# An Approach to Improve Efficiency of Traditional Automated Trust Negotiation Strategies

Bailing Liu, Chao Lei, Yanhui Li, Liting Liang

School of Information Management, Central China Normal University, China
bl_liu@mail.ccnu.edu.cn, 934273713@qq.com, yhlee@mail.ccnu.edu.cn, 1178323062@qq.com

## Abstract

Automated trust negotiation (ATN) is a successful approach to establishing mutual trust between strangers wishing to share resources or conduct business by gradually requesting and disclosing digitally signed credentials. An ATN strategy need to be adopted to determine a successful negotiation sequence based on the access control policies. Traditional ATN strategies provide the maximum degree of protection to the involved resources. However, there can be cases where efficiency is the most crucial need especially in light of widespread use of mobile device. It is preferable to adopt highly efficient ATN strategies, even if they do not maximize the protection of involved resources. Thus in this paper, we propose an approach to improve efficiency of traditional negotiation strategies by integrating sequence prediction, making a trade-off between negotiation efficiency and sensitive information protection. We validate the effectiveness of our approach by performing experiments on a mobile platform, taking the representative traditional negotiation strategy PRUENS as an example. Our experimental results show that our approach dramatically enhances efficiency of PRUNES.

**Keywords**: Automated trust negotiation, Negotiation strategy, Efficiency, Mobile phone

## 1 Introduction

Automated trust negotiation (ATN) [1-4] is an attractive approach to dynamically establishing mutual trust between strangers wishing to share resources or conduct business in open environments like the Internet. It enables strangers to establish trust by iteratively disclosing digitally signed credentials containing participants' attribute information. Usually, the digital credentials themselves contain sensitive information that a subject does not want to reveal to any strangers, so for each credential there is an access control policy (a policy for short) associated with it, governing the credential disclosure. Participants need to employ negotiation strategies [1, 5] to automatically control negotiation process, that is, which credentials are disclosed, when they are disclosed, which credentials are requested from the other subject, and when the negotiation is terminated. Performing a negotiation strategy may require heavy computational resources and much communication resources especially when the policies on both sides are complex.

Nowadays, there have been several negotiation strategies such as Prudent Negotiation Strategy (PRUNES) [6], Deterministic Finite Automation Negotiation Strategy (DFANS) [7] and so on. These negotiation strategies are the most complete strategies that assure the maximum degree of protection to the involved resources. But sometimes, efficiency is the most crucial need, especially with the widespread use of mobile networks and mobile devices, the emerging mobile applications require efficient approaches to trust negotiations. It is preferable to adopt highly efficient ATN schemes, even if they do not maximize the protection of involved resources.

To address this problem, Bertino et al. [8] proposed a negotiation strategy that predicts the whole negotiation sequence based on participants' previous interactions without running a traditional negotiation strategy. The negotiation sequence is used either in its entirety or not at all, which cannot to be used to predict portions of a negotiation. When their sequence prediction does not work, traditional negotiation strategies still need to be invoked. However, their work inspires us to integrate sequence prediction into traditional negotiation strategies to predict portions of negotiations, improving negotiation efficiency. Therefore, in this paper, we propose an approach to predict trust negotiation sequence during the enforcement of traditional negotiation strategies whenever possible, achieving a better trade-off between negotiation efficiency and sensitive information protection in mobile environments with constrained resources. We conduct experiments taking the representative traditional negotiation strategy PRUENS as an example. The experimental results show that our approach can highly enhance its efficiency.

The remainder of this paper is organized as follows. Section 2 gives an overview of ATN and describes the

*Corresponding Author: Yanhui Li; E-mail: yhlee@mail.ccnu.edu.cn
DOI: 10.3966/160792642018011901004

related work. In Section 3, we propose the method of integrating sequence prediction for molecule resources into traditional negotiation strategies. Section 4 reports our experimental results. We conclude the paper in Section 5.

## 2 Overview of ATN and Related Work

### 2.1 Overview of ATN

Winsborough *et al.* defined the model of ATN as a credential disclosure sequence [1]. The participants in a trust negotiation are the client and server. Each side owns a set of credentials, which is denoted by *ClientCreds* and *ServerCreds*, respectively. The credential disclosure sequence is defined as follow:

$$\{C_i\}_{i\in[0,2n+1]} = C_0, C_1, \ldots, C_{2n+1}$$

where $n \in N$, $C_{2i} \subseteq$ ClientCreds, $C_{2i+1} \subseteq$ ServerCreds.

As an example, Figure 1 illustrates a simple trust negotiation occurring between two entities Bob and Alice. Upon requesting access to Bob's service, Bob sends the policy guarding this service to Alice, which states that she must disclose a visa card so that she can be billed for her service usage. Alice has this credential, but cannot be disclosed unless Bob is a member of the Better Business Bureau (BBB), so she asks Bob to reveal his BBB credential. Bob then discloses his BBB credential, which satisfies Alice's policy for her Visa card. After Alice's disclosure of her visa card, trust is established between them, and the service originally requested by Alice is granted.

