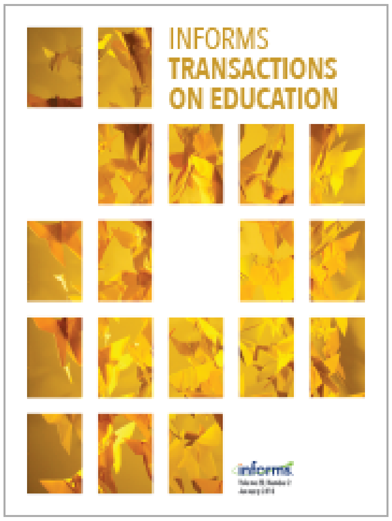

INFORMS Transactions on Education

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Puzzle—A Tank Attack Puzzle

Martin J. Chlond,

Vol. 8, No. 3, May 2008, pp. 149–152
ISSN 1532-0545 | 08 | 0803 | 0149

INFORMS Transactions on Education

informs®

DOI 10.1287/ited.1080.0011

Puzzle

A Tank Attack Puzzle

Martin J. Chlond
Lancashire Business School, University of Central Lancashire, Preston PR1 2HE, United Kingdom,
mchlond@uclan.ac.uk

Key words: integer programming; puzzle

Introduction

The writer was introduced to the following puzzle at the seventh "Gathering for Gardner." This event is held every two years in Atlanta in honor of Martin Gardner, the long time writer of the Mathematical Games column in *Scientific American*. Gardner enjoys a huge readership worldwide and the more fervent of these form the basis of the "Gathering." Attendees include puzzlists, mathematicians, scientists, artists, teachers, and magicians all of whom owe a debt of gratitude to Gardner for his inspirational writings spanning across six decades.

The puzzle originates from an international computer problem-solving competition in the late nineties where participants were emailed a number of puzzles simultaneously and had to submit solutions within 48 hours. This particular one involves a battleground represented by a square grid. Each cell on this grid contains a tank and each of these tanks is assigned a particular integer. The value of this integer represents the range of the tank such that a tank of value n may only attack tanks that are at a distance of exactly n cells along the same row or column. The objective of the puzzle is to assign values to each tank on the grid such that the number of tanks that may attack a cell is equal to the value of the tank in that cell. Note that a tank with value "zero" cannot be part of a solution as it can attack itself and would therefore have at least one attacker.

A 4×4 grid with all cells containing the value "two" and a 6×6 grid with the four center cells containing the value "four" and all other cells containing the value "two" represent simple solutions to the puzzle. No solutions exist for 1×1, 2×2, 3×3, 5×5, or 7×7 boards and no "simple" solution exists for an 8×8 board. In order to find some more interesting solutions we develop an integer program to model the puzzle.

IP Formulation

We define sets $N = \{1, \ldots, n\}$ and $M = \{1, \ldots, n-1\}$ where n is the length of a side of the board. Also, we define decision variables $x_{i,j,k} = 1$ if cell (i, j) has value k, else 0.

We need to ensure that the number of attacks on each tank is equal to the range of the tank and each tank is assigned a single value only. These conditions are achieved by the following two statements respectively:

$$\sum_{\substack{p \in N \\ p \neq j}} x_{i,p,|j-p|} + \sum_{\substack{q \in N \\ q \neq i}} x_{q,j,|i-q|} = \sum_{k \in M} k x_{i,j,k} \quad \forall i \in N,\ j \in N$$

$$\sum_{k \in M} x_{i,j,k} = 1 \quad \forall i \in N,\ j \in N$$

Results

A Mathprog implementation of the above formulation was processed overnight using the Gnu Linear Programming Kit (2007) but failed to find a solution to an 8×8 board. However, the following solution to a 6×6 board was found

1	3	2	1	5	1
5	2	2	2	2	3
1	2	3	3	2	2
2	2	3	3	2	1
3	2	2	2	2	5
1	5	1	2	3	1

and the 90 degree rotational symmetry it displays was exploited to reinvestigate the 8×8 board, that is, the search is restricted to find only solutions that display this same symmetry. This is achieved by appending the following constraints to the model.

$$x_{i,j,k} = x_{j,(n+1-i),k} \quad \forall i \in N,\ j \in N,\ k \in M$$
$$x_{i,j,k} = x_{(n+1-i),(n+1-j),k} \quad \forall i \in N,\ j \in N,\ k \in M$$

The Mathprog model for this revised formulation is in Appendix A and it was used to identify the following solution in less than 12 seconds.

2	1	3	3	1	6	3	2
3	4	2	3	3	1	4	1
6	1	2	1	4	2	2	3
1	3	4	4	4	1	3	3
3	3	1	4	4	4	3	1
3	2	2	4	1	2	1	6
1	4	1	3	3	2	4	3
2	3	6	1	3	3	1	2

No similarly symmetrical solution exists for a 10×10 board and given the experience with the 8×8 problem without symmetry constraints it was pointless to attempt to find an asymmetrical solution (if one exists) for the 10×10 board.

