Harald Nahrstedt

Das Turm-Weg-Problem (TI-59)

In Martin Gardners Buch „Mathematischer Karneval" wird ein Problem auf dem Schachbrett angedeutet, dem ich in abgewandelter Form den Namen „Turm-Weg-Problem" gegeben habe. Es lautet wie folgt:

> Ein Turm steht auf dem Königsquadrat eines Schachbrettes. Entsprechend seiner Bewegungsmöglichkeiten soll er den längsmöglichen Weg zurücklegen, ohne dabei eine Wegstrecke mehrmals zu benutzen.

Dieses Problem läßt sich (wie das 8-Damen-Problem in „Bild der Wissenschaft", Heft 10/78) auf eine Baumstruktur zurückführen, dessen Lösung man durch „Probieren und Überprüfen" erhalten kann. Die Bewegung des Turmes über mehrere Felder hinweg wird in das Vorrücken um einzelne Felder zerlegt. Damit ergeben sich für das Ausgangsfeld nach Fig. 1 drei Alternativen. Diese wiederum ergeben neue Alternativen, usw. Damit sind wir schon bei der Baumstruktur, wie sie Fig. 2 wiedergibt. Bleibt also nur noch die Frage zu klären: Nach welchem Algorithmus wird die Baumstruktur durchmustert?

Der Trick für das nachfolgende Verfahren heißt: „Immer rechts herum". Wir versuchen also immer unsere Bewegungsrichtung rechts herum forzusetzen. Erst wenn dies nicht möglich ist oder bereits praktiziert wurde, setzen wir unsere Bewegung mit fallender Priorität gerade, links herum oder rückwärts fort. Erreichen wir dabei die Gegenrichtung zu der Richtung, unter der wir unser Ausgangsfeld begonnen haben, so ist die Prozedur beendet. Auf diese Weise wird die gesamte Baumstruktur durchmustert, wie in Fig. 3 an einer einfachen Baumstruktur graphisch demonstriert ist. Das Prinzip „immer rechts herum" wird durch die gestrichelte Linie angezeigt. Alternativ wäre „immer links herum".

