Automated Synthesis of Multi–party Rational Exchange Security Protocols

Almudena Alcaide, Juan M. E. Tapiador,
Julio C. Hernandez–Castro and Arturo Ribagorda

Computer Science Department
Carlos III University of Madrid
Avda. Universidad 30, 28911, Leganés, Madrid (Spain)
Email: {aalcaide, jestevez, jcesar, arturo}@inf.uc3m.es

Abstract: It is only recently that rational exchange schemes have been considered as an alternative solution to the exchange problem. A rational exchange protocol cannot provide fairness but it ensures that, rational (i.e. self–interested) parties would have no reason to deviate from the protocol as misbehaving does not have a beneficial result. The common understanding of rationality as a security property has encouraged researchers to look for methods to formally analyze and categorize rational protocols. By contrast, in this paper we adopt a completely new approach and we present an automated tool for the design of multi–party rational exchange security protocols. Given a specific set of initial goals the tool performs a heuristic search in the space of exchanging protocols, producing a rational exchange scheme as a solution. As this is work in progress, we will present the first results obtained executing the application in a three entity environment and a total set of six exchanging items. This article extends previous work [3] including a formal study based on theory of landscapes, which will allow us to measure the effectiveness of our protocol design methodology.
Keywords: rational exchange, automated protocol synthesis, theory of landscapes.

1. Introduction

The *exchange problem* of how to design a general procedure according to which several parties can exchange items in a *fair* manner has attracted much attention throughout the years. Interest in this class of protocols stems from its importance in many applications where disputes among parties can occur, such as digital contract signing, certified e–mail, exchange of digital goods and payments, etc. In particular, assurance of fairness is fundamental when the exchanged items include some kind of non–repudiation evidence, for this constitutes a key service in most of the previously mentioned applications.

Roughly, the property of fairness means that no party should reach the end of the protocol in a disadvantageous position, e.g. having sent her item without having received anything valuable in return. Formally, there exists no protocol according to which a number of parties can exchange items in a fair manner exclusively by themselves, assuming that misbehaving parties can take part in the protocol. Pagnia and Gärtner provide a formal analysis of this problem in [16]. As a result, all fair exchange protocols require a trusted third party (TTP) –whose involvement varies from one class of protocol to another– in order to preserve fairness during the exchange.

By contrast, *rational* exchange protocols do have the enormous advantage of not needing a trusted third party. Informally, a rational exchange protocol cannot provide fairness, but it ensures that rational (i.e. self–interested) parties would have no reason to deviate from the protocol as misbehaving does not have a beneficial result. Since rational exchange protocols provide fewer guarantees, one would expect that they also demand fewer requirements, so they can be viewed as a convenient trade–off between complexity and true fairness.

1.1 Motivation and Related Work

The work presented in this paper focuses on the automated design of multi–party rational exchange security protocols (M–RES). Next we motivate the need for such an approach and introduce some related work on similar topics.

Shortage of rational proposals. As it is only recently that rational exchange schemes have been considered as an alternative solution to the exchange problem, there are very few rational exchange protocols proposed in the literature (see e.g. [7], [20]).

Multi–party environments. The design of exchange security protocols has been proven to be a very difficult, error–prone task and the challenge is even greater when considering multi–party environments. Some rational multi–party solutions do exist, although rationality is not applied to solve exchange problems but other types of questions, such as secret sharing and multi–party computation [13].

Automated analytical tools versus designing tools. With regard to security protocol formal analysis, several automated tools have been presented over the years, each one implementing a different methodology ([15], [19], [18], [21]). These tools have served to formally study classic security requirements such as confidentiality, authentication and/or integrity. More recently, other methodologies have been considered to evaluate new security properties such as rationality ([1], [4], [7]). However, in every case the focus has always been on the analysis of existing schemes. We intend to adopt a completely different approach: the automated design of rational exchange security protocols for which a formal verification is implicit in the design methodology.

Meta–heuristic search. It is clear that the number of possible protocols achieving a set of goals from a set of initial assumptions grows exponentially as the number of goals or the number of participant entities rise. Therefore, for a protocol designing technique to be scalable and feasible it

cannot be based on simple enumeration. In this context, a methodology based on a meta–heuristic search represents a good compromise between optimal solutions and computational tractability. Examples of meta–heuristics include simulated annealing (SA) [14], tabu search (TS) [11], genetic algorithms (GA) [12], and ant colony optimization (ACO) [5].

Automated protocol synthesis. Hao, Clark and Jacob were the first to show in a series of works ([9] and [8]) how meta–heuristic search (in particular, simulated annealing) can be used to automatically synthesize protocols that are demonstrably correct and satisfy various security criteria. In their work, they present an automated tool which, given a set of assumptions and goals, finds security protocols that achieve those goals from the initial assumptions. They use a well known formalism, BAN logic [6], to represent protocol assumptions as well as participants' individual goals and other security requirements. In particular, in Hao et al.'s work, a protocol is represented by a list of messages each being exchanged between two of the participant entities. Associated with every entity is a vector of its current *beliefs*. Each message contains a sequence of beliefs that one entity sends to another (by construction, senders only send beliefs they actually hold). This allows a very simple move strategy for local search which randomly changes any of the beliefs involved in any of the messages. After a message is sent, the beliefs vector for the current receiver is updated and entities check whether their goals have been satisfied. The search technique used is based on a fitness function defined to measure how close a protocol comes to achieve all the required goals. Initially, the search algorithm (simulated annealing) applies perturbations to a randomly generated protocol to generate new schemes. Sequentially, similar changes are applied to each intermediate protocol measuring their fitness in search of an optimal solution (which satisfies all requirements) or the nearest possible to the optimum. More on this heuristic algorithm is described in Section 3.4.1.

Furthermore, in later work [10] the authors apply a different heuristic technique based on genetic algorithms for the synthesis of provably secure protocols. Similar technique is also applied by Park et al. in [17] to the synthesis of cryptographic protocols for a fault–tolerant agent replication system.

