Small Steps in Heuristics for the Russian Cards Problem Protocols

Ignacio Hernández-Antón

Dept. Philosophy, Logic and Philosophy of Science
University of Seville (Spain)
iha(at)us.es

Abstract. This work[1] presents a couple of algorithmic techniques applied to the Russian Cards Problem. This problem represent an idealized scenario where Dynamic Epistemic Logic [4, 5, 8] plays an important role in secure communications analysis. This logic is in a *lower layer* below the protocol desing tasks acting as a specification and verification formal tool. This work focusses not on the logical aspects but rather on the protocol design/searching problem. It is important to have present the epistemic formal notions of that logic to fully understand the epistemic functions developed here. Secure protocols found in this card game scenario may be a good starting point for developing some aspects towards unconditional information security. Hill-Climbing and Genetic Algorithms are studied as searching techniques aimed to find optimal announcements that can be part of a secure communication protocol. Some problems as the dimension and complexity of the search space are pointed out.

Keywords: Evolutionary computing, genetic algorithm, epistemic protocols, information security.

1 Introduction

We present an algorithmic approach to search unconditionally secure protocols of communicating agents within the well-known Russian Cards Problem scenario. In public/private key approaches as AES (Advanced Encription Standard), RSA (Rivest, Shamir, Adleman technique), DSA (Digital Signature Algorithm) or ECC (Elliptic Curve Cryptography), secret information if safeguarded because of the high complexity of computational operations to decrypt the message, for instance, RSA uses the IFD, the Integer Factorization Problem [10, 11]. Instead of the computational hardness for security assurance, we focus on an information-based approach to protocol design in this work. We model the communicating agents as cards players and the communicating secret is the ownership of the cards in the game. In this scenario it is possible to define good protocols regardless of the computational complexity of cryptographic techniques [4, 5, 7].

[1] This work shares some content with [9] but add the Hill-Climber section and some conlusion remarks important to understand the nature of the search and the solution space.

R.K. Rendsvig and S. Katrenko (Eds.): *ESSLLI 2012 Student Session Proceedings*, CEUR Workshop Proceedings vol.: see http://ceur-ws.org/, 2012, pp. 43–53.

We just study the security aspects of communication in order to avoid eavesdropping; other models of attacker are intentionally ommited here, we are aware of the importance of considering them for future works in order to gain robustness though. We study how to guarantee the privacy of the message which should only be shared by those principal agents we legitimated. This will occur regardless of other agents listening passively to the information passed. There is a logical approach where this problem is formalized using dynamic epistemic logic [4, 5].

We employ genetic algorithms to search for card deal protocols [12, 16, 17]. Genetic algorithms are a bio-inspired family of computational techniques which have natural evolution as a model for encoding some critical aspects of solutions as chromosomes-like data structures [3, 15]. An initial population is transformed by genetically inspired operations in order to produce new generations. The most important ones are **selection** (of the fittest), **crossing-over**, and **mutation**, the latter ones add variety to the search producing the exploration of the solution space. We use JAVA to specify the russian cards problem and a genetic engine called *jgap* to search for protocols. For further details on the genetic engine see `http://jgap.sourceforge.net/`.

2 The Russian Cards Problem

> *From a pack of seven known cards (for instance 0-6) two players (a, b) each draw three cards and a third player (c) gets the remaining card. How can the two first players (those with three cards) openly (publicly) inform each other about their cards without cyphering the messages and without the third player learning from any of their cards who holds it?*

Although this presentation of the problem has 7 cards in the stack and the deal distribution is 3.3.1 (agent a draws three cards, b draws three and c draws just one), one may consider other scenarios, e.g., a 10-cards stack with a 4.4.2 deal. To become familiar with a basic game scenario, let us call agents a, b and c. The cards are named 0, ..., 6. Deals distributions (size) are noted as integer strings, for instance 3.3.1. Legitimated principals are a and b while c is the eavesdropper. We suppose the actual deal is 012.345.6 w.l.o.g. Communication is done by truthful and public announcements, see [13, 14]. A public announcement for an agent a is a set of a's possible set of hands (we use a simplified notation to denote set of hands, e.g., $\{012, 125, 156\}$ instead of $\{\{0,1,2\},\{1,2,5\},\{1,5,6\}\}$).

