## 1 Analysis

Here, we analyze the three steps paper from [1]. The cards leaked to the agent $C$ are tabulated in table 1 while the deals that $C$ thinks possible after each of those runs are tabulated in Table 2.

With these tables, we are able to ascertain that our tool agrees with the analysis of [1] as it is a safe protocol and hence any run taken in isolation is also guaranteed to be safe and hence $C$ does not learn any cards.

| $nCards$ | $r_0$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ |
|---|---|---|---|---|---|---|
| 0 | 5 | 5 | 5 | 5 | 5 | 5 |

Table 1: Cards Leaked

| | $r_0$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ |
|---|---|---|---|---|---|---|
| $pDeals$ | 5 | 5 | 5 | 5 | 5 | 5 |

Table 2: Possible Deals

In the second table, we see that the number of possible deals is also 5 after every run and we can in fact prove it independently that it would be exactly 5 for each run. The above follows from the properties of the first announcement made by $A$ that always consists of 15 disjuncts of a particular kind (type of 1-designs and 2-designs).

The reasoning is that since the first announcement is such that any card appears in exactly 6 disjuncts and since $C$ holds two cards, he can first eliminate 6 hands of the announcement for card $x$ that he holds. For the other card $y$, it occurs in exactly 2 hands alongwith $x$ and hence he can eliminate 4 other disjuncts leaving him with 5 possibilities. The later announcements do not reveal any additional information to $C$ and hence he will be unsure of 5 possible deals.

## 2 Other Measures

Besides the number of deals that the Eavesdropper thinks possible, we can consider other relevant measures of uncertainty after a particular run. These measures attempt to capture the remaining uncertainty as well as vulnerabilities at a more abstract level (using atomic propositions).

For instance, we consider a proposition (or card) to be "weak" if leaking the value of that proposition to the eavesdropper results in him learning the complete deal. There are no propositions of that kind for the protocol in [1] (for the deal starting at $d_0 =< 0, 1, 2, 3 \mid 4, 5, 6, 7 \mid 8, 9 >$ and run $r_0$).

```
> KL
[Kca__0, Kca__1, Kca__2, Kca__3, Kcb__4, Kcb__5, Kcb__6, Kcb__7]
> s1 = s0.execRun(r0)
> map(len, computeRemainingDeals(s1, KL))
  [3, 3, 3, 3, 3, 3, 3, 3]
```

Thus even if we update the agent $C$'s knowledge with any of the propositions, she still considers 3 other possibilities compatible with her knowledge.

We compute the same for other deals possible after $r_0$ and we get similar property that $C$ is still unsure of the exact deal for run $r_0$. However, note that the number of alternate deals varies for different deals.

```
> alternateDeals = s1.getAgtDeals(c)
> for d in alternateDeals:
     s0.deal = d
     s1 = s0.execRun(r0)
     print map(len, computeRemainingDeals(s1, KL))

[3, 3, 3, 3, 3, 3, 3, 3]
[3, 3, 3, 3, 3, 3, 3, 3]
[3, 3, 3, 3, 3, 3, 3, 3]
[3, 3, 3, 3, 3, 3, 3, 3]
[3, 3, 3, 3, 3, 3, 3, 3]
```

However, note that not all runs generate the same number of deals when revealing a particular proposition. In fact, if we take another run of the protocol $r_1$ and compute the same measure above, we get that after revealing some propositions, the number of remaining deals might be 2 for the Eavesdropper as shown below.

```
> r1[0]
('a', [[2, 3, 7, 9],  [0, 1, 7, 9],  [2, 4, 6, 7],
       [3, 5, 7, 8],  [0, 4, 5, 7],  [1, 6, 7, 8],
       [2, 5, 8, 9],  [3, 4, 6, 9],  [0, 6, 8, 9],
       [1, 4, 5, 9],  [0, 1, 2, 3],  [0, 2, 5, 6],
       [1, 2, 4, 8],  [0, 3, 4, 8],  [1, 3, 5, 6]])

> allDeals1, remDeals1, nRem1 = nDealsRemaining(r1, KL)
> nRem1
[[3, 2, 3, 2, 3, 2, 2, 3],
 [3, 2, 3, 2, 3, 2, 2, 3],
 [3, 2, 3, 2, 3, 2, 2, 3],
 [3, 2, 3, 2, 3, 2, 2, 3],
 [3, 2, 3, 2, 3, 2, 2, 3]]
```

## 2.1 Weak Pairs

We can repeat the above experiment for weak pairs

```
> pairKL
[[Kca__0, Kca__1], [Kca__0, Kca__2], [Kca__0, Kca__3], [Kca__1, Kca__3],
 [Kca__2, Kca__3]]
```