1.2 Overview of Our Work

In this paper, we describe the formal foundations for the automated synthesis of rational exchange protocols. The practical implementation of this formalism will result on an application designed to produce a multi–party cryptographic rational exchange protocol giving solution to a particular exchange problem.

As previously stated, a similar approach has been adopted for the automated synthesis of cryptographic protocols, however, that was done to solve other type of problems different from the exchange problem described in the introduction. Moreover, the proof system underneath the synthesis process that we propose will be based on game theory, and not on logics of believes as it was done in previous related works.

Although, the flow and the data structures being used are highly flexible in terms of the number of entities and number and type of exchanging items, this particular work is based on an automated tool which explores the space of 3–party exchanging protocols in search of rational solutions (3–RES protocols). The resulting solution/s will ensure protocol participants a minimum set of requirements expressed by the appropriate utility values as well as ensuring rationality.

We have explored the use of simulated annealing to search for the best possible solution/s. We could not apply standard linear optimization algorithms as the imposed restrictions on the dynamics of the exchange make this problem a non–linear optimization problem. Furthermore, we use theory of landscapes to assert the effectiveness of our methodology.

The rest of the paper is organized as follows. Section 2, formally describes the foundations of our approach. In Section 3, we apply the formalism just described to the parametrization of a particular three entity rational exchange problem. We also propose an heuristic search technique, based on simulated annealing, for the synthesis of a rational exchange protocol to give solution to the specific exchange problem. In Section 4 we use theory of landscapes to determine the level of difficulty of the task. Section 5, describes the resulting synthesized protocol and, finally, in Section 6.1 we summarize the paper by presenting the main conclusions and some future research lines.

2. Foundations for the Synthesis of M–RES Protocols

The usual three elements when dealing with any meta–heuristic search are:

1. Representation of candidate solutions: In our particular scenario, candidate solutions are exchanging protocols.
2. Fitness function: A fitness function (also called utility function) will determine how good a protocol is or, equivalently, how close a protocol is to an optimal solution. The fitness function also provides guidance to the search.
3. Search technique: The search algorithm provides a strategy, able to locate *good* protocols within reasonable amount of time and computational resources.

Next, is the formalization of each one of the aforementioned aspects in our particular model. Before that, the following definition will serve to unify notation throughout this section:

DEFINITION 2.1 (Exchange protocol). *Given a set of entities $\mathcal{P} = \{P_1, \ldots, P_v\}$ and a set of items $\mathcal{O} = \{o_1, \ldots, o_m\}$, an exchange protocol Π consists of n steps, each denoted by:*

$$(t)\ \ P_i \rightarrow P_j : o_{t_1}, \ldots, o_{t_{k_t}} \tag{1}$$

where:

- *$t = 1, \ldots, n$ is the step number,*
- *P_i, $P_j \in P$, $i \neq j$, are the sender and receiver of the message, respectively.*
- *$\{o_{t_1}, \ldots, o_{t_{k_t}}\} \subseteq \mathcal{O}$ are the items P_i sends to P_j, subject to P_i owning those items at step t of the protocol.*

2.1 Representation of Candidate Solutions

2.1.1 Protocol Matrix

In our particular scenario, an exchange protocol Π defined as in 2.1 is represented by a matrix $M^{\Pi} \in \mathcal{M}_{n\times(m+2)} = [m^{\pi}_{ij}]$

of integers. Each row is interpreted as a message in which the first two components identify the sender and the receiver of the message, respectively, and the rest of the row components represent the items being sent.
We can formalize the above in the next expressions:

- $\forall\, l \in \{1, \ldots, n\}$:
  m^{π}_{l1} represents the sender entity of message m_l,
  m^{π}_{l2} represents the receiver entity of message m_l and,
- $\forall j \in \{1, \ldots, m\}$

$$m^{\pi}_{l(2+j)} = \begin{cases} +1 & \text{iff entity } m^{\pi}_{l1} \text{ sends entity} \\ & m^{\pi}_{l2} \text{ item } o_j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

2.2 Fitness Function

A fitness function is individually defined for each participant of the protocol to evaluate the gains obtained at each step of the scheme. The search is aimed at finding exchanging schemes which maximize each individual fitness function.

In our particular model, all participants assign every item involved in the exchange a value depending on whether that entity is interested or otherwise uninterested in gaining access to that item. Those values represent each individual's set of exchange requirements and are captured in the following matrices.

2.2.1 Benefit Matrix

Matrix $B = [b_{ij}] \in \mathcal{M}_{v\times m}$ (v being the number of participant entities and m the number of items involved in the exchange) is defined as:

$$b_{ij} = \begin{cases} 1 & \text{iff item } o_j \text{ initially belongs to } P_i \\ & \text{and } o_j \text{ is part of the exchange.} \\ 0 & \text{iff item } o_j \text{ is of no value to participant } P_i. \\ > 1 & \text{represents the value item } o_j \text{ is worth to} \\ & \text{entity } P_i \text{ iff } o_j \text{ becomes accessible to } P_i. \end{cases} \tag{3}$$

Remark 1. Note that each entity P_i can compute in advance the maximum utility value $\hat{b}_i$ that can be obtained from a protocol run:

$$\hat{b}_i = \sum_{j=1}^{m} b_{ij} \tag{4}$$

2.2.2 State Matrix

The protocol is executed in a fixed number of steps $t = 1, \ldots, n$. Matrices $H(t) = [h_{ij}] \in \mathcal{M}_{v\times m}$ capture the possessions of each party at the end of step t, $0 \leqslant t \leqslant n$, where:

$$h_{ij}(t) = \begin{cases} +1 & \text{iff } P_i \text{ has gained access to item } o_j \\ -1 & \text{iff } P_i \text{ has lost control over item } o_j \\ 0 & \text{iff item } o_j \text{ is unknown to entity } P_i \end{cases} \tag{5}$$

At a initial step ($t = 0$) the matrix will represent the items an entity holds prior to the exchange.