A secure announcement in this scenario should keep c ignorant throughout the entire communication and guarantee common knowledge of this agent's ignorance. According to that approach, a good protocol comprises an announcement sequence that verifies that:

Informativeness 1: Principals a and b know each other cards.
Informativeness 2: It is common knowledge, at least for the principals, that they do know each other's cards.
Security 1: The intruder, c, remains ignorant always.
Security 2: It is common knowledge for all agents that the intruder remains ignorant.
Knowledge-based: Protocol steps are modelled as public announcements.

The reason we split the informativeness and security requirements into two parts can be found in [5]. A protocol is then a finite sequence of instructions determining sequences of announcements. Each agent chooses an announcement conditional on that

agent's knowledge. The protocol is assumed to be common knowledge among all agents following Kerckhoff's principle.

One knowledge-based protocol that consitutes a solution for the riddle is as follows. Suppose that the actual deal of cards is that agent a has $\{0, 1, 2\}$, b has $\{3, 4, 5\}$ and c has $\{6\}$.

- a says: My hand is one of $\{012, 046, 136, 145, 235\}$.
- Then, b says: c's card is 6.

After this, it is common knowledge to the three agents that a knows the hand of b, that b knows the hand of a, and that c is ignorant of the ownership of any card not held by itself. For further details on the notion of common knowledge see [6].

We can also see these two sequences as the execution of a knowledge-based protocol. Given a's hand of cards, there is a (non-deterministic) way to produce her announcement, to which b responds by announcing c's card. The protocol is knowledge-based, because the agents initially only know their own hand of cards, and have public knowledge of the deck of cards and how many cards each agent has drawn from the pack. It can be viewed as an *unconditionally secure* protocol, as c cannot learn any of the cards of a and b, no matter their computational resources. The security is therefore not conditional on the high complexity of some computation.

3 Russian Hill-Climbers

Let's begin with a simple metaheuristic technique called Hill-Climbing. It is a very simple algorithm that give us the oportunity to study the primitive behaviour of the problem representation and the nature of the search space. The Genetic Algorithms can be partially viewed as a sophistication of hill-climbing as optimization technique. The latter constitutes a point of reference because, being simpler, it can give clues about the need of using more sophisticated (hence, more resource-consuming) algorithms. Hill-Climbing is related to gradient ascent, but it does not require you to know the strength of the gradient or even its direction: you just iteratively test new candidate solutions in the region of your current candidate and adopt the new ones if they are better. This enables you to climb up the hill until you reach a local optimum. To have a first flavor about how this technique would work on the russian card problem, we implemented a specific Hill-Climber (see Algorithm 1) with the next parameters:

– 100 executions over 3.3.1.
– 100000 evaluations allowed

Algorithm 1 Russian Hill-Climber 1 (RHC_1)

1: $ANN \leftarrow$ All-hand-in initialization
2: **repeat**
3: $ANN' \leftarrow BFM_1(ANN)$
4: **if** $Quality(ANN') > Quality(ANN)$ **then**
5: $ANN \leftarrow ANN'$
6: **end if**
7: **until** 100000 generations or good protocol found
8: **return** $\boldsymbol{x}$

Algorithm 2 Single Bit Flip Mutator (BFM_1)

1: $h \leftarrow Random(PossibleHandsForANN)$
2: **if** $h \in ANN$ **then**
3: $ANN' \leftarrow$ Remove h from ANN
4: **else**
5: $ANN' \leftarrow$ Include h in ANN
6: **end if**
7: **return** ANN'

The initialization chosen for RHC_1 produces an announcement with all hands in and a single Bit Flip Mutator BFP_1 described in Algorithm 2. A random hand from all possible hand to be included into the announcement is selected to proceed as the `if` conditional states removing it if present or including it otherwise.

After 100 executions for 3.3.1 we obtained the next results: the average number of protocol evaluations (in red) was 39288.16 with a minimum of 1716 and a maximum of 130321. Find attached the whole experiment graphics in Figure 1, see in the horizontal axis the excution number and in the vertical one the generation where the RHC_1 found a global optimal solution.

