Zeitschrift: Elemente der Mathematik
Herausgeber: Schweizerische Mathematische Gesellschaft
Band: 44 (1989)
Heft: 6

Artikel: Kombinatorik mit dem Computer : Klassifikationen und damit verwandte Figuren
Autor: Jeger, M.
DOI: https://doi.org/10.5169/seals-41625

Didaktik und Elementarmathematik

Kombinatorik mit dem Computer: Klassifikationen und damit verwandte Figuren

1. Einleitung

Die *Partitionen* oder *Klassen-Zerlegungen von endlichen Mengen* werfen zahlreiche interessante Fragen kombinatorischer Art auf und es liegt in unserer Zeit auf der Hand, auch beim Studium dieses klassischen Problemfeldes den Computer heran zu ziehen. Im Vergleich mit der gängigen Behandlungsweise ergeben sich dabei einige deutliche Akzentverschiebungen. So rücken jetzt solche Lösungsansätze in den Vordergrund, die sich leicht in Algorithmen umsetzen lassen. Andererseits können gewisse Fragen überhaupt erst unter Zuhilfenahme eines Computers in einer akzeptablen Frist beantwortet werden.

Im Hinblick auf die wachsende Bedeutung der finiten Mathematik (im angelsächsischen Raum Combinatorial Mathematics genannt) ist es angezeigt, auch im Mathematik-Unterricht vermehrt kombinatorische Aktivitäten einzubringen. Dabei ergeben sich zugleich bemerkenswerte Möglichkeiten zur Pflege des algorithmischen Denkens und zum unterrichtsbegleitenden Einsatz eines Rechners. In Anlehnung an einen früheren Aufsatz mit ähnlichen Zielsetzungen [7] sollen anschliessend einige Überlegungen zur *Kombinatorik der Klassen-Zerlegungen* vorgestellt werden.

$\boldsymbol{N}$ bezeichnet in dieser Note die Menge der natürlichen Zahlen. Ferner sei $N_n = \{1, 2, \ldots, n\}$ die Menge, bestehend aus den ersten n natürlichen Zahlen.

Generell heisst jedes aus einer Menge $\boldsymbol{U}$ hervorgehende System $\Gamma(\boldsymbol{U})$ von nicht-leeren Teilmengen eine Partition oder Klassen-Zerlegung von $\boldsymbol{U}$, wenn jedes Element von $\boldsymbol{U}$ zu genau einer Teilmenge in $\Gamma(\boldsymbol{U})$ gehört.

Die Kombinatorik befasst sich ausschliesslich mit Systemen $\Gamma(\boldsymbol{U})$, die aus einer endlichen Menge $\boldsymbol{U}$ abgeleitet sind; in diesem Falle enthält auch jedes $\Gamma(\boldsymbol{U})$ nur endlich viele Elemente. Ist insbesondere $|\boldsymbol{U}| = n$ und $|\Gamma(\boldsymbol{U})| = k$, dann heisst $\Gamma(\boldsymbol{U})$ eine *k-Partition der n-Menge $\boldsymbol{U}$* beziehungsweise eine *Klassen-Zerlegung der n-Menge $\boldsymbol{U}$ mit genau k Klassen.* Da aus einer n-Menge höchstens n nicht-leere, paarweise disjunkte Teilmengen ausgeschieden werden können, gilt

$$1 \leq k \leq n\,.$$

In der Rolle kombinatorischer Figuren werden k-Partitionen einer n-Menge in der Literatur gelegentlich auch als *Klassifikationen vom Index* (n, k) bezeichnet.

Schliesslich sei noch der Hinweis angebracht, dass jede Partition einer Menge $\boldsymbol{U}$ eine *Äquivalenz-Relation auf $\boldsymbol{U}$* charakterisiert.

2. Abzählung der Klassifikationen

Im folgenden bezeichnet $S(n, k)$ die *Anzahl der Klassifikationen bei einer n-Menge mit genau k Klassen.* Die Zahlen $S(n, k)$ – man nennt sie *Stirling-Zahlen* 2. Art [1*] – prägen

ein recht umfangreiches Gebiet der abzählenden Kombinatorik. Dies hängt damit zusammen, dass die jeweils massgebenden kombinatorischen Figuren bei vielen Fragestellungen mit Klassifikationen in Verbindung gebracht werden können.
Bei der Berechnung der Stirling-Zahlen stützt man sich mit Vorteil auf die Rekursionsformel

$$S(n,k) = S(n-1,k-1) + k\,S(n-1,k) \quad \text{für } n,k>1, \tag{2,1}$$

die mit folgenden Überlegungen direkt einsichtig gemacht werden kann. Auf der linken Seite von (2.1) steht etwa die Anzahl Klassen-Zerlegungen der n-Menge N_n mit genau k Klassen. Diese Figuren setzen sich zusammen aus

a) den Klassen-Zerlegungen von N_{n-1} mit genau $k-1$ Klassen, denen jeweils noch $\{n\}$ als k-te Klasse hinzugefügt ist. Die Anzahl solcher Figuren ist $S(n-1,k-1)$.
b) den Klassen-Zerlegungen von N_{n-1} mit genau k Klassen, bei denen eine der vorhandenen Klassen um das Element n erweitert ist. Für die Unterbringung des zusätzlichen Elementes n gibt es jeweils k Möglichkeiten; die Anzahl derartiger Figuren beträgt daher $k\,S(n-1,k)$.