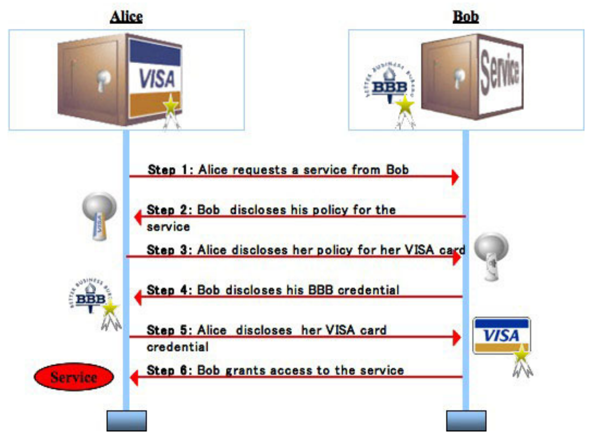

Alice
Bob
VISA
BBB
Service
Step 1: Alice requests a service from Bob
Step 2: Bob discloses his policy for the service
Step 3: Alice discloses her policy for her VISA card
Step 4: Bob discloses his BBB credential
Step 5: Alice discloses her VISA card credential
Step 6: Bob grants access to the service
VISA
Service

**Figure 1.** A simple trust negotiation scenario between Bob and Alice

### 2.2 Related Work

An automated trust negotiation strategy [1, 5] determines the search for a successful trust negotiation sequence based on the policies for sensitive credentials. It controls which credentials are disclosed, when they are disclosed, and which credentials are requested from the other participant. The negotiation strategy also determines when the negotiation is terminated. Winsborough *et al.* [1] proposed two different kinds of negotiation strategies: an eager strategy and a parsimonious strategy. The Eager strategy allows participants to disclose a credential as soon as the policy for this credential is satisfied, while the parsimonious strategy does not allow participants to disclose any credentials until it determines that trust can be established. The former negotiation strategy ensures a successful negotiation in a minimum number of rounds, but may disclose many credentials unnecessarily, which is impractical. The latter strategy can avoid unnecessary credential disclosures, but in the worst case it may bring exponential communication cost when two participants have many credentials and complex policies for these credentials.

Based on these two strategies, Yu *et al.* introduced the Prudent Negotiation Strategy (PRUNES) [6], which is a depth first search of certain areas of an AND/OR tree [9]. PRUNES guarantees no irrelevant credential disclosure, and trust can be established whenever possible based on backtracking. In the worst case, its communication cost and computational complexity are $O(n^2)$ and $O(m*n)$ respectively, where $n$ is the total number of credentials requested and $m$ is the total size of credential policies.

He and Zhu proposed an efficient trust negotiation strategy based on Negotiation Petri Net (SNPN) by combining the characteristics of Negotiation Petri Net architecture with the behaviors of trust negotiations [10]. SNPN's communication complexity is $O(n)$ and computational complexity is $O(nm)$ in the worst case. But the precondition of running this strategy is that each participant needs to have a good knowledge of the other party's policies in advance, which is impractical in an open environment.

Further, Lu and Liu developed an efficient and practical negotiation strategy Deterministic Finite Automation Negotiation Strategy (DFANS) [7] that takes advantages of deterministic finite automata and Ordered Binary Decision Diagrams (OBDDs). In the worst case, DFANS's communication complexity is $O(n)$ and computational complexity is $O(m)$ when not involving cyclic dependencies. But certain cases result in exponentially large OBDDs.

Liu *et al.* [11] proposed a semantically relevant negotiation strategy (SRNS) based on a shared ontology and ontology inference techniques. It guarantees the success of a negotiation whenever it is semantically possible. In the worst case, SRNS's communication complexity is $O$ ($m$), where $m$ is the total number of semantically relevant credentials in a negotiation. But SRNS becomes costly as the size of the ATN ontology grows.

Bertino *et al.* [8] and Squicciarini *et al.* [12] proposed trust tickets and sequence prediction module are explored. The trust tickets support fast interaction for the participants who have previously completed a similar trust negotiation. The sequence prediction module is used when the negotiation that is being carried on shows similarities with previously executed negotiations. Similarity is estimated based on the information collected during previous negotiations.

From the above work, the traditional negotiation strategies conduct negotiations by iteratively communicating both sides' policies to determine a successful negotiation sequence. Thus their computatinal cost and communication cost is based on the number of credentials requested and the size of credential policies. The use of trust tickets and sequence prediction can reduce the sequence generation's computational complex and communication complex to $O(1)$. But they cannot work when there is not an appropriate trust ticket or a sequence to be used. Thus, we propose an approach to improving the efficiency of traditional negotiation strategies taking advantages of the idea of sequence prediction. Differing from Bertino *et al.*, we cache negotiation sequence no matter whether a negotiation is successful or failure. Besides that, we explore the concept of molecule resource, it makes the length of cached sequences is shorter, and more flexible.