After each step of the protocol,

$$(t) \quad P_i \to P_j : o_{t_1}, \ldots, o_{t_{k_t}} \quad \text{with } i \neq j \text{ and } i, j \in \{1, \ldots, v\},$$

and *before* the next step is to be executed, matrix state $H(t)$ is updated as follows:

$$\begin{aligned} \forall l \in \{t_1, \ldots, t_{k_t}\} \quad & h_{il}(t) = -1 \text{ and} \\ & h_{jl}(t) = +1 \end{aligned} \tag{6}$$

In other words, P_i loses item o_l and P_j wins it.

We will now define a fitness function to compute participant fitness after each step in the protocol as well as global protocol fitness at the end of a protocol run.

2.2.3 Utilities

At each step t of the protocol (i.e. after updating state matrix $H(t)$ according to the transference of items), we shall refer to the gains achieved by a player so far as "utility" values. Each P_i's current utility at step t, ($0 \leqslant t \leqslant n$) can be computed as:

$$u_i(t) = \sum_{j=1}^{m} b_{ij} h_{ij}(t), \;\; i \in \{1, \ldots, v\} \tag{7}$$

Remark 2. Note that expression (7) implicitly includes the cost induced by losing control over a valuable item. When P_i sends o_k, h_{ik} passes from $+1$ to -1, and consequently the overall utility gets reduced by $2b_{ik}$.

2.2.4 Differential Utilities

The differential utility for a player P_i between steps t_1 and t_2, with $0 \leqslant t_1 \leqslant t_2 \leqslant n$, is defined as:

$$\begin{aligned} du_i(t_1, t_2) &= u_i(t_2) - u_i(t_1) \\ &= \sum_{j=1}^{m} b_{ij} h_{ij}(t_2) - \sum_{j=1}^{m} b_{ij} h_{ij}(t_1) \\ &= \sum_{j=1}^{m} b_{ij} \big[h_{ij}(t_2) - h_{ij}(t_1) \big] \end{aligned} \tag{8}$$

Remark 3. During a protocol execution, there may be steps at which players go into a temporarily "worst" state (i.e. $du_i(t, t+1) \leqslant 0$). The relevant fact, however, is whether at the end of the protocol run P_i gets enough differential utility:

- If $du_i(0, n) > 0$, the exchange is successful to P_i.
- If $du_i(0, n) < 0$, the exchange is unsuccessful to P_i.
- If $du_i(0, n) = 0$, the exchange is of no use to P_i.

2.2.5 Global Differential Utilities

Additionally, a global protocol fitness function will be defined to describe the overall fitness of a protocol solution M^{Π}. An overall differential utility is defined by the following equation:

$$\begin{aligned} dU(M^{\Pi}) &= \sum_{i=1}^{v} du_i(0, n) \\ &= \sum_{i=1}^{v} \sum_{j=1}^{m} b_{ij} \big[h_{ij}(n) - h_{ij}(0) \big] \end{aligned} \tag{9}$$

2.3 Search Technique – Goals and Dimension

We wish to explore the space of all exchange protocols of the form $M^{\Pi} \in \mathcal{M}_{n\times(m+2)}$ to find rational schemes for which all participants' utility values are maximum or the nearest

possible to the maximum. As mentioned before, the searching algorithm is based on simulated annealing (see Section 3.4.1 for a detailed description of this algorithm).

In order to explore such a space, we will distinguish between types of exchange protocols. We will consider an exchange protocol to be a *feasible solution* if the protocol dictates the steps of a feasible exchange of the required commodities amongst participant entities; i.e. entities following the protocol will fulfill their requirements and will gain access to the required items.

Moreover, we will further distinguish between *blind* feasible solutions and *rational* feasible solutions.

Blind protocols represent feasible solutions to an exchange problem although for these protocols to be terminated entities must be both:

1. *Rational*, aimed at maximizing their individual utility value and,
2. *Cooperative*, aimed at maximizing the overall protocol fitness. In other words, cooperative entities will also seek to satisfy other entities' requirements.

By contrast, in a rational protocol solution entities are required to be only rational being able to display selfish and non–cooperative behavior. Rationality will be ensured by forcing participants who have reached their maximum utility values in an intermediate step of the protocol to quit the exchange at that step. In other words, when an entity has fulfilled all their requirements, that entity will stop actively participating in the scheme.

Formally, the following definition will describe the goals of the current search.

DEFINITION 2.2 (Search Goal). *The goal of the search is to maximize dU, as defined in expression (9), subject to:*

- *The values $h_{ij}(t) = \pm 1$ in state matrix $H(n)$.*
  These values will depend on each message content as well as on linear and non–linear relations between the elements of each matrix $H(t)$, $t = 1, \ldots, n$. For example, a possessed item such as an encrypted token might not become available until other token/s (e.g. the decryption key) are also accessible.
- *Entities achieving their maximum utilities stop being active participants of the exchange. In other words, entities which have fulfilled all the requirements in an intermediate step in the protocol must not be the senders of any other message in the exchange.*

2.3.1 How Many Exchange Protocols Exist?

As described in Section 2.1, a protocol is represented by a matrix $M^{\Pi} \in \mathcal{M}_{n\times(m+2)}$, where n is the number of protocol steps and m is the number of tokens involved in the exchange. Each row represents a message in the protocol and the first two components of each row describe the sender and receiver of that message, respectively. We can compute an estimate of the total number of exchange protocols subject to evaluation as:

$$\mathcal{O}\big(\frac{v!}{(v-2)!}2^{nm}\big) = \mathcal{O}\big(v(v-1)2^{nm}\big) \qquad (10)$$

For example, for a 3 entity scenario, a maximum of 10 messages exchanged and a total of 8 items in each message, the search space has an estimated complexity of $\mathcal{O}(6 \cdot 2^{80})$.

