J. Parallel Distrib. Comput. 74 (2014) 2141–2151

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

# Privacy-preserving and verifiable protocols for scientific computation outsourcing to the cloud

Fei Chen [a], Tao Xiang [b,*], Yuanyuan Yang [c]

[a] Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong, China
[b] College of Computer Science, Chongqing University, Chongqing, China
[c] Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY, USA

## HIGHLIGHTS

- Point out a problem in a previous linear equation outsourcing protocol.
- Propose new and efficient protocols for *linear equation solving* and *linear programming* outsourcing to the cloud.
- New protocols improve the performance of previous protocols significantly.
- Experimental results validate the new protocols.

## ARTICLE INFO

*Article history:*
Received 19 September 2013
Received in revised form
23 November 2013
Accepted 27 November 2013
Available online 8 December 2013

*Keywords:*
Computation outsourcing
Linear equation solving
Linear programming
Cloud computing
Distributed computing

## ABSTRACT

Computation outsourcing to the cloud has become a popular application in the age of cloud computing. Recently, two protocols for secure outsourcing scientific computations, i.e., linear equation solving and linear programming solving, to the cloud were proposed. In this paper, we improve the work by proposing new protocols that achieve significant performance gains. For linear equation solving outsourcing, we achieve the improvement by proposing a completely new protocol. The new protocol employs some special linear transformations and there are no homomorphic encryptions and interactions between the client and the cloud, compared with the previous protocol. For linear programming outsourcing, we achieve the improvement by reformulating the linear programming problem in the standard and natural form. We also introduce a method to reduce the key size by using a pseudorandom number generator. The design of the newly proposed protocols also sheds some insight on constructing secure outsourcing protocols for other scientific computations. Comparisons between our protocols and the previous protocols are given, which demonstrate significant improvements of our proposed protocols. We also carry out numerical experiments to validate the efficiency of our protocols for secure linear equation solving and linear programming outsourcing.



## 1. Introduction

Computation outsourcing enables a client with relatively weak computing power to give out a computational task to some servers with more powerful computing power. Then the servers are expected to fulfill the computation and return the result to the client. Such a computing model is especially suitable for cloud computing, in which a client can buy a computation service from a cloud service provider in a pay-per-go manner. There have been several distributed computing projects that employ this computing model, such as SETI@HOME [34], FOLD@HOME [35], etc. In addition, in wireless sensor networks, this computing model can also help a computationally weak sensor node to outsource complex computations to a more powerful data sink [33].

In general, there are three key requirements in computation outsourcing, which are correctness, verifiability and privacy [2,6,11,30–32]. Correctness means that as long as the client and the server both follow the protocol, the computation outsourcing can be done. Verifiability refers to that the client has a way to figure out whether the returned result is correct or not on its own. The verification time is expected to be much shorter than the time needed to solve the outsourced problem; indeed, if not, there is no motivation for outsourcing the computation. Privacy means that the server will not gain the knowledge of the client's input and output data through computation. Meeting all these three requirements is a challenging task. Readers may refer to Section 6 for more related work in this area.

In this paper, we focus on secure scientific computation outsourcing. Recently, two proposals have been proposed for secure

* Corresponding author.
*E-mail address:* txiang@cqu.edu.cn (T. Xiang).

0743-7315/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jpdc.2013.11.007

scientific computation outsourcing. The first protocol [31,32] is designed to help a client with weak computing power to outsource the linear equation (LE) solving work to a cloud server, which has powerful computing power, in a secure, correct and verifiable way. The second protocol [30] is constructed to help outsource a linear programming (LP) work to a cloud server. Linear equation solving and linear programming are two frequently used tools in modeling various engineering problems and thus have a lot of applications. For example, linear equation is employed to model linear systems [22] and linear programming is used to solve the maximal network flow problem [17]. Therefore, understanding how to employ the cloud to solve these problems is important and interesting especially when the client has weak computing power.

The basic idea of the protocol proposed in [31,32] for outsourcing linear equation (LE) solving can be summarized as follows. Given a computation problem instance, it is first encrypted to another problem, which will be sent to the cloud for finding a solution. Then, the client and the cloud interact in a way that only the client knows whether a convergent solution has been found. After several rounds of interaction, the client will finally obtain a satisfactory solution for the problem. Verification is done by checking whether the solution is indeed correct. In this protocol, the problem solving takes $O(n^3)$ time while the verification only takes $O(n^2)$ time, where $n$ is the size of the problem outsourced. Clearly, this is a case that verifying a solution is easier than finding a solution. In many applications, the former is a polynomial problem and the latter may be an NP-hard problem. For example, verifying a subgraph is a clique with size at least $k$ is easy while finding such a clique is NP-hard [10]. As will be seen later, this protocol could be improved in several ways. First, in the protocol there exists an attack to break the input privacy. The attack can recover partial of the client's input data. The weakness of the protocol lies in an inappropriate use of the Paillier public-key cryptosystem. Second, the interaction rounds between the client and the server could be further reduced. Third, the computational overhead for the client could also be improved. In this paper, we first show the attack and then improve this work by proposing a new protocol for the linear equation solving outsourcing problem, which will be presented in Section 3.