We get the following result (for $d_0$ and run $r_0$).

```
In [1]: %time allDeals0, pairRemDeals0, nPairRem0 = nDealsRemaining(r0, pairKL)
CPU times: user 36.2 s, sys: 44 ms, total: 36.3 s
Wall time: 36.4 s

In [238]: for l in nPairRem0:
     ...:     print l
     ...:

[2, 2, 1, 1, 2, 2]
[2, 2, 1, 1, 2, 2]
[2, 2, 1, 1, 2, 2]
[2, 2, 1, 1, 2, 2]
[2, 2, 1, 1, 2, 2]

In [236]: %time allDeals0, pairRemDeals1, nPairRem1 = nDealsRemaining(r1, pairs)
CPU times: user 7.63 s, sys: 12 ms, total: 7.64 s
Wall time: 7.66 s

In [238]: for l in nPairRem1:
     ...:     print l
     ...:
[1, 2, 1, 1, 2, 1]
[1, 2, 1, 1, 2, 1]
[1, 2, 1, 1, 2, 1]
[1, 2, 1, 1, 2, 1]
[1, 2, 1, 1, 2, 1]
```

## References

[1] Hans P. van Ditmarsch, and Fernando Soler-Toscano, Three Steps. In *Proceedings of the 12th International Conference on Computational Logic in Multi-agent Systems (CLIMA'11), Barcelona, Spain, 2011.*

## A Experimental Setup

Note that the number of all hands that $A$ can have are 210 and they can be ordered in ascending order as shown in Table 4. The indices shown in that table are used in what follows.

Table 3 shows a snapshot of the runs that were used to check for leakage to agent $C$ as well as to compute the possible deals that $C$ considers possible at the end of any run.

| | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0, 26, 39, 52, 65, 82, 93, 110, 123, 136, 153, 159, 170, 181, 188 | 1, 11, 45, 58, 65, 79, 96, 113, 126, 136, 143, 153, 166, 175, 204 | 2, 7, 46, 58, 73, 76, 101, 119, 124, 133, 140, 150, 163, 189, 196 |
| $\vdots$ | $\dots$ | $\dots$ | $\dots$ |
| 11 | 12, 13, 32, 36, 75, 81, 96, 104, 114, 125, 145, 161, 178, 182, 208 | 22, 27, 35, 41, 53, 55, 92, 98, 108, 110, 151, 152, 196, 199, 209 | 3, 13, 31, 57, 71, 82, 104, 109, 114, 133, 143, 155, 169, 186, 202 |

Table 3: First Announcements for $r_k$ (where $k = 3 * i + j$)

