Trading Processor Cycles for Communication

Arnab Paul * Rahul Ray † Umakishore Ramachandran ‡

Georgia Tech Technical Report *GIT-CERCS-04-05*

Abstract

"Trade computation for communication whenever possible" has been the conventional wisdom to save bandwidth and power in wireless domain. We glanced at the merging trends of technology and the applications, and identified a number of areas where the processor cycles can be traded for network bandwidth. These areas are - file transfer, shared document edits, remote authentication, and verification of remotely available codes. We collect and present some of the theory-paradigms that have given birth to results potentially usable towards saving bandwidth at the cost of computation. We hope to make a case for incorporating the insights collected in this paper from various fields of complexity theory, into the design of network protocols for the future. In the penumbra of this over-arching goal, we also suggest how the engine of "Probabilistic Checking of Proofs" (PCP) can be used for remote authentication purposes, resulting in much smaller network overhead and leading to very efficient usage of bandwidth and power. While this seems to be an interesting route, we then present the algorithmic challenges that need be solved and argue why such a theoretical cranking may be just as worth.

1 Introduction

Exchange of information between two (or multiple) parties can be effected in various modes. On one extreme, the entire information is generated at one site and then disseminated to the rest of the group. In another extreme, every party (if possible) computes the information independent of the others and no one needs to go to the network. Real life, however, often presents scenarios situated in the middle, i.e., when parties hold partial information and they need to both *compute and communicate*. In fact, many a times, it is possible that computation can be traded off with communication and *vice versa*. It so turns out that the possibility of such a trade-off can be used to various advantages to handle resource limitations more efficiently. In this paper we explore one direction for this trade-off, *viz.* trading computation for communication. First, we survey why and when this could be helpful. And next we will suggest some theoretical techniques that, although awaiting certain mathematical and algorithmic challenges to be met, could prove extremely helpful.

*College of Computing, Georgia Tech, arnab@cc.gatech.edu
†Max-Planck-Institut fuer Informatik (MPII), rahul@mpi-sb.mpg.de
‡College of Computing, Georgia Tech, rama@cc.gatech.edu

Why Save Communication?

Although modern wireless technology has advanced quite rapidly, it is yet to catch up with the remarkable speed of the CPUs that have been achieved. The primary obstacle behind this lag is the presence of natural physical noise that any wireless signal has to wade through, especially in the domain of long range transmission. Consequently, still now, for many wireless applications, bandwidth is a more priced resource than the processor. Moreover, as computing is becoming ubiquitous in nature, and more and more pervasive applications are surfacing, there is a plethora of devices in the market that are small, wearable or hand-held, connected on wireless and operating on battery; for such devices power consumption is quite critical. Sadly, long range wireless transmission is extremely expensive compared to on-chip computing as far as the currency of energy is concerned. For example, according to an observation made by Pottie and Kaiser, that dates back couple of years, transmission of one bit over a long range wireless is equivalent to few thousands of cycles in modern processors [10]. If the scenario has changed in next few years, it is only to the favor of processing power and speed.

Where Can Communication be saved

Bandwidth and power are the two motivating factors behind conserving communication for mobile wireless systems. There are numerous situations in which communication can be saved by additional computation; here is a short list:

- Compression while transferring files : This is the most common scenario. Large files can be compressed, the

overhead is that of compression and decompression and the gains are dependent upon the efficiency of the algorithms used as well as the exact input.

- Document Exchange: Consider two parties A and B updating a single document. They may hold slightly different version of the same file; if A wants to know the contents of B's document, it may be possible to smartly compare their edits with a much lower communication overhead than transferring the whole file.

- Remote Authentication: This is a scenario which is becoming extremely commonplace given the application and technology trends. Remote users need to be authenticated quite often, and the paradigm of password based authentication is gradually being replaced by other authentication mechanisms, such as biometrics or key-stroke dynamics and so on. These authentication keys are quite large in size and typically the applications require authentication on a continual basis. Naive implementation of such systems therefore leads to huge communication overhead resulting just from the act of authentication alone. While the large size and high entropy of such keys are quite desirable security features, they poses the bandwidth bottleneck. It may be possible to improve the bandwidth requirement of such schemes by carefully modifying the associated verification protocol, which saves the bandwidth, yet at the same time, retains the benefits of a high-entropy authentication token. A point to note here is that simple cryptographic hashing of the key will not work. Because, typically, the biometric patterns generated by a person at different times and to different sensors are not identical, rather close matches. However, a cryptographic hash function does not preserve such proximities and therefore the distance between two patterns cannot be obtained by comparing their hash digests.