## 3 Integration of Sequence Prediction into Traditional Negotiation Strategy

There can be cases where efficiency is the most crucial need, especially in the resource-constrained mobile environment. It is preferable to adopt highly efficient ATN strategies, even if they do not maximize the protection of involved resources. Although trust tickets and sequence prediction proposed by Bertino et al. can replace the traditional negotiation strategies, they cannot work all the time. If there is not an appropriate trust ticket, and the sequence prediction fails, the participants still need to run a traditional negotiation strategy to carry out the negotiation. Thus it is necessary to propose an approach to speed up negotiations when running traditional negotiation strategies. In this section, we developed an approach to integrating sequence prediction into traditional negotiation strategies, speeding up negotiation process when traditional negotiation strategies are performed. Note that, our motivation is not to propose a new negotiation strategy algorithm, but rather to explore a more general-purpose approach for effectively improving the efficiency of existing negotiation strategies.

### 3.1 Cache Negotiation Sequence for Molecule Resources

As mentioned in [8], the same trust sequence can be used several times to perform similar negotiations. The notion of similarity means negotiations for the same resource that are executed by an entity with different participants having similar policies for required resources. After each successful negotiation, participants would cache a negotiation sequence to be directly reused in the subsequent negotiations. A participant caches a negotiation sequence only when the negotiation is successful and uses it either in its entirety or not at all. However, it is not flexible for predicting portions of a negotiation. Besides that, we found some information generated in failure negotiations is also useful. To fully take advantages of historical information, we propose a concept molecule resource. That is, participants cache negotiation sequences for molecule resources defined below.

**Definition 1.** During a trust negotiation, if the sum of credential disclosures and attribute disclosures is greater than or equal to $n$ ($n$ is an integer, $n \geq 1$) between the time a resource is requested and the resource is granted, then this resource is called a molecule resource in this negotiation, which is represented by *mr*.

The value of $n$ is critical to the efficiency of the sequence prediction in our approach. Since the sequence prediction itself consumes the computational and communication resources. Like the scenario in section 3, the negotiation for the employee's card is so simple that the cost of predicting the negotiation sequence may be more than the cost of performing traditional negotiation strategies.

The value of $n$ is critical to the efficiency of the sequence prediction because the sequence prediction itself consumes the computational and communication resources. If $n$ is too small, the cost of predicting a negotiation sequence may be more than the cost of performing traditional negotiation strategies. After a large number of experiments to be shown section 4, we found that the most appropriate value of $n$ for PRUNES is equal to two. That is, at least two resources need to be disclosed between the time *mr* is requested and granted.

Besides the value of $n$, hit ratio of sequence prediction for molecule resource is another crucial factor influencing the efficiency of sequence predictions. To ensure a higher hit ratio, the negotiation sequence to be cached applies to the following rule.

**Rule 1.** Suppose a participant obtains a sequence $seq=\{c_1, c_2, \ldots, c_m\}$ for a molecule resource *mr* after finishing a negotiation, where $m \in N$, $c_i$ ($i \in [1, m]$) is either a local resource or a remote resource. The 3participant would cache *seq* if and only if $c_1$ is a local unprotected resource.

The unprotected resource means the resource can be disclosed whenever requested. A negotiation cannot be successful if there is not an unprotected resource. It is similar to the sequence prediction for molecule resources. If not any a credential is disclosed before sending *seq* to the other participant, the sequence prediction must be failure if the first resource in *seq* is not unprotected. Furthermore, the *seq* will be used to predict the negotiation sequence for the disclosure of remote participant's resource. A participant usually does not have any knowledge about other participants' policies. But he knows his own resources' policies. Therefore, to ensure a higher hit ratio, in our approach a participant only caches the molecule resource's negotiation sequence whose first resource is unprotected in the local side.

A whole trust negotiation sequence would be cached when a negotiation succeeds. To enable flexible prediction for molecule resources, we divide a whole negotiation sequence by molecule resources. Thus both participants cache the trust sequence in the cache memory with the following form when a negotiation succeeds.

