A Low Communication Competitive Interactive Proof System for Promised Quadratic Residuosity*

Toshiya Itoh[1] Masafumi Hoshi[1] Shigeo Tsujii[2]

[1] Department of Information Processing,
Interdisciplinary Graduate School of Science and Engineering,
Tokyo Institute of Technology,
4259 Nagatsuta, Midori-ku, Yokohama 227, Japan.
[2] Department of Electrical and Electronic Engineering,
Disciplinary Graduate School of Science and Engineering,
Tokyo Institute of Technology,
2-12-1 O-okayama, Meguro-ku, Tokyo 152, Japan.

Abstract. A notion of "competitive" interactive proof systems is defined by Bellare and Goldwasser as a natural extension of a problem whether computing a witness w of $x \in L$ is harder than deciding $x \in L$ for a language $L \in \mathcal{NP}$. It is widely believed that quadratic residuosity (QR) does not have a competitive interactive proof system. Bellare and Goldwasser however introduced a notion of "representative" of Z_N^* and showed that there exists a competitive interactive proof system for promised QR, i.e., the moduli N is guaranteed to be the product of $k = O(\log\log|N|)$ distinct odd primes. In this paper, we consider how to reduce the communication complexity of a competitive interactive proof system for promised QR and how to relax the constraint on k from $O(\log\log|N|)$ to $O(\log|N|)$. To do this, we introduce a notion of "dominant" of Z_N^* and show that promised QR with the constraint that $k = O(\log|N|)$ has a competitive interactive proof system with considerably low communication complexity.

1 Introduction

1.1 Background and Motivation

Is proving membership harder than deciding membership? This is one of the most basic questions in theoretical computer science. It has been known that if a language L is $\mathcal{NP}$-complete then computing a witness w for $x \in L$ is polynomially equivalent to deciding $x \in L$. How about the languages that are not known to be $\mathcal{NP}$-complete? In general, it has been widely believed that this is not the case. Recently, Bellare and Goldwasser [2], [3] showed that there exists a language $L \in \mathcal{NP} - \mathcal{P}$ for which computing a witness w for $x \in L$ is exactly harder than deciding $x \in L$ if the class of deterministic double exponential

* Sponsored by Okawa Institute of Information and Telecommunication grant 93-01.

D.R. Stinson (Ed.): Advances in Cryptology - CRYPTO '93, LNCS 773, pp. 61-72, 1994.

time is not equal to the class of nondeterministic double exponential time. The language $L \in \mathcal{NP} - \mathcal{P}$ found by Bellare and Goldwasser [2], [3] satisfies the uniformly log-sparse property and thus it is somewhat unnatural. On the other hand, there exist several natural languages $L \in \mathcal{NP}$ for which computing a witness w for $x \in L$ may be harder than deciding $x \in L$, e.g., quadratic residuosity (QR), quadratic nonresiduosity (QNR), etc.

What will happen when interactions and randomization are allowed in the proving process of membership? This way of the proving process of membership is formulated by Goldwasser, Micali, and Rackoff [7] (resp. independently by Babai and Moran [4]) as interactive proof systems (resp. Arthur-Merlin games). Informally, a language L has an interactive proof system $\langle P, V \rangle$ if for the honest prover P and for any $x \in L$, the honest verifier V accepts $x \in L$ with probability at least 2/3 and for any all powerful (dishonest) prover P^* and for any $x \notin L$, the honest verifier V accepts $x \notin L$ with probability at most 1/3. Bellare and Goldwasser [2], [3] extended the problem whether computing a witness w for $x \in L$ is harder than deciding $x \in L$ for a language $L \in \mathcal{NP}$ to the case of interactive proof systems and formulated the problem to be "competitive" interactive proof systems. Informally, an interactive proof system $\langle P, V \rangle$ for a language L is *competitive* if for the (probabilistic polynomial time bounded) honest prover P with an access to the oracle L and for any $x \in L$, the honest verifier V accepts $x \in L$ with probability at least 2/3 and for any all powerful (dishonest) prover P^* and for any $x \notin L$, the honest verifier V accepts $x \notin L$ with probability at most 1/3. It should be noted that in interactive proof systems $\langle P, V \rangle$, the honest prover P is allowed to be a computationally unbounded Turing machine, while in competitive interactive proof systems $\langle P, V \rangle$, the honest prover P must be a probabilistic polynomial time bounded oracle Turing machine with an access to the underlying language as an oracle.