- Verification of Mobile Codes and Remote Computations: Internet cookies are no strangers to any one now. However, installing programs downloaded from remote locations can be quite sensitive to the security and privacy issues. It would be really nice if a remote code came with a certification that it does not perform harmful action. As of today, the only certification attached to such a code is the trustworthiness of the party from which it has been downloaded. However, a more convincing proof will be a complete listing of its behavior on different input condition that a user can verify. Quite naturally, such a proof will be really long will take a cause the user a long time to verify. A similar situation would arise in cyber-foraging which is often discussed as an upshot of computing becoming pervasive. In cyber-foraging, a small device with limited computation hands over a heavy-duty processing to a larger machine. When the result is handed back, it is not clear how efficiently the small device could be assured of the correctness of the result. Again, a proof, which would be unreasonably long, could be attached, but the verification of such a proof will not only eat away all the processor cycles that were to be saved at the first place, but will also choke the bandwidth completely because of its exponential size.

Goals of this paper

The ambition in this position-paper is four-fold. First, we start on the note of a conventional wisdom that communication should be purchased at the cost of additional processing where bandwidth and power are scarce, and identify the areas falling within the scope of such a trade-off. Second, we collect and present some of the theory-paradigms (*viz.*, Communication Complexity Theory, Randomized Approximation Algorithms, and Probabilistic Checking of Proofs) that have given birth to results potentially usable in this regard. And we believe that such paradigms have a lot more to offer to the applied-research community. Third, we make an interesting observation that the techniques of Probabilistic Checking of Proofs can be exploited to develop a new remote authentication protocol that results in huge savings in bandwidth(and power). We show how to transform a remote authentication problem to benefit from the proof-checking framework. And finally, we bring out the algorithmic challenges that need be handled. In part, these hurdles are theoretical, and in part, these are the implementation difficulties of an extremely messy and intricate system.

We first present a quick and cursory survey of how in the various situations discussed above, communication can be saved. While data compression is too diverse in nature over various application domains, the subject is quite well-studied at the same time, and hence lies outside the scope of this article. We briefly mention some of the results derived by the theoreticians in the domain of document exchange. One of our goals is to stimulate the interest of applied research community to integrate these results into real life systems. Such a task is often quite non-trivial and comes only after overcoming a series of engineering hurdles. Next, we discuss the issue of remote authentication. Here we first point out an interesting solution, *viz.* Probabilistic Checking of Proofs (PCP) [1] that can be applied as a generic mechanism to save bandwidth and power. For Remote Authentication, we propose a new protocol that uses the PCP-trick so that the entire authentication-key need not be transmitted. Instead,

a very small fraction of the key would be enough for the verifier at a remote end to validate a user's identity.

The paradigm of proof checking has deeper connections with complexity theory and error correction. In a nutshell, any problem in NP [1] can be cast as a problem where a prover is producing some proof and a verifier is verifying the same, and in this process the verifier is looking into only a small fraction of it instead of the whole proof. This is very counter intuitive, and will be discussed with little more details in section 3. Authentication however, seems to readily fit into a prover-verifier framework. But we reformulate it in a slightly different way in section 4. The reason for reformulating the problem is to make it resemble the prover-verifier version of an NP problem. That helps us use the trick of checking only a small fraction of bits in the proof, i.e., in our case, it suffices to transmit only a small fraction of a large authentication key over the network. It is worth mentioning that **P**CP builds upon the results from Theory of Error Correction which in turn is based on polynomials over mathematical structures called Finite Fields. The techniques used in probabilistic checking of a proof relies on a bunch of other results related to many properties of low degree polynomials [12, 11, 4, 6]. The interesting challenge from a systems point of view is to integrate these theoretical notions into an implementable authentication system.