**Definition 2.** A cached sequence *scs* is stored as a 5-tuple ($R$(*count*), *mrc*, *ts*, *cond*, *date*), in which $R$ is the resource originally requested to invoke a negotiation, and *count* records the number of times the trust negotiation sequence for $R$ has been successfully used; *mrc* is a set $\{mr_1(count_1, date_1), \ldots, mr_n(count_n, date_n)\}$ containing all the molecule resources $mr_i$ ($1 \leq i \leq n$) in $P_R$, $P_R$ represents the policy for $R$, $n$ is the number of molecule resources in $P_R$, $count_i$ counts the number of times the trust negotiation sequence for $mr_i$ has been successfully solely used, and $date_i$ remembers the last date the cached sequence for $mr_i$ was successfully solely used; *ts* is a trust negotiation sequence divided by molecule resources mentioned in $P_R$, which takes the form $\{s_1, s_2, \ldots, s_m\}$, $s_i$ ($1 \leq i \leq m$) is either a resource name or the form $(mrts)mr$, in which *mrts* is a trust negotiation sequence for *mr* contained in $P_R$; *cond* identifies the conditions under which *ts* can be used; *date* represents the last date *ts* was successfully used.

In this definition, the trust negotiation sequence *ts* for granting $R$ is the sequence composed of all the resource names needed to obtain $R$. For distinguishing the owners of resources in *ts*, the resources disclosed by one who caches the sequence are enclosed by square brackets. The cached sequence for $R$ not only can be wholly used when $R$ is requested, but also can be partially used to predict the trust negotiation sequence for disclosing $mr_i$ contained in *ts*. Therefore, the difference between *count* and $count_i$ is that when *ts* is used as a whole, *count* increases one, $count_i$ remains unchanged; when $mrts_i$ for $mr_i$ is used independently, $count_i$ increases one, *count* and $count_j$ ($1 \leq j \leq n$ and $j \neq i$) remain unchanged. Note that *count* is initialized as one while each $count_i$ is initialized as zero. Sometimes, there may be more than one sequence can be used to predict the negotiaiton for a resource, the parameters *count* and *date* will help users to choose a latest most commonly used sequence. *Cond* can be used to verify whether *ts* can be usable, which is a set of requirements such as credentials, attributes and preferences that have to be satisfied by the other side before *ts* is successfully selected.

Consider the scenario described in the following example:

**Example 1.** An online bank named Bob offers a special rate loan to anyone who has an employee card, a credit score more than 1000, and an age under 35. For Bob, he has a good security process credential and a bank license that can be disclosed to anyone, a credit license that can be released if the other party has a credit report issued by the bank CoS. Alice, who is an applicant, would like to know whether she is eligible for such a special rate loan using her mobile phone. She can prove to anyone that she possesses a driver license, a credit report and an employee card. She considers her credit score (1500) included in the credit report to be a sensitive attribute that can be disclosed only to the one who possesses a bank license and a credit license. Her DoB (1980_5_20) contained in her driver license and employee card cannot be disclosed unless the opposing side has a good security process credential. She deems the issuer (CoS) in her credit report as non-sensitive information.

Referring to example 1, suppose that Alice and Bob find a trust negotiation sequence by running a negotiation strategy. When the negotiation successfully finishes, Bob caches the trust negotiation sequence as the following form (special-rate loan(1), {credit score (1)}, {employee's card, ([bank license], credit report, [credit license]) credit score, [goodSecProcess], age}, employee's card, *date*) in which the credit score is a molecule resource, and the sequence for the molecule resource consists of the credentials bank license, credit report and credit license. Since the bank license and credit license are disclosed by Bob, each of them is enclosed by a square bracket. The special-rate loan is only for the one who has a job, so *cond* asks the other participant to reveal her employee's card. *cond* is verified before choosing *ts*. *date* represents the date this negotiation succeeds.

To fully take advantages of historical information, participants would cache useful information even when a negotiation fails. Thus, trust negotiation sequences for molecule resources are cached in the following form when a negotiation fails.

**Definition 3.** A cached sequence *fcs* is stored as a 4-tuple (*mr*, *fts*, *count*, *date*), where *mr* is a molecule resource contained in the policy for the originally requested resource invoking a trust negotiation; *fts* is a trust negotiation sequence for disclosing *mr*; *count* remembers the number of times *fts* has been successfully used; *date* represents the last date *fts* was successfully used.

*fcs* is a cached sequence for a molecule resource in the policy for the originally requested resource, and each molecule resource itself can be *scs.cond*. Therefore, unlike *scs*, the element *cond* is unnecessary in *fcs*. For distinguishing the owners of resources in *fts*, the resources disclosed by the one who caches the *fcs* are enclosed by square brackets. Note that *count*'s initial value in *fcs* is zero which is equal to *scs*.

Consider the scenario described in example 1, suppose that Alice is older than 35. Although the trust negotiation fails finally, the negotiation for Alice's employee's card and credit score is successful. Because credit score is a molecule resource in the policy for the special-rate loan, and according to Rule 1, the first resource in its negotiation sequence is Bob's unprotected credential bank license. Bob caches the sequence *fcs* = (credit score, {[bank license], credit report, [credit license]}, 0, *date*). *date* represents the date when the negotiation generating *fts* finishes.