Then is proving membership still harder than deciding membership in competitive interactive proof systems? In some cases, the interactions and the randomization alleviate the proving task, but in another cases, they may not. To see this more precisely, let us first consider the language QNR. It has not been known that computing a witness w for $x \in$ QNR is polynomially equivalent to deciding $x \in$ QNR. Indeed it is believed that computing a witness w for $x \in$ QNR may be harder than deciding $x \in$ QNR. Goldwasser, Micali, and Rackoff [7] however showed that QNR has a competitive interactive proof system and this implies that the (honest) prover P suffices to have the computational ability of deciding $x \in$ QNR in order to prove membership of $x \in$ QNR in an interactive and a randomized manner. Next let us consider the language QR. It is also believed that computing a witness w for $x \in$ QR may be harder than deciding $x \in$ QR. Contrary to QNR, in all known interactive proof systems $\langle P, V \rangle$ for QR (see, e.g., [10], [6]), the (honest) prover P requires to have at least the computational ability of computing square roots modulo a composite number N (equivalently the computational ability of factoring a composite number N).

Bellare and Goldwasser [2], [3] observed that the interactions and the randomization do not necessarily alleviate the proving task and showed that there

exists a language $L \in \mathcal{NP} - \mathcal{BPP}$ that does not have a competitive interactive proof system if the class of nondeterministic double exponential time is not included in the class of bounded probabilistic double exponential time. (Independently, Beigel and Feigenbaum [1] showed for a different purpose that there exists an *incoherent* language $L \in \mathcal{NP}$ if the class of nondeterministic triple exponential time is not included in the class of bounded probabilistic triple exponential time.) Again, the language $L \in \mathcal{NP} - \mathcal{BPP}$ shown by Bellare and Goldwasser [2], [3] satisfies the uniformly log-sparse property and thus it is also somewhat unnatural. This result only guarantees the existence of a language $L \in \mathcal{NP} - \mathcal{BPP}$ that does not have a competitive interactive proof system under the complexity assumption but does not necessarily imply that QR never has a competitive interactive proof system. Then is it possible to construct a competitive interactive proof system for QR like in the case for QNR?

This has not been solved yet but is believed that this is not the case. To affirmatively solve this open problem, Bellare and Goldwasser [3] investigated QR in a *promised* form. Intuitively, a promise problem (see, e.g., [5], [9], etc) is specified by a pair of disjoint sets A and B and for $x \in A \cup B$ we have to decide whether $x \in A$ or $x \in B$. It should be noted that the promise problem is different from the language membership problem, because the former imposes restrictions on inputs but the latter does not. In this setting, Bellare and Goldwasser [3] introduced a notion of *representative* of Z_N^* and showed that there exists a competitive interactive proof system for promised QR, i.e., the moduli N is guaranteed to be the product of $k = O(\log\log|N|)$ distinct odd primes. Informally, a vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_{2^k-1})$ over Z_N^* is said to be *representative* of Z_N^* if each y_i $(1 \le i \le 2^k - 1)$ belongs to a distinct residue class except for quadratic residues modulo N. The basic idea behind the result above is to use the fact that there exist $2^k = O(\log|N|)$ distinct residues classes under a relation appropriately defined on Z_N^* and to reduce a quadratic residuosity test to a collection of quadratic nonresiduosity tests. Then the protocol following this idea requires about 2^{2k} quadratic nonresiduosity tests and thus the communications complexity of the resulting protocol is comparatively large — in the protocol, the prover P sends to the verifier V about $2^k(|N| + 2^k)$ bits and the verifier V sends to the prover P about $2^{2k}|N|$ bits.