Dieser kurze und elementare Beweis der Rekursionsformel (2,1) ist der neueren Kombinatorik-Literatur entnommen (vgl. etwa [2], p. 209; [4], p. 57; [5], p. 253).
Die Berechnung der Stirling-Zahlen aufgrund von (2,1) setzt noch die Kenntnis der Randwerte

$$\begin{aligned} &S(n,1) = 1 \\ &S(n,k) = 0 \quad \text{für } k>n \end{aligned} \tag{2,2}$$

voraus, die unmittelbar aus der Definition von $S(n,k)$ hervorgehen.
Die Gesamtzahl der Klassifikationen einer n-Menge beträgt

$$B_n = \sum_{k=1}^{n} S(n,k)\,. \tag{2,3}$$

Die B_n werden als *Bell-Zahlen* [2*] bezeichnet.
Die beiden anschliessend in Form von Rechner-Programmen in COMAL formulierten Algorithmen liefern die Stirling-Zahlen 2. Art und die Bell-Zahlen für sämtliche Index-Werte $n \leq q$. Der Algorithmus in Fig. 1 a folgt direkt der Rekursionsformel (2,1). Die Fig. 1 b zeigt eine Modifikation, die mit wesentlich weniger Speicher-Platz auskommt. Es genügt nämlich, jeweils eine Zahlen-Zeile bis zu deren Ausdruck zu speichern. Dabei ist aber zu beachten, dass wegen der besonderen Struktur der Rekursionsformel (2,1) das Überschreiben einer Zahlen-Zeile von rechts her erfolgen muss (FOR $k := n$ TO 2 step -1 DO).
COMAL [3*] ist eine spezifisch auf den Unterricht zugeschnittene Programmier-Sprache, die also von ihrer primären Zielsetzung her gut in den Rahmen der vorliegenden Studie passt. Sie repräsentiert eine Art verlängertes BASIC, das an PASCAL und an LOGO heranführt: So ist sie einerseits ausgesprochen Prozedur-orientiert, erlaubt aber andererseits auch den direkten Zugang zur Bildschirm-Graphik, insbesondere zur sog Turtle-

```
INPUT q

DIM s(q,q)

FOR n:=1 TO q DO

  s(n,1):=1; b:=1

  IF n>1 THEN
    FOR k:=2 TO n DO
      s(n,k):=s(n-1,k-1)+k*s(n-1,k)
      b:=b+s(n,k)
    ENDFOR k
  ENDIF

  zeiledrucken

ENDFOR n

END
```

Figur 1 a.

```
INPUT q

DIM s(q)

s(1):=1
FOR n:=1 TO q DO

  b:=1
  IF n>1 THEN
    FOR k:=n TO 2 STEP -1 DO
      s(k):=s(k-1)+k*s(k)
      b:=b+s(k)
    ENDFOR k
  ENDIF

  zeiledrucken

ENDFOR n

END
```

Figur 1 b.

```
Tabelle
der Stirling-Zahlen 2.Art S(n,k) und der Bell-Zahlen B(n)
```

Stirling-Zahlen										Bell-Zahlen
n/k	1	2	3	4	5	6	7	8	9	10	
1	1										1
2	1	1									2
3	1	3	1								5
4	1	7	6	1							15
5	1	15	25	10	1						52
6	1	31	90	65	15	1					203
7	1	63	301	350	140	21	1				877
8	1	127	966	1701	1050	266	28	1			4140
9	1	255	3025	7770	6951	2646	462	36	1		21147
10	1	511	9330	34105	42525	22827	5880	750	45	1	115975

Figur 2.

Graphik. Es soll hier die Gelegenheit benutzt werden, COMAL in dieser Zeitschrift in Verbindung mit einem konkreten Problemfeld vorzustellen. Ein weiterer Grund für die Ausrichtung dieser Note auf COMAL liegt bei der Tatsache, dass diese Programmier-Sprache weitgehend selbsterklärend ist; in COMAL notierte Algorithmen dürften auch solchen Lesern verständlich sein, die sich bislang nur am Rande mit Computern befasst haben. Zudem lassen sich COMAL-Programme leicht in andere Programmier-Sprachen übertragen.