Although, the aforementioned expression gives an estimate of how many exchange protocols there are, it is difficult to determine how many of these protocols represent *feasible* solutions to the specific exchange problem, and even more challenging to estimate how many of those feasible solutions represent a rational exchange.

3. Automated Design of 3–RES Solutions

As it was previously stated, parameters such as the number of entities, number of exchanging items or entities' requirements can be configured within the model to describe a specific exchanging scenario. For the purpose of this study, we will be focusing on a particular three–entity exchange problem.

For every participant entity we will be giving a series of initial assumptions and goals which will be represented using the matrices previously described. The search will then try to resolve the problem by looking for a rational scheme in which entities will see all their requirements fulfilled.

3.1 Initial Assumptions

- The specific exchange problem will consist of an entity P_0 which aims to collect a series of electronic items from entities P_1 and P_2, delivering the appropriate tokens in return. All entities, P_0, P_1 and P_2, are considered to be rational.
- None of the collected items in isolation is of any value to entity P_0. In other words, P_0 is interested in collecting all or none of these items.
- Additionally, the nature of these items is such that their utilities only become available when the corresponding tokens are delivered in return. Although this restriction seems hard and unrealistic, there are a few real life examples where items are of this nature. For example, P_0 could be a user trying to book a holiday package consisting of accommodation, flights and tickets for a local tourist attraction. User P_0 needs either all or none of the required items and, additionally no item becomes available unless the providers of the required services have received payment.
- Participant entities P_1 and P_2 are part of a visible and recognizable PKI (Public Key Infrastructure). No other trusted or semi–trusted parties can be involved in the scheme. By contrast, this is not a restriction on entity P_0, who can maintain anonymous his/her real identity.
- All messages sent by P_1 and P_2 must be signed with their corresponding private keys and all messages received by these entities must be encrypted with their corresponding public keys.

3.2 Data Representation for 3–RES Protocol Synthesis

3.2.1 Items

The following is the list of all items involved in the scheme:

(o_1) Request for $item_1$: $desc\_item_1$.
It contains a description of $item_1$ required from entity P_1.
(o_2) Request for $item_2$: $desc\_item_2$.
It contains a description of $item_2$ required from entity P_2.

(o_3) Item $item_1$:
Customized item issued by entity P_1 as specified by P_0 in $desc\_item_1$.
(o_4) Item $item_2$:
Customized item issued by entity P_2 as specified by P_0 in $desc\_item_2$.
(o_5) Return token $return\_item_1$:
Token issued by P_0 for P_1 in return for $item_1$.
(o_6) Return token $return\_item_2$:
Token issued by P_0 for P_2 in return for $item_2$.

Remark 4. At a initial state (t=0), only P_0 holds descriptions for the required items. These items must be individually tailored by P_1 and P_2 to satisfy requirements specified in tokens $desc\_item_1$ and $desc\_item_2$. In a similar way, until these items have been issued, entity P_0 will not hold the appropriate return tokens. P_0 will read the specifications to produce the return tokens within the items received.

Remark 5. The application will assign a *non–accessible* status to items $item_1$ and $item_2$, until the return tokens are received by the appropriate entities.

3.3 Computing Fitness

Utility values are calculated as defined in equation (7) at each step in the protocol and for every participant entity. In a similar way, the overall protocol fitness value is computed as defined in equation (9) at the end of the protocol run. However, when evaluating a given protocol, the fitness taken will be the maximum utility value obtained along the whole execution.

Moreover, when evaluating a given protocol, if an entity reaches her maximum utility value in an intermediate step, that entity must quit the protocol. The continuation protocol is only considered a possible feasible solution when such an entity is not the sender of any further messages.

3.4 3–RES Search Algorithm

In this work, an heuristic algorithm is used for the automated synthesis of rational protocols of the form $M^{\Pi} \in \mathcal{M}_{n\times(m+2)}$. The algorithm is based on simulated annealing. Next we provide a brief description of its operation.

3.4.1 Simulated Annealing

Simulated annealing ([14]) is a search heuristic inspired by the cooling processes of molten metals. It merges a basic hill–climbing technique with a probabilistic acceptance of non–improving solutions which allows the search to escape from local optima. Fig. 1 details the basic scheme.

Roughly, this technique requires a first individual S_0 which is randomly generated and presented for evaluation. Each individual's evaluation consists of computing its fitness value $F(S)$. There is also a control parameter $T \in \mathbb{R}^+$ known as temperature, which takes an initial value T_0 and which dynamically decreases its value during the annealing process. The first randomly generated S_0 is evolved to a different solution C in the neighborhood of S_0. The new scheme is also evaluated. If the new individual C reaches a higher level of overall fitness than the original one, then it is accepted as a new valid scheme from which to generate the next one. If the new individual represents a solution worse than the previous one, the new scheme could only be accepted if the control parameter T is above a specific value. In other words, better solutions are always accepted and worse solutions are accepted when the temperature is still above a certain threshold. The process is repeated a fixed number of times depending on the initial temperature and the number of solutions being accepted.

```
1  S ← S_0
2  T ← T_0
3  repeat until stopping criterion is met
4        repeat MIL times
5              Pick C ∈ N(S) with uniform probability
6              Pick U ∈ (0, 1) with uniform probability
7              if F(C) > F(S) + T ln U then
8                    S ← C
9        T ← αT
```

where:
S_0 represents the initial individual
T_0 is the initial temperature
MIL defines the number of moves in the inner loop.
$N(S)$ represents S's neighborhood
$F(\cdot)$ represents the fitness function
$0 < \alpha < 1$ is the cooling factor

Fig. 1. Basic simulated annealing algorithm.

3.4.2 Search Operators

The following routines are candidate mutation operators to obtain neighbor individuals, as required by the simulated annealing algorithm. What follows is a brief description of each one of them:

– **Random Mutation:** A random mutation is a modification of a random number of elements in the protocol.
– **Permutation:** This routine is a permutation of the protocol message order.
– **Expansion:** Consists in adding new random messages to every shortened feasible solution in search of higher fitness values.

