The Russian Cards problem

Project report

Group 11: s3249212, s3749185 and s3533603

1 Introduction

For our project, we created a solver that finds solutions to variations of the Russian Cards problem. The Russian Cards problem describes a situation with three card players, called Anne, Bill, and Cath, who have been given 3, 3, and 1 numbered cards, respectively. The goal is for Anne to communicate her hand of cards to Bill via a public announcement in a way that keeps Cath ignorant of the specific cards Anne and Bill are holding.

There is no particular reason why the (3,3,1) variant should be the only form of the problem. In our research project, we set out to explore variations where a different number of cards is divided, such as (4,4,1), (2,2,1), or (5,5,2).

2 Theory

2.1 Original problem

In the original Russian cards problem, agent A (Anne) and agent B (Bill) receive three cards each, while agent C (Cath) receives one. The goal is for Anne to communicate her hand to Bill without Cath finding out Anne's hand. This must be done through truthful public announcements.

The dealing of the cards is seen as a function d that maps cards Q to players P, i.e. $d : Q \to P$. Two deals d, e are indistinguishable for a player when the number of cards in the deal $\#d$ is equivalent to $\#e$ and the player has received the same hand.

Deals can be modeled as epistemic worlds. A deal is defined to be *Russian* when all players initially only know their own cards and every card is held by exactly one player.

Since the exact cards that have been dealt to the players do not matter, the cards are simply represented by numbers. The 'true' deal can then be represented by 012.345.6, i.e. $h_A = \{0, 1, 2\}$, $h_B = \{3, 4, 5\}$ and $h_C = \{6\}$, where h_a represents the hand of player a. All other possible card deals are represented by permutations of this sequence of numbers. The fact that a player holds a specific card is denoted by q_a, where $q \in Q$ is the card and $a \in P$ is the player. For example, 3_B means that Bill holds card 3.

A description of a deal $d = 012.345.6$ is shortened with the notation

$$\delta^d = 0_A \wedge 1_A \wedge 2_A \wedge \neg 3_A \wedge ... \wedge \neg 0_B \wedge ... \wedge \neg 5_C \wedge 6_C$$

Define D to be the epistemic model. Then in each epistemic state $(D, d) \models \delta^d$. This δ^d can be restricted to a δ^d_a for a player a, such that

$$\delta^d_A = 0_A \wedge 1_A \wedge 2_A \wedge \neg 3_A \wedge \neg 4_A \wedge \neg 5_A \wedge \neg 6_A$$

Note that $\delta^d = \delta^d_A \wedge \delta^d_B \wedge \delta^d_C$.

Define $\mathcal{D}$ to be the set of all Russian card deals. The Russian card problem is solved in the epistemic state (D, d) where the following three conditions hold:

1. Anne knows the hand of Bill.

$$\mathsf{AknowsB} = \bigwedge_{e \in \mathcal{D}(D)} (\delta_B^e \to K_A \delta_B^e) \tag{1}$$

2. Bill knows the hand of Anne.

$$\mathsf{BknowsA} = \bigwedge_{e \in \mathcal{D}(D)} (\delta_A^e \to K_B \delta_A^e) \tag{2}$$

3. Cath does not know the hand of Anne, nor that of Bill.

$$\mathsf{Cignorant} = \bigwedge_{q \in Q} (\neg K_C q_A \wedge \neg K_C q_B) \tag{3}$$

A solution to the Russian Cards problem is a sequence of public announcements that changes the model such that it is common knowledge that Anne and Bill know each other's hands (or, less strictly, it is common knowledge for Anne and Bill that they know each other's hands). Each public announcement by a player a needs to have the form $K_a\phi \wedge [K_a\phi]C_{ABC}\mathsf{Cignorant}$. This is called a *safe announcement*.

Note, however, that $\mathsf{BknowsA}$ can already be ensured after Anne's first public announcement. Because Bill then knows all cards, he can simply communicate Cath's hand to ensure $\mathsf{AknowsB}$. So the sequence of public announcements always has a length of two. Only the first announcement by Anne is of interest.