3. Auflistung von Klassifikationen, Reim-Schemen

Zu den zentralen Aufgaben der Kombinatorik gehört neben dem Abzählen auch das *Aufzählen* oder *Auflisten von Figuren-Mengen.* So können z. B. kombinatorische Optimierungsprobleme meist nur über konkrete Auflistungen angegangen werden. Da für das Auflisten von Figuren-Mengen der Computer als adäquates Werkzeug erst seit geraumer

Zeit zur Verfügung steht, ist dieses Problemfeld fast ganz in den neueren Entwicklungen der Kombinatorik angesiedelt. Auflistungen werfen verschiedene interessante mathematische – vor allem algorithmische – Probleme auf, die anschliessend am Beispiel der Klassifikationen aufgezeigt und diskutiert werden sollen.

Zunächst ist bei einer Auflistung zu überlegen, wie man die massgebenden Figuren auf den Computer bringt. Im Falle der Klassifikationen treten bereits in dieser Abklärungsphase erste Schwierigkeiten auf; auch wenn man als Träger-Mengen die Zahlen-Mengen N_n heranzieht, ist keine Übertragung zu erkennen, die sich unmittelbar anbietet. Es liegt daher nahe, nach geeigneten Ersatz-Figuren Ausschau zu halten, d.h. nach Figuren, mit denen Klassifikationen Computer-gerecht beschrieben werden können. Dazu eignen sich die sog. *Reim-Schemen.*

Zur Erklärung der Reim-Schemen möge die 1. Strophe eines bekannten, von Schubert vertonten Goethe-Gedichtes dienen.

	Zeile	Reim-Nummer
Sah ein Knab ein Röslein steh'n	Z_1	1
Röslein auf der Heiden;	Z_2	2
War so jung und morgenschön,	Z_3	1
lief er schnell, es nah zu seh'n	Z_4	1
sah's mit vielen Freuden.	Z_5	2
Röslein, Röslein, Röslein rot,	Z_6	3
Röslein auf der Heiden.	Z_7	2

Numeriert man die vorkommenden Reime von oben her, dann lässt sich die *Reim-Struktur* dieses 7-Zeilers mit dem Reim-Schema

$$1211232 \qquad [4^*]$$

erfassen. Die Zeilen Z_1, Z_3, Z_4 bilden den ersten, die Zeilen Z_2, Z_5, Z_7 den zweiten und die Zeile Z_6 für sich den dritten Reim.

Die vorliegende Reim-Struktur kennzeichnet nun eine bestimmte Klassen-Zerlegung der Zeilen-Menge $\{Z_1, Z_2, Z_3, Z_4, Z_5, Z_6, Z_7\}$, die dadurch zustande kommt, dass man Zeilen mit derselben Reim-Nummer je zu einer Klasse zusammenfasst:

$$1211232 \mapsto \{\{Z_1, Z_3, Z_4\}, \{Z_2, Z_5, Z_7\}, \{Z_6\}\}\,.$$

Generell impliziert jedes n-stellige Reim Schema über dem Alphabet $1, 2, \ldots, n$ eine Relation «gleicher Reim» auf der entsprechenden Zeilen-Menge

$$\{Z_1, Z_2, \ldots, Z_n\}\,.$$

Diese Relation ist reflexiv, symmetrisch und transitiv, also klassenbildend. Umgekehrt lässt sich aus jeder Klassen-Zerlegung der n-Menge $\{Z_1, Z_2, \ldots, Z_n\}$ ein n-stelliges Reim-Schema konstruieren: Der Reim 1 gehört zur Klasse, die das Element Z_1 enthält, der Reim 2 zur Klasse in der das Element Z_i mit dem nächstgrösseren, noch nicht erfassten Index i enthalten ist, und so fort. Damit ist eine Zuordnung nachgewiesen, die eine bijektive Abbildung α_n der Menge aller n-stelligen Reim-Schemen auf die Menge aller Klassen-Zerlegungen der n-Menge $\{Z_1, Z_2, \ldots, Z_n\}$ darstellt. Die beiden genannten Fi-

guren-Mengen sind daher gleich mächtig. Dieser Zusammenhang zwischen Reim-Schemen und Klassifikationen ist bekannt (vgl. etwa [3] und [10]), die Abbildung α_n scheint aber bislang bei der Auflistung der Klassifikationen einer n-Menge (Klassifikationen vom Index n) kaum verwendet worden zu sein.
Wir fügen jetzt diesen Überlegungen noch eine Anpassung an den Rechner hinzu, indem wir das Element Z_i durch i ersetzen. Die n-stelligen Reim-Schemen über dem Alphabet $1, 2, \ldots, n$ werden dann durch α_n auf die Menge der Klassifikationen bei der Standard n-Menge N_n bezogen. Die folgenden Beispiele zeigen einen Ausschnitt aus der Abbildung α_9:

$$111222333 \mapsto \{\{1, 2, 3\}, \{4, 5, 6\}, \{7, 8, 9\}\}$$
$$123243131 \mapsto \{\{1, 7, 9\}, \{2, 4\}, \{3, 6, 8\}, \{5\}\}$$
$$122341523 \mapsto \{\{1, 6\}, \{2, 3, 8\}, \{4, 9\}, \{5\}, \{7\}\}$$

