Optimal Perfectly Secure Message Transmission

K. Srinathan†, Arvind Narayanan, and C. Pandu Rangan

Department of Computer Science and Engineering, Indian Institute of Technology, Madras, Chennai-600036, India.

email: ksrinath@cs.iitm.ernet.in, arvindn@meenakshi.cs.iitm.ernet.in, rangan@iitm.ernet.in

Abstract. In the *perfectly secure message transmission* (PSMT) problem, two synchronized non-faulty players (or processors), the *Sender* **S** and the *Receiver* **R** are connected by n wires (each of which facilitates 2-way communication); **S** has an ℓ-bit message that he wishes to send to **R**; after exchanging messages in phases[1] **R** should *correctly* obtain **S**'s message, while an *adversary* listening on and *actively* controlling any set of t (or less) wires should have *no information* about **S**'s message.
We measure the quality of a protocol for securely transmitting an ℓ-bit message using the following parameters: the number of wires n, the number of phases r and the total number of bits transmitted b. The optima for n and r are respectively $2t+1$ and 2. We prove that any 2-phase reliable message transmission protocol, and hence any secure protocol, over n wires out of which at most t are faulty is required to transmit at least $b = \left(\frac{n\ell}{n-2t}\right)$ bits. While no known protocol is simultaneously optimal in both communication and phase complexity, we present one such optimum protocol for the case $n = 2t+1$ when the size of message is large enough, viz., $\ell = \Omega(t \log t)$ bits; that is, our optimal protocol has $n = 2t+1$, $r = 2$ and $b = O(n\ell)$ bits. Note that privacy is *for free*, if the message is large enough.
We also demonstrate how randomness can effectively improve the phase complexity. Specifically, while the (worst-case) lower bound on r is 2, we design an efficient optimally tolerant protocol for PSMT that terminates in a *single* phase with arbitrarily high probability.
Finally, we consider the case when the adversary is *mobile*, that is, he could corrupt a different set of t wires in different phases. Again, the optima for n and r are respectively $2t+1$ and 2; However we show that $b \geq \left(\frac{n\ell}{n-2t}\right)$ bits irrespective of r. We present the first protocol that is (asymptotically) optimum in b for $n = 2t+1$. Our protocol has a phase complexity of $O(t)$.

1 Introduction

Consider a synchronous network $\mathcal{N}(\mathcal{P}, \mathcal{E})$ represented by an undirected graph where $\mathcal{P} = \{P_1, P_2, \ldots, P_N\} \cup \{\mathbf{S}, \mathbf{R}\}$ denotes the set of players (nodes) in

† Financial support from Infosys Technologies Limited, India, is acknowledged.

[1] A phase is a send from **S** to **R** or from **R** to **S** or both simultaneously.

the network that are connected by 2-way communication links as defined by $\mathcal{E} \subset \mathcal{P} \times \mathcal{P}$. The players **S** and **R** do not trust the network connecting them. Nevertheless, the sender **S** wishes to *securely* send a message to the receiver **R** through the network. Security here means that **R** should receive exactly what **S** sent to him while other players should have no information about it, even if up to t of the players (excluding **S** and **R**) collude and behave maliciously. This problem, known as *perfectly secure message transmission* (PSMT), was first proposed and solved by Dolev *et al.*[3]. In essence, it is proved in [3] that PSMT from **S** to **R** across the network $\mathcal{N}$, tolerating a static[2] adversary that corrupts up to t players (nodes), is possible if and only if $\mathcal{N}$ is at least $(2t+1)$-(**S**,**R**)-connected.[3] We use the approach of [3] and abstract away the network entirely and concentrate on solving the PSMT problem for a single pair of synchronized processors, the *Sender* **S** and the *Receiver* **R**, connected by some number n of wires denoted by $w_1, w_2, \dots w_n$. We may think of these wires as a collection of vertex-disjoint paths between **S** and **R** in the underlying network.[4]

The PSMT problem is important in its own right as well as a very useful primitive in various secure distributed protocols. Note that if **S** and **R** are connected directly via a private and authenticated link (like what is assumed in generic secure multiparty protocols [15, 6, 1, 12]), secure communication is trivially guaranteed. However, in reality, it is not economical to directly connect *every* two players in the network. Therefore, such a complete network can only be (virtually) realized by simulating the missing links using SMT protocols as primitives.

In this paper, we shall use the simple and standard model of a synchronous network wherein any communication protocol evolves as a series of *phases*, during which the players (**S** or **R**) send messages, receive them and perform (polynomial time) local computations according to the protocol.

There are three basic aspects contributing to the quality of an algorithm for PSMT: the maximum tolerable number t of faulty wires, the number of phases r, and the total number of bits sent b. The optima for the above quality parameters are as follows: $n = 2t+1$ [3], $r = 2$ [13]. The lower bound for b is proved in this work to be $\left(\frac{n\ell}{n-2t}\right)$ bits when $r = 2$ for any $n \geq 2t+1$.

In the last few years, there have been some attempts toward improving the quality of protocols. All protocols proposed so far, securely communicate an element of a finite field $\mathbb{F}$; extending this to securely communicate ℓ field elements would result in a proportional increase of communication complexity. Dolev *et al.* [3] proposed three protocols: the first one with $n = 2t+1$, $r = t+1$, $b = O(t^3\ell)$ field elements, the second one with $n = 2t+1$, $r = 3$, $b = O(t^5\ell)$ field elements

---

[2] By static adversary, we mean an adversary that decides on the set of players to corrupt before the start of the protocol.

[3] We say that a network $\mathcal{N}$ is κ-(P_i, P_j)-connected if the deletion of no $(\kappa - 1)$ or less nodes from $\mathcal{N}$ disconnects P_i and P_j.

[4] The approach of abstracting the network as a collection of n wires is justified using Menger's theorem [8] which states that a graph is c-(**S**,**R**)-connected if and only if **S** and **R** are connected by at least c vertex-disjoint paths.

and the third one with $n = 2t + 1$, $r = 2$ and b not polynomial in n. This was substantially improved by the protocol of [13], that has $n = 2t + 1$, $r = 2$ and $b = O(t^3\ell)$ field elements. The protocol of [14] has $n = 2t + 1$, $r = \log t$ and $b = O(t^2 \log t\ell)$ field elements.

However, no known protocol is simultaneously optimal in both b and r. In this paper, we present (in Section 4.1) an (asymptotically) optimal protocol to perfectly securely transmit a message consisting of ℓ field elements, viz., our protocol has $n = 2t + 1$, $r = 2$ and $b = O(t\ell)$ field elements, if $\ell = \Omega(t)$. Since we require the field size to be at least n, this means that the message size is $\Omega(t \log t)$ bits.