The rest of the paper is structured as follows. The next section describes the paradigm of communication complexity; we discuss very briefly the main objective of this theory and how it connects to the areas where communication can be saved at the cost of additional computation. In section 3 we introduce the preliminary concept of a prover-verifier game that is at the basis of PCP. In the same section we define the notion of PCP as well. In section 4, we cast the remote authentication problem as an instance of prover-verifier game and demonstrate how the fruits of PCP-theorem can be harnessed in that setting. Our discussion on PCP culminates in section 5 where we present the PCP-based protocol for remote authentication. In section 6 we present the algorithmic design challenges that arise in the context of implementing PCP-based protocols. We finally conclude in section 7.

2 Communication Complexity

Communication Complexity Theory [5] is a relatively younger branch of theoretical computer science that offers a different angle to understand many complexity theoretic questions such as lower bounds of problems, circuit depth etc. However, researchers gradually realized the immense scope that the subject opens up even for practical real life problems. In this section, we start with a quick tour of Communication Complexity, and then discuss how the fruits of this theory can be used for areas such as Document Exchange and Remote Authentication. Fundamentally we deal with the following abstract question :

Suppose A and B are two parties trying to compute a function $f(x, y)$ of two variables x and y. A has x in possession, but not y. Similarly, B has y, but not x. There is no limitation assumed on the computational power of A and B. Without any loss of generality we can assume that the size of these two inputs are same, *i.e.*, $|x| = |y| = n$. The question is: "what is the minimum number of bits that need to be exchanged between A and B so that both can compute f" ?

An obvious and trivial answer is n, because both A and B can send their respective inputs to each other and thereafter compute the function privately. But it turns out that depending on the nature of the function (f) to be computed, one can take smarter steps and reduce the required communication significantly. As a trivial, yet illustrative, example, if x and y are two lists of numbers, ($[x_1, \ldots x_n], [y_1 \ldots y_n]$) and the function f to be computed is the average of all these numbers, i.e., $f(x, y) = \frac{\Sigma_i x_i + \Sigma_i y_i}{2n}$, then quite clearly there is no need for sending the complete lists across the network.

The basic formulation of the communication complexity question, however abstract, directly appeals to the need for saving communication. Drawing upon our discussion in section 1, these insights can be applied to many problems that we mentioned. In particular, Document Exchange and Remote Authentication are two areas where it has been applied quite successfully, although in a *randomized* setting. Randomized communication complexity in principle deals with the same question, except the fact that guarantees for the answers are relaxed to high-probability cases as opposed to complete determinism. For example, we may ask, whether two strings x and y are identical with a high probability guarantee, say 0.99. Another possibility is that we want an approximation of the function at the cost of complete determinism. For example x and y are two images and the function f is a distance between these images quantifying how different they are; in this case, it may be useful just to compute an approximate distance $\hat{f} \approx f$ provided communication can be saved substantially.

Now we present some examples of where the notion communication complexity has been applied. One of them is Document Exchange. In this case, it is mostly not necessary to transfer whole documents between two parties. This problem has been studied by quite a few researchers. For example, Orlitsky [7] showed that this

[1] The class of problems solvable by Nondeterministic Turing Machine in Polynomial Time

can be achieved in $\Omega(h.\log n)$ time provided the two files do not differ in more than h positions (the number of positions where they differ is also known as the Hamming Distance). In a more recent result, Cormode *et al.* showed that it is useful to conceive of a new distance between the documents, called the *edit distance* which quantifies the number of edits necessary to transform one copy of the document to the other [3]. The edit distance is a metric. However, instead of directly computing the edit distance, it turns out easier to transform this distance into Hamming Distance and then proceed thenceforth.