Ein n-stelliges Wort $f_1 f_2 \ldots f_n$ über dem Alphabet $1, 2, \ldots, n$ ist genau dann ein Reim-Schema, wenn

$$\begin{aligned} &f_1 = 1 \\ &f_{i+1} \leqq \max(f_1, f_2, \ldots, f_i) + 1 \quad \text{für} \quad 1 \leqq i \leqq n-1 \end{aligned} \tag{3,1}$$

ist.
Es soll nun zunächst ein Algorithmus zur Auflistung der n-stelligen Reim-Schemen entwickelt werden. Durch den späteren Einbau einer Prozedur, welche die Abbildung α_n realisiert, kann man dann hieraus zu einem Algorithmus für die Auflistung der Klassifikationen bei der Menge N_n gelangen.
Die Auflistung einer endlichen Figuren-Menge basiert immer auf einer wohldefinierten Anordnung ihrer Elemente. Das Kern-Problem bei einem Auflist-Algorithmus besteht generell immer darin, für eine bestimmte Figur in der Liste deren Nachfolger zu konstruieren. Bei der Festlegung der Reihenfolge ist daher darauf zu achten, dass die entsprechende Nachfolger-Konstruktion algorithmisch einigermassen praktikabel ist.
In vielen Fällen ist mit der aufzulistenden Figuren-Menge bereits auch eine bestimmte Reihenfolge vorgezeichnet. Dies trifft z. B. bei allen Wörter-Mengen über einem Zahlen- oder Buchstaben-Alphabet zu. Durch die Ordnung im betreffenden Alphabet ist bei solchen Figuren-Mengen die sogenannte *lexikographische Anordnung* gewissermassen vorgespurt. Wie sich gleich zeigen wird, ist diese Art der Aufreihung bei den Reim-Schemen auch in algorithmischer Hinsicht ausgesprochen gutartig.
Zum Auflisten der n-stelligen Reim-Schemen über dem Alphabet $1, 2, \ldots, n$ fassen wir nun eine einzelne Figur $f_1 f_2 \ldots f_n$ als Zahlwort mit der Stellenbelegung $f_1, f_2, \ldots, f_n$ auf. Die Nachfolger-Figur $\bar{f}_1 \bar{f}_2 \ldots \bar{f}_n$ bei lexikographischer Anordnung kann dann durch eine Art Zähl-Prozess erhalten werden: Man erhöhe die letzte Stelle um eine Einheit und nehme dies zum Anlass eines stellenweisen zyklischen Zählens mit Überträgen auf die vorangehenden Stellen; die jeweilige Zyklen-Länge in der i-ten Stelle ist durch die Ungleichung

$$f_i \leqq \max(f_1, f_2, \ldots, f_{i-1}) + 1$$

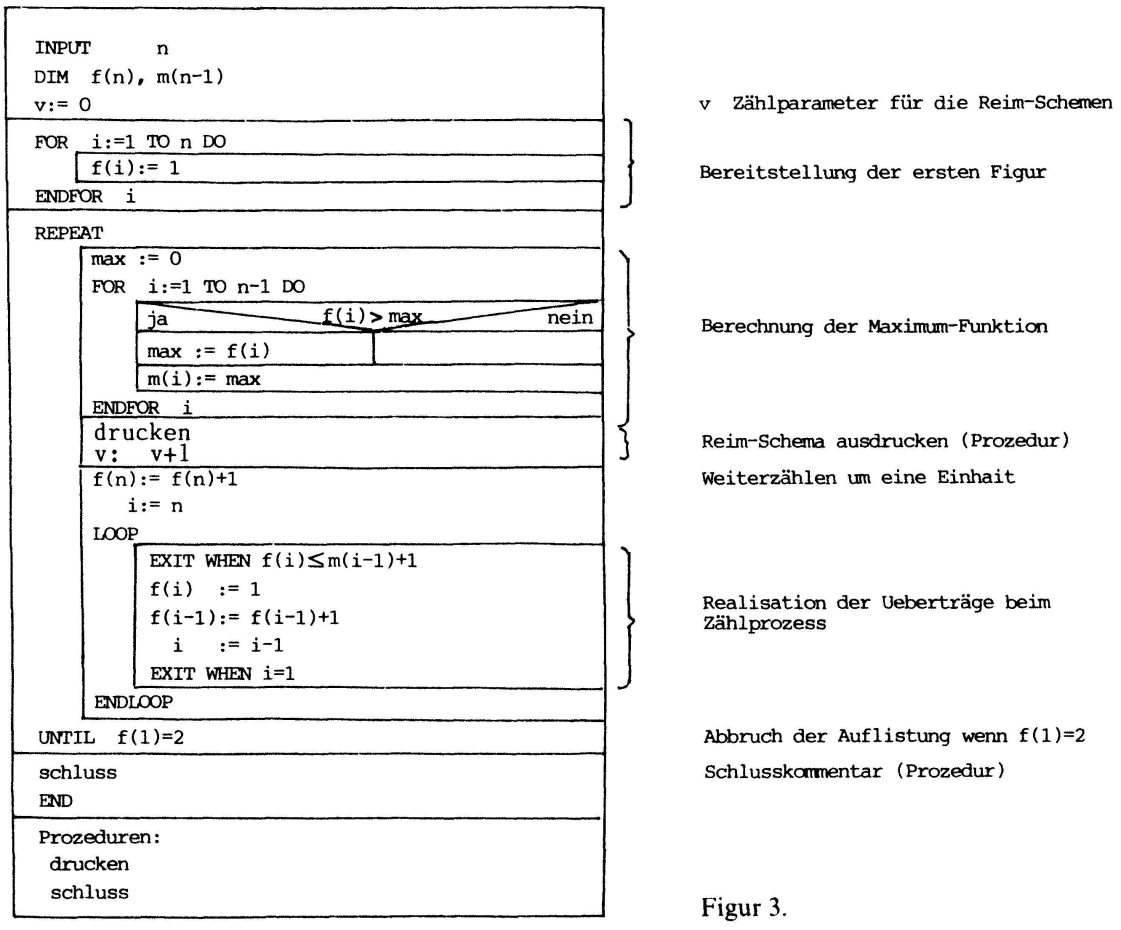
bestimmt. Dieses Vorgehen setzt zunächst einmal voraus, dass die Reim-Schemen mit den Parametern $f_1, f_2, \ldots, f_n$ beschrieben werden; zu Beginn ist also mit der Anweisung