It is interesting to note that an infinite grid with the number four in each cell represents a solution to the problem as does a finite grid of 4×4 or greater wrapped around a torus.

The question arises as to whether less monotonous solutions exist for an infinite grid. While the limitations of current IP solvers prevent us from modeling such an infinite grid we may look for solutions on the surface of a torus. A tank that may reach another square from two opposite directions will count as two attacks. If found, such solutions may be unwrapped and used to tile the plane to provide solutions for an infinite grid.

The Mathprog model was amended to search for toroidal solutions but none were found. A further amendment was made to the model to allow for empty squares. Such squares do not contain a tank and therefore do not attack any others. Furthermore there is no restriction on the number of tanks that attack such a square. We are now looking to maximize the occupied squares and an appropriate objective function is required. The finished toroidal model incorporating these changes is in Appendix B and was used to produce the following.

6	6	3	2	3	6	6
6	6	3	2	3	6	6
3	3	4	3	4	3	3
2	2	3		3	2	2
3	3	4	3	4	3	3
6	6	3	2	3	6	6
6	6	3	2	3	6	6

3	4	3	4	2	4	3	3
3	2	7	7	2	3	2	4
4	3	4	1	6	4	7	3
2	2	6			1	7	4
4	7	1			6	2	2
3	7	4	6	1	4	3	4
4	2	3	2	7	7	2	3
3	3	4	2	4	3	4	3

As mentioned above, these flat boards are not solutions in themselves but may be used to tile the plane thereby producing solutions for an infinite grid.

Reference

Gnu Linear Programming Kit. 2007. http://www.gnu.org/software/glpk/, last accessed on September 24, 2007.

Appendix A

```
# model name   : tank.mod
# description  : tank puzzle (courtesy of Adam Atkinson G4G7)
# written by   : Martin Chlond
# date written : 6/2/08

param n := 8;

# indices
set N := 1..n;
set M := 1..n-1;

# variables
var x{i in N,j in N,k in M} binary;  # x[i,j,k] = 1 if number on tank {i,j} = k

subject to

  # attacks on tank = number on tank
  att{i in N,j in N}: sum{p in N : p <> j} x[i,p,abs(j-p)] + sum{q in N : q <> i} x[q,j,abs(i-q)]
  = sum{k in M} k*x[i,j,k];

  # one number on each tank
  one{i in N,j in N}: sum{k in M} x[i,j,k] = 1;
  # symmetry constraints - otherwise too slow
  syca{i in N,j in N,k in M}: x[i,j,k] = x[(n+1-i),(n+1-j),k];
  sycb{i in N,j in N,k in M}: x[i,j,k] = x[j,(n+1-i),k];

solve;

# print results

printf "\n";
for {i in N} {
  for {j in N} {
    printf sum{k in 1..n-1} k*x[i,j,k];
    printf " ";
  }
  printf "\n";
}

end;
```

Appendix B

```
# model name   : tanktorus.mod
# description  : tank puzzle (torus version - with holes)
# written by   : Martin Chlond
# date written : 19/3/08

param n := 7;

# indices
set N := 1..n;
set M := 1..n;

# variables
var x{i in N,j in N,k in M} binary;  # x[i,j,k] = 1 if number on tank {i,j} = k

maximize tocc: sum{i in N,j in N,k in M} x[i,j,k];

subject to

  # exclude trivial solution
  nf: x[1,1,4] = 0;

  # attacks on tank = number on tank (only for occupied squares)
  atta{i in N,j in N}: sum{p in N : p <> j}(x[i,p,abs(j-p)]+x[i,p,n-abs(j-p)])+sum{q in N : q <> i}
  (x[q,j,abs(i-q)]+x[q,j,n-abs(i-q)])<=sum{k in M} k*x[i,j,k]+n*(1-sum{k in M} x[i,j,k]);

  attb{i in N,j in N}: sum{p in N : p <> j}(x[i,p,abs(j-p)]+x[i,p,n-abs(j-p)])+sum{q in N : q <> i}
  (x[q,j,abs(i-q)]+x[q,j,n-abs(i-q)])>=sum{k in M} k*x[i,j,k];

  # one number on each tank
  one{i in N,j in N}: sum{k in M} x[i,j,k] <= 1;

  # symmetry constraints
  syca{i in N,j in N,k in M}: x[i,j,k] = x[(n+1-i),(n+1-j),k];
  sycb{i in N,j in N,k in M}: x[i,j,k] = x[j,(n+1-i),k];

solve;

# print results

printf "\n";
for {i in N} {
  for {j in N} {
    printf sum{k in 1..n-1} k*x[i,j,k];
    printf " ";
  }
  printf "\n";
}

end;
```