## 3.2 Predict Sequence During Negotiations

Caching trust negotiation sequences for molecule resources makes the integration of sequence predictions into the traditional negotiation strategies applicable. Figure 2 shows the pseudo code of integrating sequence prediction for molecule resources into traditional negotiation strategies.

We take the server side as an example. Suppose a server receives a message, he determines a response *m* by employing a traditional negotiation strategy. If *m* is a request for a resource *c*, the server checks whether there exists a negotiation sequence for *c* in his cache memory. If yes, he selects the most commonly used sequence *fs* from his cache memory, and sends it to the client for predicting the negotiation for granting *c*. To further improve hit ratio of sequence prediction, the server first sends a name list of local unprotected resources $Cred_{unprotected}$ to the client. When the client receives *fs*, she determines whether to accept *fs* based on her policies, $Cred_{unprotected}$ and the credentials that have been granted during the negotiation. If the client agrees with the sequence, she sends *true* to the server, and the negotiation process is accelerated by omitting the negotiation for granting *c*. Otherwise, the server draws another negotiation sequence using the same way whenever possible. Since prediction process consumes computational and communication resources. We restrict the maximal number of times of prediction for each resource is three. The negotiation process is presented in Figure 3.

## 4 Performance Evaluation

We have performed our experiments on a Huawei C8815 Smartphone running Android operating system 2.3. We take PRUNES as an example since it is a well-known and representative negotiation strategy. For conciseness, PRUNES incorporating our approach is called Pred-PRUNES.

In our experiments, each server has forty resources such as credentials, services and so on, and the number of clients' credentials varies between ten and fifteen. Their policies are generated randomly. For each policy, there are at most two clauses, and for each clause, a server has at most four remote credentials and a client has at most three remote credentials. For guaranteeing similarities between policies for a certain credential, requirements in the policies for a same credential are randomly chosen from the certain six randomly generated credentials. Each of the server and client has three unprotected local credentials.

First, we determine the most appropriate value *n* for Pred-PRUENS mentioned in Definition 1. For each different value *n*, ten trials have been executed, and for each trial, one thousand clients initiate negotiations to a server. We recorded the execution time of negotiations that succeed using both Pred-PRUNES and PRUNES. The performance is measured in terms of CPU time (in milliseconds).

Figure 4 compares the average of the execution time obtained from the ten trials when varying the value of *n* from two to four. From this figure, we can observe the average execution time cost by Pred-PRUNES increases with the value of *n*. Table 1 shows the statistical results of the two-samples t-test for Figure 4. As shown in table 1, the differences in mean between Pred-PRUNES and PRUENS (mean=-427.752, sig=.000) is the greatest when *n* is equal to two. When *n* is equal to three and four, the differences are not significant. This is because the number of negotiation sequence for molecule resources cached in the cache memory decreases when *n* increases. If the number of cached sequence is fewer, the probability that the server gets the appropriate sequence for prediction will be lower. Furthermore, the sequence prediction itself consumes the computational and communication resources, thus the hit ratio of sequence prediction is critical to the efficiency. However, the more resources required from a client are contained in a cached sequence, the harder the sequence would be accepted by a client. The number of client's resources contained in a sequence increases when the value of *n* increases, leading to a lower hit ratio. We thus have some evidence concluding that the most appropriate value of *n* for Pred-PRUNES is equal to two.

When *n* is equal to 2, Figure 5 shows the distributions of usage number of times of different lengths of cached sequences, in which *m* represents the length of cached sequences. From this figure, we can observe that cached sequences with the shortest length used most frequently. It is reasonable. The successful usage of a cached sequence needs to satisfy that the policy for each resource contained in the sequence is satisfied. Thus a cached sequence involving fewer resources is easier to be successfully used.