DIM $f(n)$

ein entsprechendes Register-Feld bereitzustellen. Ferner ist für jedes feststehende Reim-Schema die Maximum-Funktion

$$m(i) = \max(f_1, f_2, \ldots, f_i); \quad i = 1, 2, \ldots, n-1$$

zu bestimmen, die bei der Nachfolger-Konstruktion benötigt wird. Da die Maximum-Funktion mit jeder neuen Figur ändert, kann von einem dynamischen Zählprozess gesprochen werden.
Start-Figur bei der Auflistung ist das n-stellige Reim-Schema 1 1 1 ... 1. Andererseits ist 1 2 3 ... n die letzte Figur in der Liste; sie ist etwa daran zu erkennen, dass beim Weiterzählen um eine Einheit erstmals ein Übertrag auf die erste Stelle anfällt. Damit steht in $f(1) = 2$ ein einfaches *Abbruch-Kriterium* zur Verfügung.
Nach dieser verbalen Umschreibung sei der Auflist-Algorithmus anschliessend auch noch in der präziseren Form eines Struktogrammes notiert, das auf die Programmier-Sprache COMAL abgestimmt ist.



Figur 3.

Die Anweisung EXIT innerhalb einer LOOP-Schleife bewirkt einen Sprung in der Verarbeitung auf die unmittelbar an ENDLOOP anschliessende Zeile (Ausstieg aus einer LOOP-Schleife).
Die Prozedur «drucken» kann so ausgelegt werden, dass für $n=5$ der folgende Output entsteht.

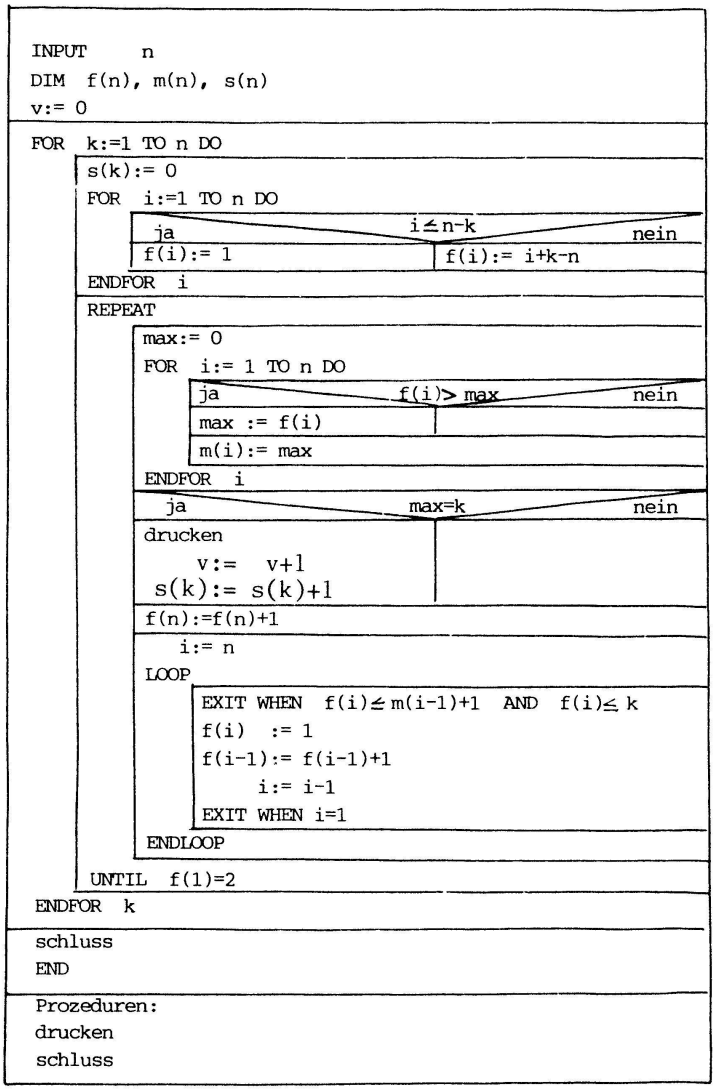
```
Auflistung der Reim-Schemen
vom Index n

n=5

Figuren:

 11111 11112 11121 11122 11123 11211 11212 11213 11221 11222
 11223 11231 11232 11233 11234 12111 12112 12113 12121 12122
 12123 12131 12132 12133 12134 12211 12212 12213 12221 12222
 12223 12231 12232 12233 12234 12311 12312 12313 12314 12321
 12322 12323 12324 12331 12332 12333 12334 12341 12342 12343
 12344 12345

Anzahl der Figuren: 52

Rechenzeit: 6 sk
```