Unfortunately, due to the stringent requirements of privacy and reliability, (even optimal) PSMT protocols are not always as efficient as we would like them to be in practice. Therefore, one often relaxes either the reliability or the privacy requirement, or both, and tries to achieve *statistical* reliability/privacy.

Thus, we look for protocols in which the probability that **R** will receive the correct message is $1 - \delta$ and the probability that the adversary will learn the message is ϵ for arbitrarily small δ and ϵ. Of course, PSMT is the case $\epsilon = \delta = 0$; broadcast satisfies $\epsilon = 1, \delta = 0$, and so on. In [5], a $(0, \delta)$-protocol with $n = 2t+1$, $r = 3$ and $b = O(t^2)$ field elements was presented to securely communicate one field element. For an extensive discussion of (ϵ, δ)-secure protocols see [5].

In this paper we introduce a new way of relaxing the requirements. We study the *average case* efficiency of SMT protocols, rather than the worst case. We do not require that the worst case complexity be polynomially bounded, or even finite; we feel that nonterminating protocols that nevertheless complete quickly with high probability and have perfect security and reliability are very useful constructions.

In Section 5 we present an optimally fault-tolerant protocol that terminates in a *single* with high probability, and having $b = O(t)$ field elements. We note that the significance of a single phase protocol is more than merely a gain in efficiency (in terms of network latency): **S** and **R** are not required to be on the network at the same time for executing a single phase protocol, and therefore they are applicable in a much bigger set of scenarios than are multi-phase protocols.

Most of the results in the literature model the sender's distrust in the network via a centralized static adversary that can corrupt up to t of the n wires and assume the worst-case that the adversary can completely control the behavior of the corrupted wires [3, 13]. In line with this, we assume up to section 6 that the adversary is static, i.e., he (a) decides on the set of t wires to corrupt before the start of the protocol and (b) a wire once corrupted remains so subsequently.

However, in practice the bound on the number of corrupted wires may depend on the total time of the protocol execution. Thus motivated, in section 6 we model the faults via a *mobile* adversary, in line with [10]. In this model, the adversary can corrupt any set of wires in the lifetime of the protocol but is constrained to corrupt at most t wires in any single phase of the protocol.

We show that our ideas in the case of static adversaries can be extended to withstand mobile adversaries. We prove that the lower bound of $b = (\frac{nl}{n-2t})$

for reliable message transmission holds for mobile adversaries irrespective of the number of rounds. We also give a bit-optimal protocol when $n = 2t + 1$.

2 Preliminaries

Notation. Throughout the paper, we use $\mathbf{M}$ to denote the message that $\mathbf{S}$ wishes to securely communicate to $\mathbf{R}$. The message is assumed to be a sequence of ℓ elements from the finite field $\mathbb{F}$. The only constraint on $\mathbb{F}$ is that its size must be no less than the number of wires n. Since we measure the size of the message in terms of the number of field elements, we must also measure the communication complexity in units of field elements; we follow this convention in the rest of the paper. We assume that there exists a publicly specified one-to-one mapping $\alpha : \{1, 2, \ldots, n\} \to \mathbb{F}$. For convenience, we use α_i to denote $\alpha(i)$.

We say that a wire is *faulty* if it is controlled by the adversary; all other wires are called *honest*. A faulty wire is *corrupted* in a specific phase if the value sent along that wire was changed. When the context makes clear which phase is being referred to, we simply say that a wire is *corrupted*. Observe that a wire may be faulty but not corrupted in a particular phase.

2.1 Efficient single phase reliable communication

To *reliably* communicate a message $\mathbf{m}$, a sequence of k field elements, to $\mathbf{R}$, one simple way is for $\mathbf{S}$ to send $\mathbf{m}$ along each wire – i.e, broadcast. However, when $n > 2t + 1$, where t out of n wires are corrupted, broadcast, requiring $O(nk)$ field elements, is not the most efficient method of (single phase) reliable communication. Instead, it is possible to use an *error-correcting code* to improve the communication complexity of reliable communication to $\frac{nk}{n-2t}$ field elements. A *block* error correcting code encoding a *message* of k field elements to a *codeword* of n symbols is an injective mapping $\mathcal{C} : \mathbb{F}^k \to \mathbb{F}^n, (n > k)$. The encoding function is used in conjunction with a decoding function $\mathcal{D} : \mathbb{F}^n \to \mathbb{F}^k$ with the property that if its input differs from a valid codeword in at most t field elements, then $\mathcal{D}$ outputs the message corresponding to that codeword. We say that the code corrects t errors. Clearly, such a decoding function will always exist if any two valid codewords differ in at least $2t+1$ symbols, that is, the distance of the code $d \geq 2t + 1$.

The efficiency of an error correcting code is subject to the *Singleton bound*:

Lemma 1. *Let C be a block code which reliably transmits k field elements by communicating a total of n field elements and has a* distance *of d. Then $n \geq k + d - 1$.*

We observe that for a t-error correcting code, the distance d (which is the minimum Hamming distance between any two codewords) is at least $2t+1$. Thus we have

Corollary 1. *Let C be a t-error correcting block code as in lemma 1. Then $k \leq n - 2t$.*

We now consider a special class of error correcting codes called Reed-Solomon codes (RS codes).

Definition 1. *Let $\mathbb{F}$ be a finite field and $\alpha_1, \alpha_2, \ldots \alpha_n$ be a collection of distinct elements of $\mathbb{F}$. Given $k \leq n \leq |\mathbb{F}|$, and a block $\mathbf{B} = [m_0\ m_1\ \ldots\ m_{k-1}]$ the encoding function for the Reed-Solomon code $RS(n,k)$ is defined as $[p_{\mathbf{B}}(\alpha_1)\ p_{\mathbf{B}}(\alpha_2)\ \ldots\ p_{\mathbf{B}}(\alpha_n)]$ where $p_{\mathbf{B}}(x)$ is the polynomial $\sum_{i=0}^{k-1} m_i x^i$.*

Theorem 1 ([7]). *The Reed-Solomon code meets the Singleton bound.* □

The following special property of the RS-code will be of use in our subsequent discussion:

Lemma 2. *Let $[p_{\mathbf{B}}(\alpha_1)\ p_{\mathbf{B}}(\alpha_2)\ \ldots\ p_{\mathbf{B}}(\alpha_n)]$ be an $RS(n,k)$-encoding of $\mathbf{B}$. Then for any $n' < n$, any subsequence of $[p_{\mathbf{B}}(\alpha_1)\ p_{\mathbf{B}}(\alpha_2)\ \ldots\ p_{\mathbf{B}}(\alpha_n)]$ of length n' forms a valid $RS(n',k)$-encoding of $\mathbf{B}$.*

Proof: Easy observation. □

Constructing message transmission protocols using error correcting codes is a typical application, for example see [2, 11]. We now describe REL-SEND$(\mathbf{m}, k)$, a protocol for reliable communication obtained by using the corresponding Reed-Solomon code RS(n,k). REL-SEND$(\mathbf{m}, k)$ will be used as a sub-protocol later on.