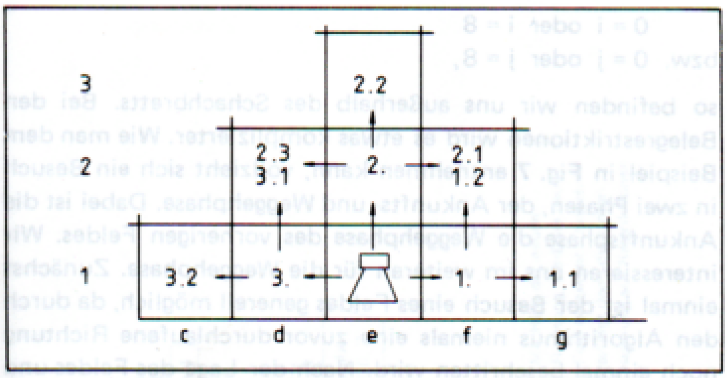


Fig. 1

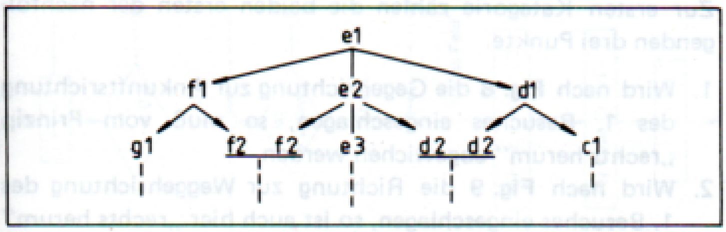


Fig. 2

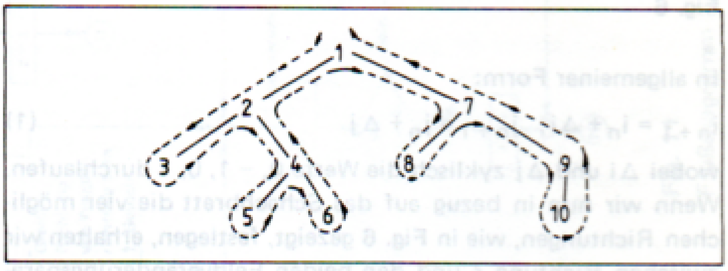


Fig. 3

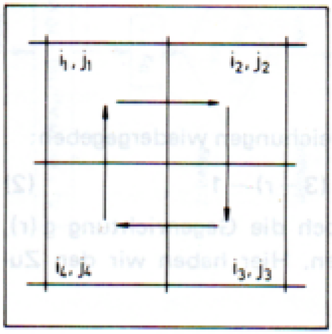


Fig. 4

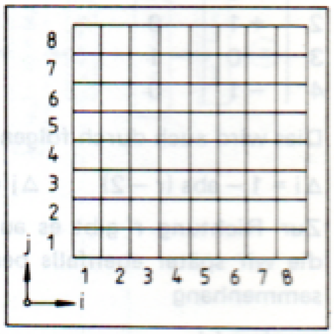


Fig. 5

Die so entworfene Idee müssen wir nun an der Hardware (sprich Schachbrett und programmierbarer Taschenrechner) konkretisieren. Zu diesem Zweck legen wir das Schachbrett in ein Koordinatensystem und erhalten so nach Fig. 4 die Feldparameter i (Zeilenwert) und j (Spaltenwert). Mit der Angabe der aktuellen Parameter (i, j) ist also der jeweilige Stand auf dem Schachbrett definiert. Das Prinzip „immer rechts herum" ist damit für ein Ausgangsfeld (i_1, j_1) durch die Veränderung der Parameter auf folgende Art, wie Fig. 5 zeigt, gekennzeichnet:

$i_2 = i_1 \quad ; j_2 = j_1 + 1$
$i_3 = i_2 - 1; j_3 = j_2$
$i_4 = i_3 \quad ; j_4 = j_3 - 1$
$i_1 = i_4 + 1; j_1 = j_4$

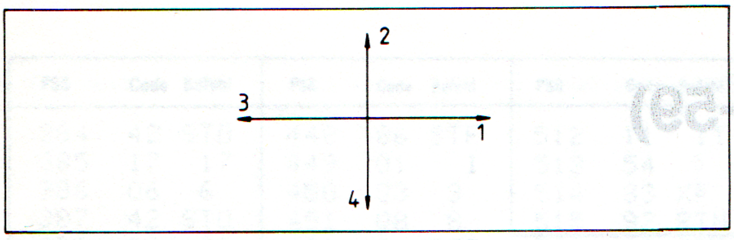


Fig. 6

In allgemeiner Form:

$i_{n+1} = i_n + \Delta i; \quad j_{n+1} = j_n + \Delta j$ (1)

wobei Δi und Δj zyklisch die Werte 0, −1, 0, 1 durchlaufen. Wenn wir nun in bezug auf das Schachbrett die vier möglichen Richtungen, wie in **Fig. 6** gezeigt, festlegen, erhalten wir zwischen Richtung r und den beiden Feldveränderungsparametern Δi und Δj, folgenden funktionalen Zusammenhang:

r	Δi	Δj
1	0	+1
2	+1	0
3	0	−1
4	−1	0

Dies wird auch durch folgende Gleichungen wiedergegeben:

$\Delta i = 1 - \mathrm{abs}(r-2) \qquad \Delta j = \mathrm{abs}(3-r) - 1$ (2)

Zur Richtung r gibt es auch noch die Gegenrichtung g (r), die wir später ebenfalls benötigen. Hier haben wir den Zusammenhang

r	g (r)
1	3
2	4
3	1
4	2

Diesen ergibt auch die Gleichung

$g(r) = r + \mathrm{sgn}(2.5 - r) \times 2$ (3)

Weiterhin müssen wir jeden Besuch eines Feldes codiert registrieren. Dazu ordnen wir jedem Schachbrettfeld ein Datenregister zu. Da diese fortlaufende Adressen besitzen, muß sich das entsprechende Register

$R(n) = R(i, j)$

aus den aktuellen Feldparametern bestimmen. Sei dazu a die niedrigste Adresse der Register. Dann ist eine mögliche Zuordnung