2.2 Extension on distribution of cards

This project considers an extension of the Russian Cards problem that changes the constant distribution (3,3,1) to a varying number of cards distributed to each person. Let us define the number of cards that Anne, Bill and Cath have as n_A, n_B and n_C, and $n = n_A + n_B + n_C$. Then the number of possible deals d is $\binom{n_A+n)B+n_C}{n_A} \cdot \binom{n_B+n_C}{n_B} \cdot \binom{n_C}{n_C} = \frac{(n_A+n_B+n_C)!}{n_A!n_B!n_C!}$.

Cath can not distinguish the cards that Anne and Bill have. Given a hand C that Cath has received, there are initially $\binom{n_A+n_B}{n_A} = \frac{(n_A+n_B)!}{n_A!n_B!}$ worlds that Cath can not distinguish. Similarly for Anne and Bill.

2.2.1 Extension of problem definition

For a random player $p \in \{a, b, c\}$ it holds in each possible epistemic state (D, d), for an epistemic model D and a card deal d and for all cards $q \in Q$, that

$$(D, d) \models q_p \bigwedge_{r \in \{a,b,c\} \setminus p} \neg q_r$$

In other words, a card can only be assigned to one agent, and *must* be assigned to one agent exactly. The δ^d defined in section 2.1 then becomes

$$\delta^d = \bigwedge_{q \in Q} q_p \bigwedge_{r \in \{a,b,c\} \setminus p} \neg q_r$$

where $p \in \{a, b, c\}$ is the player for which q_p holds.

2.2.2 Constraints

There are a few restrictions on the input n_A, n_B, and n_C that must be satisfied for the problem to have at least one solution, though satisfying these restrictions does not *guarantee* that the problem has a solution.

The restrictions $n_A > 0$, $n_B > 0$ and $n_C > 0$ are considered to be evident.

Constraint 1.

$$n_A > n_C \tag{4}$$

Proof: Every public announcement that Anne makes to Bill is a list of hands, of which one is her true hand. For example, she could communicate $012_a \vee 023_a \vee 016_a$. To guarantee BknowsA, Anne needs to make sure that every 'decoy hand', i.e. each hand that is not her true hand, contains at least one card of Bill's. Note that 016_a is therefore not a good decoy hand, though 023_a and 026_a would be. Since she does not know Bill's cards, this means that she should have at least $C+1$ cards in each decoy hand that are not in her own hand. Since each decoy hand contains n_A cards, it needs to hold that $n_A \geq n_C + 1$, i.e. $n_A > n_C$, at least.

There is a similar constraint on n_B.

Constraint 2.

$$n_B > n_C \tag{5}$$

Proof: Since Anne needs to communicate a list of hands, of which one is her true hand, the condition Cignorant only holds if for Cath there is more than one hand that does not contain one of Cath's cards, while BknowsA only holds when for Bill there is exactly one hand in Anne's announcement which does not contain one of Bill's cards. The more cards a player has, the fewer hands in the announcement a player considers a possibility for Anne's hand. Therefore, it should hold that $n_B > n_C$.

3 Algorithm

To find a solution to the extended Russian Card Problem, we only need to find a list of hands that Anne communicates to Bill. This is due to the following reason.

For all decoy hands $h \in H^{decoy} \subseteq Q_A^n$, we define $\phi = h_a \vee (\bigvee_{h \in H^{decoy}} h)$. The first action is that Anne gives the public announcement $K_A\phi \wedge [K_A\phi]C_{ABC}$Cignorant, as mentioned in section 2.1. Moreover, as said in the same section, we know then that Bill knows all cards, and therefore the second public announcement by Bill is simply Cath's hand, after which all three conditions of the Russian Card Problem hold. So the only thing we need to solve is ϕ.

The algorithm performs a **depth-first search** over the sets of possible hands. It uses two rules to determine for each set of hands if this could lead to a valid solution.

Rule 1.

Any pair of hands in the statement share at most $n_A - n_C - 1$ cards.