Then, we compare the negotiation efficiency between Pred-PRUENS and PRUNES when *n* is equal to two. The efficiency of a negotiation strategy involves two aspects [3], which are the computational cost and communication cost. The performances of computational cost and communication cost have been measured in terms of CPU time (in milliseconds) and the number of negotiation rounds respectively. Since policies are generated randomly, not all the negotiations can be successful. According to our experiments,

```
pred_strategy ($L$, $R$, $Cred_{unprotected}$)
Input: $L$ is the set of local resources.
            $R$ is the resource to which access was originally requested in a negotiation.
Output: the negotiation result and a trust negotiation sequence.
Let $M$ be an empty disclosure message sequence.
$r$←NOT_TERMINATED.
If ($R$ is a local resource)
Send $Cred_{unprotected}$ to the other party.  // $Cred_{unprotected}$ is a set of names of local unprotected resources.
then $r$← prediction_for_molecule resource ($M$, $L$, $R$).
If ($r$ = TRUST or $r$ = DISTRUST)
then return ($r$, $M$).
While ($r$ = NOT_TERMINATED)
Receive message $m$ from the other party.
If ($m$ is a sequence prediction $fs$) then
    $ack$←check whether the trust negotiation sequence $fs$ can be accepted according to the policies for $L$, which can
    either be $true$ or $false$.
    Send $ack$ to the other party.
    Continue.
$M$←$M \cup m$.
$r$←check_for_termination($m$, $R$).
If ($r$ = TRUST or $r$ = DISTRUST)
then return ($r$, $M$).
$r$← prediction_for_molecule resource ($M$, $L$, $R$).
If ($r$ = TRUST or $r$ = DISTRUST)
then return ($r$, $M$).
End of pred_strategy.

check_for_termination($m$, $R$)
If ($m = \phi$ )
then return DISTRUST.
If ($R \in m$)
then return TRUST.
Return NOT_TERMINATED.
End of check_for_termination.

prediction_for_molecule resource ($M$, $L$, $R$)
$S_m$←local_strategy($M$, $L$, $R$).
//call a negotiation strategy adopted by a system, such as eager strategy, PRUNES and so on. $S_m$ contains candidate
messages that the local strategy suggests.
Choose any single message $m$ from $S_m$.
If ($m$ is a request for a resource $c$) then
pred_count=0.  // pred_count counts the number of times of sequence prediction for $c$
Load a set of sequences $FS$ from local cache memory, in which $fcs.mr = c$ or $scs.mrc.mr = c$.
Choose a sequence $fs$ whose $count$ is the largest in $FS$.
While ($fs \neq \phi$ & pred_count<3)
Send $fs$ to the other party.
$ack$←wait for a response for $fs$ from the other party.
If ($ack$)
$m$←$m \cup fs$.
$fs.count$++, $fs.date$=current date.
Break.
   pred_count++.
Choose the next latest most commonly used sequence $fs$ from $FS$.
$M$←$M \cup m$.
$r$←check_for_termination($m$, $R$).
Send $m$ to the other party.
Return $r$.
End of prediction_for_molecule resource.
```

**Figure 2.** The pseudo code for the integration of sequence prediction for molecule resource into traditional negotiation strategies

approximately seventy percent negotiations can succeed. Thus Four hundred negotiations are executed in which the execution time and the number of negotiation rounds of the first two hundred and twenty-five successful negotiations are recorded. We divide these negotiations into fifteen groups in sequence. For each group, we compute the average execution time and number of negotiation rounds. The results are shown in Figure 6. Each data point in this figure represents the average of fifteen negotiations. The reason we do not choose more number of negotiations such as twenty or thirty negotiations to represent a point is to better reveal the variation trend of Pred-PRUNES's efficiency. At the beginning, since the server's cache memory has few sequences, Pred-PRUNES's negotiation efficiency is almost the same as PRUNES's. But the difference between their efficiency grows with the number of negotiations.

These experiment results show that our approach can effectively enhance the efficiency of PRUNES when $n$ is set as two.

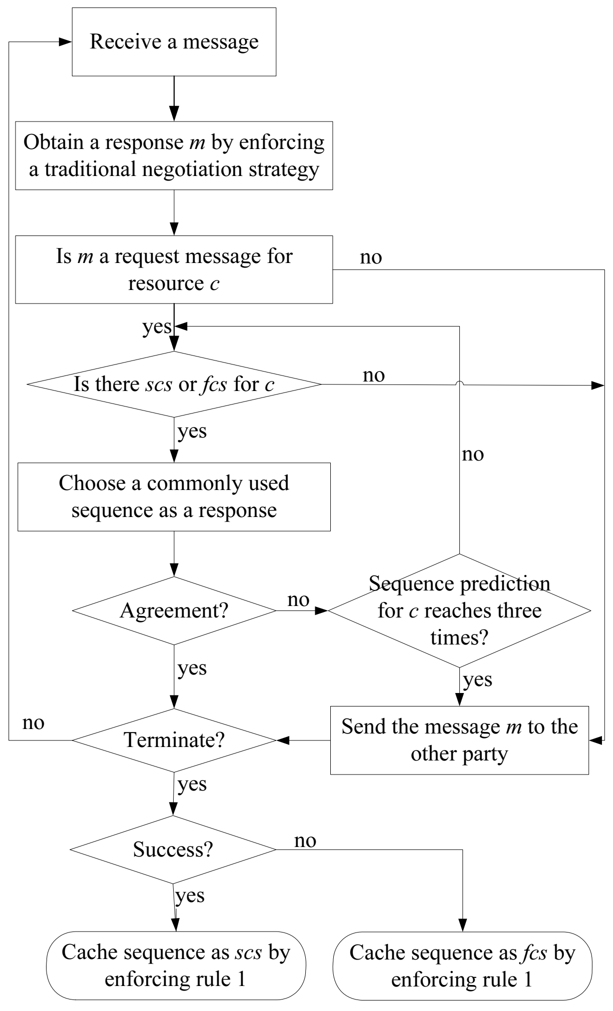

Receive a message
Obtain a response *m* by enforcing a traditional negotiation strategy
Is *m* a request message for resource *c*
no
yes
Is there *scs* or *fcs* for *c*
no
yes
no
Choose a commonly used sequence as a response
Agreement?
no
Sequence prediction for *c* reaches three times?
yes
yes
no
Terminate?
Send the message *m* to the other party
yes
Success?
no
yes
Cache sequence as *scs* by enforcing rule 1
Cache sequence as *fcs* by enforcing rule 1