Protocol REL-SEND: optimal single phase reliable message transmission of $\mathbf{m}$.

- $\mathbf{S}$ breaks up $\mathbf{m}$ into blocks of k field elements.
- For each block $\mathbf{B} = [m_0\ m_1\ \ldots\ m_{k-1}]$:
  - $\mathbf{S}$ computes RS(n,k) to obtain $[p_{\mathbf{B}}(\alpha_1)\ p_{\mathbf{B}}(\alpha_2)\ \ldots\ p_{\mathbf{B}}(\alpha_n)]$.
  - $\mathbf{S}$ sends $p_{\mathbf{B}}(\alpha_i)$ along the wire w_i.
  - $\mathbf{R}$ receives the (possibly corrupted) $p_{\mathbf{B}}(\alpha_i)$'s and applies the decoding algorithm and constructs $\mathbf{B}$.
- $\mathbf{R}$ concatenates the $\mathbf{B}$'s to recover the message $\mathbf{m}$.

We note that the resulting protocol is a single phase protocol. The reverse process is equally valid - given a single phase reliable communication protocol, we can convert it into a block error correcting code. Thus, the maximum attainable efficiency for single phase reliable communication is also subject to the Singleton bound.

Remark: This conversion to an error correcting code is straightforward if the messages sent along each wire in the protocol are of the same length. Suppose, however, that there is exists a protocol Π that does not have this symmetry property and beats the Singleton bound. Then consider the protocol Π' which consists of n sequential executions of protocol Π with the identities or numbers of the wires being "rotated" by a distance of i in the i^{th} execution. Clearly, this protocol achieves the symmetry property by "spreading the load"; further its message expansion factor is equal to that of Π'. It therefore beats the Singleton bound as well, which is a contradiction.

Lemma 3. *Suppose that the receiver* **R** *knows f faults among the n wires, and t' be the number of faulty wires apart from those f. Then REL-SEND*$(\mathbf{m}, k)$ *works if $n - f \geq k + 2t'$.*

Proof: Since **R** knows f faults, he simply ignores those wires; and by lemma 2, this converts the code into an RS code with parameters $n - f$ and k. The result now follows from lemma 1 and theorem 1. □

2.2 Extracting randomness

In several of our protocols we have the following situation: **S** and **R** by some means agree on a sequence of n numbers $\mathbf{x} = [x_1, x_2, \ldots x_n] \in \mathbb{F}^n$ such that

- The adversary knows $n - f$ of the components of $\mathbf{x}$
- The adversary has no information about the other f components of $\mathbf{x}$
- **S** and **R** do not necessarily know which values are known to the adversary.

The goal is for **S** and **R** to agree on a sequence of f numbers $y_1, y_2, \ldots y_f \in \mathbb{F}$ such that the adversary has no information about $y_1, y_2, \ldots y_f$. This is achieved by the following algorithm:

Algorithm EXTRAND$_{n,f}(\mathbf{x})$. Let **V** be a $n \times f$ Vandermonde matrix with members in $\mathbb{F}$. This matrix is published as a part of the protocol specification. **S** and **R** both locally compute the product $[y_1\ y_2\ \ldots\ y_f] = [x_1\ x_2\ \ldots\ x_n]\mathbf{V}$.

Lemma 4. *The adversary has no information about $[y_1\ y_2\ \ldots\ y_f]$ in algorithm EXTRAND.*

Proof. We need to show that there is a bijective mapping between the f tuple of values that are not known to the adversary and the f tuple $y_1, y_2, \ldots y_f$. But this is a direct consequence of the fact that every f-subdeterminant in an $n \times f$ Vandermonde matrix is nonzero.

3 Lower bound on communication complexity

Theorem 2. *Any 2-phase perfectly reliable message transmission (PRMT) of ℓ bits requires communicating $(\frac{nl}{n-2t})$ bits.*

We first observe that a probabilistic polynomial time (PPT) protocol for PRMT with a worst-case communication complexity of b bits exists if and only if there is a deterministic protocol with the *same* communication complexity. Since perfect reliability is required, the algorithm must succeed for every possible choice of coin tosses; in particular, it must succeed when all the random bits of **S** and **R** are zeroes. Thus we convert any PPT protocol into a deterministic protocol by fixing the sequence of coin-tosses to all zeros. Hence, we assume that **S** and **R** are deterministic polynomial time algorithms.

We recall that in a phase, both **S** and **R** may simultaneously send messages to the other player. If this happens we call it a *bidirectional* phase. On the other hand, if only one of the players sends a message, we call it a *unidirectional* phase.

Without loss of generality we assume that in the first phase communication is from **R** to **S** and in the second phase it is from **S** to **R**. (Clearly there is no point in communication from **R** to **S** in the second phase; similarly if **S** sends any messages to **R** in the first phase we can consider these to be part of the second phase as well.) In the rest of the paper we assume that communication in each phase is unidirectional.

We prove the stronger statement that any 2-phase PRMT of ℓ bits requires communicating t least $(\frac{nl}{n-2t})$ bits even against a weaker adversary, namely, one that is passive in the first phase.

Thus, let Π be a two phase protocol in which M_1 is the totality of messages sent by **R** to **S** in phase I and M_2 the totality of messages sent by **S** to **R** in phase 2. The steps of Π are as follows:

1. **R** computes M_1 and sends it to **S**.
2. **S**, using M_1 and the message computes M_2 and sends it to **R**.
3. **R** recovers the message using M_2.

In the above protocol, we see that step 1 is "useless": consider the protocol Π' in which step 1 is replaced with the following step:

S and **R** both locally compute M_1 by simulating **R**'s execution in step 1 of Π. (Since the adversary is passive, M_1 is guaranteed to have been received by **S** in the first phase of Π.)

It is clear that Π' succeeds whenever Π succeeds.