4. Theory of Landscapes

Before presenting our final results we will make use of a combinatorial technique based on theory of landscapes to proof the efficiency of our global formalism. We will study the fitness landscape generated by the different search operators defined in previous Section 3.4.2. This information will assist us in determining which one of those routines is more effective to be used as a neighboring operator, in the simulated annealing algorithm.

4.1 Fitness Landscape and Random Walk Techniques

Taken from biology, the notion of *fitness landscape* has become an important concept in evolutionary computation. The relationship between genotypes, determined by an evolutionary operator, and the fitness assigned to each genotype via some mapping provided by the fitness function, constitutes the landscape. Recently, the landscapes of a range of problems have been analyzed in an attempt to determine the relation

between fitness landscape structure and the performance of a particular heuristic technique. In other words, the analysis of landscape structures could allow us to determine the difficulty of a task and hence apply the most appropriate heuristic search algorithm.

The *random walk technique* produces a number of consecutive neighboring points within a solution space for which to evaluate their fitness. In the initial phase of the random walk a random individual is produced and evaluated. Then, using the neighbor relation a neighbor individual is generated and calculated its fitness too. The same step is repeated for k number of times, which represents the length of the walk.

A number of techniques for the analysis of landscape structures exist. In particular, Weinberger in [22] investigated how the *autocorrelation function* of the fitness values along the steps of a *random walk* relates to the ruggedness of the examined landscape.

Intuitively, one can assume that if there is no variation between the fitness of two neighbor points, this will correspond to high correlation of those values and there will be very few opportunities for improvement as the search algorithm progresses. If, on the other hand, there is no correlation between the fitness values of consecutive neighbor points, this will be representative of rugged fitness landscapes wherein any guided search will likely degenerate into a random search.

We will used the study of the fitness landscape to justify the parameters of the search algorithm, at the same time we will analyze the results in order to establish the difficulty of our task.

4.2 Fitness Autocorrelation

The autocorrelation function of a set $F = \{f_1, \ldots, f_k\}$ of fitness values is defined by the following equation:

$$r(s,t) = \frac{E(f_t * f_{t+s}) - E(f_t) * E(f_{t+s})}{var(F)} \qquad (11)$$

where $s = 0, \ldots, k-1$ and $t = 1, \ldots, k-s$. Likewise, $E(x)$ represents the expected value of x and $var(x)$ its variance.

Some characteristic shapes for the autocorrelation function are described below:

- Most correlation values are close to zero. This is a characteristic of a random landscape where there is no correlation between fitness values. In this case, heuristic search algorithms are not expected to obtain better results and/or in less time than classic random search.
- The second characteristic form of an autocorrelation function is the slow decaying form. In this case, correlation between fitness values is usually representative of *even* fitness landscapes. In other words, neighbor solution points offer very similar fitness which will probably make any guided search quite slow.
- A third characteristic form is the fast decaying form. It is usually representative of *rugged* fitness landscapes, in which often appropriate directed heuristic search algorithms can serve as guidance to obtain good solutions in less time than a classic random search.
- Finally, a constant or periodic autocorrelation function often indicates a badly defined neighboring operator which would periodically generate individuals which are too similar to one another.

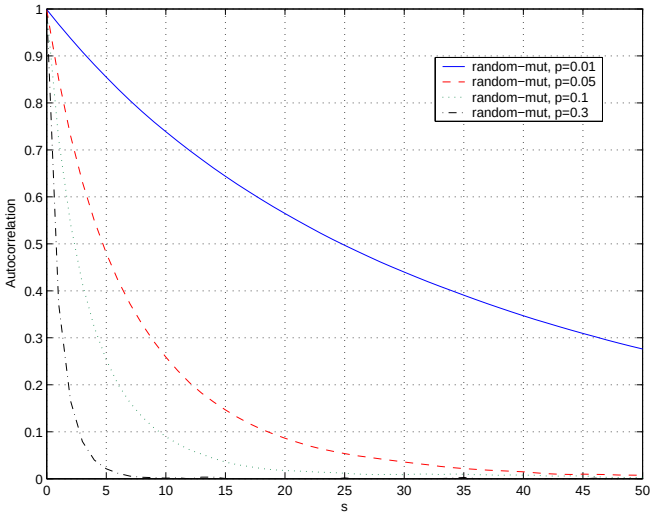
4.3 3–RES Fitness Landscape

Each particular exchange problem will have a certain landscape structure. As mentioned before, measuring this will assist in determining the effectiveness of the search technique, as well as ensuring that candidate representation, mutation operator and fitness function are appropriately defined in the formalism. However, as previously mentioned, apart from clear extreme cases (very high or very low autocorrelation values), this study will only serve as an approximation of the difficulty of the task and the rate of success we expect to obtain with our particular search methodology.

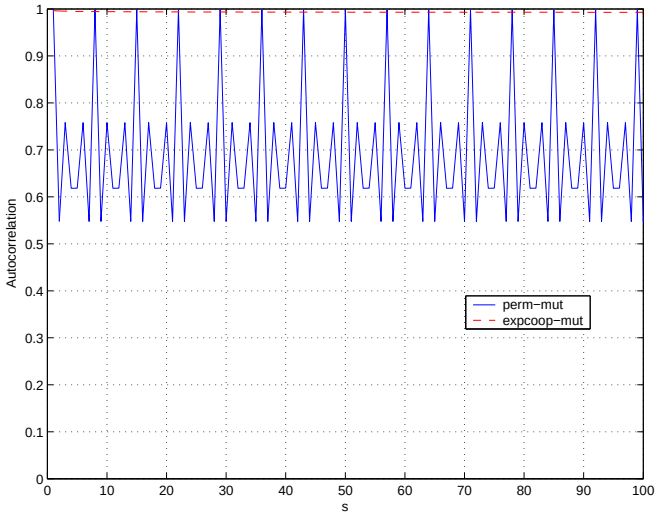
For the purposes of our current work, we are presenting the autocorrelation functions obtained when evaluating the operators defined in Section 3.4.2 and a linear combination of them (in all the cases, the autocorrelation is computed by averaging 400 random walks of length 1000 individuals each):

