197–204.
[7] Schreiber, P.: Theorie der geometrischen Konstruktionen. VEB Deutscher Verlag der Wissenschaften, Berlin 1974.
[8] Thiele, H.: Über einen sequentiellen Prädikatenkalkül als Grundlage für den Aufbau problemorientierter algorithmischer Sprachen. Wiss. Z. TU Dresden *17* (1968), 1122–1126.

Manuskripteingang: 28. 8. 1975

VERFASSER:

Susanne Harms und Wolf-Dieter Klix, Sektion Mathematik der Technischen Universität Dresden, Bereich Mathematische Kybernetik und Rechentechnik