Π' being a single phase protocol, can also be viewed as an error correcting code. That is, the concatenation of the data sent along all the wires forms the codeword. Let $\mathbb{S}_i$ be the set of possible values of the data sent along the wire w_i. Thus, each codeword is of length at least $\sum_{i=1}^{n} \log |\mathbb{S}_i|$, consisting of n elements one from each $\mathbb{S}_i$, $1 \leq i \leq n$. Now, the removal of any $2t$ elements from each of the codewords should result in shortened codewords that are all distinct. For if any two were identical, the original codewords could have differed only among at most $2t$ elements implying that there exist two original codewords c_1 and c_2 and an adversarial strategy such that the receiver's view is the *same* on the receipt of either c_1 or c_2. In more detail, without loss of generality assume that c_1 and c_2 differ only in their last $2t$ elements. That is, $c_1 = \alpha \circ \beta$ and $c_2 = \alpha \circ \gamma$, where $\circ$ denotes concatenation and $|\beta| = |\gamma| = 2t$ elements. Let β_1 denote the first t elements of β, while β_2 be the last t elements. That is, let $\beta = \beta_1 \circ \beta_2$, $|\beta_1| = |\beta_2| = t$elements. Similarly, let $\gamma = \gamma_1 \circ \gamma_2$, $|\gamma_1| = |\gamma_2| = t$. Now, consider the two cases: (a) c_1 is sent and the adversary corrupts it to (by corrupting the last t wires and changing β_2 to γ_2) $\alpha \circ \beta_1 \circ \gamma_2$ and (b) c_2 is sent and the adversary corrupts it to (by corrupting the penultimate set of t wires and changing γ_1 to β_1) $\alpha \circ \beta_1 \circ \gamma_2$. Thus, the receiver cannot distinguish between the receipt of c_1 and c_2, which violates the reliable communication property. Therefore, all shortened codewords are distinct and there are as many shortened codewords as original codewords. But the number of shortened codewords is at most the minimum $\prod_{j=1}^{n-2t} |\mathbb{S}_{i_j}|$ among all $(n-2t)$ sized subsets $(i_1, i_2, \ldots, i_{n-2t})$ of $1, 2, \ldots, n$. Thus we may sort the $|\mathbb{S}_i|$'s in a non-decreasing order and multiply the first $(n-2t)$ values to obtain the number of original codewords denoted by C. Thus, reliable communication of $k = \log C$ bits incurs a communication cost of at least $\sum_{i=1}^{n} \log |\mathbb{S}_i|$ bits. But $\log C = \sum_{j=1}^{n-2t} |\mathbb{S}_{i_j}|$. Thus, in the best case all the domains are of equal size and is thus subject to the Singleton bound. By corollary to lemma 1, reliable communication of $k = n - 2t$ bits incurs a communication cost of n bits. Since Π' communicates an l bit message, it follows that Π' has a communication complexity of $(\frac{nl}{n-2t})$ bits. $\square$

Corollary 2. *Any 2-phase perfectly reliable message transmission (PRMT) of ℓ field elements requires communicating $(\frac{nl}{n-2t})$ field elements.*

The above corollary follows from the fact that a field element can be represented as a string of $\lceil \log |\mathbb{F}| \rceil$ bits.

4 An Optimal Protocol for PSMT

4.1 The PSMT Protocol

In this section, we present our 2-phase protocol for PSMT for any message that is a sequence of ℓ field elements, with $n = 2t + 1$ and $b = O(t\ell)$ field elements, for a sufficiently large l. It turns out that we require $\ell = \Omega(t)$ bits.

Suppose that there exists a protocol that securely transmits a message consisting of $\Omega(t)$ field elements with $n = 2t+1$, $r = 2$ and $b = O(nt)$ field elements. It is evident that for any integer $j \geq 1$, tj field elements can be sent in $b = O(ntj)$ field elements whilst maintaining $n = 2t + 1$ and $r = 2$; this is because we can run the j sub-protocols in parallel. Setting $j = \lceil \frac{\ell}{t} \rceil$, we obtain a protocol that communicates ℓ field elements by sending $O(n\ell)$ field elements. Thus, our goal now reduces to the *design of a protocol that achieves secure transmission of $\Omega(t)$ field elements over a network of $n = 2t + 1$ wires, in two rounds by communicating $b = O(nt)$ field elements.* We now present one such protocol. Specifically, our protocol sends $\lfloor \frac{t}{3} \rfloor$ field elements by sending $O(nt)$ field elements.

In our protocol, the first phase is a send from **R** to **S** and the second phase is from **S** to **R**. We denote the set of n wires by $\mathcal{W} = \{w_1, w_2, \ldots, w_n\}$. We assume that **S** wishes to communicate a block, denoted by $\mathbf{m}$, that consists of $\lfloor \frac{t}{3} \rfloor$ field elements from $\mathbb{F}$.

Phase I (R to S)

The receiver **R** selects at random n polynomials p_i, $1 \leq i \leq n$ over $\mathbb{F}$, each of degree t.
Next, through each wire w_i, **R** sends the following to **S**:

– The polynomial p_i.[5]

[5] We assume that the polynomial is sent by sending a $(t + 1)$-tuple of field elements.

– For each j, $1 \leq j \leq n$, the value of $p_j(\alpha_i)$ (which we denote by r_{ij}), where α_i's are arbitrary distinct publicly specified members of $\mathbb{F}$.

Phase II (S to R)

S begins this phase after receiving what **R** has sent in the first phase. Let **S** receive the polynomial p'_i and the values r'_{ij} along the wire w_i. In this phase, **S** must locate all the corruptions that occurred in the previous phase, communicate the corruptions and send the message securely. A naive and straightforward way of doing this is as follows: communicate the list of $\Theta(n^2)$ contradictions (we say that wire i contradicts wire j if $r'_{ij} \neq p'_j(\alpha_i)$) among the wires (in the worst case). However, there are two problem with this approach: (a) the method requires communicating $\Theta(n^3)$ field elements, and (b) such an approach necessitates more than two phases.

We solve the former problem by using a two step technique to communicate the list of contradictions. In the first step we broadcast a selected set of contradictions; this will enable the players to find sufficiently many faults to facilitate sending the remaining contradictions in $O(n^2)$ field elements using the REL-SEND protocol in the second step. The second problem is solved using some new techniques described in the sequel.

S's computation.