Approximating the Hamming Distance is an interesting question. In a rather negative result Pang and Gamal showed that the communication complexity of this problem is $\Omega(n)$, even in a probabilistic case [8]. Therefore, it is theoretically impossible to obtain a two sided ϵ-bound, *i.e.*,it is impossible to achieve an estimation $\hat{h}$ (the estimated hamming distance) of h (the original distance) so that $(1-\epsilon)h \leq \hat{h} \leq (1+\epsilon)h$ holds for all $0 \leq \epsilon \leq 1$ with arbitrary high probability. However, the results in [3] how a one sided bound can be tightly achieved; in particular these results describe an approximation scheme that underestimates the Hamming Distance between two strings only with a negligible probability.

Such results are quite theoretical in nature. And, as with the case of most randomized systems, a naive implementation is often fraught with an unacceptable abundance of negative outcomes. Such systems need extensive experimentation and parameter tuning before real-life deployment. In a very recent work, Paul *et al.* showed how the approximation technique for Hamming Distance can be applied to Remote Authentication Systems [9]. They borrow the theoretical notion of approximating Hamming Distance computation from [3] and describe a structured, systematic experimental methodology on how to deploy the theory in practice, in addition to proving other theoretical bounds. It turns out that Hamming Distance computing fits in the core of a diverse range of pattern-matching algorithms used in many biometric devices, such as finger print identifiers or iris-scan recognizers and so on. Their results, although not extremely accurate and should be taken more as ball-park figures, show that such randomized Hamming Distance computation can yield a power savings of up to 80% in a Remote Authentication system, while degrading the security guarantee for the authentication only negligibly.

The moral of the whole story that bred itself in our discussion of Communication Complexity Theory, is that there have already been quite a few encouraging results that can be directly borrowed while developing several components of computer networks of the future, and that to the strongest belief of these authors, there are many more problems that deserve attention from this new theoretical angle offering practical gains.

3 PCP Preliminaries

In this section we keep our focus on remote authentication. However, we want to suggest a new solution framework, *viz.*, the Probabilistic Checking of Proofs (PCP), for this domain. We first show that the problem of remote authentication can be reformulated as a two party Prover-Verifier game. In this framework we can immediately apply the notion of PCP and the associated techniques to illustrate how to cut down the number of transmitted bits. As a final assimilation we describe our protocol. We start with defining the infrastructure, viz., the Prover-Verifier game.

3.1 Prover-Verifier Game and PCP

Any computational decision problem essentially means deciding a question like $\mathbf{x} \in \mathbf{L}?$, where x is a string and L is a language, both over the same alphabet [2]. A machine essentially decides this set inclusion problem. The hardness of the problem depends on the lower bound on the time the machine would take in deciding the question. Our two party setting consists of the following : (i) The Prover (**P**r), who has an unlimited computational power, and (ii) The Verifier (**V**f) who has polynomial amount of computational resources. The game is the following: **P**r is trying to prove to **V**f that some string x belongs to some language L. **P**r can produce evidence in the form of bit strings and **V**f will use his limited computational power (in terms of space and time) and verify the question if $\mathbf{x} \in \mathbf{L}?$.

Definition 1 *NP is the class of languages, such that $\forall L \in NP$, and for a string $x \in L$, **P**r can always present a string y of length $\mathcal{O}(poly(n))$ (n is the length of x), such that taking x and y as inputs, **V**f can verify the claim (viz., $x \in L$) in $\mathcal{O}(poly(n)$ time. y is called the certificate for x.*

The above definition is exactly equivalent to the standard definition that **N**P is the set of all languages that can be decided by a Non Deterministic Turing Machines in polynomial time.

PCP has a randomized setting. This is almost identical to the Prover-Verifier game that we described before. However, in addition to the certificate string y (and the original input string x), **V**f generates a random bit string r of polynomial length. **V**f is going to take three strings as input, viz., r, y, x but while deciding if $\mathbf{x} \in \mathbf{L}$, it is not going to use all the bits of y but only a selective few (depending on r) and still decides with *high probability* if $\mathbf{x} \in \mathbf{L}$.

[2] An alphabet is a set of symbols

Definition 2 ***P****CP*$(r(n), q(n))$ *denotes the set of languages for which the random string* r *has length* $r(n)$ *and* ***V****f can look only* $q(n)$ *number of bits from the certificate string* y *and still be able to decide with high probability if* $x \in L$.