**Figure 3.** The negotiation process using our approach that integrates sequence prediction for molecule resources into traditional trust negotiation strategy

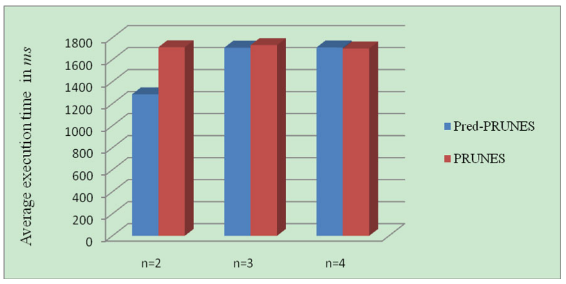


**Figure 4.** Comparisons of average execution time cost by Pred-PRUNES and PRUNES when $n$ =2, 3, 4

**Table 1.** Statistical results of the two-samples test

| | Mean | Std. Deviation | $t$-value | Sig. (2-tailed) |
|---|---|---|---|---|
| $n$=2 | -427.752 | 1178.239 | -17.509 | .000 |
| $n$=3 | -22.288 | 1071.435 | -.729 | .466 |
| $n$=4 | 9.092 | 1573.226 | .275 | .784 |

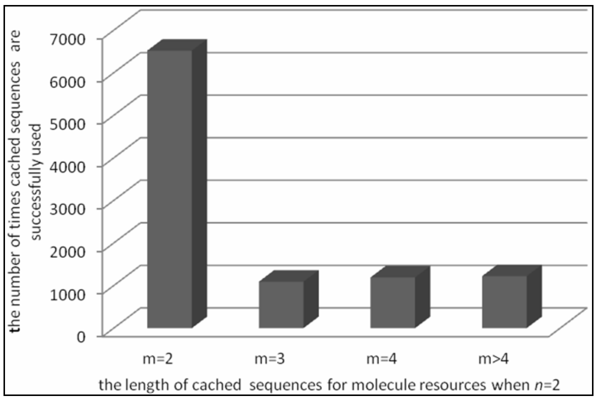


**Figure 5.** Distributions of usage number of times of different lengths of cached sequences when $n$=2

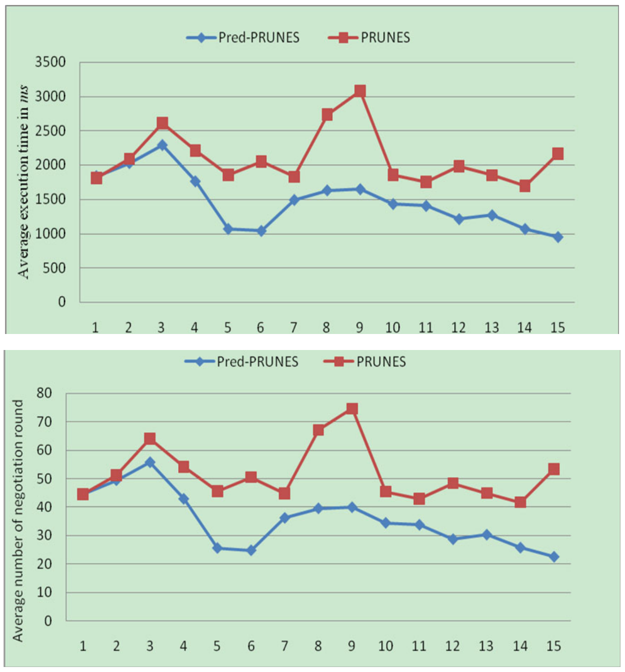


**Figure 6.** Comparisons of average execution time and number of rounds of the first two hundred and twenty-five negotiations that succeed using both PRUNES and Pred-PRUNES when $n$ =2

## 5 Conclusion

Automated trust negotiation has shown to be a successful approach to establishing mutual trust between strangers wishing to share resources or conduct business. Carrying out trust negotiations may need expensive computational resources and much communication resources. But in some circumstances, efficiency is the most crucial need, especially in light of widespread use of mobile devices. It is preferable to adopt highly efficient ATN strategies, even if they do not maximize the protection of involved resources. In this paper, we propose an approach to integrating sequence prediction for molecule resources into traditional negotiation strategies, speeding up negotiation process during the enforcement of traditional negotiation strategies. We explore the concept of molecule resource to help negotiators determine which sequences need to be cached, introduce the rule for caching sequence for molecule resources, and develop the method to predict sequences for molecule resources during negotiations when performing traditional negotiation strategies. To validate the effectiveness of our approach, we take PURENS, the representative traditional negotiation strategy, as an example and compare the efficiency between PRUNES and Pred-PRUNES (incorporates our approach) on the mobile platform. Our experimental results show that our approach dramatically enhances PRUNES' s negotiation efficiency.