1. A **random mutation** operator, consisting in randomly changing a percentage of elements in the protocol matrix M^{Π}.
   Fig. 2 (a) shows the autocorrelation function when the neighboring operator is a random mutation with different mutating rates (1%, 5%, 10% and 30%). The curve showing rate 1% corresponds to a slow decaying shape where correlation between fitness values is quite high. This indicates a flat fitness landscape, often too slow to explore applying any guided search algorithm. On the other hand, a mutation rate of 30% results in most correlation values below $0,5$ and close to zero. As mentioned before, this would represent uneven fitness landscapes. In this case, the rate of mutation seems to be sufficiently high to obtain similar results than with a completely random search algorithm. Finally, the curves described when rates are of 5% and 10% result in rapid decaying shapes where a guided search is expected to outperform any random exploration.
2. A simple **message permute operator** as described in Section 3.4.2.
   Fig. 2 (b) shows a periodic and hence strongly correlated curve. This corresponds to a cyclic landscape. This is an indicator of a badly defined neighboring operator and/or an inadequate candidate representation and/or an invalid fitness function. However, as we will see later, this operator could enhance the performance of others when used in combination.
3. The **expansion routine**, also described in Section 3.4.2.
   In this case, Fig. 2 (b) shows the curve (plotted as a dashed line) of such a neighboring operator. The curve is constant and close to 1.0, indicating a strong correlation between fitness values and therefore a very flat landscape. Again, this could be a consequence of the inadequacy of one or more aspects of the problem representation.
4. A **combination of all previous three**. Different linear combinations of the three previous operators offer the results shown in Fig. 2 (c). The curves described correspond to different values for combining the three different routines with a mutation operator of the form:

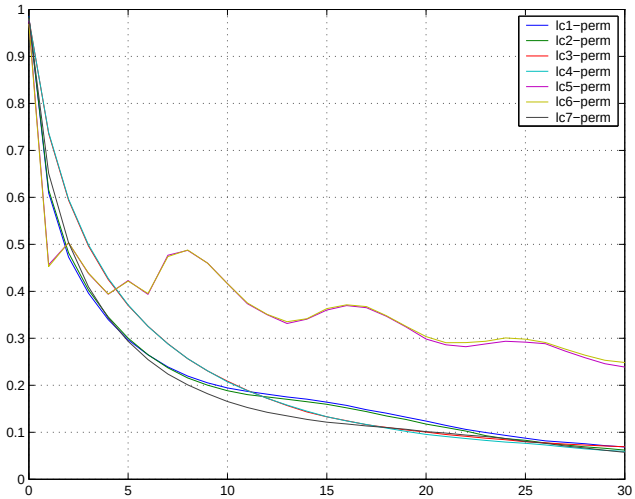
$$lc = \begin{cases} \text{Apply } RandomMutation \text{ with probability } p_1 \\ \text{Apply } PermuteMutation \text{ with probability } p_2 \\ \text{Apply } ExpansionMutation \text{ with probability } p_3 \end{cases} \qquad (12)$$



(a)



(b)



(c)

Mutation Operator	Value of s_0	Function Shape
Permute	–	Periodic (Strongly Correlated)
Expansion	–	All values close to one (Strongly Correlated)
Random Mutation (rate 1%)	25	Slow decaying shape
Random Mutation (rate 5%)	5	Moderate decaying shape
Random Mutation (rate 10%)	3	Rapid decaying shape
Random Mutation (rate 30%)	1	All values close to zero (Non Correlated)
Linear Combination	$[2, 3]$	Rapid decaying shape

(d)

Fig. 2. Fitness Autocorrelation for different mutation operators.

subject to $p_1 + p_2 + p_3 = 1$.

Specifically, the explored combinations are shown in the next table:

LC	p_1	p_2	p_3
lc1	0.1	0.4	0.5
lc2	0.1	0.4	0.5
lc3	0.1	0.2	0.7
lc4	0.1	0.2	0.7
lc5	0.1	0.8	0.1
lc6	0.1	0.8	0.1
lc7	0.0	0.3	0.7

(Note that pairs $lc1/lc2$, $lc3/lc4$ and $lc5/lc6$ correspond to different sets of random walks with the same combining probabilities, hence the similarity).

Perhaps the most remarkable result here is the significantly different curves obtained in $lc5 - perm$ and $lc6 - perm$, when the permutation routing is applied proportionally more often than the others.

Finally, the table shown in Fig. 2 (d) summarizes the results previously described. The value s_0 represents the value for s (number of steps in the random walk) for which the correlation function drops below the value $0,5$. As mentioned before, only in extreme cases for which the correlation function is constantly close to one, periodic or constantly close to zero, correlation can be determined in absolute terms: strongly correlated or non–correlated. In any other case, correlation can only be considered in terms relative to the problem to be solved and the correlation curve can only serve as an indicative.

5. Experiments and Results

A M–RES search algorithm (in particular 3–RES search) has been designed and implemented, based on simulated annealing and combining the different routines described in Section 3.4.2, for the synthesis of a rational exchange protocol, giving solution to the exchange problem established in Section 3. Table 1 shows a complete list of the parameters used for the search.

Furthermore, 3–RES search has been compared to other two search schemes: a random search in which a mutation routine operates with a rate of 10% and a search algorithm combining the permuting and expanding routines previously described. The following graphs show how our approach displays better rate of success in every case.

- Table 2 shows a comparative between different search algorithms. The results are based on a trial of 100 executions, each performing a search for a rational exchange

Table 1. Simulated Annealing parameters.