Proof: Anne wants to guarantee that Bill knows her cards after her announcement. For this to happen, Bill needs to be able to rule out all the decoy hands that Anne included in the set of options. After he ruled out all the decoys, he has one hand remaining: Anne's true hand.

Bill can rule out a decoy hand if that hand includes one of his cards, as he knows the card can not be held by him and Anne at the same time. Therefore, Anne wants every decoy to have at least one card of Bill.

However, she does not know which cards that do not belong to her are Bill's and which are Cath's. Thus she needs to ensure that in the worst-case scenario - when a decoy hand contains all of Cath's cards - the decoy still has one card of Bill. Therefore, she has to make sure there number of cards that do not belong to her in a decoy is (at least) one more than the number of cards Cath has.

If there is a minimum to the number of cards that are not Anne's in a decoy, that means there is also a maximum of cards that *do* belong to Anne, which is the total number of cards in a decoy minus that minimum.

The decoys are the same size as Anne's real hands so the maximum number of Anne's cards in a decoy is: *Anne's hand size - (Cath's hand size + 1)*.

Now, you might be saying: good explanation but the rule actually says something different.

This is true and due to Cath's presumed knowledge of the protocol. Cath knows that Anne's true hand can have at most *Anne's hand size - (Cath's hand size + 1)* cards in common with any other option in the set.

So, when she sees two options with more than that number of cards in common, she knows neither can be Anne's true hand. She can cross off both hands from her list of options. It could well be that, if only one of the two conflicting options were included, Cath could not have crossed it off. So, by adding one card, Cath can scratch off two; Cath's insecurity might actually decrease by adding more decoys!

To prevent this, we set as a rule that no hand can have more than *Anne's hand size - (Cath's hand size + 1)* cards in common with any other in the list of options.

This rule is used to check if the decoy hand, which is to be added by the algorithm at a specific node in the tree, satisfies this condition. The algorithm creates a table of all pairs of cards as rows and one column, which contains

Note that due to constraint 1, $n_A - n_C - 1 \geq 0$.

Rule 2.

> For any combination of n_C cards, for any card not in the combination,
> there exists a hand in which the card is present and none of the combination's cards is present and
> there exists a hand in which the card is not present and none of the combination's cards is present.

Proof: After Cath crosses off the hands that include cards that belong to her, she is not allowed to know the distribution of any of the other cards.

The cards that Cath cannot cross off are the ones in which none of Cath's cards are present.

Not knowing the owner of a certain card means that out of the hands she could not cross off, at least one does include that card and at least one does not.

This gives rise to the following rule:

> For any card not in Cath's hand,
> there exists a hand in which the card is present and none of the Cath's cards is present and

there exists a hand in which the card is not present and none of the Cath's cards is present.

Since Anne does not know Cath's cards, we could change the rule to:

For any combination that Anne considers a possible hand of Cath,
for any card not in the combination,
there exists a hand in which the card is present and none of the combination's cards is present and
there exists a hand in which the card is not present and none of the combination's cards is present.

However, similarly to the initial version of Rule 1, this can give Cath some information about which hands Anne does not have. Cath would know that Anne would make sure the rule holds, so if the rule does not hold for one of the hands Cath considers possible for Anne, she can deduce that that card cannot be Anne's hand.

Therefore, we need to make sure that the rule holds for any possible hand Anne might have, which in practice comes down to Rule 2.

This formulation causes some duplicate requirements. Namely, both when Cath's hand is [6,7] and the card not in Cath's hand is 0, and when Cath's hand is [0,6] and the card not in Cath's hand is 7, there is a requirement that there is a hand that does not include 0 nor 6 nor 7.

This is why the rule is worked out slightly differently in the code. It removes the duplications. The rule is formulated in this way for the report because it makes more intuitive sense and has the same outcome.

Search for a Solution

The program goes over all possible decoy hands, adding a hand when Rule 1 allows, i.e. when it is different enough from all of the hands already in the set. When it is added, Rule 2 is checked to see if there is enough insecurity to guarantee cignorant. If so, we have a solution; if not, the program moves on to the next possible decoy. If it has no possible decoys left to add and Rule 2 is not yet satisfied, it backtracks by removing the last added hand. It will loop over the possible hands again, with the one difference that it does not include that hand this time.