$R(i, j) = R(a + (j-1) + 8(i-1)) = R(a - 9 + j + 8i)$ (4)

Für i = j = 8 ergibt sich so

$R(8, 8) = R(a - 9 + 8 + 64) = R(a + 63)$

und damit die besagten 64 Felder des Schachbretts. Beim Besuch eines Feldes kann dieser codiert in dem so zugeordneten Register notiert werden. Dies geschieht von Ankunfts- und Weggehrichtung mit steigender Zehnerpotenz. Danach bedeutet z. B. der Inhalt eines Registers von

1 4 3 2

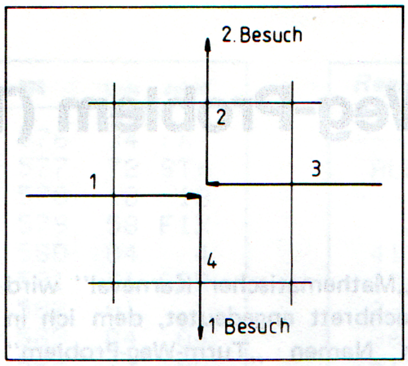


Fig. 7

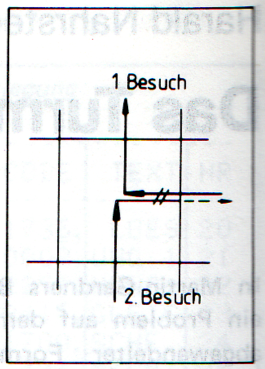


Fig. 8

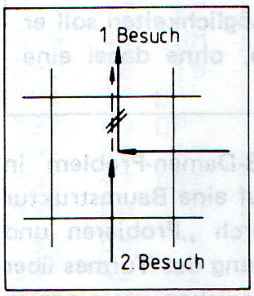


Fig. 9

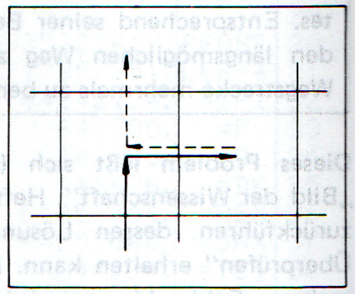


Fig. 10

die in **Fig. 7** wiedergegebene Bewegung. Dieses Beispiel macht darüber hinaus deutlich, daß ein Feld höchstens zweimal besucht werden kann. Bei 64 Schachbrettfeldern ergibt sich damit eine theoretische Lösung unseres Problems von 128 Feldern. Dies wird jedoch praktisch nicht möglich sein.

Kommen wir zum schwierigsten Teil des Problems, den Restriktionen. Diese unterteilen sich in die Feld- und Belegrestriktionen. Die ersteren sind leicht erklärt. Gilt

$0 = i$ oder $i = 8$

bzw. $0 = j$ oder $j = 8$,

so befinden wir uns außerhalb des Schachbretts. Bei den Belegrestriktionen wird es etwas komplizierter. Wie man dem Beispiel in **Fig. 7** entnehmen kann, vollzieht sich ein Besuch in zwei Phasen, der Ankunfts- und Weggehphase. Dabei ist die Ankunftsphase die Weggehphase des vorherigen Feldes. Wir interessieren uns im weiteren für die Weggehphase. Zunächst einmal ist der Besuch eines Feldes generell möglich, da durch den Algorithmus niemals eine zuvor durchlaufene Richtung noch einmal beschritten wird. Nach der Lage des Feldes und den vorhandenen Belegungen unterscheiden sich die Belegrestriktionen in diejenigen, die den Schritt „links herum" bewirken und in diejenigen, die einen Rückschritt einleiten. Zur ersten Kategorie zählen die beiden ersten der nachfolgenden drei Punkte.

1. Wird nach **Fig. 8** die Gegenrichtung zur Ankunftsrichtung des 1. Besuches eingeschlagen, so muß vom Prinzip „rechts herum" abgewichen werden.
2. Wird nach **Fig. 9** die Richtung zur Weggehrichtung des 1. Besuches eingeschlagen, so ist auch hier „rechts herum" nicht möglich.