Having set up the framework, we now introduce the celebrated PCP-theorem [1] that we are going to use to design our authentication protocol.

Theorem 1

$$\boldsymbol{N}P = \boldsymbol{P}CP(\mathcal{O}(\log n), \mathcal{O}(1))$$

In other words, the verifier ***V****f uses only* $\mathcal{O}(\log n)$ *random bits (*r*) and samples only* ***c****onstant number of bits from the certificate string provided by* ***P****r , to resolve if* $x \in L$.

Why PCP ?

PCP theorem stated above is remarkably strong. Once the prover prepares a proof (or certificate) with appropriate encoding, the verifier needs only to randomly look into a few (constant number) of its bits and will be able to decide with a very high probability if $x \in L$. We can use this trick for remote authentication. It is intuitively clear that we are trying to design a Verifier for our remote authentication system that will need to sample only a few bits (constant number - to be exact) instead of the entire authentication key which may be quite large. To be able to exploit this principle, we need to reformulate our authentication mechanism as a **P**CP instance, which we do in the next section. There are couple of advantages in using PCP-based protocols for remote authentication, over the techniques suggested in [9]. First, the estimated transmission requirement is $\mathcal{O}(1)$ (constant) in the case of PCP, as opposed to $\mathcal{O}(\log n)$ in [9]. And second, a PCP-based protocol is quite generic in nature, while the other is limited to the approximation of Hamming Distance.

We feel it may be helpful for the reader's understanding to include an intuitive discussion on how **P**CP works. Consider the verifier's task in the normal case. The prover provides a certificate y, and the verifier computes the verification function $v(y)$; if $v(y) = 1$, the verifier accepts that $x \in L$ else it rejects. Now, it turns out that the prover can encode the certificate y in a special way. Error Correction Codes come in pretty handy in such encoding. Typically, an error correcting code maps smaller dimensional strings (messages) to strings of larger dimensions (codewords) ensuring that the codewords are sufficiently far from each other. In the case of PCP, we resort to a special kind of error correction code that also takes into account the verification function $v(y)$. Which works in the following way. Consider two strings α and β, so that $v(\alpha) = 1$ and $v(\beta) = 0$. In other words, α is a proper certificate while β is not. Now, we can design a special kind of code that maps α and β far apart from each other. Suppose the encoding function is E and $E(\alpha) = \alpha'$ and $E(\beta) = \beta'$. Now, that α' and β' are sufficiently different from each other, it becomes possible to distinguish between them by sampling much smaller number of bits (than their original lengths). Although the exact construction of such a code is quite complicated and beyond the scope of this paper, in a nutshell, this captures the intuition behind the **P**CP framework. The prover uses the encoding E to encode its certificate. The verifier is able to distinguish an honest certificate (encoded) from a bad one by sampling only a constant number of bits.

4 Remote Authentication as a Prover Verifier game

In this section we cast the problem of remote authentication as a two party game. In particular we consider the scenario of a user remotely authenticating himself to a server, using a large token such as a biometric sample. The user provides a user id, say U and a token b. The server can access a database which has a template b_U known to be the identifier for U. Once presented with b, the server computes the distance $d(b, b_U)$ between the two samples, and if $d(b, b_U)$ is small enough, it approves U. Notice that $d()$ is specific to the nature of the pattern used.

Now, we can think of the aforementioned scenario as a language recognition problem as follows :

Let L be the set of all possible tokens that can be generated by U. If the token is a fingerprint, then L consists of all possible fingerprints that U may generate. Typically, the fingerprints taken from a person are not exactly identical to each other; they vary slightly depending on the inherent sensor inaccuracy, exact condition of the thumb and so on. When U requests the server for authentication, the question the server really asks is - if $\mathbf{b_U} \in \mathbf{L}$?. The answer is obviously affirmative if the user is really U. However, if the person is an imposter and not U then he would potentially generate another language $L_v \neq L$ so that $b_U \notin L_v$. To prove the identity, the user gives one biometric sample of his own, which plays his certificate to the server. If the user is really U, then he can give a sample b such that $d(b, b_U)$ is quite small. However for some other person $V \neq U$, it won't be possible to give such a biometric sample. To summarize, the process of remote authentication can be thought of as a two party game, where the server plays the role of **V**f , the user plays the role of **P**r , the template token b_U in the server database is the input string (called x in Definition 1). The user (or the prover **P**r) provides the certificate string b (denoted by y in Definition 1).