4 Modelling cards protocols with Genetic Algorithms

The final objective of modelling cards protocols with genetic algorithms is to find protocols for card deal sizes where an analytic solution is lacking. We reinvestigate protocols for 3.3.1 with genetic algorithms in order to study the behaviour of the relationship between the problem representation and the fitness function used. This way we will be able to polish them for more complex scenarios. The use of genetic algorithms requires to satisfy two conditions: possible solutions can be represented by chromosomes and an evaluation function can be defined in order to assign a value to each chromosome. Regarding this encoding representation we observe that the set of possible hands of an agent can be arranged in lexicographic order, e.g., for Russian

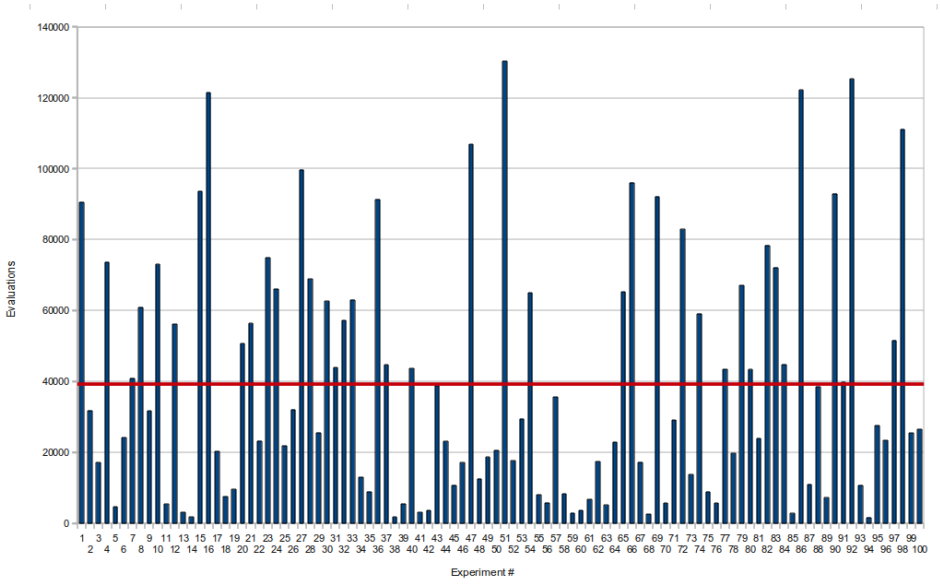
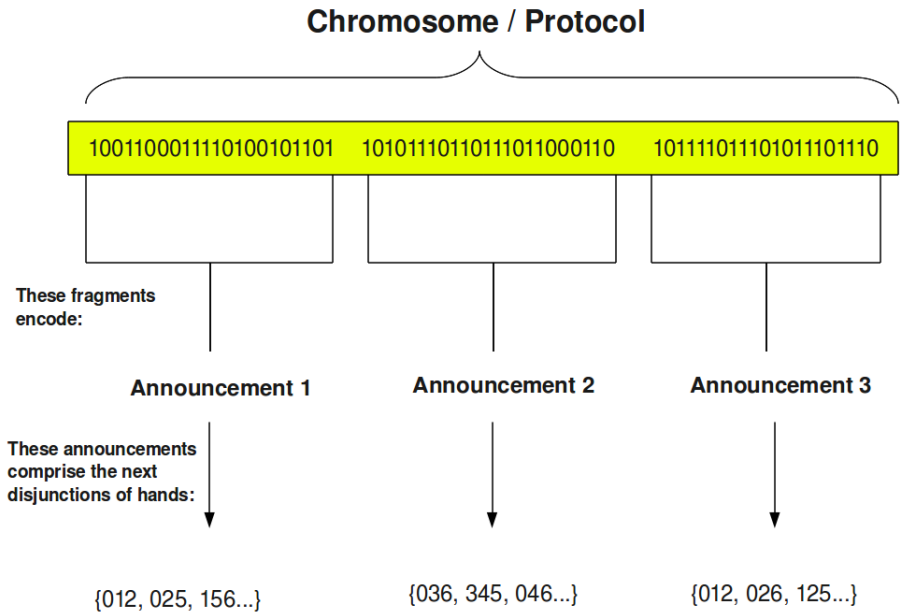