Figur 4.

Es sind auch Algorithmen zur lexikographischen Auflistung der n-stelligen Reim-Schemen mitgeteilt worden, denen ganz andere mathematische Überlegungen zugrunde liegen (vgl. etwa [6], p. 318–326; [9], p. 91).
Eine weitere, in den Anwendungen oft bevorzugte Auflistung der n-stelligen Reim-Schemen beruht auf einer Blockbildung nach zunehmender Reimzahl k, wobei innerhalb der einzelnen Blöcke lexikographisch aufgereiht wird. Ein Algorithmus, der dies leistet, kann leicht aus dem bereits vorliegenden Algorithmus erhalten werden, indem man die darin enthaltene REPEAT-Schleife n mal durchläuft und nacheinander die Reim-Schemen mit den Reimzahlen $1, 2, \ldots, k, \ldots, n$ aussiebt. Ohne besondere Vorkehren ist bei diesem Konzept – im Vergleich mit dem Primär-Algorithmus – etwa die n-fache Laufzeit zu erwarten. Es ist daher naheliegend, noch nach Modifikationen zu suchen, die den Prozessablauf beschleunigen. Eine Verkürzung der Laufzeit lässt sich einmal damit erreichen, dass man vor jedem Durchlauf der REPEAT-Schleife das erste Wort im betreffenden Block konstruiert, also nicht jedesmal wieder mit dem Reim-Schema $1\,1\,1 \ldots 1$ startet. Andererseits kann man ebenso problemlos in den Algorithmus einfliessen lassen, dass die Stellenwerte bei den Wörtern des k-ten Blocks die Zahl k nicht überschreiten, also die Zählzyklen dort höchstens die Länge k aufweisen. Dies bringt zusätzlich eine Verkürzung der Rechenzeit.
Es sei noch darauf hingewiesen, dass zur Aussiebung der Reim-Schemen mit der Reimzahl k eine Berechnung der Maximum-Funktion über die ganze Wortlänge erforderlich ist:

$$m(i) = \max(f_1, f_2, \ldots, f_i); \quad i = 1, 2, \ldots, n\,.$$

Auch dieser zweite Auflist-Algorithmus sei anschliessend noch in der Gestalt eines Struktogrammes formuliert, wobei wir uns wiederum auf den eigentlichen mathematischen Kern beschränken (d. h. Legenden und Druck-Anweisungen sind weggelassen).



Figur 5.

Drucker-Output für $n=5$

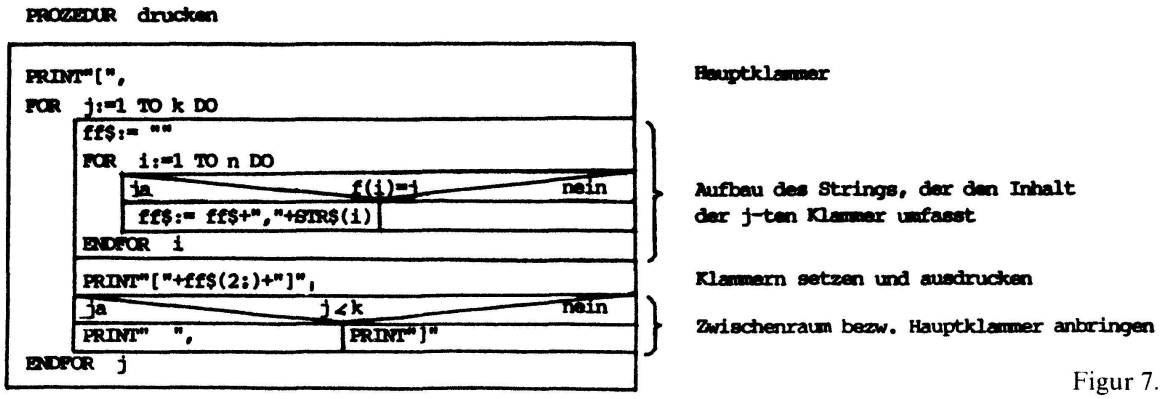
```
Auflistung der Reim-Schemen
vom Index n

n=5

Figuren:

 11111

 11112 11121 11122 11211 11212 11221 11222 12111 12112 12121
 12122 12211 12212 12221 12222

 11123 11213 11223 11231 11232 11233 12113 12123 12131 12132
 12133 12213 12223 12231 12232 12233 12311 12312 12313 12321
 12322 12323 12331 12332 12333

 11234 12134 12234 12314 12324 12334 12341 12342 12343 12344

 12345

Anzahl der Figuren:

1+15+25+10+1=52

Rechenzeit: 9 sk
```