Now we can present the above scenario in a **P**CP setting. The question asked is - if $b_U \in L$? The work required to resolve the question is computing the distance

$d(b, b_U)$. This computation is typically a finger print recognition or a scanned retinal image matching or something similar depending upon the kind of pattern being used. There are polynomial time algorithms available for these pattern matching problems. So the decision question is in **NP** . Hence, We can now appeal to Theorem 1 which asserts the existence of a randomized algorithm A such that given a random string r of logarithmic length, **V**f (the server) can run A on r, b_U and only $\mathcal{O}(1)$ number of bits of b and decide with very high probability if $b_U \in L$, i.e., approve (or disapprove) the user at the remote end. This is the focal point of this paper. Once **P**r and **V**f agrees on the random string r, **P**r needs to present to **V**f only a very small number of bytes from its authentication token. And by the strength of Theorem 1, **V**f will still be able to decide whether **P**r is an authorized person or not. Normally the biometric sample will be quite large in size, usually a few hundreds of kilobytes. Instead, in our setting only a very small fraction of that needs to be communicated over the network. The string r is also logarithmically small compared to b resulting in a very small overhead of communication. In fact, if **P**r and **V**f share the same pseudo-random-number generator (PRNG), then even that communication becomes unnecessary. This leads to the final protocol which we summarize in the following section.

5 The Remote Authentication protocol

All the previous discussion culminates in constructing our protocol. We stick to the same notation used so far to denote the parties and input strings of our problem.

1. **P**r sends his user-id U to **V**f .
2. **V**f generates a small number of random bits (logarithmic in the size of the expected biometric sample) and sends this string (r) to **P**r . This step is not required if the two parties share the same PRNG.
3. **P**r computes from r and the encoded version of b, a small sample b_c (of $\mathcal{O}(1)$ length) and sends this to **V**f .
4. **V**f runs a computation with r, b_u and b_c as inputs and decides the authorization. This becomes possible by the result of **P**CP theorem.

The above protocol has the following characteristics.

- The communication overhead associated with a remote authentication is reduced by a significant amount.
- Since an additional level of encryption of information takes place in order to transmit the data, this will offer at least one extra level of protection on top of the normal one offered by a public key infrastructure.
- The whole process of computing the small subset of bits to be communicated is usually done through techniques of Error Correcting Codes which heavily uses polynomials and an associated mathematical structure called Galois field. Although Galois field operations are somewhat expensive, they are doable in reasonable time; in fact such operations are used for error correction in network transmission, or digital storage systems quite heavily.

6 Discussion: Directions and Challenges

It should be clear by now that PCP technology offers a neat and attractive design principle for Remote Authentication. In fact, the same principle has been proposed in the recent past as an alternative for quick verification of remotely executable codes [2]. The main research challenge however is to effectively design the proofs and the verification procedure that can be used in real time. Though it is gratifying to note that the proof of PCP-theorem is quite constructive [1], we are still at a distance from using the technique directly for a remote-authentication or a mobile-code verification. The reason is the following. The entire proof of PCP-theorem is based on a well known problem, *viz.*, 3-CNF-satisfiability [3]. 3-CNF being an NP-complete problem, ensures that for any other problem in NP, an equivalent algorithm exists. In our case the verification algorithm typically does some pattern recognition in polynomial time. And all we know for sure by the strength of **P**CP theorem is that there exists another equivalent algorithm that performs the same task, but looks only into a small fraction of the pattern. However, no one to our knowledge has ever constructed such an algorithm. There are a few possible directions that can be taken to engineer the solution.