– **S** initializes his *fault-list*, denoted by L_{fault}, to $\emptyset$.
– **S** constructs a directed graph $G = (\mathcal{W}, A)$ where arc $(w_i, w_j) \in A$ if $r'_{ij} \neq p'_j(\alpha_i)$.
– Let $H = (\mathcal{W}, E)$ be the undirected graph based on G; that is, $(w_i, w_j) \in E$ if $(w_i, w_j) \in A$ or $(w_j, w_i) \in A$.
– For each i, $1 \leq i \leq n$, such that the degree of node w_i in the graph H constructed above is greater than t (i.e., $degree(w_i) \geq t + 1$), **S** adds w_i to L_{fault}.
– Let $H' = (\mathcal{W}', E')$ be the induced subgraph of H on the vertex set $\mathcal{W}' = (\mathcal{W} \setminus L_{fault})$.
– Next, **S** finds a maximum matching[6] $M \subseteq E'$ of the graph H'; this can be done efficiently using the algorithms of [4, 9].
– For each arc (w_i, w_j) in G that does not belong to M, **S** associates the four-tuple $\{\alpha_i, \alpha_j, r'_{ij}, p'_j(\alpha_i)\}$. Let $\{a_1, a_2, \ldots, a_N\}$ be the arcs in G that are not in M. Replacing each arc with its associated 4-tuple, **S** gets a set of $4N$ field elements, $X = \{X_1, X_2, \ldots, X_{4N}\}$.
– Let $u = \left\lfloor \frac{t}{3} \right\rfloor$. Next, **S** creates the $2t + u$-degree message-carrying polynomial $s(x) = \sum_{i=0}^{2t+u} k_i x^i$ as follows: let $\mathbf{m} = [m_0 m_1 \ldots m_{u-1}]$.

$$\text{Assign } k_i = \begin{cases} m_i & \text{if } 0 \leq i < u. \\ 0 & \text{if } w_{i-u+1} \in L_{fault}. \\ p'_{i-u+1}(0) & \text{otherwise.} \end{cases}$$

[6] A subset M of the edges of H, is called a *matching* in H if no two of the edges in M are adjacent. A matching M is called *maximum* if H has no matching M' with a greater number of edges than M has.

– **S** initializes the set Y as follows:

$$Y = \{s(\alpha_1), s(\alpha_2), \ldots, s(\alpha_{t+1})\}$$

– Finally, the sender **S** selects at random n polynomials q_i, $1 \leq i \leq n$ over $\mathbb{F}$, each of degree t, such that the values $q_i(0)$ lie on a polynomial of degree $\lfloor \frac{4t}{3} \rfloor$.
– **S** computes $\mathbf{y} = [y_0\ y_1\ \ldots\ y_{u-1}] = \text{EXTRAND}_{n,u}([q_1(0)\ q_2(0)\ \ldots\ q_n(0)])$.
– For each j, $1 \leq j \leq n$, let v_{ij} denote the value of $q_j(\alpha_i)$. With each of the $N+|M|$ arcs in the graph G, **S** associates the four-tuple $\{\alpha_i, \alpha_j, v_{ij}$ and $q_j(\alpha_i)\}$. He initiates the set Z in similar lines as X to contain $4(N + |M|)$ field elements.

S's communication.

S sends the following to **R** through all the wires:

1. The blinded message $\mathbf{m} \oplus \mathbf{y}$.
2. The set L_{fault}.
3. For each edge $(w_i, w_j) \in M$, the following four field elements: $\{\alpha_i, \alpha_j, r'_{ij}$ and $p'_j(\alpha_i)\}$.

Along each wire w_i, **S** sends the following as specified by the REL-SEND$(\cdot, \cdot)$ Algorithm:

1. REL-SEND$(X, |M| + |L_{fault}| + 1)$.
2. REL-SEND$(Y, |M| + |L_{fault}| + 1)$.
3. REL-SEND$(Z, |M| + |L_{fault}| + 1)$.

Again, through each wire w_i, **S** sends the following to **R**:

– The polynomial q_i.
– For each j, $1 \leq j \leq n$, the value of $v_{ij} = q_j(\alpha_i)$.

Message recovery by R.

R receives what **S** sent in the second phase and locally deciphers the message **m** as follows:

1. **R** reliably receives L_{fault} and knows that the wires in this set are faulty. He initializes $L^{\mathbf{R}}_{fault} = L_{fault}$.
2. For each arc $(w_i, w_j) \in M$, **R** reliably receives $\{\alpha_i, \alpha_j, r'_{ij}$ and $p'_j(\alpha_i)\}$. He locally verifies: $r'_{ij} \stackrel{?}{=} r_{ij}$ and $p'_j(\alpha_i) \stackrel{?}{=} p_j(\alpha_i)$. If the former check fails (that is the values are unequal), then **R** adds w_i to $L^{\mathbf{R}}_{fault}$. If the latter check fails, then **R** adds w_j to $L^{\mathbf{R}}_{fault}$ (note that both w_i and w_j may be identified as faulty; in any case, at least one of them is guaranteed to be found faulty). Thus, at least $|M|$ new faults are caught in this step.

3. From Lemma 5, it is clear that $\mathbf{R}$ receives the set X reliably. Again $\mathbf{R}$ locally verifies for each arc's (say (w_i, w_j)) 4-tuple: $r'_{ij} \stackrel{?}{=} r_{ij}$ and $p'_j(\alpha_i) \stackrel{?}{=} p_j(\alpha_i)$. If the former check fails (that is the values are unequal), then $\mathbf{R}$ adds w_i to $L^{\mathbf{R}}_{fault}$. If the latter check fails, then $\mathbf{R}$ adds w_j to $L^{\mathbf{R}}_{fault}$. At the end of this step, *all* the faults that occurred during transmission in **Phase I** are guaranteed to have been identified (see Lemma 6).
4. We know from Lemma 5 that the $\mathbf{R}$ receives the set Y correctly. If the number of faults (which are not in L_{fault}) that occurred in **Phase I** was $\leq \frac{2t}{3}$, then in the polynomial $s(x)$, $\mathbf{R}$ has $\leq t$ unknowns and $t+1$ equations (which are bound to be consistent whatever the number of unknowns may be, since all faults have been eliminated). Thus, in this case, $\mathbf{R}$ obtains the message $\mathbf{m}$.
5. Similarly, from Lemma 5, we know that $\mathbf{R}$ receives the set Z correctly. If the number of faults (which are not in L_{fault}) that occurred in **Phase I** was $> \frac{2t}{3}$, then from Lemma 9, we know that $\mathbf{R}$ can obtain the message using the polynomials $q(\cdot)$ and Z. Thus, in this case too, $\mathbf{R}$ obtains the message $\mathbf{m}$.

Lemma 5. $\mathbf{R}$ *is guaranteed to receive the sets X, Y and Z correctly.*

Proof: From lemma 3, the REL-SEND$(\cdot, k)$ protocol succeeds provided that $n - f \geq k + 2(t-f)$; here, $k = |M| + |L_{faults}| + 1$ and $n = 2t+1$. Therefore, REL-SEND succeeds if $(2t+1) - f - (|M| + |L_{faults}| + 1) \geq 2t - 2f$, or if, $f \geq |M| + |L_{faults}|$. Since, $\mathbf{R}$ is guaranteed to have identified at least $|M| + |L_{faults}|$ faulty wires at this stage, the lemma follows. □

Lemma 6. *If the set X was received correctly, then* $\mathbf{R}$ *can find all the corruptions that occurred during* **Phase I**.

