Elaborato IA

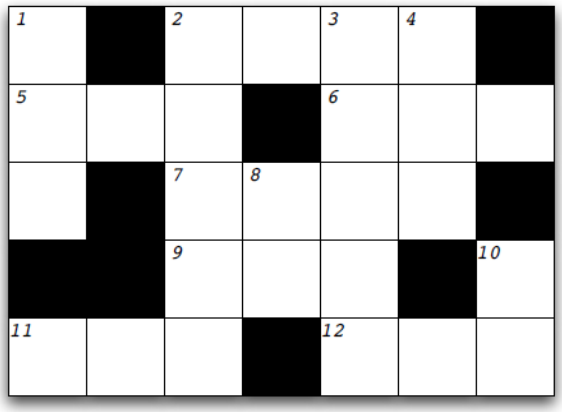
Gian Maria Pandolfi

1 Introduzione

In questo elaborato studiamo un modello per risolvere un problema di soddisfacimento di vincoli che riguarda una variante del classico cruciverba, dove le caselle bianche dello schema *n x m* devono essere riempite con numeri compresi tra due interi assegnati *r > 0* e *s* in modo che il prodotto dei numeri (in orizzontale o verticale) produca la definizione. Nel problema è assegnato lo schema del cruciverba e un set S di interi distinti che rappresentano i valori da usare come definizioni. Si devono costruire le definizioni in modo da usare esattamente una volta ciascun elemento di S e riempire le caselle in modo da rispettare le definizioni. Riportiamo il seguente esempio:

Set assegnato:

$S = \{6,8,24,28,30,36,42,54,64,70,84,90,96\}$



2 Modello

Il modello è stato sviluppato con *Minizinc* ed ha le seguenti caratteristiche:

- un array *board nxm*, formato dalle caselle X di dominio $D = \{s..r\}$, che rappresenta la griglia del cruciverba e nel quale le caselle nere sono rappresentate come *z*eri.

- un set di interi distinti *solutions* che rappresenta il set S assegnato dal problema.

- una matrice *segments* nella quale ogni riga rappresenta un segmento di caselle contigue, infatti ogni segmento è composto dalle posizioni delle caselle nell'array board. Questa matrice contiene sia i segmenti in orizzontale che in verticale e, ovviamente, possiede un numero di righe pari alla grandezza del set *solutions*.

- un array *sol* che ha come dominio il set *solutions* e contiene un numero di elementi pari a quest'ultimo. L'array sarà utilizzato per contenere il prodotto di ogni elemento appartenente allo stesso segmento.

Riportiamo un esempio di *board* e di *segments*, che vengono passati come parametri in input assieme all'intervallo *s…r*, riferiti al problema mostrato precedentemente:

```
board = [
 _,  0,  _,  _,  _,  _,  0,
 _,  _,  _,  0,  _,  _,  _,
 _,  0,  _,  _,  _,  _,  0,
 0,  0,  _,  _,  _,  0,  _,
 _,  _,  _,  0,  _,  _,  _,
];

r=1;
s=9;
```

```
segments = [|
%across
 3,   4,   5,  6, 0|
 8,   9,  10,  0, 0|
12,  13,  14,  0, 0|
17,  18,  19, 20, 0|
24,  25,  26,  0, 0|
29,  30,  31,  0, 0|
33,  34,  35,  0, 0|
%down
 1,   8, 15,  0,  0|
 3,  10, 17, 24, 30|
18,  25,  0,  0,  0|
 5,  12, 19, 26, 33|
 6,  13, 20,  0,  0|
28,  35,  0,  0,  0|
|];
```

Possiamo notare che i segmenti hanno una lunghezza massima ed alcuni segmenti hanno una lunghezza inferiore rispetto ad essa quindi si utilizzano gli zeri per completare la matrice *segments*.

I vincoli che vengono utilizzati dal modello sono 2:

1. il primo impone che tutti gli elementi dell'array *sol* siano diversi tra loro in modo tale che l'array contenga tutti gli elementi del suo dominio che è il set di soluzioni assegnato.
   .
2. il secondo vincola la produttoria di ogni segmento ad essere uguale ad un valore del set assegnato. In questo vincolo si usano i segmenti per accedere alle effettive posizioni delle caselle nel *board*.

L'output del nostro modello ci mostra la composizione finale del *board* e l'array *sol* in modo tale da poter controllare che la nostra soluzione rispetti le richieste del problema.

Si riporta l'output relativo al nostro problema:

```
1 0 1 3 1 2 0
6 8 2 0 2 5 7
7 0 1 4 7 3 0
0 0 9 6 1 0 1
5 2 9 0 2 4 8

[6, 96, 70, 84, 54, 90, 64, 42, 36, 24, 28, 30, 8]
```

3 Test

Si riportano 2 istanze sulle quali è stato testato il nostro modello, tralasciando le matrici *segments*.

3.1 Test 1

Input:

```
board = [
 _,  _,  0,  _,  _,  _,  _,  _
 0,  _,  0,  _,  _,  _,  0,  _,
 _,  _,  _,  _,  0,  _,  0,  _,
 _,  0,  _,  _,  _,  0,  0,  0,
 _,  _,  _,  0,  _,  0,  _,  0,
 0,  0,  _,  _ , _,  _,  _,  _
];
```

```
solutions = {2, 3, 14, 18, 27, 33, 36, 42, 45, 54, 70, 77, 80, 99, 100, 110};
r=1;
s=19;
```

Output:

```
9 5 0 1 2 1 7 3
0 5 0 1 18 3 0 1
1 4 2 10 0 11 0 9
2 0 7 11 1 0 0 0
9 11 1 0 3 0 2 0
0 0 1 1 1 5 1 14

[45, 42, 54, 80, 77, 99, 70, 100, 110, 36, 33, 27, 18, 14, 3, 2]
```

3.2 Test 2

Input:

```
board = [
 _,  _,  _,  _,  _,  _,  0,  _,
 _,  _,  _,  0,  _,  _,  _,  _,
 0,  0,  _,  0,  0,  _,  _,  _,
 _,  _,  _,  _,  _,  _,  0,  0,
 _,  _,  0,  _,  _,  _,  _,  _,
];

solutions = {2, 7, 22, 24, 27, 33, 35, 39, 44, 56, 65, 78, 88, 91, 99,
120, 154, 220};

r=1;
s=15;
```

Output:

```
1 3 1 1 9 1 0 11
2 11 4 0 11 1 5 4
0 0 14 0 0 7 13 1
4 3 1 2 5 1 0 0
6 13 0 11 7 1 1 2

[27, 88, 220, 91, 120, 78, 154, 2, 33, 56, 99, 7, 44, 65, 24, 39, 22, 35]
```