3.4.2. Das Problem der stabilen Heirat

Gegeben seien zwei verschiedene Mengen A und B mit gleicher Kardinalität. Gesucht ist eine Menge von n Paaren <a,b>, so dass a IN A, b IN B, und gewisse Nebenbedingungen erfüllt sind. Eine solche Nebenbedingung wird durch die *Regel der stabilen Heirat* gegeben und geht aus folgendem Beispiel hervor:

Sei A eine Menge von Männern und B eine Menge von Frauen. Jeder Mann und jede Frau hat eine Wunschliste der Partner in der Reihenfolge ihrer Bevorzugung aufgestellt. Wenn die n Ehepaare so gewählt werden, dass es einen Mann und eine Frau gibt, die nicht miteinander verheiratet sind, die sich aber gegenseitig vor ihren tatsächlichen Ehepartnern den Vorzug geben, dann heisst die Wahl unstabil. Existiert kein solches Paar, so ist die Wahl stabil.

Diese Situation ist typisch für viele ähnliche Probleme, bei denen die Wahl je nach Bevorzugung zu treffen ist, wie z.B. die Wahl einer Schule durch Studenten, die Wahl der verschiedenen Zweige der Armee durch Rekruten, usw. Das Beispiel des Heiratens ist besonders anschaulich; zu beachten ist jedoch, dass die aufgestellte Liste der Bevorzugungen invariant ist, d.h. sich nicht ändert, nachdem eine bestimmte Zuordnung einmal getroffen wurde. Diese Regel vereinfacht das Problem; ihr wohnt aber gleichzeitig eine Verzerrung der Wirklichkeit (also eine Abstraktion) inne.

Ein Weg zur Suche einer Lösung ist der Versuch, Mitglieder aus beiden Mengen nacheinander zusammenzuführen, bis beide Mengen leer sind. Sollen *alle* stabilen Zuordnungen gefunden werden, so können wir eine Lösung leicht unter Verwendung von Programmschema (3.35) als Schablone skizzieren. Bezeichnet try(m) den Algorithmus zum Finden eines Partners für den Mann m, und erfolgt diese Suche in der Reihenfolge der Liste der Bevorzugungen des Mannes, so ergibt sich eine erste Version in (3.36).

```
PROCEDURE try(m: man);                                    (3.36)
   VAR r: rank;
BEGIN
   FOR r := 1 TO n DO
   BEGIN "greife r-te Bevorzugung des Mannes m heraus";
      IF "annehmbar" THEN
      BEGIN "notiere Heirat";
         IF "m ist nicht letzter Mann" THEN try(succ(m))
            ELSE "notiere stabile Menge";
         "annuliere Heirat"
      END
   END
END
```

Hier sind wir nun wieder an einem Punkt angelangt, an dem wir ohne weitere Entscheidungen über die Darstellung der Daten nicht weiterkommen. Wir führen

drei skalare Typen ein, deren Werte der Einfachheit halber die ganzen Zahlen 1 bis n sind. Obwohl diese drei Typen formal gleich sind, trägt die unterschiedliche Bezeichnung wesentlich zur Klarheit bei. Es wird dadurch offensichtlich, wozu eine Variable verwendet wird.

```
TYPE man = 1..n;                                    (3.37)
     woman = 1..n;
     rank = 1..n
```

Die Anfangswerte sind durch zwei Matrizen dargestellt, die die Bevorzugungen der Männer und Frauen enthalten.

```
VAR wmr: ARRAY [man, rank] OF woman;                (3.38)
    mwr: ARRAY [woman, rank] OF man
```

Folglich bezeichnet wmr[m] die Liste der Bevorzugungen von Mann m, d.h. wmr[m,r] ist die Frau, die auf dem r-ten Platz in der Liste des Mannes m figuriert. Entsprechend ist mwr[w] die Liste der Bevorzugungen der Frau w, und mwr[w,r] ist ihre r-te Wahl.