Proof: Suppose wire w_i was corrupted in **Phase I**, i.e, $p'_i \neq p_i$. Then the two polynomials can intersect in at most t points. Since there are $t+1$ honest wires, there is guaranteed to be at least one honest wire which contradicts w_i. Since the correct values corresponding to every contradiction have been received by $\mathbf{R}$, $\mathbf{R}$ can find all the corruptions. □

Lemma 7. *If the number of corruptions that occurred in* **Phase I** *was* $\leq \frac{2t}{3}$, $\mathbf{R}$ *obtains the polynomial $s(x)$ correctly.*

Proof: To find the message $\mathbf{R}$ must find the polynomial $s(x)$. To find $s(x)$, $\mathbf{R}$ must find k_i for $0 \leq i \leq 2t+u$. Of these $\mathbf{R}$ does not yet know k_i for $0 \leq i < u$ and does not know $\leq \frac{2t}{3}$ of the $p'_i(0)$, a total of at most $\lfloor \frac{t}{3} \rfloor + 1 + \frac{2t}{3} = t+1$ field elements. But the set Y gives $\mathbf{R}$ $t+1$ values of $s(x)$, which yield $t+1$ linear equations on the coefficients, and using these values $\mathbf{R}$ can determine all k_i. □

Lemma 8. *For some i, if the wire w_i was corrupted in* **Phase I** *then* $\mathbf{R}$ *can correct any corruption of the corresponding q_i in* **Phase II**.

Proof: Let the in-degree of w_i in G be d. Then there must have been at least $t-d+1$ corruptions in **Phase I** (since the number of honest wires is $t+1$). Thus in the second phase, there are at most $2t+1-(t+1-d)=t+d$ legitimate wires. The maximum number of faults that **R** needs to correct in this phase is $t-(t+1-d)=d-1$. We verify that these parameters satisfy the constraint in lemma 3, and therefore **R** will be able to correct all corruptions of q_i. □

Lemma 9. *If the set Z was received correctly and if the number of faults that occurred in* **Phase I** *was $> \frac{2t}{3}$, then* **R** *can obtain the message* **m**.

Proof: By lemma 8, **R** can correct all except a maximum of $\frac{t}{3}$ of the q_i's. The degree of the polynomial that they lie on is $\lfloor \frac{4t}{3} \rfloor$; since these parameters satisfy the constraints of lemma 3, it follows from the correctness of the REL-SEND protocol that **R** can obtain all the q_i's and hence **m**. □

Theorem 3. *The protocol presented in Section 4.1 achieves perfect reliability.*

Proof: Perfect reliability is a consequence of lemmas 7 and 9. □

Lemma 10. *For every honest wire w_i, the adversary has no information about $p_i(0)$ and $q_i(0)$.*

Proof: Obvious. p_i is a random polynomial of degree t but the adversary has seen only t points on it. The same argument holds for q_i as well. □

Theorem 4. *The protocol presented in Section 4.1 achieves perfect security.*

Proof (sketch): First we prove that the adversary has no information about the coefficients $k_0, k_1, \ldots k_{u-1}$ of the polynomial $s(x)$. There are at least $t+1$ values of $p_i(0)$ which are not known to the adversary. The adversary obtains $t+1$ linear equations on the coefficients k_j by knowing the values of $s(\alpha_1), s(\alpha_2), \ldots s(\alpha_{t+1})$ which are sent reliably by **S**. Thus the adversary has $t+1$ linear equations on $u+t+1$ unknowns, which implies that he has no information about any u-tuple of them.

Next we observe that among the $\lfloor \frac{4t}{3} \rfloor$ values of $q_i(0)$, the adversary knows at most t, and hence by the security of the EXTRAND algorithm (lemma 4) it follows that the adversary gets no information about **m**. □

4.2 Performance

Theorem 5. *Given an undirected graph $H=(V,E)$, with a maximum degree of Δ, and a maximum matching M, the number of edges $|E|$ is less than or equal to $(2|M|^2+|M|\Delta)$.*

Proof: We first fix a representation of the maximum matching M as a set of *ordered* pairs of vertices as described below.

We say that a vertex i belongs to vertex-set of the matching, denoted by $Vertex(M)$, if there exists another vertex j such that the edge $(i,j) \in M$. A vertex $i \in Vertex(M)$ is called the *match-vertex* if the degree of i in the subgraph H'_i induced by H over the vertices $i \cup (V \setminus Vertex(M))$ is ≤ 1.

Given a maximum matching M, a match-vertex i may have at most one incident edge $e_i = (i, \cdot)$ in H'_i. We call the edge e_i as a *match-edge* (corresponding to the match-vertex i). We now define X to be the set of all match-edges (corresponding to each of the match-vertices in M).

Claim. Every edge $(i, j) \in M$ has at least one match-vertex.

Proof: On the contrary, if neither i nor j was a match-vertex, then, both i and j are adjacent to at least two vertices in $(V \setminus Vertex(M))$. Let i be adjacent to vertices u and v in $(V \setminus Vertex(M))$ and let j be adjacent to a vertex w $(\neq u)$ in $(V \setminus Vertex(M))$. Now, removing the edge (i, j) from M and adding the edges (u, i) and (j, w) to M gives rise to a new matching in H of size $|M| + 1$ which contradicts the maximality of the matching M. Hence the claim holds.

Hereafter, we represent every edge in $M \cup X$ as (i, j) if and only if i is a match-vertex; in case of both i and j being match-vertices, i is the one with the lower number of corresponding match-edges (ties broken by random choice). We fix one such representation of the edges in $M \cup X$. To avoid confusion between the unordered pair (i, j) and the ordered pair used in the representation of an edge in $M \cup X$ hereafter, we denote the *ordered* pair as $\langle i, j \rangle$. A vertex i belonging to $Vertex(M)$ is called a *left-vertex* if, in the representation of M that we fixed earlier, there exists a vertex j such that $\langle i, j \rangle \in M$. We call all the non-left-vertices belonging to M as *right-vertices*.

Note that the number of left-vertices is equal to the number of right-vertices is equal to $|M|$. Also note that by definition, every left-vertex is a match-vertex.