Fig. 1. 100 runs of RHC_1 over 3.3.1; BFM_1

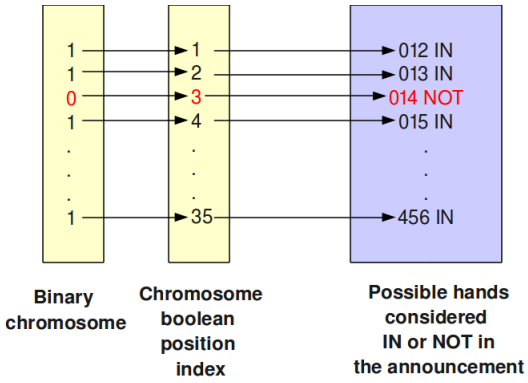
Cards, the 35 hands are listed as $012, 013, \ldots, 456$. Then we assign a binary gene to each possible hand. An announcement is then represented by a 35-bitstring. To illustrate a mapping like this see Figure 2 (a), we can encode several announcements into one chromosome as in Figure 2 (b) representing this way an composite protocol. In the 3.3.1 case as it is proved that there is a minimal protocol where just one announcement is needed, we restrict the encoding representation to just agent a announcement, but if several announcements are required, this representation is flexible enough to be adapted to it.

Algorithm 3 Genetic algorithm schema

1: $t \leftarrow 0$
2: Population initialization $P(0) : \{p_1(0), \ldots p_k(0)\}$
3: Evaluation of $P(0)$: $\Phi(p_1(0)), \ldots, \Phi(p_k(0))$
4: **while** $\theta(P(t) \neq$ true) **do**
5: Recombination $P'(t) \leftarrow r(P(t))$
6: Mutation $P'(t) \leftarrow m(P'(t))$
7: Evaluation $P'(t) : \Phi(p'_1(0)), \ldots, \Phi(p'_m(0))$
8: Selection $P(t+1) \leftarrow s(P'(t))$
9: **end while**
10: **return** $P(t+s)$



(a) Mapping announcements



(b) Allele decoding

Fig. 2. Chromosome general structure

To evolve the population a fitness function assigns to each announcement a value. As the protocols are knowledge-based, this function needs some epistemic aspects to be considered (implemented). As we are looking for a two-step protocol [2], we only need one announcement. The fitness function evaluates possible announcements. Note that if D is the set of cards, these announcements are elements of $\mathcal{P}(\mathcal{P}(D))$ with certain properties.

The first epistemic function is $compCardsGivenAnnounce(Ann, Hand)$ which returns, for a given announcement Ann, the set of possible hands that an agent having

$Hand$ considers for the agent making the announcement:

$$compCardsGivenAnnounce : \mathcal{P}(\mathcal{P}(D)) \times \mathcal{P}(D) \mapsto \mathcal{P}(\mathcal{P}(D))$$
$$compCardsGivenAnnounce(Ann, Hand) = \{h \in Ann \mid h \cap Hand = \emptyset\}$$

Now we define $whatAgentLearnsFromAnnounce(Ann, Hand)$ that returns the set of cards that an agent having $Hand$ learns from the agent. The function is defined as:

$$whatAgentLearnsFromAnnounce : \mathcal{P}(\mathcal{P}(D)) \times \mathcal{P}(D) \mapsto \mathcal{P}(D)$$
$$whatAgentLearnsFromAnnounce(Ann, Hand) = \bigcap compCardsGivenAnnounce(Ann, Hand)$$

As an example, consider that, in the 3.3.1 setting with deal 012.345.6, a announces that her hand is one of $\{012, 016, 234\}$. Then, a's compatible hands for b are $\{012, 016\}$ and b learns that a has $\{0, 1\}$ in this case. However an agent may learn cards not only from the agent making the announcement, but also from the remaining one. In our example, as c's hand is $\{6\}$, c can also apply the previous function to learn that a holds card 2. But not that a's compatible hands for c are $\{012, 234\}$, so c learns that b holds cards 5. So c learns two cards, one from a and the other from b. The following function calculates this:

$$howManyAgentLearnsFromDeal : \mathcal{P}(\mathcal{P}(D)) \times \mathcal{P}(D) \mapsto \mathbb{N}$$
$$howManyAgentLearnsFromDeal(Ann, Hand) = |whatAgentLearnsFromAnnounce(Ann, Hand)| + |D| - |Hand| - |\bigcup compCardsGivenAnnounce(Ann, Hand)|$$