Das Ergebnis steht in einem Array x von Frauen, und x[m] bezeichnet die Partnerin des Mannes m. Um die Symmetrie zwischen Mann und Frau aufrechtzuerhalten - die oft zitierte "Gleichberechtigung" - wird ein zusätzlicher Array y eingeführt, so dass y[w] den Partner der Frau w angibt.

```
var x: ARRAY [man] OF woman;                        (3.39)
    y: ARRAY [woman] OF man
```

Es ist klar, dass y nicht unbedingt erforderlich ist, da es bereits in x vorhandene Information enthält. Es gelten folgende Beziehungen

```
x[y[w]] = w,        y[x[m]] = m                     (3.40)
```

für alle verheirateten m und w. Somit kann der Wert y[w] durch einfaches Durchsuchen von x erhalten werden; der Array y erhöht aber die Effizienz des Algorithmus. Die durch x und y dargestellte Information wird zur Bestimmung der Stabilität einer vorgelegten Menge von Ehepaaren benötigt. Da diese Menge schrittweise durch Verheiraten eines Paares und anschliessendem Testen der Stabilität konstruiert wird, werden x und y sogar benötigt, bevor alle ihre Komponenten definiert sind. Um die bereits definierten Komponenten festzuhalten, könnten wir Boole'sche Arrays einführen

```
singlem : ARRAY [man] OF BOOLEAN                    (3.41)
singlew : ARRAY [woman] OF BOOLEAN
```

wobei aus "NOT singlem[m]" folgt, dass x[m] definiert ist, und aus "NOT singlem[w]" folgt, dass y[w] definiert ist.

Betrachtet man aber den vorgeschlagenen Algorithmus, so erkennt man schnell, dass der Zivilstand eines Mannes durch den Wert m auf einfachere Art bestimmt

ist, nämlich durch die Beziehung

```
NOT singlem[k]  =  (k < m)                                    (3.42)
```

Damit kann auf den Array ***singlem*** verzichtet werden, und der Name ***singlew*** wird vereinfacht zu ***single***.

Diese Abmachungen ergeben die in (3.43) gezeigte Verfeinerung; das Prädikat "annehmbar" wird als Konjunktion "single[w] AND stabil" ausgedrückt, wobei "stabil" eine weiter auszuarbeitende Funktion ist.

```
PROCEDURE try(m: man);                                        (3.43)
   VAR r: rank; w: woman;
BEGIN
  FOR r := 1 TO n DO
  BEGIN w := wmr[m,r];
    IF single[w] AND "stabil" THEN
    BEGIN x[m] := w; y[w] := m; single[w] := FALSE;
      IF m < n THEN try(succ(m))
               ELSE "notiere stabile Menge";
      single[w] := TRUE
    END
  END
END
```

In dieser Fassung ist die starke Ähnlichkeit der Lösung mit Programm 3.5 immer noch festzustellen, da beide aus demselben Schema hervorgegangen sind. Die wesentliche Aufgabe ist nun die Verfeinerung des Algorithmus zur Bestimmung der Stabilität. Leider kann die Stabilität nicht durch einen ähnlich einfachen Ausdruck dargestellt werden wie die Sicherheit der Damen in Programm 3.5. Zunächst sollte man beachten, dass sich die Stabilität nach Definition aus Vergleichen von Bevorzugungen oder Reihenfolgen ergibt. Die Reihenfolge von Männern oder Frauen ist aber in der bisherigen Aufstellung der Daten nicht explizit verfügbar. Sicherlich kann der Platz der Frau w in der Einschätzung von Mann m berechnet werden, aber dies erfordert ein aufwendiges Suchen von w in wmr[m].

Da die Berechnung der Stabilität eine äusserst häufige Operation ist, soll diese Information direkt zur Verfügung gestellt werden. Dazu führen wir zwei Matrizen ein:

```
rmw : ARRAY [man, woman] OF rank;                             (3.44)
rwm : ARRAY [women, man] OF rank
```

so dass rmw[m,w] den Platz der Frau w in der Liste der Bevorzugungen von Mann bezeichnet und rwm[w,m] den Platz von Mann m in der Liste der Frau w. Offensichtlich sind die Werte dieser Hilfs-Arrays konstant und können zu Begin aus den Werten von wmr bestimmt werden.

Der Prozess zur Bestimmung des Prädikats "stabil" läuft nun entsprechend der ursprünglichen Definition ab. Bekanntlich untersuchen wir die Möglichkeit der Heirat zwischen m und w, wobei w = wmr[m,r], d.h. w den Platz r in der Liste der Bevorzugungen von m einnimmt. Als Optimisten nehmen wir zunächst an, dass die Stabilität erhalten bleibt, und versuchen dann, mögliche Quellen für Schwierigkeiten zu finden. Wo könnten sie verborgen sein? Es gibt zwei symmetrische Möglichkeiten:

1. Es könnte eine Frau pw (preferred woman) geben, die von m seiner Frau w vorgezogen wird, und diese Frau selbst zieht m ihrem Mann vor.