Figur 6.

Ein Vergleich der Rechenzeiten in den beiden Drucker-Outputs (Fig. 4 und Fig. 6) zeigt, dass die vorgenommenen Ergänzungen zur Verkürzung der Laufzeit tatsächlich greifen [5*].

Sollen jetzt anstelle der n-stelligen Reim-Schemen über dem Alphabet $1, 2, \ldots, n$ die entsprechenden Klassifikationen der n-Menge N_n ausgegeben werden, dann ist noch die Abbildung α_n in die Druck-Prozedur einzubauen. Zur Darstellung von Klassifikationen mit dem Computer benutzen wir Klammer-Figuren, wie etwa

$$[[1, 4, 7], [2, 10], [3], \ldots, [\ldots]] \qquad [6^*].$$



Figur 7.

```
Auflistung der Klassifikationen einer
n-Menge

n=5

Figuren:

[[1,2,3,4,5]]

[[1,2,3,4] [5]]   [[1,2,3,5] [4]]   [[1,2,3] [4,5]]   [[1,2,4,5] [3]]
[[1,2,4] [3,5]]   [[1,2,5] [3,4]]   [[1,2] [3,4,5]]   [[1,3,4,5] [2]]
[[1,3,4] [2,5]]   [[1,3,5] [2,4]]   [[1,3] [2,4,5]]   [[1,4,5] [2,3]]
[[1,4] [2,3,5]]   [[1,5] [2,3,4]]   [[1] [2,3,4,5]]

[[1,2,3] [4] [5]]   [[1,2,4] [3] [5]]   [[1,2] [3,4] [5]]   [[1,2,5] [3] [4]]
[[1,2] [3,5] [4]]   [[1,2] [3] [4,5]]   [[1,3,4] [2] [5]]   [[1,3] [2,4] [5]]
[[1,3,5] [2] [4]]   [[1,3] [2,5] [4]]   [[1,3] [2] [4,5]]   [[1,4] [2,3] [5]]
[[1] [2,3,4] [5]]   [[1,5] [2,3] [4]]   [[1] [2,3,5] [4]]   [[1] [2,3] [4,5]]
[[1,4,5] [2] [3]]   [[1,4] [2,5] [3]]   [[1,4] [2] [3,5]]   [[1,5] [2,4] [3]]
[[1] [2,4,5] [3]]   [[1] [2,4] [3,5]]   [[1,5] [2] [3,4]]   [[1] [2,5] [3,4]]
[[1] [2] [3,4,5]]

[[1,2] [3] [4] [5]]   [[1,3] [2] [4] [5]]   [[1] [2,3] [4] [5]]   [[1,4] [2] [3] [5]]
[[1] [2,4] [3] [5]]   [[1] [2] [3,4] [5]]   [[1,5] [2] [3] [4]]   [[1] [2,5] [3] [4]]
[[1] [2] [3,5] [4]]   [[1] [2] [3] [4,5]]

[[1] [2] [3] [4] [5]]


Anzahl der Figuren:

1+15+25+10+1=52

Rechenzeit: 13 sk
```

Figur 8.

Zum Aufbau derartiger Zeichenreihen ist eine sog. String-Variable erforderlich, die im Struktogramm (Fig. 7) mit ff$ bezeichnet ist.
In einem Reim-Schema des k-ten Blocks kommen die Reim-Nummern $1, 2, \ldots, j, \ldots, k$ vor. Zu jeder dieser Nummern gehört eine bestimmte Klasse (Klammer) in der betreffenden Klassen-Zerlegung (Klammer-Figur). Die Elemente in der j-ten Klammer gehen aus den Nummern jener Plätze im ursprünglichen Reim-Schema hervor, die mit der Zahl j belegt sind.
Der Drucker-Output der Figur 8 zeigt die Umsetzung der 5stelligen Reim-Schemen in Klassen-Zerlegungen der Menge N_5 (die Formatierung mit 4 Figuren pro Zeile ist im Struktogramm weggelassen).