Now, it is easy to place an upper bound on $|E|$ as follows: the maximum number of edges among the $2|M|$ vertices in $Vertex(M)$ is $|M|(2|M|-1)$. Again, the maximum number of edges from the left-vertices to $(V \setminus Vertex(M))$ is $|M|$, since each left vertex is a match-vertex (having at most one edge to $(V \setminus Vertex(M)))$ and there are $|M|$ left vertices. Furthermore, each right vertex can have at most $\Delta - 1$ edges to $(V \setminus Vertex(M))$ (by the definition of Δ). Thus, $|E| \leq (|M|(2|M| - 1) + |M| + |M|(\Delta - 1)) \leq (2|M|^2 + |M|\Delta)$. □

Theorem 6. *The PSMT protocol presented in Section 4.1 communicates $O(t^2)$ field elements in order to securely transmit $\left\lfloor \frac{t}{3} \right\rfloor$ field elements.*

Proof: We have already proved that the protocol securely transmits $\left\lfloor \frac{t}{3} \right\rfloor$ field elements. From the description of the protocol, it is easy to verify that all steps except possibly the invocations of REL-SEND$(X, .)$ and REL-SEND$(Z, .)$ in Phase II have a communication cost of $O(t^2)$ field elements. Since the maximum degree of a node in H' is at most $t+1$ and $|M|$ is also at most $t+1$, from theorem 5 it follows that $|X|$ (and hence also $|Z|$) is $O((|M| + |L_{fault}|)t)$. Since the efficiency of REL-SEND$(X, |M| + |L_{fault}| + 1)$ is $O(\frac{|X|t}{|M|+|L_{fault}|})$, the theorem follows. □

The main result now follows from the discussion at the beginning of Section 4.1:

Corollary 3. *There exists a 2-round PSMT protocol that securely communicates a message consisting ℓ field elements and has a communication complexity of $O(n\ell)$ field elements when $n = 2t + 1$, if $\ell = \Omega(t)$.*

5 A Las Vegas Single Phase Protocol

In this section we present an optimally tolerant ($n = 2t + 1$) PSMT protocol which terminates in a single phase with (arbitrarily) high probability. We represent the block of field elements $\mathbf{m}$ that $\mathbf{S}$ wishes to send to $\mathbf{R}$ as $\mathbf{m} = [m_0\ m_1\ \dots\ m_t]$.

The Single Phase Protocol

1. $\mathbf{S}$ selects at random n polynomials $p_i, 1 \leq i \leq n$ over $\mathbb{F}$, each of degree t.
2. For each wire w_i
   - $\mathbf{S}$ sends the polynomial p_i through w_i
   - For each j, $\mathbf{S}$ randomly selects one of the t points of intersection of p_i and p_j (denote the selected point by r_{ij}). $\mathbf{S}$ ensures that $r_{ij} \neq r_{ji}$. $\mathbf{S}$ sends r_{ij} through w_i.
3. $\mathbf{S}$ computes $\mathbf{y} = [y_0\ y_1\ \dots\ y_t] = \text{EXTRAND}_{n,t+1}$ $([p_1(0)\ p_2(0)\ \dots\ p_n(0)])$ and broadcasts $\mathbf{m} \oplus \mathbf{y}$.
4. Let p'_i and r'_{ij} be the values received by $\mathbf{R}$. We say that wire i *contradicts* wire j if p'_i and p'_j do not intersect at r_{ij}.
5. $\mathbf{R}$ checks if there is a wire contradicted by at least $t+1$ wires. All such wires are removed.
6. If there is at least one contradiction among the remaining wires, $\mathbf{R}$ broadcasts "FAILURE"; $\mathbf{S}$ and $\mathbf{R}$ now execute the PSMT protocol of section 4.1.
7. If there is no contradiction, $\mathbf{R}$ corrects the polynomials $p_i(.)$ of each corrupted wire w_i (i.e, he "corrects" those wires) using the values of r_{ij} received along the uncorrupted wires. $\mathbf{R}$ now knows all the polynomials p_i.
8. $\mathbf{R}$ computes $\mathbf{y} = [y_0\ y_1\ \dots\ y_t] = \text{EXTRAND}_{n,t+1}([p_1(0)\ p_2(0)\ \dots\ p_n(0)])$ and recovers $\mathbf{m} = (\mathbf{m} \oplus \mathbf{y}) \oplus \mathbf{y}$.

Let ϵ be a bound on the probability that the protocol does not terminate in a single phase. We require that the size of the field $\mathbb{F}$ be $\Omega(\frac{Q(n)}{\epsilon})$, for some polynomial $Q(n)$, but this is of course acceptable since the complexity of the protocol increases logarithmically with field size. We now discuss the correctness of the protocol.

Lemma 11. $\mathbf{R}$ *will never output an incorrect value.*

Proof. Since any corruption involves changing the polynomial corresponding to that wire, it is clear that no corrupted wire can escape contradiction by at least one other wire. If p and p' agree on $t+1$ points (corresponding to the $t+1$ honest wires) then p and p' must be equal. Therefore, at the start of step 7, all the wires which were used in calculation of the output could not have corrupted their values. This guarantees that $\mathbf{R}$'s output in step 8 is correct. □

Lemma 12. *The protocol terminates in a single phase with high probability.*

Proof. Since no uncorrupted wire changes the value sent on the wire, it follows that no honest wire can contradict another honest wire. Thus, if wire i contradicts wire j, then either wire i or wire j is faulty. From this it is easy to see that an honest wire can be contradicted by at most t other wires, and therefore any wire that is contradicted $t+1$ or more wires has to be faulty. Hence **R** can be sure that all the wires removed by him are indeed faulty.

We need to show that if a wire is corrupted, then it will be contradicted by all the honest players with high probability. Let π_{ij} be the probability that the corrupted wire j will not be contradicted by i. This means that the adversary can ensure that $p_j(r_{ij}) = p'_j(r_{ij})$ with a probability of π_{ij}. Since there are only t points at which these two polynomials intersect, this allows the adversary to guess the value of r_{ij} with a probability of at least $\frac{\pi_{ij}}{t}$. But since r_{ij} was selected uniformly in $\mathbb{F}$, the probability of guessing it is at most $\frac{1}{|\mathbb{F}|}$. Therefore we have $\pi_{ij} \leq \frac{t}{|\mathbb{F}|}$ for each i, j. Thus the total probability that the adversary can find i, j such that corrupted wire j will not be contradicted by i is at most $\sum_{i,j} \pi_{ij} \leq \frac{n^2 t}{|\mathbb{F}|}$.

