The chosen problem is **065: Optimal Financial Portfolio Design**, also referred to as **OPD**, the problem was proposed by Pierre Flener, Jean-Noël Monette and is presented in an abstract fashion trough a simple to explain requirement:

Given a matrix of a certain size, containing only the numbers 0 and 1, with each row having a specific number of 1’s, find a configuration for the matrix such that the maximum dot product between any 2 rows is minimized.

The problem has only 3 input parameters, **V**, **B** and **R**.

**V** and **B** are the matrix dimensions and **R** is the number of 1’s we must place on each row, making the problem become:

Given a 0/1 matrix of size **V**x**B**, with exactly **R** 1’s on each row, find a configuration for the matrix such that the maximum dot product between any 2 rows is minimized. This objective value is called **λ**.

The problem model uses **MiniZinc 2.3.2** as the optimization language and **Gecode 6.1.1** for the solver.

The **first iteration** of the model lacked any optimization at all and simply had the constraint that ensured the number of 1’s per row and the minimization objective and used an unoptimized implementation of the dot product.

The **second iteration** improved the dot product implementation using more features of the MiniZinc standard library and eliminating unnecessary bool->int->bool conversions.

The **third iteration** introduced additional constraints to break symmetry in the solution space for both columns and rows as duplicate rows would produce an objective of exactly R, the maximum, and worst, value possible.

The symmetry breaking costraints are marked as *symmetry\_breaking* for the solver because of additional optimizations that can be used by Gecode on constraints marked as such. The constraints are that each row and column has to be lexicographically greater or equal than the one before it.

Because of some implementation details of Gecode, while the median solve time decreased for the third model, some spikes appeared with up to 1000 times higher run times, spiking the average time. As the target **λ** decreases, the points that quickly lead to a solution also become rarer.

The **fourth iteration** changed the symmetry constraints to use greater not greater or equal comparison. The input dimensions are guaranteed to be small enough so that we do not „run out” of possible configurations of 1’s.

The problem instance on which the tests were run was **V=10, B=37, R=14** and the models were timed according to how fast on average they would achieve a certain **λ** value. The possible values for **λ** are 5..14.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Target λ** | | | | | | | | | |
|  | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** |
| **Time (s) (100 runs) ---------------- Model Iteration** | **Mean** | **Mean** | **Mean** | **Mean** | **Mean** | **Mean** | **Mean** | **Mean** | **Mean** | **Mean** |
| **1** | 0.10 | 0.11 | 0.17 | 0.54 | 2.10 | 10.57 | 56.15 | **100+** |  |  |
| **2** | 0.12 | 0.12 | 0.13 | 0.13 | 0.13 | 0.15 | 0.29 | 1.15 | **100+** |  |
| **3** | 0.13 | 0.24 | 0.26 | 0.26 | 0.27 | 0.29 | 1.00 | 1.04 | 27.59 | **100+** |
| **4** | 0.99 | 1.00 | 0.15 | 0.11 | 0.09 | 0.35 | 0.55 | 0.58 | 5.87 |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Target λ** | | | | | | | | | |
|  | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** |
| **Time (s) (100 runs) ---------------- Model Iteration** | **Median** | **Median** | **Median** | **Median** | **Median** | **Median** | **Median** | **Median** | **Median** | **Median** |
| **1** | 0.09 | 0.09 | 0.15 | 0.53 | 2.06 | 10.59 | 59.26 | **100+** |  |  |
| **2** | 0.11 | 0.11 | 0.12 | 0.12 | 0.12 | 0.13 | 0.29 | 1.18 | **100+** |  |
| **3** | 0.06 | 0.07 | 0.07 | 0.13 | 0.14 | 0.14 | 0.18 | 0.87 | 16.37 | **100+** |
| **4** | 0.98 | 0.94 | 0.15 | 0.09 | 0.08 | 0.18 | 0.18 | 0.44 | 0.31 |  |

As the target **λ** decreases, the points that quickly lead to a solution also become rarer. For the 3rd iteration of the model about 30-40% of the runs resulted in spikes in the runtime. The problem seemed to decrease to only about 5% for the 4th iteration of the model.