In the fitness function we have to consider not only one deal (as that in our example) but all possible deals, in order to ensure that the announcement is unconditionally secure. The following function calculates, for a given announcement Ann and an agent having x cards, the minimum number of cards that the agent learns from the deal, by considering all possible agent's hands consistent with Ann:

$$minLearn : \mathcal{P}(\mathcal{P}(D)) \times \mathbb{N} \mapsto \mathbb{N}$$
$$minLearn(Ann, x) = min(\{howManyAgentLearnsFromDeal(Ann, H) \mid H \subseteq D, |H| = x, compCardsGivenAnnounce(Ann, H) \neq \emptyset\})$$

In the same way, we can define a function $maxLearn$ to calculate the maximum number of cards that an agent may learn. Then, the fitness function, given that the size of the deal is $n.m.k$, is defined as:

$$fitness : \mathcal{P}(\mathcal{P}(D)) \times \mathbb{N} \times \mathbb{N} \mapsto \mathbb{Z}$$
$$fitness(Ann, wI, wS) = wI \cdot minLearn(Ann, m) - wS \cdot maxLearn(Ann, k)$$

The two values wI and wS are two natural numbers which measure the relevance of b's knowledge and c's ignorance, respectively. Obviously, the maximum value of the fitness function is obtained when b learns $n+k$ cards (all a's and c's cards) and c learns nothing. Then, the value of the fitness function is $wI(n + k)$. The minimum value is $-wS(n + m)$.

In the Java implementation we work with *jgap*'s chromosomes, that are sequences of binary values. Prior to apply the fitness function we need to decode the binary sequence into an announcement, as explained above. As the fitness function is only allowed to return non-negative values, $S(n + m)$ is added to every result of our previous $fitness$ function.

5 Weighing informativeness and ignorance

It seems reasonable to suppose that there is a correspondence between weighing informativeness and security, on the one hand, and the efficiency in time of the search, on the other hand. In this section we demonstrate by statistical analysis that this is not the case.

Weighing the fitness function with wI and wS we can influence the search, prioritizing one aspect or the other. If we consider informativeness more important than security, the algorithm lets survise to the next generation announcements where c could learn some cards. But if we focus on security and wish to avoid that situation, we will prioritize the security weight in order to devaluate the announcement where c can learn.

Apart from the number of experiments, the size of the populations, and the number of generations set in the algorithm, the different weight combinations also allow us to create different scenarios and configurations. Those will be useful to collect data for statistical analysis. We wish to determine if there is a combination of weights that makes the search go faster.

The results of the algorithm search are stored and a statistic data analyzer goes through them in order to find relevant information. Figure 3 shows an 3.3.1-case sample summary of the output of this analysis.

```
Best weighing MEAN preformance: 7,1
minTime sPR TimeOfSearching 465,
announcement: [[0, 2, 5], [0, 4, 6], [1, 3, 4], [1, 5, 6], [2, 3, 6]],
wInformativeness: 9,
wSecurity: 9,
Deal: 3.3.1
maxTime sPR TimeOfSearching 48875,
announcement: [[0, 1, 2], [0, 3, 4], [0, 3, 6], [1, 2, 6], [1, 3, 5],
[1, 3, 6], [2, 3, 4], [2, 3, 5], [4, 5, 6]],
wInformativeness: 6,
wSecurity: 7,
Deal: 3.3.1
(max-min) range time sPR 48410.0
Whole experiment mean timeOfSearching: 4031.433
Data analyzer timing: 915
```

Fig. 3. Stats analyzer output for 30-runs experiment $wI, wS = [1..10]$

The first line of Figure 3 represents the weights that have the best mean search time. Those weights could be good candidates to weigh other card deals (than 3.3.1), in order to investigate if there is a relation between weighing and search time. Expression `minTime sPR TimeOfSearching 465` denotes the lowest time, in millis, (the fastest protocol found) in the experiment that is associated to the announcement `[[0, 2, 5], [0, 4, 6], [1, 3, 4], [1, 5, 6], [2, 3, 6]]` that has the weight $wI =$ `wInformativeness: 9`, $wS =$ `wSecurity: 9`. We also see similar parameters for the protocol with the highest search time. At the end, the search time range and mean of the entire experiment.