2. Es könnte einen Mann pm (preferred man) geben, der von w ihrem Mann m vorgezogen wird, während pm selbst w seiner Frau vorzieht.

Verfolgen wir Fall 1 und vergleichen die Plätze rwm[pw,m] und rwm[pw,y[pw]] für alle Frauen, die m der Frau w vorzieht, d.h. für alle pw = wmr[m,i] mit i < r. Wir wissen, dass alle in Frage kommenden Frauen bereits verheiratet sind, da eine ledige schon vorher für m ausgesucht worden wäre. Der hier beschriebene Vorgang kann als einfacher linearer Suchprozess formuliert werden; dabei steht s für Stabilität.

```
s := TRUE; i := 1;                                          (3.45)
WHILE (i < r) AND s DO
BEGIN pw := wmr[m,i]; i := i+1;
   IF NOT single[pw] THEN s := rwm[pw,m] > rwm[pw,y[pw]]
END
```

Verfolgen wir Fall 2, so müssen wir alle Kandidaten pm untersuchen, die w ihrem gegenwärtig angetrauten m vorzieht, d.h. in Frage kommen alle vorgezogenen Männer pm = wmr[w,i] mit i < rwm[w,m]. In Analogie zu Fall 1 sind Vergleiche zwischen den Plätzen rmw[pm,w] und rmw[pm,x[pm]] notwendig. Wir müssen dabei vorsichtig vorgehen und die Vergleiche mit x[pm] auslassen, wenn pm noch ledig ist. Dies erfordert einen Test pm < m, da bekanntlich alle Männer vor m bereits verheiratet sind.

Programm 3.6 enthält den vollständigen Algorithmus. Tabelle 3.3 gibt einen Satz von Eingabe-Werten für die Arrays wmr und mwr, und Tabelle 3.4 schliesslich stellt die zugehörigen neun berechneten stabilen Lösungen dar.

```
PROGRAMM marriage(input,output);
{Problem der stabilen Heirat}
CONST n = 8;
TYPE man = 1..n; woman = 1..n; rank = 1..n;

VAR m: man; w: woman; r: rank;
   wmr: array [man, rank] OF woman;
   mwr: ARRAY [woman, rank] OF man;
   rmw: ARRAY [man, woman] OF rank;
```

```
   rwm: ARRAY [woman, man] OF rank;
   x:   ARRAY [man] OF woman;
   y:   ARRAY [woman] OF man;
   single: ARRAY [woman] OF BOOLEAN;

PROCEDURE print;
   VAR m: man; rm, rw: INTEGER;
BEGIN rm := 0; rw := 0;
   FOR m := 1 TO n DO
   BEGIN write(x[m]:4);
      rm := rm + rmw[m,x[m]]; rw := rw + rwm[x[m],m]
   END ;
   writeln(rm:8, rw:4)
END {print} ;

PROCEDURE try(m: man);
   VAR r: rank; w: woman;

   FUNCTION stable: BOOLEAN;
      VAR pm: man; pw: women;
          i, lim: rank;  s: BOOLEAN;
   BEGIN s := TRUE; i := 1;
      WHILE (i < r) AND s DO
      BEGIN pw := wmr[m,i] ; i := i+1;
         IF NOT single[pw] THEN
            s := rwm[pw,m] > rwm[pw,y[pw]]
      END ;
      i := 1; lim := rwm[w,m];
      WHILE (i < lim) AND s DO
      BEGIN pm := mwr[w,i] ; i := i+1;
         IF pm < m THEN s := rmw[pm,w] > rmw[pm,x[pm]]
      END ;
      stable := s
   END {stable} ;

BEGIN {try}
   FOR r := 1 to n DO
   begin w := wmr[m,r];
      IF single[w] THEN
         IF stable THEN
         BEGIN x[m] := w; y[w] := m; single[w] := FALSE;
            IF m < n THEN try(succ(m)) ELSE print;
            single[w] := TRUE
         END
   END
```

```
END {try} ;

BEGIN
   FOR m := 1 TO n DO
      FOR r := 1 TO n DO
      BEGIN read(wmr[m,r]); rmw[m,wmr[m,r]] := r
      END ;
   FOR w := 1 TO n DO
      FOR r := 1 TO n DO
      BEGIN read(mwr[w,r]); rwm[w,mwr[w,r]] := r
      END ;
   FOR w := 1 TO n DO single[w] := TRUE;
   try(1)
END .
```