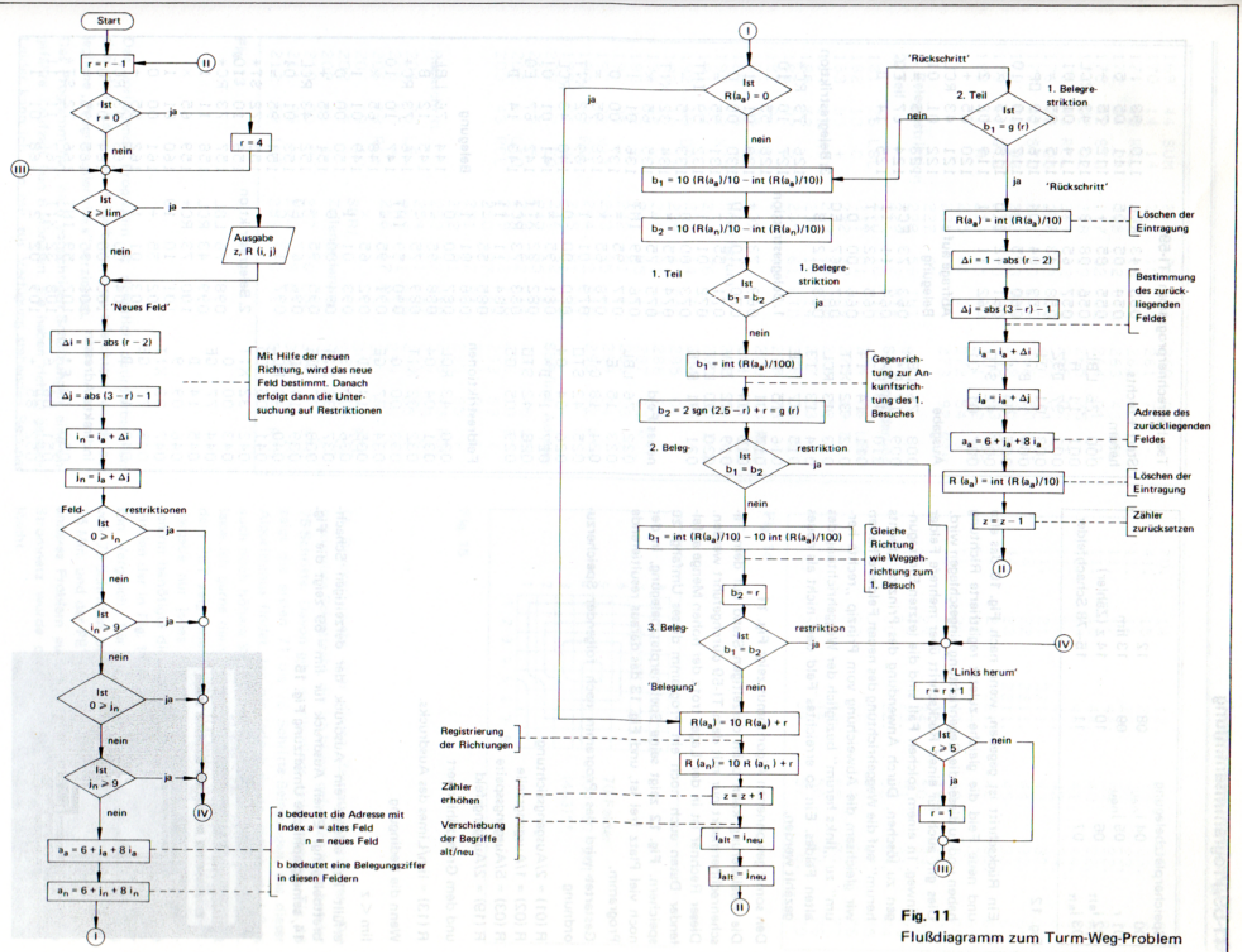


Fig. 11
Flußdiagramm zum Turm-Weg-Problem

Speicherplatzbelegung			
00	04 i_{neu}	08	12
01 r	05 j_{neu}	09	13 lim
02 i_{alt}	06	10	14 z (Zähler)
03 j_{alt}	07	11	15–78 Schachfelder