There are also other 7-cards distributions where the genetic approach can search for protocols. Considering the constraints among a, b, c cards presented in [1], we obtain just two 7-cards scenarios where there exist protocols to search, namely: 3.3.1 and 4.2.1. Another case is 2.4.1. Then, there is no protocol where first a makes the announcement. But swapping the roles of a and b we can apply a 4.2.1 protocol.

The first experiment we did was a 30-runs from 1 to 10 different combinations of weights. We can represent that as (`SC` $30, 500, 200, wI, wS, 3, 3, 1$) where:

- SC stands for scenario configuration
- 500 is the maximun number of generations allowed to evolve
- 200 is the initial population size
- 30 means thirty runs of the algorithm
- wI is the weight for informativeness
- wS is the weight for security
- $3, 3, 1$ is the deal distribution

Notice that wI and wS will vary from 1 to 10, generating different scenarios configurations in order to search for weight with the fastest result. The first three results from a 30-run sample experiments with $wI = 4$ and $wS = 7$ are depicted in Figure 4.

```
[[0, 2, 5], [0, 3, 4], [1, 3, 5], [1, 4, 6], [2, 3, 6]]:
58.0, 58.0, 73, 6464, 4, 0, 500, 200
[[0, 1, 6], [0, 2, 3], [1, 2, 4], [2, 5, 6], [3, 4, 5]]:
58.0, 58.0, 24, 2587, 4, 0, 500, 200
[[0, 1, 5], [0, 3, 4], [1, 2, 6], [2, 4, 5], [3, 5, 6]]:
58.0, 58.0, 48, 4478, 4, 0, 500, 200
. . .
```

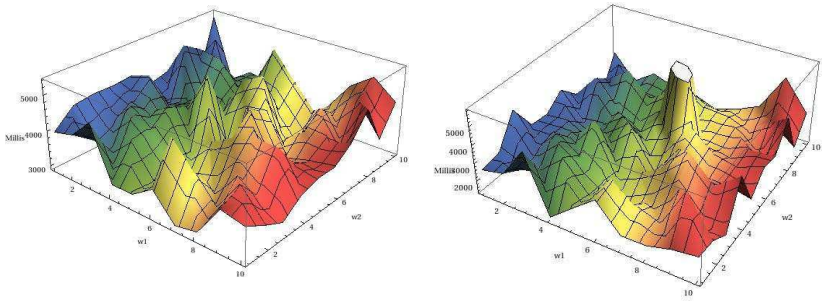
Fig. 4. Partial output of SC $30, 500, 200, 4, 7, 3, 3, 1$

Each two lines represent the execution of the serach for a 3.3.1 scenario. For example, considering the first protocol result, those figures mean:

- `[[0, 2, 5], [0, 3, 4], [1, 3, 5], [1, 4, 6], [2, 3, 6]]` is the announcement sequence. So a annouces $\{025, 034, 135, 146, 236\}$.
- `58.0` means the value the fitness functions assigned to that protocol.
- `58.0` (the second occurrence) represents the maximum fitness value that can be reached for any protocol.
- `73` is the generation where this protocol was found.
- `6464` is the time of searching in milliseconds.
- `4` means the minimum number of cards b can learn using this protocol.
- `0` is the maximum number of cards c can learn.
- `500` is the maximun number of generations allowed to evolve.
- `200` is the initial population size.

We executed a 30-runs experiment varying both weights from 1 to 10 generating a total of $30 \times 10 \times 10 = 3000$ protocol results. Figure 5 depicts information about the runtime for different weights for 3.3.1: on the left, the relation between the differents weights assignments and the search mean time of those; on the right, the standard

deviation extracted from the first. At first sight one can infer there is not a regular relation among those parameters. Hence we think there is no use in extracting the best weights for ignorance and informativeness from this experiment and scale them up to larger deals.