Programm 3.6: Stabile Heiraten

Reihenfolge	1	2	3	4	5	6	7	8
Mann 1 wählt Frau	7	2	6	5	1	3	8	4
2	4	3	2	6	8	1	7	5
3	3	2	4	1	8	5	7	6
4	3	8	4	2	5	6	7	1
5	8	3	4	5	6	1	7	2
6	8	7	5	2	4	3	1	6
7	2	4	6	3	1	7	5	8
8	6	1	4	2	7	5	3	8
Frau 1 wählt Mann	4	6	2	5	8	1	3	7
2	8	5	3	1	6	7	4	2
3	6	8	1	2	3	4	7	5
4	3	2	4	7	6	8	5	1
5	6	3	1	4	5	7	2	8
6	2	1	3	8	7	4	6	5
7	3	5	7	2	4	1	8	6
8	7	2	8	4	5	6	3	1

Tabelle 3.3

Lösung	x1	x2	x3	x4	x5	x6	x7	x8	rm	rw	c
1	7	4	3	8	1	5	2	6	16	32	21
2	2	4	3	8	1	5	7	6	22	27	449
3	2	4	3	1	7	5	8	6	31	20	59
4	6	4	3	8	1	5	7	2	26	22	62

5	6	4	3	1	7	5	8	2	35	15	47
6	6	3	4	8	1	5	7	2	29	20	143
7	6	3	4	1	7	5	8	2	38	13	47
8	3	6	4	8	1	5	7	2	34	18	758
9	3	6	4	1	7	5	8	2	43	11	34

c = Anzahl der Tests auf Stabilität
Lösung 1 = optimale Lösung für die Männer
Lösung 9 = optimale Lösung für die Frauen

Tabelle 3.4

Dieser Algorithmus stützt sich wesentlich auf das Backtracking-Schema. Seine Effizienz hängt primär davon ab, wie weit das Schema zur Beschneidung des Lösungsbaumes entwickelt ist. Ein etwas schnellerer, aber komplexerer und undurchsichtigerer Algorithmus wurde von McVitie und Wilson vorgelegt ([21] und [22]), die ihn auch auf verschieden grosse Mengen (von Männern und Frauen) ausgedehnt haben.

Algorithmen von der Art der beiden letzten Beispiele, die *alle* möglichen Lösungen (unter gewissen Nebenbedingungen) generieren, werden oft verwendet, um eine oder mehrere, in gewisser Beziehung *optimale* Lösungen auszuwählen. Bei diesem Beispiel könnte man an der Lösung interessiert sein, die dem Wunsch der Männer, oder der Frauen, oder aller Personen im Mittel am besten gerecht wird.

Dazu enthält Tabelle 3.4 die Summe der Plätze aller Frauen in der Liste der Bevorzugungen ihrer Ehemänner und die Summen der Plätze aller Männer in der Liste ihrer Ehefrauen. Dies sind die Werte

$$rm = \text{Summe}(rmw[m,x[m]]),\quad rw = \text{Summe}(rwm[x[m],m]) \qquad (3.46)$$

wobei m jeweils von 1 bis n läuft. Die Lösung mit dem kleinsten Wert rm heisst "für die Männer optimale stabile Lösung", die mit dem kleinsten rw ist die "für die Frauen optimale stabile Lösung". Durch die Art der gewählten Suchstrategie werden die für die Männer günstigen Lösungen zuerst generiert, die aus der Perspektive der Frauen günstigen Lösungen erscheinen am Ende. Der Algorithmus bevorzugt in diesem Sinn die männliche Bevölkerung. Dies ist (in der Abstraktion) durch systematisches Vertauschen der Rolle von Mann und Frau leicht zu ändern, nämlich durch Vertauschen von mwr mit wmr und rmw mit rwm.

Wir sehen davon ab, dieses Programm weiter auszubauen und behandeln die Suche nach einer optimalen Lösung im nächsten und letzten Beispiel eines Algorithmus mit Backtracking.