General	
Max. No. inner loops	100
Max. No. moves in inner loop	50
Max. No. failed inner loops	3
Initial temperature	200
Cooling rate	0.98
PRNG	Mersenne Twister
M–RES Problem	
No. parties (v)	3
Max. No. messages per protocol (n)	10
Max. No. items per message (m)	6
Total No. of items to exchange	6

Table 2. Results obtained over 100 executions.

Mutation Operator	**Average No. of Protocols Evaluated**	**Average of success**
Permute & Expand	53927	$0,03\%$
Random Mutation	3885	$0,42\%$
3–RES search: Permute $(0,1\%)$ Expand $(0,8\%)$ Random Mut. $(0,1\%)$	1825	$0,94\%$

scheme to give solution to the exchange problem parameterized in Section 3.

- Similar results are shown in Fig. 3. This figure serves to compare the 3–RES search (based on routines expansion, permutation and random mutation of rate 10%) with a simple random mutation search algorithm (also mutating rate of 10%). The graph shows the first three cycles of the simulated annealing algorithm. At the end of these three cycles, 3–RES search has found a rational protocol (fitness value of 20) whereas the random mutation search will demand larger number of cycles to succeed (for this case, up to 48 cycles until reaching a valid protocol).

5.1 Synthesized 3–RES protocol

The following rational protocol is synthesized by the 3–RES search algorithm:

(1) Entity P_0 sends entity P_1, a message including $desc\_item_1$ and $desc\_item_2$, descriptions of the required items.
(2) Entity P_1 produces, according to the appropriate description, a customized $item_1$ destined to P_0.
(3) Entity P_1 sends P_2 a message containing $item_1$ and the description token $desc\_item_2$.
(4) Entity P_2 sends P_0 items $item_1$ and $item_2$.
(5) Participant P_0 sends P_1 a message including $return\_item_1$ and $return\_item_2$, the return–tokens for the items received.
(6) P_1 receives a message with two return–tokens. It takes $return\_item_1$ and sends entity P_2, the token $return\_item_2$.

Figures 4 (a) and 4 (b) represent the flow of messages and items exchanged between the participant entities of the two–phase synthesized protocol. Description of every item o_j involved in the scheme is given in Section 3.2.

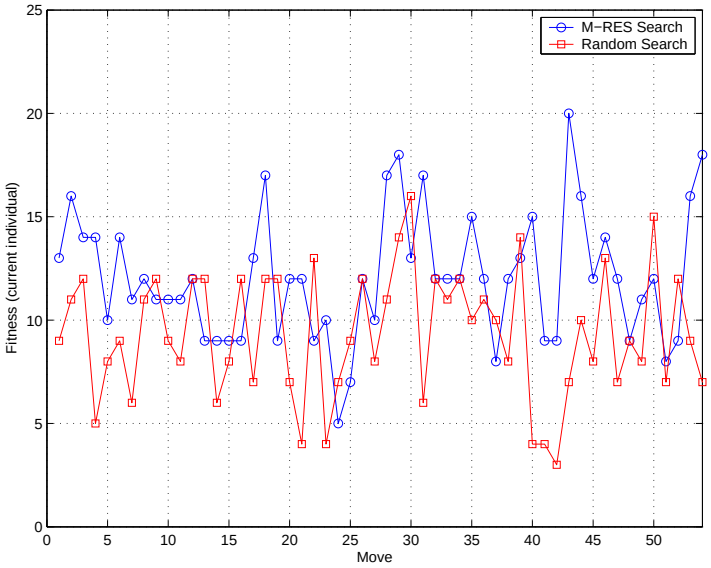
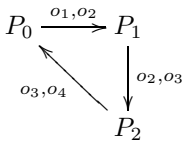
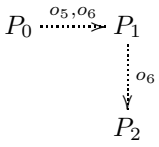


Fig. 3. Behavior of two different search algorithms for the 3–RES exchange problem. The maximum fitness is reached by 3–RES search and it represents the fitness of a rational exchange protocol.



(a)



(b)

Fig. 4. Synthesis of a three entity rational protocol. The protocol run in two phases described in (a) and (b).

$$
\begin{array}{lll}
(1) & P_0 \rightarrow P_1: & \{desc\_item_1, desc\_item_2\}_{K_{P_1}} \\
(2) & P_1 \rightarrow P_2: & \{\{item_1, desc\_item_2\}_{K_{P_1}^{-1}}\}_{K_{P_2}} \\
(3) & P_2 \rightarrow P_0: & \{item_1, item_2\}_{K_{P_2}^{-1}} \\
(4) & P_0 \rightarrow P_1: & \{return\_item_1, return\_item_2\}_{K_{P_1}} \\
(5) & P_1 \rightarrow P_2: & \{\{return\_item_2\}_{K_{P_1}^{-1}}\}_{K_{P_2}}
\end{array}
$$

Fig. 5. A three entity rational exchange protocol (3–RES).

As established in the initial assumptions of the problem, all messages sent by entities P_1 and P_2 must be signed with the corresponding private keys ($K_{P_1}^{-1}$ and $K_{P_2}^{-1}$), as well as all messages being received by these entities must be encrypted with the appropriate public keys (K_{P_1} and K_{P_2}). The final resulting scheme is described in Fig. 5.

5.1.1 Rationality

Rationality of the scheme previously described, can be directly inferred by the methodology used in the synthesis of the protocol. Techniques, based on game theory and backward induction, can be applied to formally proof rationality. See [2] for a detailed description of this formal proof.

The following is a list of those aspects of the formalism which ensure that the scheme is a feasible rational solution satisfying all participants' sets of requirements:

- **From entity's P_0 point of view.** As stated in the initial assumptions, items $item_1$ and $item_2$ are of no use to entity P_0 until the corresponding return items $return\_item_1$ and $return\_item_2$ have reached entities P_1 and P_2 respectively. To this regard, and since entity P_0 requires either all or none of these items, entity P_0 is *rationally* forced to reply with items $return\_item_1$ and $return\_item_2$ to entity P_1 and P_2 respectively.
- **From entity's P_1 point of view.** As previously mentioned, entity P_0 requires either all or none of these items. Again, this assumption forces entity P_1 to send P_2 messages (2) and (5).
- **From entity's P_2 point of view.** Similar rationale will force entity P_2 to send P_0 messages (3).

