University of St Andrews

CS4402 - Constraint Programming

Assignment: P1 - Late Binding Solitaire

Matriculation Number: 160001362

March 2020

Contents

1 Overview

1.1 Aim of Practical

The aim of this practical was to formulate a model of the game Late Binding Solitaire using the Essence Prime constraint modelling language and to test the performance of running this model on a number of supplied problem instances.

2 Execution Instructions

For convenience a script called 'run.sh' has been developed to make it easier to run Savile Row with the Essence Prime model and a provided parameter file.

To run the essence prime model using this script, perform the following steps:

1. Open a terminal window in the 'CS4402-P1-Solitaire/src/' directory
2. Run the following comand:

```
./run.sh <parameter file name>
```

For example:

```
./run.sh LBS9_0.param
```

The Savile Row directory has been omitted from this submission for obvious reasons. If the reader should desire to run any examples using the implemented Essence Prime model, they should include the Savile Row directory in the 'CS4402-P1-Solitaire/' directory.

3 Design Decisions

The Essence Prime model implemented for this practical is contained within the 'lbs.eprime' file in the 'CS4402-P1-Solitare/src/' directory.

3.1 Variables and Domains

To model the sequences of piles of cards after each move (i.e. to model the decision variables), a 2D matrix of integers was used. The matrix is represented explicitly as each element in the matrix corresponds directly to a playing card from 0 to 51 (or -1 for no card). Each of these cards represents the top card of a pile in the sequence. After a move is made using the current sequence of piles, the rightmost non-empty pile is replaced by -1, marking that pile as empty, and one of the cards (referred to as the source card) takes the position of the (destination) card which is no longer present in the sequence. The first row of the matrix is the same as the list of piles provided in supplied parameter file.

This design decision was made as the specification stated that this sort of problem is a planning problem, and is typically modelled using a sequence of moves. It was also useful to be able to debug the produced solution by visualising the sequence of piles after each move.

3.2 Constraints

Each of the following constraints correspond to a constraint in the 'lbs.eprime' file (which have been numbered for ease of comparison in the code comments):

1. The sequence of piles in the first row of the sequences matrix must match the cards provided by the parameter file.
2. Each additional sequence in the sequences matrix has an additional empty pile appearing where the rightmost non-empty pile of the previous sequence occurred. The empty piles are denoted by -1.
3. (a) For each sequence (except the final sequence)
   (b) there exists a source pile (which is not the first pile in the sequence)
   (c) and a destination pile (which is either one or three piles to the left of the source pile)
   (d) where the value at the destination index of the subsequent sequence is equal to the value at the source index of the current sequence.
   (e) In addition, the value at the destination and source indices in the current sequence must either be the same rank
   (f) or the same suit
   (g) Finally, for all other piles in the sequence
   (h) each value not at the source or destination indices either shifts down by one index in the next sequence (if its index is greater than the source index)
   (i) or stays at the same index in the next sequence (if its index is less than the source index).

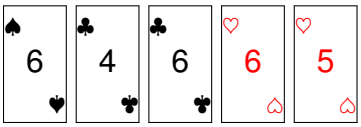
4 Example Solutions

The following are the resulting sequences of cards produced by running Savile Row with the essence prime model and the provided examples, LBS5_0, LBS7_47, and LBS9_0 respectively:

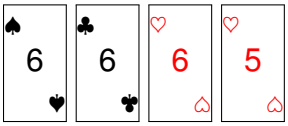
```
language ESSENCE' 1.0
$ Minion SolverNodes: 2
$ Minion SolverTotalTime: 0.034374
$ Minion SolverTimeOut: 0
$ Savile Row TotalTime: 0.413
letting seqs be [[44, 29, 31, 5, 4],
[44, 31, 5, 4, -1],
[31, 5, 4, -1, -1],
[5, 4, -1, -1, -1],
[4, -1, -1, -1, -1]]
```

Figure 1: Sequences of cards produced by running the essence prime model with LBS5_0.param

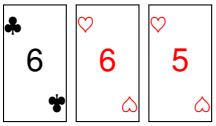
4.1 LBS5_0



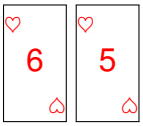
Move the 6 of clubs on top of the 4 of clubs:



Move the 6 of clubs on top of the 6 of spades:



Move the 6 of hearts on top of the 6 of clubs

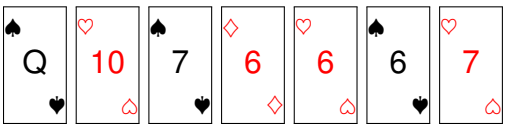


Move the 5 of hearts on top of the 6 of hearts:

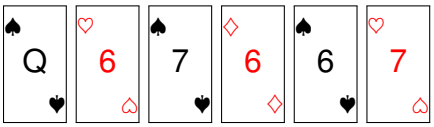
```
language ESSENCE' 1.0
$ Minion SolverNodes: 73
$ Minion SolverTotalTime: 0.039795
$ Minion SolverTimeOut: 0
$ Savile Row TotalTime: 0.289
letting seqs be [[50, 9, 45, 18, 5, 44, 6],
[50, 5, 45, 18, 44, 6, -1],
[50, 5, 6, 18, 44, -1, -1],
[50, 5, 6, 44, -1, -1, -1],
[44, 5, 6, -1, -1, -1, -1],
[5, 6, -1, -1, -1, -1, -1],
[6, -1, -1, -1, -1, -1, -1]]
```

Figure 2: Sequences of cards produced by running the essence prime model with LBS7_47.param

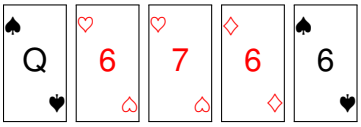
4.2 LBS7_47



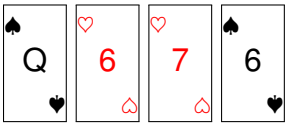
Move the 6 of hearts on top of the 10 of hearts:



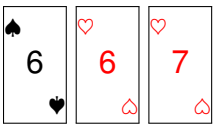
Move the 7 of hearts on top of the 7 of spades:



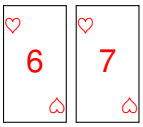
Move the 6 of spades on top of the 6 of diamonds



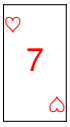
Move the 6 of spades on top of the queen of spades:



Move the 6 of hearts on top of the 6 of spades:



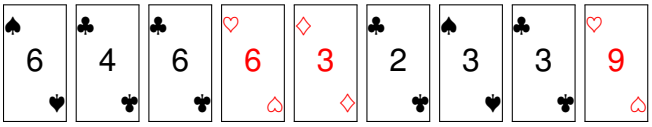
Moves the 7 of hearts on top of the 6 of hearts:



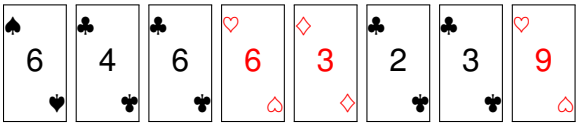
```
language ESSENCE' 1.0
$ Minion SolverNodes: 3213
$ Minion SolverTotalTime: 0.057226
$ Minion SolverTimeOut: 0
$ Savile Row TotalTime: 0.319
letting seqs be [[44, 29, 31, 5, 15, 27, 41, 28, 8],
[44, 29, 31, 5, 15, 27, 28, 8, -1],
[44, 29, 31, 5, 15, 28, 8, -1, -1],
[44, 29, 31, 5, 28, 8, -1, -1, -1],
[44, 28, 31, 5, 8, -1, -1, -1, -1],
[44, 31, 5, 8, -1, -1, -1, -1, -1],
[31, 5, 8, -1, -1, -1, -1, -1, -1],
[5, 8, -1, -1, -1, -1, -1, -1, -1],
[8, -1, -1, -1, -1, -1, -1, -1, -1]]
```

Figure 3: Sequences of cards produced by running the essence prime model with LBS9_0.param

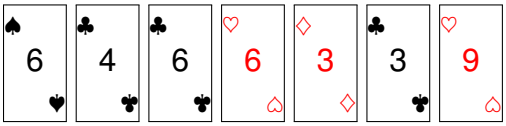
4.3 LBS9_0



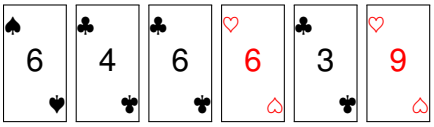
Move the 3 of clubs on top of the 3 of spades:



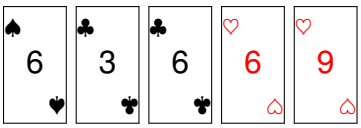
Move the 3 of clubs on top of the 2 of clubs:



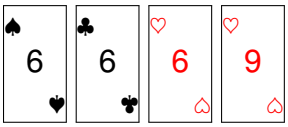
Move the 3 of clubs on top of the 3 of diamonds:



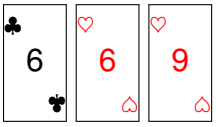
Move the 3 of clubs on top of the 4 of clubs:



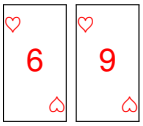
Move the 6 of clubs on top of the 3 of clubs:



Move the 6 of clubs on top of the 6 of spades:



Move the 6 of hearts on top of the 6 of clubs:



Move the 9 of hearts on top of the 6 of hearts:

5 Empirical Evaluation

5.1 Experimental Setup

There is little to say about the experimental setup used for this coursework assignment. The solvable and unsolvable files were run for 3 repeat readings by hand using Savile Row and the implemented Essence Prime model *lbs.eprime*. The results for each were recorded from the relevant .info file into the spreadsheet *results.ods*, and the average values of these repeat readings were calculated for use in this report.

5.2 Provided Examples

The model produced a solution for all of the solvable instances, and produced no solutions for any of the unsolvable instances. The observed number of solver nodes, solver solve times, and Savile Row total times are recorded below, for both the solvable and unsolvable instances:

5.2.1 Solvable

Parameter File	Solver Nodes	Solver Solve Time (s)	Savile Row Total Time (s)
LBS4_13	1	0.0466	0.166
LBS4_27	3	0.0437	0.155
LBS4_33	3	0.0468	0.225
LBS5_0	2	0.0444	0.211
LBS5_30	4	0.0613	0.174
LBS5_44	3	0.0422	0.184
LBS7_47	73	0.0508	0.259
LBS8_47	184	0.0475	0.336
LBS9_0	3213	0.0696	0.393
LBS9_12	93	0.0547	0.410
LBS9_16	10259	0.100	0.382
LBS9_21	6635	0.0918	0.441
LBS9_40	631	0.0682	0.392
LBS9_41	206	0.0457	0.365
LBS10_12	467	0.0526	0.496
LBS10_16	201	0.0505	0.479
LBS10_19	1382	0.0593	0.522
LBS10_21	30176	0.247	0.467
LBS10_39	11270	0.124	0.533

Figure 4: Results of running Savile Row with the Essence Prime model and the solvable parameter instances

5.2.2 Unsolvable

Parameter File	Solver Nodes	Solver Solve Time (s)	Savile Row Total Time (s)
LBS6_12	0	0.00000567	0.196
LBS6_20	571	0.00200	0.254
LBS7_20	518	0.0125	0.326
LBS7_21	327	0.00304	0.294
LBS8_20	3113	0.0179	0.357
LBS8_8	3885	0.0338	0.354

Figure 5: Results of running Savile Row with the Essence Prime model and the unsolvable parameter instances

5.2.3 Discussion

From figures 4 and 5, we can make some general statements about the nature of the Late Binding Solitaire problem and how Savile Row uses the implemented model to solve (or not solve) provided problem instances. In general we can say that as the length of the provided sequence of cards, n, increases, the number of nodes used by the Savile Row to find a solution (or no solution) increases. We can also say that, in general, the solve time, and total time also increase given an increase in the length of n.

For example, the solvable length 4 and 5 problem instances use between 1 and 4 nodes, and each have a solve time of between $\approx$0.04-0.06s. The Savile Row total times for these instances are between $\approx$0.15-0.2s. These solutions clearly take less time, and use fewer nodes than the problem instances of length 9 or length 10, which use between 93 and 10259, and 201 and 30176 nodes respectively. The length 9 and 10 solutions also have solve times anywhere from just under 0.05s to just under 0.25s, and have Savile Row total times of just over 0.35s to just over 0.5s.

However, this is not strictly true for all problem instances, and the difficulty of a solution must also be accounted for. If we take the length 9 and 10 instances *LBS9_16* and *LBS10_16* as examples, we can observe a huge disparity in the number of nodes used and time taken to solve each instance. With the length 9 instance using 10,259 nodes, with a solve time of a tenth of a second, where as the length 10 instance only uses 201 nodes, and has a solve time of about half that of the former instance, of 0.0505s.

Finally, the general increasing trend for the number of nodes and solve time as n is increased appears to hold true for the unsolvable instances as well. The length 8 solutions use more nodes than the length 6 and 7 instances and also take more time (solver time and Savile Row total time).