- One straightforward way would be to reduce the problem of pattern recognition into 3CNF and then apply the algorithm known for 3CNF on the new reduced form of the problem. This is doable because anything in NP is reducible to 3CNF.
- Modifying a standard algorithm such as fingerprint recognition) into one that our protocol can use. This is based on the same intuition on which the very proof of PCP-theorem is designed. The idea is the following: Take a proof and blow up it up in size using a suitable error correction code such as multivariate Reed-Muller code (which is used in the PCP-theorem). Clearly, any arbitrary code will not work; the structure of the code has to be intrinsically related to the exact verification function that the verifier would use in a normal setting. This is an open problem for the theoreticians.

[3] Conjunctive Normal Form

- A third design approach is presented in [2]. In this work, the authors argue, and reasonably so, that it is more convenient to view a verification algorithm in the RAM model (random access machine) than in the Turning Machine model. In a RAM model, a program becomes a sequence of assembly level instructions, such as LOAD, STORE, ADD, MOV and so on. At this point, the authors show how to convert a set of RAM instructions into a 3-CNF expression. The rest of the routine falls in place following the constructive proof of the **P**CP -theorem [1].

7 Conclusions

We have focused on the problem of trade-off between computation and communication. The benefit for trading computation for communication is two-fold, savings in bandwidth, and savings in power. For small and mobile devices operating on wireless, especially in the long range (such as cell-phones), these two resources are very critical. We glanced at the merging trends of the technology and the applications and identified a number of areas where the processor cycles can be traded for network bandwidth. These areas are - file transfer, shared document edits, remote authentication, and verification of remotely available codes. We hope to make a case for incorporating the insights collected in this paper from various fields of complexity theory, into the design of the network protocols for the future. In the penumbra of this over-arching goal, we focus in particular into an interesting possibility that a protocol for remote authentication can take advantage of. This is the framework of Probabilistic Checking of Proofs. We show how a remote-authentication can be viewed as a PCP instance through an intuitive yet somewhat non-obvious formulation, which is small yet rather a focused technical contribution of this paper. Finally, we present the algorithmic challenges that come on the way. It is not clear which of the paths prescribed in section 6 would be the one offering least resistance. One of these paths mandates designing new algorithms, which is a theoretical task, yet, if solved, seems to be the neatest of all. The other two are quite messy in terms of the coding-complexity but seem to present less of theoretical barrier otherwise.

References

[1] S. Arora and S. Safra. Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998.

[2] T. Batu, R. Rubinfeld, and P. White. Runtime verification of remotely executed code using probabilistically checkable proof systems.

[3] G. Cormode, M. Paterson, S. C. Sahinalp, and U. Vishkin. Communication complexity of document exchange. In *Symposium on Discrete Algorithms*, pages 197–206, 2000.

[4] K. friedel and M. Sudan. Some improvements to total degree tests. In *Proc. of the Third Israil Symposium on Theory and Computing Systems*, 1995.

[5] E. Kushilevitz and N. Nisan. *Communication Complexity*. cambridge University Press, 1997.

[6] C. Lund, L. Fortnow, H. J. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. In *IEEE Symposium on Foundations of Computer Science*, pages 2–10, 1990.

[7] A. Orlitsky. Interactive communication: Balanced distributions, correlated files, and average-case complexity. In *IEEE Symposium on Foundations of Computer Science*, pages 228–238, 1991.

[8] K. Pang and A. El-Gamal. Communication complexity of computing the hamming distance. *SIAM Journal on Computing*, 15(4):932–947, 1986.

[9] A. Paul and U. Ramachandran. Computation-communication trade-off for power and bandwidth savings in remote authentication over wireless networks. *Georgia Instute of Technology Technical Report*, GIT-CERCS-04-01, January 2004.

[10] G. J. Pottie and W. J. Kaiser. Embedding the internet: wireless integrated network sensors. *Communications of the ACM*, 43(5):51–58, May 2000.

[11] R. Rubinfeld and M. Sudan. Robust characterizations of polynomials with applications to program testing. In *TR RC-19156, IBM Research Division, T. J. Watson Research Center, Yorktown Heights, NY 10598*, September 1993.

[12] R. Rubinfeld and M. Sudan. Testing polynomial functions efficiently and over rational domains. In *Proc. of the Third Symposium on Discrete Algorithms*, 1994.