The basic idea of the protocol in [30] for outsourcing a linear programming (LP) computation can be briefly described as follows. First, a transformation is applied to the original LP problem to form a new LP problem, where the solutions of both problems have some relationship that is only known to the client. Then the new LP problem is outsourced to the cloud; the cloud solves the transformed LP problem and its duality problem; and then returns the solutions to the client. Finally, the client verifies whether the returned solution is correct or not using the well-known LP duality theorem [4]. This protocol could also be improved in several ways. First, the formulation of the LP problem in the solution is not a standard LP problem as in many LP solvers and textbooks on optimization. We find that the secure linear programming outsourcing problem can also be done in the standard form by reformulating the linear programming model. After the reformulation, the performance of the protocol can be significantly improved. This way, the length of the secret key, computation overhead of the client, and the communication between the client and the server can be greatly reduced. Second, besides the improvement in the problem reformulation, the length of the secret key can also be reduced further by using a pseudorandom number generator [3,16].

We summarize our contribution in this paper as follows.

- We propose a new protocol for linear equation (LE) solving outsourcing. The new protocol is completely different from that in [31,32], and has much better performance.
- We reformulate the linear programming (LP) problem considered in [30] into a standard form, which is widely adopted by many LP solvers and textbooks, and improve the performance of the previous protocol.
- We validate our new protocols through extensive experiments. All the experimental results can be reproduced using our source code which is posted online.

The remainder of the paper proceeds as follows. Section 2 formulates the LE and LP outsourcing problems and briefly describes the previous protocols. Section 3 proposes a new protocol for the secure LE outsourcing problem. It also gives the security analysis and discusses the detailed improvements. Section 4 presents the improvements on the secure LP outsourcing problem by reformulating the problem, together with the security analysis and discussion of detailed improvements. Section 5 analyzes the performance of the new protocols in both theory and experiments. Section 6 reviews recent related work on computation outsourcing. Finally, we conclude the paper with some remarks in Section 7.

## 2. Problem formulation and previous protocols

In this section, we formulate the problem and describe the previous solutions to the secure linear equation solving and linear programming outsourcing problems in [30–32]. In the meantime, we also discuss where the previous solutions could be improved, which motivates this paper.

### 2.1. System model and design goals

Fig. 1 shows our system model. There are two main entities: a client and a cloud. The client has an LE or LP problem $\Phi$. Due to the limited computing power, the client wants to outsource problem $\Phi$ to the cloud. To protect privacy, the client first encrypts the original problem $\Phi$ with a secret key $K$ to obtain a new problem $\Phi_K$. The encryption does not change the problem structure, but transforms the problem into another related LE or LP problem. Then, the new problem $\Phi_K$ is given to the cloud for finding a solution. The cloud solves the encrypted problem $\Phi_K$ and then returns the solution to $\Phi_K$ together with a proof that the solution is correct. Using the secret key $K$, the client verifies the solution. If it is correct, the client decrypts it to get the solution for the original problem $\Phi$. If not, the client rejects the solution. The client and the cloud could also have several rounds of communication to solve the problem.

We assume the cloud may be malicious, which is standard in the community [2,6,11,30–32]. It may be dishonest in the computation by returning a false solution to the client's problem. There are huge economic incentives for the cloud to behave this way. The cloud may tend to provide most of its resources to serve those clients who pay much more to the cloud. Besides, the cloud may also return a false solution due to various software and hardware failures. The cloud may also try to learn what the client's data is, in the input or output (i.e., the solution of the outsourced problem), because the data may contain valuable information. In summary, rational users who value their data very much need to handle the case when the cloud is malicious.

A secure LE and LP outsourcing protocol should have the following properties.

- *Correctness.* If the client and the cloud both follow the designed protocol, the returned solution should be decrypted to a correct solution of the original problem.
- *Soundness.* If the cloud tries to fool the client, its success probability that the client accepts a false solution is very small.
- *Privacy.* After returning the solution to the client, the cloud's knowledge on the original problem and the corresponding solution, which we call input privacy and output privacy, respectively, is very small.