1.2 Results

In this paper, we consider how to reduce the communication complexity of a competitive interactive proof system for promised QR and how to relax the constraint on k from $O(\log\log|N|)$ to $O(\log|N|)$. For this purpose, we first introduce a notion of *dominant* of Z_N^*, which plays a role very similar to a basis in a linear space over $GF(2)$. Informally, a vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$ over Z_N^* is said to be *dominant* of Z_N^* if for any vector $e = (e_1, e_2, \ldots, e_k) \in \{0,1\}^k$ such that $e \neq 0$, $z \equiv y_1^{e_1} y_2^{e_2} \cdots y_k^{e_k} \pmod N$ is not a square modulo N. Then we investigate several properties of a dominant vector of Z_N^* and show that promised QR, with the constraint that $k = O(\log|N|)$ has a competitive interactive proof system in which the prover P sends to the verifier V about $k|N|$ bits and the

verifier V sends to the prover P about $4|N|$ bits. The basic idea behind the result here is to use the fact that if the moduli N is guaranteed to be the product of $k = O(\log |N|)$ distinct odd primes then there exist sufficiently many (samplable) vectors $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$ over Z_N^* to uniquely specify 2^k residue classes under a relation appropriately defined on Z_N^*. The idea here is inspired by the one due to Bellare and Goldwasser [3] but its use enables us to avoid 2^{2k} invocations of quadratic nonresiduosity tests. Thus the resulting protocol based on this idea considerably reduces the communication complexity.

2 Preliminaries

In this section, we present definitions and notation necessary in the sequel.

Let $\langle P, V\rangle$ be an interactive protocol. Informally, an interactive protocol $\langle P, V\rangle$ is said to be an interactive proof system for a language L if for the honest prover P and for any $x \in L$, the honest verifier V accepts $x \in L$ with probability at least $2/3$ and for any all powerful dishonest prover P^* and for any $x \notin L$, the honest verifier V accepts $x \notin L$ with probability at most $1/3$. For further details on this, see, e.g., [7], [8], etc.

Definition 1 [2, 3]. An interactive proof system $\langle P, V\rangle$ for a language L is said to be competitive if

- **Completeness:** For any $x \in L$, $\text{Prob}\{\langle P^L, V\rangle \text{ accepts } x\} \geq 2/3$, where the prover P is a probabilistic polynomial time oracle Turing machine;
- **Soundness:** For any $x \notin L$ and any all powerful dishonest prover P^*, $\text{Prob}\{\langle P^*, V\rangle \text{ accepts } x\} \leq 1/3$,

where the probabilities are taken over all possible coin tosses of P and V.

It is already known that there exist competitive interactive proof systems for quadratic nonresiduosity [7], for graph nonisomorphism [8], and for graph isomorphism [8], [10], however, quadratic residuosity is believed not to have a competitive interactive proof system.

Let $\langle A, B\rangle$ be a pair of disjoint sets. Intuitively, the problem $\langle A, B\rangle$ is said to be *promised* if the inputs are guaranteed to be in $A \cup B$. Associated to the promise problem $\langle A, B\rangle$, we define a promise oracle that returns correct answers only when the queries are in $A \cup B$.

Definition 2 [3]. A promise problem is a pair of disjoint sets $\langle A, B\rangle$. A promise oracle for $\langle A, B\rangle$ is an oracle that given $q \in A \cup B$, returns 1 if $q \in A$ and returns 0 if $q \in B$.

Informally, a promise problem $\langle A, B\rangle$ has a a competitive interactive proof system $\langle P, V\rangle$ if for the (probabilistic polynomial time bounded) honest prover P with an access to the promise oracle for $\langle A, B\rangle$ and for any $x \in A$, the honest verifier V accepts $x \in A$ with probability at least $2/3$ and for any all powerful dishonest prover P^* and for any $x \in B$, the honest verifier V accepts $x \in B$ with probability at most $1/3$.

Definition 3 [3]. A promise problem (A, B) is said to have a competitive interactive proof system if

- **Completeness:** For any $x \in A$ and any promise oracle O for (A, B), Prob$\{(P^O, V)$ accepts $x\} \geq 2/3$, where P is a probabilistic polynomial time oracle Turing machine;
- **Soundness:** For any $x \in B$ and any all powerful dishonest prover P^*, Prob$\{(P^*, V)$ accepts $x\} \leq 1/3$,

where the probabilities are taken over all possible coin tosses of P and V.

A language quadratic residuosity (QR) is defined to be QR $= \{\langle x, N\rangle \mid x \in Z_N^*$ is a square modulo $N\}$ and a language quadratic nonresiduosity QNR is defined to be QNR $= \{\langle x, N\rangle \mid x \in Z_N^*$ is not a square modulo $N\}$. The problem that we are interested in is when the moduli N is guaranteed to be the product of $k \geq 1$ distinct odd primes. In the following, we define the problem "promised QR" that will be investigated in this paper.