It will keep doing this - looping through the hands in order up to and including the last hand (for 3-3-1 is this 456 for example), and backtracking - until either it finds a solution that satisfies Rule 2 or it tries to remove A's true hand from the set of options. The former means a solution is found; the latter means no solution could be found that includes A's true hand and therefore that no solution exists to the posed problem.

4 Results

We ran the program with various inputs. First, we will study the class of problems where $n_a = n_b$, like in the default distribution. Results are mapped on in the table below for $1 < n_a < 6$ and $0 < n_c < 5$, with a Y denoting a solution exists, and a N denoting no solution exists. Squares where the condition $n_a > n_c$ fails to hold are left blank.

$n_a \setminus n_c$	1	2	3	4
2	N			
3	Y	N		
4	Y	N	N	
5	Y	Y	N	N

To show the typical input and output for such a run, we present an example. The four parameters used for input are here 4, 4, 1, and 0123, denoting that agents A and B both receive four cards, agent C receives one, and A's hand is 0123.

The output, edited for readability, is: (0123, 0145, 0167, 0246, 0257, 0347, 0356, 1247, 1256, 1346, 1357, 2345, 2368, 4578).

This is followed by a list of the hands that B and C could possibly have (including possibilities that A considers false, such as C having 0), and represents their uncertainty over A's hand by listing all hands that they consider possible for A to have. For example, if B had hand 5678, he would consider only 0123 a possible hand for A to have. If C had hand 4, he would consider 0123, 0167, 0257, 0356, 1256, 1357, and 2368 all possible. Combining the results for every possible hand B and C might have revealed that *BknowsA* and *Cignorant* are both true.

Note the diagonal, representing all problems where $n_a = n_b = n_c + 1$, is uniformly unsolvable. At first glance, one might be surprised to see this: do A and B not have more information than C? Is there no theoretically possible announcement that lets A and B figure out each other's hands, with C not having enough information to figure it out? The answer is: yes, but A and B cannot make this announcement.

Assume that in such a situation, A would propose a decoy hand containing one of her actual cards, and $n_a - 1$ cards that are not in her hand. However, at this point A has no guarantee that these $n_a - 1$ cards will contain at least one card that is present in B's hand. After all, $n_a - 1 = n_c$, so the remaining $n_a - 1$ cards could, by chance, be the exact contents of C's hand. B would not be able to differentiate this hand from the true hand, and a direct exchange would fail.

In other words, if $n_a = n_b = n_c + 1$, A may not add cards from her own hand to decoy hands. That means that an announcement either contains A's hand and some number of hands that C does not consider possible, or it contains A's hand, B's hand, and some number of hands that C does not consider possible. In the first case, it is obvious that Cignorant fails to hold. In the second case, this update fails for the same reason that $(012_a \vee 345_a)$ fails in the original problem: the knowledge that A's announcements attempt to preserve Cignorant.

We now explore situations where $n_a \neq n_b$. In the first table, assume that C is always dealt a single card. Note that the table is not symmetric over the diagonal, because it assumes that the first announcement will be made by A.

$n_a \setminus n_b$	2	3	4	5	6
2	N	N	N	N	N
3	N	Y	Y	Y	Y
4	Y	Y	Y	Y	Y
5	N	Y	Y	Y	Y
6	Y	Y	Y	Y	Y

Situations where $n_a = n_c + 1$ are impossible, as we proved before. Situations where $n_a \geq 3$ and $n_b \geq 3$ seem to always be solvable if $n_c = 1$, which agrees with our intuitions on the matter.

5 Conclusion

We find that for many variations of the original problem, no solution exists. The (3,3,1) version is the simplest (that is, with the least overall cards) possible solvable version of the problem where each player receives at least one card. We find that variations such as (4,4,1) are similarly solvable, but other configurations like (2,2,1) are not. We specifically prove the impossibility of solving a variation of the form (x,x,x-1). We find that variations where c receives two or more cards are

possible and solvable, citing (5,5,2) as an example. We also make the interesting discovery that a subset of problems is solvable if A makes the first announcement but unsolvable if B does so.