## Acknowledgements

This research is supported by the National Natural Science Foundation of China (No. 71571082, 71101061, 71171093, 71471073), self-determined research funds of CCNU from the colleges' basic research and operation of MOE (No. CCNU15A02046, CCNU14Z02016).

## References

[1] W. H. Winsborough, K. E. Seamons, V. E. Jones, Automated Trust Negotiation, *DARPA Information Survivability Conference and Exposition*, Hilton Head, SC, 2000, pp. 88-102.

[2] B.-L. Liu, F. Ge, F. Xiao, A Peer-to-peer Framework for Accelerating Trust Establishment in Automated Trust Negotiation, *Journal of Internet Technology*, Vol. 11, No. 5, pp. 643-650, September, 2010.

[3] A. C. Squicciarini, E. Bertino, A. Trombetta, S. Braghin, A Flexible Approach to Multisession Trust Negotiations, *IEEE Transactions on Dependable and Secure Computing*, Vol. 9, No. 1, pp. 16-29, January-February, 2012.

[4] E. Caprin, Y. Zhang, Negotiation Based Framework for Attribute-Based Access Control Policy Evaluation, *the 7th International Conference on Security of Information and Networks*, Glasgow, UK, 2014, pp. 122-127.

[5] T. Yu, M. Winslett, K. E. Seamons, Interoperable Strategies in Automated Trust Negotiation, *the 8th ACM Conference on Computer and Communications Security*, Philadelphia, PA, 2001, pp. 146-155.

[6] T. Yu, X. Ma, M. Winslett, PRUNES: An Efficient and Complete Strategy for Automated Trust Negotiation over the Internet, *the 7th ACM Conference on Computer and Communications Security*, Athens, Greece, 2000, pp. 210-219.

[7] H. Lu, B. Liu, DFANS: A Highly Efficient Strategy for Automated Trust Negotiation, *Computers & Security*, Vol. 28, No. 7, pp. 557-565, October, 2009.

[8] E. Bertino, E. Ferrari, A. C. Squicciarini, Trust-x: A Peer-to-peer Framework for Trust Establishment, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 16, No. 7, pp. 827-842, July, 2004.

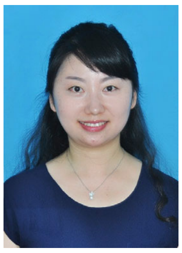
[9] R. Shinghal, *Formal Concepts in Artificial Intelligence: Fundamentals*, Chapman & Hall, 1992.

[10] Y. He, M. Zhu, A Complete and Efficient Strategy Based on Petri Net in Automated Trust Negotiation, *the 2nd International Conference on Scalable Information Systems*, Suzhou, China, 2007, pp. 1-7.

[11] X. Liu, S. Tang, Q. Huang, Z. Yu, An Ontology-based Approach to Automated Trust Negotiation, *Computer Standards & Interfaces*, Vol. 36, No. 1, pp. 219-230, November, 2013.

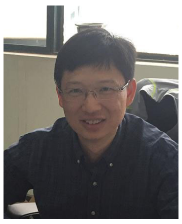
[12] A. C. Squicciarini, E. Bertino, E. Ferrari, F. Paci, B. M. Thuraisingham, PP-trust-x: A System for Privacy Preserving Trust Negotiations, *ACM Transactions on Information and System Security*, Vol. 10, No. 3, pp. 1-48, July, 2007.

## Biographies

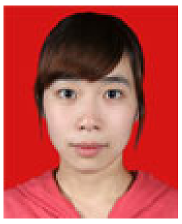


**Bailing Liu** is an associate professor in the School of Information and Management, Central China Normal University, China. She received the Ph.D. degree in computer science from Huazhong University of Science and Technology, China. Her research interests include information security and privacy, management information systems.

**Chao Lei** is a graduate of Central China Normal University, China. His major is management science and engineering.



**Yanhui Li** is a professor and the vice dean of School of Information Management at Central China Normal University, China. He received the Ph.D. degree in management science and engineering form Huazhong University of Science and Technology, China. His research interests are in the areas of logistics system engineering and supply chain management.



**Liting Liang** is a graduate of Central China Normal University, China. Her major is management science and engineering.