Fig. 12

3. Ein Rückschritt ist gegeben, wenn nach **Fig. 10** das alte und neue Feld die gleiche zuletzt registrierte Richtung haben und zu dieser die Gegenrichtung eingeschlagen wird. Dies gilt auch für einen Rückschritt über mehrere Felder hinweg. In einem solchen Fall sind die letzten Eintragungen zu löschen. Durch Anwendung des Prinzips „rechts herum" auf die Weggehrichtung des neuen Feldes erhalten wir gleichsam die Abweichung vom Prinzip „rechts herum" zu „links herum" bezüglich der Weggehrichtung des alten Feldes. Ein so erreichtes Feld darf nicht als neues gezählt werden.

Den somit gewonnenen Algorithmus zeigt **Fig. 11**.

Die Realisierung kann beim derzeitigen Stand auf dem Taschenrechnermarkt nur mit dem TI-59 durchgeführt werden. Dieser Rechner ist in der Lage, trotz der hohen Menge anfallender Daten auch noch ein Programm dieses Umfangs zu speichern. **Fig. 12** zeigt seine Speicherplatzbelegung, in der noch viel Platz frei ist, und **Fig. 13** das daraus resultierende Programm.

Gestartet wird das Programm nach folgender Speicherzuordnung

R (01) = 2/Ausgangsrichtung
R (02) = 1/Ausgangszeile
R (03) = 5/Ausgangsspalte
R (19) = 2/Ausgangsfeld

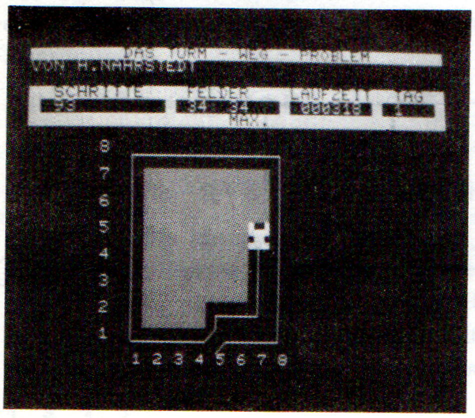
und dem Grenzzählerwert

R (13) = lim/Limes des Ausdrucks.

Wenn die Bedingung

$lim < z$

erfüllt ist, erfolgt ein Ausdruck der derzeitigen Schachbrettbelegung. Einen Ausdruck für lim = 69 zeigt die **Fig. 14**, seine graphische Umsetzung **Fig. 15**.



Taschenrechnerprogramm (TI-59)

Start + rechts herum

```
000  76  LBL
001  11   A
002  97  DSZ
003  01   01
004  17  B'
005  04   4
006  42  STO
007  01   01
```

Ausgabe

```
008  76  LBL
009  17  B'
010  43  RCL
011  14   14
012  32  X:T
013  43  RCL
014  13   13
015  77   GE
016  15   E
017  01   1
018  04   4
019  22  INV
020  90  LST
021  91  R/S
```

neues Feld

```
022  76  LBL
023  15   E
024  18  C'
025  42  STO
026  04   04
027  19  D'
028  42  STO
029  05   05
```

Feldrestriktionen

```
030  43  RCL
031  04   04
032  32  X:T
033  00   0
034  77   GE
035  14   D
036  09   9
037  32  X:T
038  77   GE
039  14   D
040  43  RCL
041  05   05
042  32  X:T
043  00   0
044  77   GE
045  14   D
046  09   9
047  32  X:T
048  77   GE
049  14   D
```

indirekte Adressen

```
050  10  E'
051  06   6
052  85   +
053  43  RCL
054  05   05
055  85   +
056  08   8
057  65   ×
058  43  RCL
059  04   04
060  95   =
061  42  STO
062  11   11
```

Abfrage auf Belegung

```
063  73  RC*
064  11   11
065  32  X:T
066  00   0
067  67   EQ
068  12   B
```