Definition 4 [3]. A promised QR is a pair of disjoint sets $(\mathrm{QR}_k, \mathrm{QNR}_k)$, where $\mathrm{QR}_k = \{\langle x, N\rangle \in \mathrm{QR} \mid N$ is the product of k distinct odd primes$\}$, $\mathrm{QNR}_k = \{\langle x, N\rangle \in \mathrm{QNR} \mid N$ is the product of k distinct odd primes$\}$, and $k \geq 1$.

3 Known Results

We overview the result by Bellare and Goldwasser [3], i.e., if $k = O(\log\log|N|)$, then the promised QR $(\mathrm{QR}_k, \mathrm{QNR}_k)$ has a competitive interactive proof system.

Lemma 5 [3]. *If* $k = O(\log\log|N|)$, *then* promised QR $(\mathrm{QR}_k, \mathrm{QNR}_k)$ *has a competitive interactive proof system.*

Here we overview the protocol given by Bellare and Goldwasser [3]. In the competitive interactive proof system for promised QR [3], Protocol QNR is used as a subprotocol.

Protocol QNR: A "Competitive" IP for QNR

common input: $\langle x, N\rangle$ and 1^s, where s is the security parameter.

V1: V chooses $c_i \in_R \{0, 1\}$, $r_i \in_R Z_N^*$ and computes $z_i \equiv x^{c_i} r_i^2 \pmod N$ $(1 \leq i \leq s)$.

$V \rightarrow P$: $\langle z_1, z_2, \ldots, z_s\rangle$.

P1: For each i $(1 \leq i \leq s)$, if $z_i \in Z_N^*$ is a square modulo N, then P sets $d_i = 0$; otherwise P sets $d_i = 1$.

$P \rightarrow V$: $\langle d_1, d_2, \ldots, d_s\rangle$.

V2: V accepts iff $c_i = d_i$ for each i $(1 \leq i \leq s)$.

It is easy to see that the protocol above is a competitive interactive proof system for quadratic nonresiduosity (QNR).

To show the protocol by Bellare and Goldwasser [3], we present a notion of "representative vector" of Z_N^* and several technical lemmas on its properties.

Definition 6 [3]. Let N be the product of k distinct odd primes. A vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_{2^k-1})$ over Z_N^* is said to be representative of Z_N^* if (1) for each i $(1 \leq i \leq 2^k - 1)$, $y_i \in Z_N^*$ is not a square modulo N and (2) for each i, j $(1 \leq i < j \leq 2^k - 1)$, $z_{ij} \equiv y_i y_j \pmod{N}$ is not a square modulo N.

The following is the key proposition on the reduction of a quadratic residuosity test to a collection of quadratic nonresiduosity tests.

Proposition 7 [3]. *Let N be the product of k distinct odd primes and let $\boldsymbol{y} = (y_1, y_2, \ldots, y_{2^k-1})$ be representative of Z_N^*. Then $\langle x, N\rangle \in \mathrm{QR}_k$ iff $w_i \equiv x y_i \pmod{N}$ is not a square modulo N for each i $(1 \leq i \leq 2^k - 1)$.*

Bellare and Goldwasser [3] showed an efficient way to find a representative vector $\boldsymbol{y}$ of Z_N^*, i.e., if $k = O(\log\log|N|)$, then there exists a probabilistic polynomial time oracle Turing machine with an access to the promise oracle for $(\mathrm{QR}_k, \mathrm{QNR}_k)$ that samples with probability at least 3/4 a representative vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_{2^k-1})$ of Z_N^*.

Proposition 8 [3]. *If $k = O(\log\log|N|)$, then exists a probabilistic polynomial time oracle Turing machine R with an access to the promise oracle for $(\mathrm{QR}_k, \mathrm{QNR}_k)$ that on input $\langle x, N\rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$ outputs either a representative vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_{2^k-1})$ of Z_N^* with probability at least 3/4 or $\perp$ with probability at most 1/4.*

The basic idea behind the result by Bellare and Goldwasser [3] is as follows: (1) The prover P generates a representative vector $\boldsymbol{y}$ of Z_N^* (see Proposition 8); (2) The prover P shows to the verifier V that the vector $\boldsymbol{y}$ is really representative of Z_N^* (see Definition 6) by the interactive proof system for QNR [7]; and (3) The prover P shows to the verifier V that $\langle x, N\rangle \in \mathrm{QR}_k$ (see Proposition 7) by the interactive proof system for QNR [7].

