**Non-Transitive Dice: An Investigation into Constraint Programming**

**1 Introduction**

Non-transitive dice are any set of dice where Dice n rolls a higher number than Dice n+1 more than half the time but Dice n=0 does not roll higher than Dice nmax. The non-transitive dice problem fits well to constraint programming (and constraint satisfaction problems) due to a well defined set of rules governing winning and losing. A basic example set of non-transitive dice are given below in figure 1.

The aim of this report was to use the non-transitive dice puzzle as a building block, asking a number of questions into dice combinations and rules as a tool to gain an understanding of constraint programming and the techniques involved. The purpose of the experiments conducted within this report was simply to attempt to gain an answer to questions in as interesting a way as possible. A number of techniques reviewed and attempted in this report include eliminating symmetry in solutions, preprocessing, variable and value ordering, and simply reformulating initial constraints.

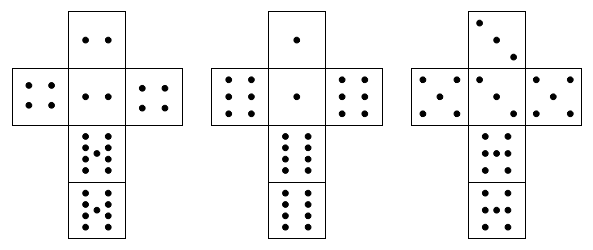


Figure 1. Basic Dice

**2 Modeling the Basic Problem**

As an introduction to constraint programming it was important to start with the most basic model of non-transitive dice. The set shown in figure 1 was used as a base and a set of constraints were extracted to create this set of dice.

Variables:

* Parameter declarations
  + Integer N, determines how many sides on each dice
  + Integer MAX, determines the maximum value in the domain of values to put on the dice
* Constant declarations
  + Values over a domain ranging from 1 to MAX
* Decision variables
  + 1D matrix for each dice being modelled: D1, D2, D3
    - Indexed by an integer from 1 to N
    - with an integer domain of values from 1 to MAX

Constraints:

* each 2 faces of each die must be equal e.g.
  + D1[1]=D1[2],
  + D1[3]=D1[4],
  + D1[5]=D1[6]
* Each pair of faces on a dice must be different e.g.
  + allDiff([D1[1], D1[3], D1[5]])
* 2 sides of D1 must be greater than 2 sides of D2
  + D1[1] > D2[1],
  + D1[5] > D2[5]
* 2 sides of D2 must be greater than 2 sides of D3
  + D2[3] > D3[3],
  + D2[5] > D3[5]
* 2 sides of D3 must be greater than 2 sides of D1
  + D3[1] > D1[1],
  + D3[3] > D1[3]
* Each dice must be ordered in ascending order in order to eliminate symmetry
  + D1[1] < D1[3],
  + D1[3] < D1[5]

**Experiments**

Once the original problem had successfully been modelled as a CSP in Essence’, the problem was run 3 times to obtain an average time and number of backtracks taken. Once the problem had been expanded to include symmetry breaking constraints it was run again 3 times to obtain values. These values are shown below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Problem Instance** | **No efficiency improvements** | | **Symmetry Breaking** | |
| **Backtracks** | **Time(secs)** | **Backtracks** | **Time(secs)** |
| **Non-transitive dice instance 1** |  |  |  | 0.166 |
| **Non-transitive dice instance 2** |  |  |  |  |
| **Non-transitive dice instance 3** |  |  |  |  |

Analysis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Problem Instance** | **No efficiency improvements** | | **Reformulate Constraints** | |
| **Backtracks** | **Time(secs)** | **Backtracks** | **Time(secs)** |
| **Non-transitive dice instance 1** |  |  |  |  |
| **Non-transitive dice instance 2** |  |  |  |  |
| **Non-transitive dice instance 3** |  |  |  |  |

Analysis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Problem Instance** | **Symmetry Breaking** | | **Reformulate Constraints** | |
| **Backtracks** | **Time(secs)** | **Backtracks** | **Time(secs)** |
| **Non-transitive dice instance 1** |  |  |  |  |
| **Non-transitive dice instance 2** |  |  |  |  |
| **Non-transitive dice instance 3** |  |  |  |  |

Analysis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Problem Instance** | **No efficiency improvements** | | **Dynamic Variable Ordering** | |
| **Backtracks** | **Time(secs)** | **Backtracks** | **Time(secs)** |
| **Non-transitive dice instance 1** |  |  |  |  |
| **Non-transitive dice instance 2** |  |  |  |  |
| **Non-transitive dice instance 3** |  |  |  |  |

Analysis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Problem Instance** | **Symmetry Breaking** | | **Dynamic Variable Ordering** | |
| **Backtracks** | **Time(secs)** | **Backtracks** | **Time(secs)** |
| **Non-transitive dice instance 1** |  |  |  |  |
| **Non-transitive dice instance 2** |  |  |  |  |
| **Non-transitive dice instance 3** |  |  |  |  |

Analysis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Problem Instance** | **Reformulate Constraints** | | **Dynamic Variable Ordering** | |
| **Backtracks** | **Time(secs)** | **Backtracks** | **Time(secs)** |
| **Non-transitive dice instance 1** |  |  |  |  |
| **Non-transitive dice instance 2** |  |  |  |  |
| **Non-transitive dice instance 3** |  |  |  |  |

Analysis