Since $\mathbb{F}$ is chosen such that $|\mathbb{F}| \geq \frac{Q(n)}{\epsilon}$, it follows that the protocol terminates in a single phase with probability $\geq 1 - \epsilon$ if we set $Q(n) = n^3$. □

Lemma 13. *The adversary gains no information about the message.*

Proof: We observe that the adversary has no information about $p_i(0)$ for each honest wire w_i. This is because p_i is a random polynomial of degree t and the adversary has seen only t points on it (one corresponding to each faulty wire.) The proof now follows from lemma 4. □

6 Mobile Adversaries

6.1 Lower Bound on Bit Complexity

The lower bound on the bit complexity of perfectly reliable message transmission proved in section 3 holds for mobile adversaries with no restriction on the number of phases. We give below a brief sketch of the proof.

Since the adversary can corrupt a different set of wires in each phase, the protocol cannot adapt as it finds corrupted wires; thus it can be considered to be memoryless. Therefore the total number of bits transmitted reliably is no more than the sum of the number of bits transmitted reliably in each phase; we have already shown in section 3 that the Singleton bound implies a lower bound of $\frac{nl}{n-2t}$ for a single phase; therefore this bound holds for multiple phase protocols as well.

6.2 An Optimal Protocol

The protocol with optimal fault tolerance and optimal communication complexity is presented below. Let **m** be a block of $t+1$ field elements that **S** wishes to communicate securely to **R**.

The Optimal Protocol

1. **S** selects at random n polynomials p_i, $1 \leq i \leq n$ over $\mathbb{F}$, each of degree t and sends through each wire w_i, the polynomial p_i and the set $\{p_j(\alpha_i)\}_{1 \leq j \leq n}$.
2. After the completion of Phase I, **R** has computed the directed graph G of contradictions among the wires. The rest of the protocol involves finding all the wires that were corrupted in phase I using the graph G as follows:
   While there exists a wire w_j in G that contradicts another wire w_i
   - **R** broadcasts i, j, $p'_i(\alpha_j)$ and r'_{ij} to **S**.
   - **S** uses this information to determine which of w_i and w_j is faulty and broadcasts the identity of the faulty wire to **R**.
   - **R** removes the faulty wire from the protocol, i.e, sets G to the projection of G onto the remaining wires.
3. **R** now knows all the wires which were corrupted in phase I. Thus he can now use the r_{ij} values from the uncorrupted wires to find the correct values of the corrupted wires.
4. **S** and **R** both compute $\mathbf{y} = [y_0\ y_1\ \ldots\ y_t] = \text{EXTRAND}_{n,t+1}([p_1(0)\ p_2(0)\ \ldots\ p_n(0)])$.
5. **S** broadcasts $\mathbf{m} \oplus \mathbf{y}$ to **R**. **R** recovers $\mathbf{m} = (\mathbf{m} \oplus \mathbf{y}) \oplus \mathbf{y}$.

Theorem 7. *The adversary gains no information about the message.*

Proof: First we note that at the end of the first phase, the adversary has no information about $p_i(0)$ for each honest wire w_i. This is because p_i is a random polynomial of degree t and the adversary has seen only t points on it (one corresponding to each faulty wire.) Furthermore, the adversary gains no new information in step 2. This can be seen as follows: each phase in step 2 involves broadcast of the 4-tuple $(i, j, p_i(\alpha_j), r'_{ij})$. Since either wire w_i or wire w_j is faulty, this information is already known to the adversary. The other information that is broadcasted is which of wire w_i and wire w_j is faulty, which is also known to the adversary. The theorem follows. $\square$

6.3 Complexity

The first phase of the protocol involves communication of $O(n^2)$ field elements. In step 2, each phase of communication results in the elimination of one wire. Therefore the number of phases in this step is $O(n)$. Since each phase involves the broadcast of $O(n)$ field elements, step 2 has a communication complexity of $O(n^2)$ field elements. The final phase involves broadcast a string of length $O(t)$ field elements. Therefore the entire protocol has communication complexity $O(n^2)$ field elements. Thus the protocol for a message **M** consisting of an arbitrary number ℓ of field elements obtained by executing this protocol in parallel $\lceil \frac{\ell}{t+1} \rceil$ times has a communication complexity of $O(\frac{n^2 \ell}{t})$ field elements, which is optimal when $n = 2t + 1$.

7 Conclusion

In this paper we have contributed significantly to the progress of the state of the art in the problem of Perfectly Secure Message Transmission. The protocol of section 4.1 constitutes a major improvement over existing protocols tolerating static adversaries; in fact we have achieved the optimal communication complexity when $n = 2t + 1$ and $r = 2$. The protocol can be extended to achieve the optimal communication complexity when $n > 2t + 1$ as well, though we have not presented it here. It would be interesting to see if the lower bound we have proved in section 3 holds when $r > 2$ as well (for PSMT); we conjecture that it does.

Perhaps our most interesting result is the average case single phase PSMT protocol. It is in fact surprising that such a protocol even exists; in addition our protocol is also very efficient in terms of communication complexity.

References

1. M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *20th ACM STOC*, pages 1–10, 1988.
2. Y. Desmedt and Y. Wang. Perfectly secure message transmission revisited. In *EUROCRYPT '02*, volume 2332 of *LNCS*, pages 502–517. Springer-Verlag, 2002.
3. D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *JACM*, 40(1):17–47, January 1993.
4. J. Edmonds. Paths, trees and flowers. *Canadian Jl. of Math.*, 17:449–467, 1965.
5. M. Franklin and R.N. Wright. Secure communication in minimal connectivity models. *Journal of Cryptology*, 13(1):9–30, 2000.
6. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *19th ACM STOC*, pages 218–229, 1987.
7. F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error Correcting Codes*. North Holland Publishing Company, 1978.
8. K. Menger. Zur allgemeinen kurventheorie. *Fundamenta Mathematicae*, 10:96–115, 1927.
9. S. Micali and V. Vazirani. An $O(\sqrt{|V|}\ |E|)$ algorithm for maximum matching in general graphs. In *21st IEEE FOCS*, pages 17–27, 1980.
10. R. Ostrovsky and M. Yung. How to withstand mobile virus attacks. In *10th ACM PODC*, pages 51–61, 1991.
11. M.O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *JACM*, 36:335–348, 1989.
12. T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *21st ACM STOC*, pages 73–85, May 1989.
13. H. Sayeed and H. Abu-Amara. Efficient perfectly secure message transmission in synchronous networks. *Information and Computation*, 126(1):53–61, 1996.
14. K. Srinathan, V. Vinod, and C. Pandu Rangan. Brief announcement: Efficient perfectly secure communication over synchronous networks. In *22nd ACM PODC*, page 252, 2003.
15. A. C. Yao. Protocols for secure computations. In *23rd IEEE FOCS*, pages 160–164, 1982.