1. Belegrestriktion

```
069  73  RC*
070  10   10
071  55   ÷
072  01   1
073  00   0
074  95   =
075  75   -
076  59  INT
077  95   =
078  65   ×
079  01   1
080  00   0
081  95   =
082  32  X:T
083  73  RC*
084  11   11
085  55   ÷
086  01   1
087  00   0
088  95   =
089  75   -
090  59  INT
091  95   =
092  65   ×
093  01   1
094  00   0
095  95   =
096  67   EQ
097  13   C
```

2. Belegrestriktion

```
098  76  LBL
099  43  RCL
100  73  RC*
101  10   10
102  55   ÷
103  01   1
104  00   0
105  00   0
106  95   =
107  59  INT
108  32  X:T
109  02   2
110  93   .
111  05   5
112  75   -
113  43  RCL
114  01   01
115  95   =
116  69  OP
117  10   10
118  65   ×
119  02   2
120  85   +
121  43  RCL
122  01   01
123  95   =
124  67   EQ
125  14   D
```

3. Belegrestriktion

```
126  73  RC*
127  10   10
128  55   ÷
129  01   1
130  00   0
131  95   =
132  59  INT
133  75   -
134  32  X:T
135  65   ×
136  01   1
137  00   0
138  95   =
139  32  X:T
140  43  RCL
141  01   01
142  67   EQ
143  14   D
```

Belegung

```
144  76  LBL
145  12   B
146  73  RC*
147  10   10
148  65   ×
149  01   1
150  00   0
151  85   +
152  43  RCL
153  01   01
154  95   =
155  72  ST*
156  10   10
157  73  RC*
158  11   11
159  65   ×
160  01   1
161  00   0
162  85   +
163  43  RCL
164  01   01
165  95   =
166  72  ST*
167  11   11
168  01   1
```

```
169  44 SUM
170  14  14
171  43 RCL
172  04  04
173  42 STO
174  02  02
175  43 RCL
176  05  05
177  42 STO
178  03  03
179  61 GTO
180  11  A
```

1. Belegrestriktion
2. Teil

```
181  76 LBL
182  13  C
183  85  +
184  32 X:T
185  53  (
186  02  2
187  93  .
188  05  5
189  75  -
190  32 X:T
191  54  )
192  69 OP
193  10  10
194  65  ×
195  02  2
196  95  =
197  32 X:T
198  43 RCL
199  01  01
200  67 EQ
201  95  =
202  61 GTO
203  43 RCL
```

Rückschritt

```
204  76 LBL
205  95  =
206  73 RC*
207  10  10
208  55  ÷
209  01  1
210  00  0
211  95  =
212  59 INT
213  72 ST*
214  10  10
215  18 C'
216  42 STO
217  02  02
218  19 D'
219  42 STO
220  03  03
221  10 E'
222  73 RC*
223  10  10
224  55  ÷
225  01  1
226  00  0
227  95  =
228  59 INT
229  72 ST*
230  10  10
231  01  1
232  22 INV
233  44 SUM
234  14  14
235  61 GTO
236  11  A
```

links herum

```
237  76 LBL
238  14  D
239  43 RCL
240  01  01
241  85  +
242  01  1
243  95  =
244  42 STO
245  01  01
246  32 X:T
247  05  5
248  32 X:T
249  22 INV
250  77 GE
251  17 B'
252  01  1
253  42 STO
254  01  01
255  61 GTO
256  17 B'
```

SBR/
Zeilenänderung

```
257  76 LBL
258  18 C'
259  53  (
260  01  1
261  75  -
262  53  (
263  43 RCL
264  01  01
265  75  -
266  02  2
267  54  )
268  50 I×I
269  85  +
270  43 RCL
271  02  02
272  54  )
273  92 RTN
```

SBR/
Spaltenänderung

```
274  76 LBL
275  19 D'
276  53  (
277  53  (
278  03  3
279  75  -
280  43 RCL
281  01  01
282  54  )
283  50 I×I
284  75  -
285  01  1
286  85  +
287  43 RCL
288  03  03
289  54  )
290  92 RTN
```