Again we can observe that instances with a lower value of *n* also sometimes take longer than instances with a larger value of *n*, presumably due to the difficulty of the problem. In this case, the file *LBS6_20* uses 571 nodes, as opposed to the 518 and 327 nodes used by the length 7 instances. However it actually took less time to solve than both of the length 7 instances. Perhaps this is due to some optimisation performed by Minion during the search process. It is conceivable that the total number of nodes may not all be traversed, for example in the case of symmetry where branches may be pruned, and so Minion may save time not having to execute certain branches of the search tree to find a solution.

5.3 Generated Examples

The following instances were generated using the provided Java program. As the longest length example provided in the solvable solutions is 10, instances have been generated starting at length 11. At each length, examples were generated starting at a seed of 0, increasing the value of the seed if the instance was unsolvable or timed out within 10 minutes. As soon as a solvable instance was found at a particular length, the node number, solve time, and Savile Row total time were recorded and then testing proceeded using the next length. From lengths 11 to 17, solvable solutions were able to be found without any instances timing out. The first two instances of length 18 timed out (seeds 0 and 1), as did the first two instances of length 19, 20, and 21, but solutions were found for these lengths using the seed 2. The first instance of length 22 timed out (seed 0), but a solution was found for a seed value of 1. At this point, running instances with an increasing value for *n* was taking too long and time was running short, so 22 was the highest value of *n* tested.

Generated Parameter File	Solver Nodes	Solver Solve Time (s)	Savile Row Total Time (s)
LBS11_9	32319	0.277076	0.514
LBS12_5	56695	0.412421	0.637
LBS13_4	5179	0.079131	0.778
LBS14_2	138633	0.919916	0.846
LBS15_0	1305849	9.88291	1.024
LBS16_0	842427	5.95109	1.114
LBS17_0	1881961	14.2455	1.161
LBS18_2	127624	0.980536	1.537
LBS19_2	71014	0.774316	1.389
LBS20_2	1474181	10.744	1.497
LBS21_2	1463439	11.1908	1.91
LBS22_1	49981176	419.457	1.764

Figure 6: Results of running Savile Row with the Essence Prime model and generated instances

5.3.1 Discussion

From the results in figure 6 it is clear that the general assertions made about the Late Binding Solitaire model hold true. That being that if the length of the problem is increased, in general, an increase in the number of solver nodes, solver solve time, and Savile Row total time is observed. Again the caveat that this is not universal must be stated, due to it being impossible to control for the difficulty of the generated instances when using the random instance generator.

It is worth observing from the results of figure 6 that some instances, even for a large value of n, ran very quickly when finding a solution. This is the case for *LBS18_2* and *LBS19_2* which both ran in less than a second. This is likely due to the generator producing a solution which required little search to find. An example of this might be an instance with an entire suit of cards present or many cards with the same rank. Even the longest valid solution here (*LBS22_1* took a total of 419s (just under 7 minutes). This is still relatively far below the specified timeout time of 10 minutes.

Without further testing at a higher timeout time this is impossible to know for certain, but these results may suggest that whenever the solver timed out during this phase of testing, the instance it was running on was unsolvable. This would make sense intuitively, as the solver would have to exhaust every possible path down the search tree to determine that no solutions were present, but finding a single solution allows the solver to stop without performing an exhaustive search.

In addition, it should be pointed out that the size of the search tree grows exponentially as the length of n is increased (assuming an exhaustive search needs to be performed), as every previous path would still need to be traversed, but now for all of the valid permutations of the newly added pile of cards. This large increase in the size of the search tree is perhaps demonstrated by the change in the time required to find a solution by *LBS21_2* ($\approx$11s) and *LBS22_1* ($\approx$ 419s). Of course, we can't make a definitive statement in this regard due to our lack of ability to control the difficulty of the instances.

6 Conclusion

In conclusion an Essence Prime specification has successfully been formulated to model Late Binding Solitaire. An empirical evaluation of the performance of this model using provided, as well as generated, instances has been carried out. The model when run using Savile Row and the generated instances seems to perform relatively well up to instances of length 22, however given more time, instances of a longer length would be experimented with to attempt to better understand the performance of this model. In addition, it would be interesting given more time to observe the effect of adding more explicit constraints to attempt to improve the model's performance as well as experimenting with

different variants of solitaire.