4. Systeme von k nicht-leeren, paarweise disjunkten Teilmengen aus N_n

Solche Mengen-Systeme hängen eng zusammen mit den Klassen-Zerlegungen der $(n+1)$-Menge N_{n+1}, die genau $k+1$ Klassen umfassen. Lässt man nämlich bei diesen Klassen-Zerlegungen jeweils die Klasse weg, die das Element $n+1$ enthält, dann bleiben gerade die Systeme von k nicht-leeren, paarweise disjunkten Teilmengen aus N_n stehen. Es gibt daher genau $S(n+1, k+1)$ Mengen-Systeme dieser Art (vgl. [8]).

```
PROC drucken
  PRINT "[",
  h:=0
  FOR j:=1 TO k DO
    IF j<>f(n) THEN
      ff$:=""; h:=h+1
      FOR i:=1 TO n-1 DO
        IF f(i)=j THEN
          ff$:=ff$+","+STR$(i)
        ENDIF
      ENDFOR i
      PRINT "["+ff$(2:)+"]",
      IF h<k-1 THEN PRINT " ",
    ENDIF
  ENDFOR j
  PRINT "]"
ENDPROC drucken
```

```
Auflistung der Systeme von k
nicht-leeren, paarweise disjunkten
Teilmengen einer Menge vom Umfang n

n=4
k=2

Figuren:

[[1,2,3] [4]]
[[1,2,4] [3]]
[[1,2] [3,4]]
[[3] [4]]
[[1,2] [4]]
[[1,2] [3]]
[[1,3,4] [2]]
[[1,3] [2,4]]
[[2] [4]]
[[1,3] [4]]
[[1,3] [2]]
[[1,4] [2,3]]
[[1] [2,3,4]]
[[2,3] [4]]
[[1] [4]]
[[1] [2,3]]
[[2] [3]]
[[1,4] [3]]
[[1,4] [2]]
[[2,4] [3]]
[[1] [3]]
[[1] [2,4]]
[[2] [3,4]]
[[1] [3,4]]
[[1] [2]]

Anzahl der Figuren: 25

Rechenzeit: 5 sk
```

Figur 9.

Unser Auflist-Algorithmus für die Klassen-Zerlegungen von N_{n+1} kann leicht so abgeändert werden, dass die zur Diskussion stehenden Teilmengen-Systeme von N_{n+1} im Output erscheinen. Zunächst fällt die k-Schleife dahin. Nach der Eingabe der Parameter n und k sind die Ersetzungen

$$n := n+1$$
$$k := k+1$$

vorzunehmen. Anschliessend ist dann die Druck-Prozedur noch so zu modifizieren, dass die Klasse, in der das Element n enthalten ist, nicht ausgedruckt wird. In der Prozedur gemäss Listing in Fig. 9 wird dies über den markierten IF THEN-ENDIF-Abschnitt realisiert.
h ist ein Zählparameter für die anfallenden Klassen; er wird für das Setzen der Zwischenräume beim Ausdrucken der einzelnen Figuren benötigt.

M. Jeger, ETH-Zürich

LITERATURVERZEICHNIS

1 Andrews G. E.: The theory of partitions. Encyclopedia of mathematics and its applications, vol. 2. London, Amsterdam, Don Mills, Sydney, Tokyo (1976).
2 Comtet L.: Advanced combinatorics. Dordrecht, Boston (1974).
3 Gardner M.: Bellsche Zahlen. Spektrum der Wissenschaft. *9*, 78–83 (1979).
4 Halder H. R., Heise W.: Einführung in die Kombinatorik. München, Wien (1976).
5 Jacobs K.: Einführung in die Kombinatorik. Berlin, New York (1983).
6 Jeger M.: Computer-Streifzüge. Basel, Boston, Stuttgart (1986).
7 Jeger M.: Kombinatorik mit dem Computer: Partitionen und Frankaturen. El. Math. *42*, 156–172 (1987).
8 Jeger M.: Aufgabe 995, El. Math. *43*, 158 (1988) und *44*, 140 (1989).
9 Nijenhuis A., Wilf H. S.: Combinatorial algorithms. New York, San Francisco, London (1978).
10 Rota C. G.: The number of partitions of a set. Amer. Math. Monthly *71*, 498–504 (1964).

ANMERKUNGEN

[1*] James Stirling (1692–1770).
[2*] Eric Temple Bell (1883–1960).
[3*] Abkürzung für *Com*mon *A*lgorithmic *L*anguage.
[4*] Wenn die Anzahl der Zeilen kleiner als 10 ist, können die Reim-Schemen problemlos als Wörter geschrieben werden. Andernfalls ist die Vektor-Notation angezeigt.
[5*] Die Laufzeiten beziehen sich auf den benutzten Rechner Commodore C 64.
[6*] Der verwendete Rechner lässt keine geschweiften Klammern zu.

 0013-6018/89/060154-11$1.50 + 0.20/0