(a) weighing and mean time (b) weighing and deviation time

Fig. 5. 3.3.1 weighing and mean stats graphics

6 Conclusions and future work

We confirmed the possibiliy of modelling unconditionally secure protocols search using Genetic Algorithms. The statistical analyzer showed the non-regular relation between weighing protocol (main) requirements and the mean time of searching. Althouth the genetic engine has been used for small deals, it has now been studied in order to improve the time/memory efficiency to use it for larger deals where several announcements comprise a protocol and a huge number of operations is presumed to be executed. In fact considering the exponential nature of the search space based on the cards distribution, our actual concern is focussed on a possible future guidance of the fitness function and the other variation operators i.e: think that a 5.5.1 scenario with a binary announcement representation has a search space cardinality around 2^{462}. Using the RHC_1 we can observe the epistatic nature of the search space where a slight modification of the chromosome leads to a huge penalty on the solution fitness assignment. We project new features regarding not just the statistical analysis over the protocols found but the search for symmetry properties in protocols, features like a non-exhaustive fitness function and alternative protocol representation that can give a clue about a possible analytic solution for larger and more complex deals.

References

1. Albert, M., Aldred, R., Atkinson, M., van Ditmarsch, H., Handley, C.: Safe communication for card players by combinatorial designs for two-step protocols. Australasian Journal of Combinatorics 33, 33–46 (2005)
2. Albert, M., Cordón, A., van Ditmarsch, H., Fernández, D., Joosten, J., Soler, F.: Secure communication of local states in interpreted systems. In: Abraham, A., Corchado, J., González, S., De Paz Santana, J. (eds.) International Symposium

on Distributed Computing and Artificial Intelligence. pp. 117–124. Advances in Intelligent and Soft Computing, Vol. 91, Springer (2011)
3. Coello, C.A.C., Lamont, G.B., Veldhuizen, D.A.V.: Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation). Springer-Verlag New York, Inc., Secaucus, NJ, USA (2006)
4. van Ditmarsch, H.: The Russian cards problem. Studia Logica 75, 31–62 (2003)
5. van Ditmarsch, H., van der Hoek, W., Kooi, B.: Dynamic Epistemic Logic, Synthese Library, vol. 337. Springer (2007)
6. Fagin, R., Halpern, J., Moses, Y., Vardi, M.: Reasoning about Knowledge. MIT Press, Cambridge MA (1995)
7. Fischer, M., Wright, R.: Bounds on secret key exchange using a random deal of cards. Journal of Cryptology 9(2), 71–99 (1996)
8. Gochet, P., Gribomont, P.: Epistemic logic. In: Gabbay, D.M., Woods, J. (eds.) Logic and the Modalities in the Twentieth Century. Handbook of the History of Logic, vol. 7, pp. 99–195. North-Holland (2006)
9. Hernández-Antón, I., Soler-Toscano, F., van Ditmarsch, H.P.: Unconditionally secure protocols with genetic algorithms. In: PAAMS (Special Sessions). pp. 121–128 (2012)
10. Kumar, A., Ghose, M.K.: Overview of information security using genetic algorithm and chaos. Information Security Journal: A Global Perspective 18(6), 306–315 (2009)
11. Menezes, A.J., Vanstone, S.A., Oorschot, P.C.V.: Handbook of Applied Cryptography. CRC Press, Inc., Boca Raton, FL, USA, 1st edn. (1996)
12. Ocenasek, P.: Evolutionary approach in the security protocols design. In: Blyth, A. (ed.) EC2ND 2005, pp. 147–156. Springer London (2006)
13. Plaza, J.: Logics of public communications. In: Emrich, M., Pfeifer, M., Hadzikadic, M., Ras, Z. (eds.) Proceedings of the 4th International Symposium on Methodologies for Intelligent Systems: Poster Session Program. pp. 201–216. Oak Ridge National Laboratory (1989)
14. Plaza, J.: Logics of public communications. Synthese 158(2), 165–179 (Sep 2007), http://dx.doi.org/10.1007/s11229-007-9168-7
15. Sivanandam, S.N., Deepa, S.N.: Introduction to Genetic Algorithms. Springer Publishing Company, Incorporated, 1st edn. (2007)
16. Whitley, D.: A genetic algorithm tutorial. Statistics and Computing 4, 65–85 (1994)
17. Zarza, L., Pegueroles, J., Soriano, M.: Evaluation function for synthesizing security protocols by means of genetic algorithms. In: Second international Conference on Availability, Reliability and Securiy (ARES'07). IEEE (2007)