Therefore, no entity would unilaterally deviate from the 3–RES protocol as they could not obtain better utility value in doing so. The scheme is then a rational solution.

6. Conclusions and Future Work

Traditionally, automated tools have always been applied to the analysis of security protocols. In this paper we have adopted a completely new approach, ensuring rationality as part of the automated design of an exchange scheme.

For the purposes of this work, we have designed and implemented a 3–RES search algorithm based on simulated annealing. Moreover, the formal foundations of our methodology ensure high levels of flexibility and scalability and have served to develop the structure of any global M–RES search for any multi–party rational exchange problem.

Finally, the synthesized 3–RES protocol does also present high levels of scalability and a whole n–RES family of protocols can be easily derived from this three–entity scheme (see [2] for further details).

6.1 Future Work

The experimental work carried out for this study is preliminary. Further experiments will include:

- An increase in the number of entities.
- An increase in the number of tokens.
- Changing the exchange problem by modifying the initial state matrix $H(0)$ and the benefit matrix B.
- Adding extra meaning to the values in matrix B to elaborate a taxonomy of problems and synthesized schemes.

With this further work we intent to:

- Automatically synthesize provable multi–party rational exchange protocols (the proof system is based on game theory) so we resolve the problem of manually designing such schemes and,
- Evidence the effectiveness of our methodology based on simulated annealing, over other search algorithms.

References

[1] A Alcaide, J Estévez-Tapiador, J Hernandez Castro and A Ribagorda, Rational exchange– a formal model based on game theory. Proceedings ETRICS'06 2006, Springer-Verlag, LNCS Vol. 3995/2006, pp. 396–408.

[2] A Alcaide, J Estévez-Tapiador, J Hernandez Castro and A Ribagorda, A multi-party rational exchange protocol. Proceedings OTM Conferences (1) 2007, LNCS Vol. 4805, pp. 42–43.

[3] A Alcaide, J Estévez-Tapiador, J Hernandez Castro and A Ribagorda, Towards automated design of multi-party rational exchange security protocols. Web Intelligence/IAT Workshop RRS 2007, pp. 387–390.

[4] A Alcaide, J Estévez-Tapiador, J Hernandez Castro and A Ribagorda, Bayesian rational exchange. International Journal of Information Security, Vol. 7, No. 1, 2008, pp. 85–100.

[5] E Bonabeau, M Dorigo and G Theraulaz, Swarm Intelligence: From Natural to Articial Systems, Oxford University press Inc., 1999, USA. ISBN 0-19-513159-2.

[6] M Burrows, M Abadi and R Needham, A logic of authentication. ACM Transactions on Computer Systems, Vol. 8, No. 1, 1990, pp. 18–36.

[7] L Buttyán, Building blocks for secure services: Authenticated key transport and rational exchange protocols, Tech. rep., Swiss Federal Institute of Technology. Lausanne (EPFL), 2001, Ph.D. Thesis No.2511.

[8] H Chen, J Clark and J J., Automatic design of security protocols. Computational Intelligence, Vol. 20, No. 3, 2004, pp. 503–516, Special Issue on Evolutionary Computing in Cryptography and Security.

[9] J Clark and J J., Protocols are programs too: the meta-heuristic search for security protocols. Information and Software Technology, Vol. 43, No. 14, 2001, pp. 891–904.

[10] J Clark and J Jacob, Searching for a solution: engineering tradeoffs and the evolutionof provably secure protocols. Proceedings IEEE Symposium on Security and Privacy 2002.

[11] D de Werra, A Hertz and E Taillard, A tutorial on tabu search. Proc. of Giornate fi Lavaro AIRO'95 1995, pp. 13–24.

[12] D Goldberg, Genetic Algorithms in Search, optimization and Machine Learning, Addison-Wesley, 1989.

[13] J Halpern and V Teague, Rational secret sharing and multiparty computation: Extended abstract. Proceedings of the thirty-sixth annual ACM symposium on Theory of computing STOC '04 2004, ACM 1-58113-852-0/04/0006.

[14] S Kirkpatrick, G C. and V M., Optimization by simulated annealing. Science, Vol. 220, No. 4598, 1983, pp. 671–680.

[15] G Lowe, Casper: a compiler for the analysis of security protocols. Simon Foley, editor, 10th Computer Security Foundations Workshop 1997, pp. 18–30, Rockport, Massachussets, USA. IEEE Computer Society Press.

[16] H Pagnia and F Gärtner, On the impossibility of fair exchange without a trusted third party, Tech. rep., Darmstadt University of Technology, Department of Computer Science, 1999.

[17] K Park and C Hong, Cryptographic protocol design concept with genetic algorithms. KES (2) 2005, pp. 483–489.

[18] L C Paulson, The inductive approach to verifying cryptographic protocols. Journal of Computer Security, Vol. 6, No. 1-2, 1998, pp. 85–128.

[19] A W Roscoe, Modelling and verifying key-exchange protocols using csp and fdr. Proceedings of the 8th IEEE Computer Security Foundations Workshop 1995, IEEE Computer Society Press, pp. 98–107.

[20] P Syverson, Weakly secret bit commitment: Applications to lotteries and fair exchange. Proceedings of the 11th IEEE Computer Security Foundations Workshop 1998, pp. 2–13.

[21] P Syverson and P C van Oorschot, On unifying some cryptographic protocol logics. IEEE Symposium on Research in Security and Privacy 1994, IEEE Computer Society Press, pp. 14–28, Oakland, CA. IEEE COmputer Society, Technical Committee on Security and Privacy.

[22] E D Wenberger, Correlated and uncorrelated fitness landscape and how to tell the difference. Biological Cybernetics, Vol. 63, 1990, pp. 325–336.