Finally, the output obtained for card leakage for the 36 runs as well as the actual deals that $C$ considers possible at the end of each run (for any deal that can truthfully announce run) are tabulated below.

```
> ut.allHands(4, range(10))
[[0, 1, 2, 3], [0, 1, 2, 4], [0, 1, 2, 5], [0, 1, 2, 6], [0, 1, 2, 7],
 [0, 1, 2, 8], [0, 1, 2, 9], [0, 1, 3, 4], [0, 1, 3, 5], [0, 1, 3, 6],
 [0, 1, 3, 7], [0, 1, 3, 8], [0, 1, 3, 9], [0, 1, 4, 5], [0, 1, 4, 6],
 [0, 1, 4, 7], [0, 1, 4, 8], [0, 1, 4, 9], [0, 1, 5, 6], [0, 1, 5, 7],
 [0, 1, 5, 8], [0, 1, 5, 9], [0, 1, 6, 7], [0, 1, 6, 8], [0, 1, 6, 9],
 [0, 1, 7, 8], [0, 1, 7, 9], [0, 1, 8, 9], [0, 2, 3, 4], [0, 2, 3, 5],
 [0, 2, 3, 6], [0, 2, 3, 7], [0, 2, 3, 8], [0, 2, 3, 9], [0, 2, 4, 5],
 [0, 2, 4, 6], [0, 2, 4, 7], [0, 2, 4, 8], [0, 2, 4, 9], [0, 2, 5, 6],
 [0, 2, 5, 7], [0, 2, 5, 8], [0, 2, 5, 9], [0, 2, 6, 7], [0, 2, 6, 8],
 [0, 2, 6, 9], [0, 2, 7, 8], [0, 2, 7, 9], [0, 2, 8, 9], [0, 3, 4, 5],
 [0, 3, 4, 6], [0, 3, 4, 7], [0, 3, 4, 8], [0, 3, 4, 9], [0, 3, 5, 6],
 [0, 3, 5, 7], [0, 3, 5, 8], [0, 3, 5, 9], [0, 3, 6, 7], [0, 3, 6, 8],
 [0, 3, 6, 9], [0, 3, 7, 8], [0, 3, 7, 9], [0, 3, 8, 9], [0, 4, 5, 6],
 [0, 4, 5, 7], [0, 4, 5, 8], [0, 4, 5, 9], [0, 4, 6, 7], [0, 4, 6, 8],
 [0, 4, 6, 9], [0, 4, 7, 8], [0, 4, 7, 9], [0, 4, 8, 9], [0, 5, 6, 7],
 [0, 5, 6, 8], [0, 5, 6, 9], [0, 5, 7, 8], [0, 5, 7, 9], [0, 5, 8, 9],
 [0, 6, 7, 8], [0, 6, 7, 9], [0, 6, 8, 9], [0, 7, 8, 9], [1, 2, 3, 4],
 [1, 2, 3, 5], [1, 2, 3, 6], [1, 2, 3, 7], [1, 2, 3, 8], [1, 2, 3, 9],
 [1, 2, 4, 5], [1, 2, 4, 6], [1, 2, 4, 7], [1, 2, 4, 8], [1, 2, 4, 9],
 [1, 2, 5, 6], [1, 2, 5, 7], [1, 2, 5, 8], [1, 2, 5, 9], [1, 2, 6, 7],
 [1, 2, 6, 8], [1, 2, 6, 9], [1, 2, 7, 8], [1, 2, 7, 9], [1, 2, 8, 9],
 [1, 3, 4, 5], [1, 3, 4, 6], [1, 3, 4, 7], [1, 3, 4, 8], [1, 3, 4, 9],
 [1, 3, 5, 6], [1, 3, 5, 7], [1, 3, 5, 8], [1, 3, 5, 9], [1, 3, 6, 7],
 [1, 3, 6, 8], [1, 3, 6, 9], [1, 3, 7, 8], [1, 3, 7, 9], [1, 3, 8, 9],
 [1, 4, 5, 6], [1, 4, 5, 7], [1, 4, 5, 8], [1, 4, 5, 9], [1, 4, 6, 7],
 [1, 4, 6, 8], [1, 4, 6, 9], [1, 4, 7, 8], [1, 4, 7, 9], [1, 4, 8, 9],
 [1, 5, 6, 7], [1, 5, 6, 8], [1, 5, 6, 9], [1, 5, 7, 8], [1, 5, 7, 9],
 [1, 5, 8, 9], [1, 6, 7, 8], [1, 6, 7, 9], [1, 6, 8, 9], [1, 7, 8, 9],
 [2, 3, 4, 5], [2, 3, 4, 6], [2, 3, 4, 7], [2, 3, 4, 8], [2, 3, 4, 9],
 [2, 3, 5, 6], [2, 3, 5, 7], [2, 3, 5, 8], [2, 3, 5, 9], [2, 3, 6, 7],
 [2, 3, 6, 8], [2, 3, 6, 9], [2, 3, 7, 8], [2, 3, 7, 9], [2, 3, 8, 9],
 [2, 4, 5, 6], [2, 4, 5, 7], [2, 4, 5, 8], [2, 4, 5, 9], [2, 4, 6, 7],
 [2, 4, 6, 8], [2, 4, 6, 9], [2, 4, 7, 8], [2, 4, 7, 9], [2, 4, 8, 9],
 [2, 5, 6, 7], [2, 5, 6, 8], [2, 5, 6, 9], [2, 5, 7, 8], [2, 5, 7, 9],
 [2, 5, 8, 9], [2, 6, 7, 8], [2, 6, 7, 9], [2, 6, 8, 9], [2, 7, 8, 9],
 [3, 4, 5, 6], [3, 4, 5, 7], [3, 4, 5, 8], [3, 4, 5, 9], [3, 4, 6, 7],
 [3, 4, 6, 8], [3, 4, 6, 9], [3, 4, 7, 8], [3, 4, 7, 9], [3, 4, 8, 9],
 [3, 5, 6, 7], [3, 5, 6, 8], [3, 5, 6, 9], [3, 5, 7, 8], [3, 5, 7, 9],
 [3, 5, 8, 9], [3, 6, 7, 8], [3, 6, 7, 9], [3, 6, 8, 9], [3, 7, 8, 9],
 [4, 5, 6, 7], [4, 5, 6, 8], [4, 5, 6, 9], [4, 5, 7, 8], [4, 5, 7, 9],
 [4, 5, 8, 9], [4, 6, 7, 8], [4, 6, 7, 9], [4, 6, 8, 9], [4, 7, 8, 9],
 [5, 6, 7, 8], [5, 6, 7, 9], [5, 6, 8, 9], [5, 7, 8, 9], [6, 7, 8, 9]]
```

Table 4: All possible hands for A