In addition to those specific findings, we present an algorithmic solver that can generate direct exchange solutions, where those exist, for variations of the Russian cards problem.

Appendix

Output for original Russian Cards problem

Below is the complete output of `dfs_approach.py 3 3 1`.

```
solution found! (dfs approach)
        [0, 1, 2]
        [0, 3, 4]
        [0, 5, 6]
        [1, 3, 5]
        [1, 4, 6]
        [2, 3, 6]

Possible hand of B 0: [0, 1, 2]
no options left

Possible hand of B 1: [0, 1, 3]
no options left

Possible hand of B 2: [0, 1, 4]
[2, 3, 6]

Possible hand of B 3: [0, 1, 5]
[2, 3, 6]

Possible hand of B 4: [0, 1, 6]
no options left

Possible hand of B 5: [0, 2, 3]
[1, 4, 6]

Possible hand of B 6: [0, 2, 4]
[1, 3, 5]

Possible hand of B 7: [0, 2, 5]
[1, 4, 6]

Possible hand of B 8: [0, 2, 6]
[1, 3, 5]

Possible hand of B 9: [0, 3, 4]
no options left

Possible hand of B 10: [0, 3, 5]
[1, 4, 6]

Possible hand of B 11: [0, 3, 6]
no options left

Possible hand of B 12: [0, 4, 5]
[2, 3, 6]

Possible hand of B 13: [0, 4, 6]
[1, 3, 5]

Possible hand of B 14: [0, 5, 6]
no options left

Possible hand of B 15: [1, 2, 3]
[0, 5, 6]

Possible hand of B 16: [1, 2, 4]
[0, 5, 6]

Possible hand of B 17: [1, 2, 5]
[0, 3, 4]

Possible hand of B 18: [1, 2, 6]
[0, 3, 4]

Possible hand of B 19: [1, 3, 4]
[0, 5, 6]

Possible hand of B 20: [1, 3, 5]
no options left

Possible hand of B 21: [1, 3, 6]
no options left

Possible hand of B 22: [1, 4, 5]
[2, 3, 6]

Possible hand of B 23: [1, 4, 6]
no options left

Possible hand of B 24: [1, 5, 6]
[0, 3, 4]

Possible hand of B 25: [2, 3, 4]
[0, 5, 6]

Possible hand of B 26: [2, 3, 5]
[1, 4, 6]

Possible hand of B 27: [2, 3, 6]
no options left
```

```
Possible hand of B 28: [2, 4, 5]
no options left

Possible hand of B 29: [2, 4, 6]
[1, 3, 5]

Possible hand of B 30: [2, 5, 6]
[0, 3, 4]

Possible hand of B 31: [3, 4, 5]
[0, 1, 2]

Possible hand of B 32: [3, 4, 6]
[0, 1, 2]

Possible hand of B 33: [3, 5, 6]
[0, 1, 2]

Possible hand of B 34: [4, 5, 6]
[0, 1, 2]

Possible hand of C 0: [0]
[1, 3, 5]
[1, 4, 6]
[2, 3, 6]

Possible hand of C 1: [1]
[0, 3, 4]
[0, 5, 6]
[2, 3, 6]

Possible hand of C 2: [2]
[0, 3, 4]
[0, 5, 6]
[1, 3, 5]
[1, 4, 6]

Possible hand of C 3: [3]
[0, 1, 2]
[0, 5, 6]
[1, 4, 6]

Possible hand of C 4: [4]
[0, 1, 2]
[0, 5, 6]
[1, 3, 5]
[2, 3, 6]

Possible hand of C 5: [5]
[0, 1, 2]
[0, 3, 4]
[1, 4, 6]
[2, 3, 6]

Possible hand of C 6: [6]
[0, 1, 2]
[0, 3, 4]
[1, 3, 5]
```