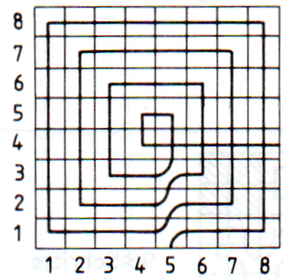
SBR/ind. Adresse

```
291  76 LBL
292  10 E'
293  53  (
294  06  6
295  85  +
296  43 RCL
297  03  03
298  85  +
299  08  8
300  65  ×
301  43 RCL
302  02  02
303  54  )
304  42 STO
305  10  10
306  92 RTN
```

Fig. 13

70.	14	12.	36	33.	58
41.	15	22.	37	33.	59
11.	16	22.	38	23.	60
11.	17	44.	39	22.	61
11.	18	44.	40	22.	62
2112.	19	44.	41	44.	63
11.	20	41.	42	34.	64
11.	21	2211.	43	33.	65
12.	22	2211.	44	33.	66
44.	23	2211.	45	33.	67
41.	24	221.	46	33.	68
11.	25	44.	47	23.	69
11.	26	44.	48	22.	70
2112.	27	44.	49	34.	71
11.	28	34.	50	33.	72
12.	29	23.	51	33.	73
22.	30	22.	52	33.	74
44.	31	22.	53	33.	75
44.	32	22.	54	33.	76
41.	33	44.	55	33.	77
11.	34	44.	56	23.	78
2112.	35	34.	57	0.	79

Fig. 14



70-Felder
[00 : 02 : 20]*
(453)**

Fig. 15

Das vorliegende Programm ist sicherlich optimierbar. Um aber dem Leser eine Einarbeitung zu erleichtern, wurde auf jegliche „Programmiertricks" verzichtet.

Das Programm hat, nach vorsichtigen Schätzungen, eine ungefähre Laufzeit von 6 Monaten. Sie haben richtig gelesen. Leider kann ich den mir zur Verfügung stehenden Rechner (+ Drucker) nicht über diesen Zeitraum entbehren. Die maximalste Schrittzahl, die ich bisher gefunden habe, beträgt 76-Schritte. Davon gibt es allerdings einige tausend. **Fig. 16** zeigt die ersten 71 bis 76 Schritte-Belegungen, die dieser Algorithmus findet. Ich bin jedoch davon überzeugt, daß es auch noch höhere Schritte-Belegungen gibt. Das Finden überlasse ich gerne dem Leser. Ein Versuch der Berechnung mit dem PET zeigt ebenfalls lange Rechenzeiten. Bei einer Wiedergabe nur jedes tausendsten Durchlaufes auf dem Bildschirm benötigte dieser eine Zeit von 643 Std. 25 Min. zum Finden der in **Fig. 17** wiedergegebenen 76-Schritte-Belegung. Ein Vergleich: Für die in **Fig. 16** wiedergegebene Belegung von 70 Schritten benötigte der TI-59 eine Zeit von 1 Std. 15 Min. und der PET 2,5 Min. Beim PET trat auch noch ein anderes Problem auf. Durch kurzfristige Schwankungen im Stromnetz wurde das Programm im Rechner zweimal gelöscht.

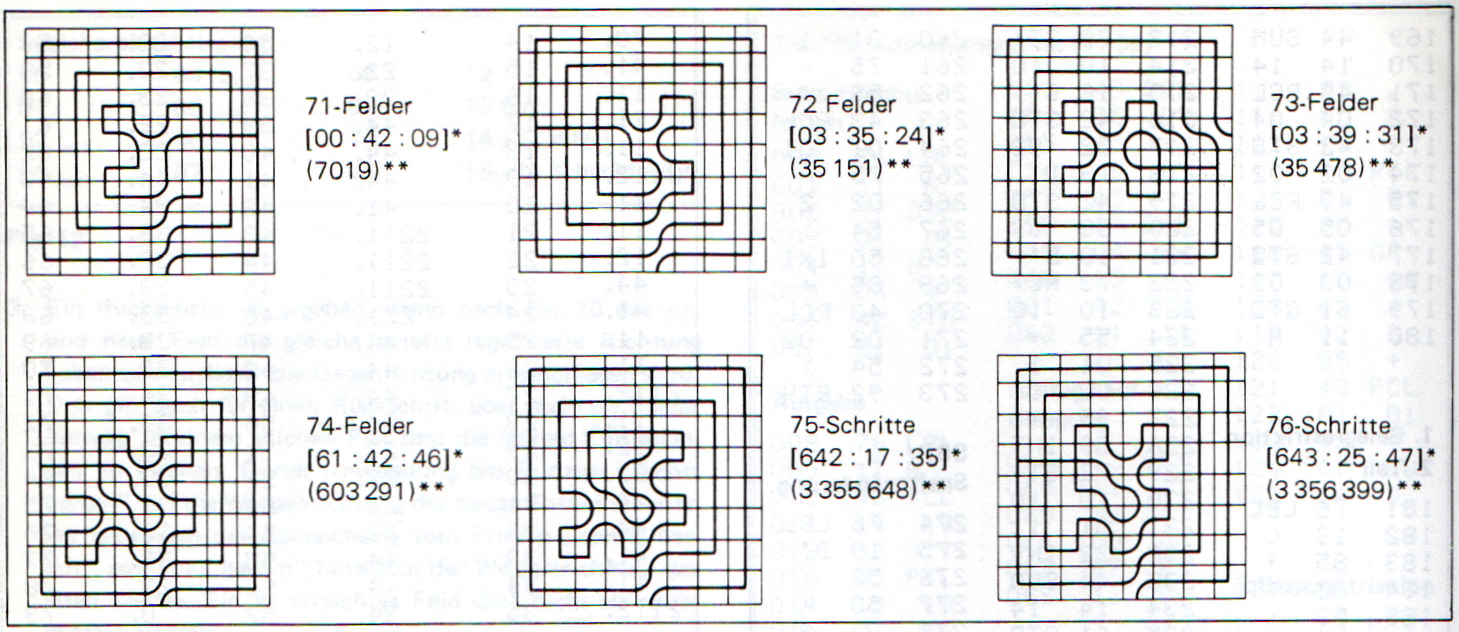


Fig. 16

Zu diesem Zeitpunkt hatte der Rechner 41 900 verschiedene 70-Schritte-Belegungen gefunden.

Die Berechnung auf dem PET 2001 Commodore ergab:
*** Zeit in [Std. : Min : Sek]**
**** (Anzahl der Durchläufe im Programm)**

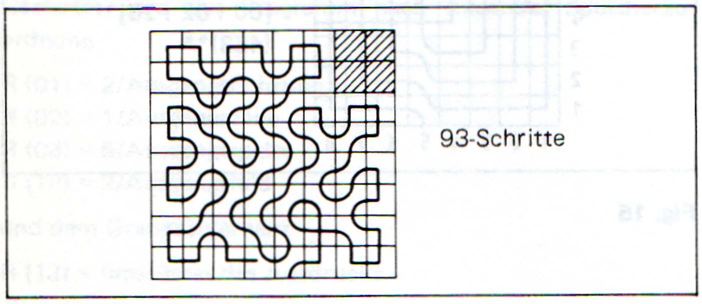


Fig. 17

Zum Abschluß möchte ich noch aufzeigen, zu welcher weiteren Nutzung dieses Programm beitragen kann. Es läßt sich nämlich zu einem Analyseprogramm für Schachprobleme ausbauen. Statt der leeren Schachfelder können diese mit Ziffern belegt werden, die einer Schachfigur zugeordnet sind und deren Größe die Priorität (= Bedeutung) der Figur darstellt. Auf diese Weise lassen sich für jede eigene Figur durch Bestimmung des Ausgangsfeldes alle Zugmöglichkeiten untersuchen. Die Bearbeitungszeit richtet sich dann nach der Belegung. Sie ist jedoch sicherlich nicht so lang wie die des vorliegenden Problems. Und etwas ist sicher: Die Analyse eines solchen Programms zeigt den oder die optimalen Wege und nicht, wie die auf dem Markt befindlichen Schachcomputer, nur Standardzüge. Es ist eben alles eine Frage der Zeit.

P.S.
Nach neuesten vorsichtigen Schätzungen glaube ich eher an eine Laufzeit von 12 Monaten. Daß es noch höhere Schritt-Belegungen gibt, zeigt die durch Versuche gewonnene Belegung in Fig. 17.