The following is the competitive interactive proof system for promised QR [3] under the constraint that $k = O(\log\log|N|)$.

Protocol PQR-1: A Competitive IP for Promised QR

common input: $\langle x, N\rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$, where $k = O(\log\log|N|)$.

P1: P runs the machine R to sample a vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_{2^k-1})$ as a candidate of representative of Z_N^*

$P \rightarrow V$: $\boldsymbol{y} = (y_1, y_2, \ldots, y_{2^k-1})$.

V1: If V receives $\perp$ from P, then V halts and rejects $\langle x, N\rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$; otherwise V continues.

$P \leftrightarrow V$: P shows to V by Protocol QNR with $s = 2$ that y_i is not a square modulo N for each i $(1 \leq i \leq 2^k - 1)$.

V2: If V does not accept $\langle y_i, N\rangle$ for some i $(1 \leq i \leq 2^k - 1)$, then V halts and rejects $\langle x, N\rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$; otherwise V continues.

$P \leftrightarrow V$: P shows to V by Protocol QNR with $s = 2$ that $z_{ij} \equiv y_i y_j \pmod{N}$ is not a square modulo N for each i, j $(1 \leq i < j \leq 2^k - 1)$.

V3: If V does not accept $\langle z_{ij}, N\rangle$ for some i, j $(1 \le i < j \le 2^k - 1)$, then V halts and rejects $\langle x, N\rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$; otherwise V continues.

$P \leftrightarrow V$: P shows to V by Protocol QNR with $s = 2$ that $w_i \equiv xy_i \pmod N$ is not a square modulo N for each i $(1 \le i \le 2^k - 1)$.

V4: If V does not accept $\langle w_i, N\rangle$ for some i $(1 \le i \le 2^k - 1)$, then V halts and rejects $\langle x, N\rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$; otherwise V halts and accepts $\langle x, N\rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$.

The correctness of Protocol PQR-1 follows from that of Protocol QNR [3].

4 Main Results

In this section, we show that if $k = O(\log |N|)$, then promised QR $(\mathrm{QR}_k, \mathrm{QNR}_k)$ has a competitive interactive proof system with much lower communication complexity than the one by Bellare and Goldwasser [3].

4.1 Technical Lemmas

Let $M \ge 2$ be an odd integer. For any $x \in Z_M^*$, let $Q_M(x) = 0$ if $x \in Z_M^*$ is a square modulo M and let $Q_M(x) = 1$ if $x \in Z_M^*$ is not a square modulo M. Let $N = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, where $p_1, p_2, \ldots p_k$ are distinct odd primes and $\alpha_i \ge 1$ for each i $(1 \le i \le k)$. For any $x, y \in Z_N^*$, define a binary relation $\simeq$ on Z_N^* to be $x \simeq y$ iff $Q_{p_i}(x) = Q_{p_i}(y)$ for each i $(1 \le i \le k)$.

It is easy to see that the relation $\simeq$ on Z_N^* is an equivalence relation on Z_N^*. The equivalence class $R_N(x)$ of $x \in Z_N^*$ under the relation $\simeq$ on Z_N^*, i.e., $R_N(x) = \{y \in Z_N^* \mid x \simeq y\}$, is called to be a residue class of $x \in Z_N^*$.

Definition 9. Let $N = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$ be the product of k distinct odd primes. Then for $x \in Z_N^*$, a vector $c_x = (c_1^x, c_2^x, \ldots, c_k^x) \in \{0,1\}^k$ is said to be associated with $x \in Z_N^*$ if $c_i^x = Q_{p_i}(x)$ for each i $(1 \le i \le k)$.

The following lemmas show basic properties of a vector $c_x \in \{0,1\}^k$ associated with $x \in Z_N^*$.

Lemma 10. *Let N be the product of k distinct odd primes. For any $x, y \in Z_N^*$, let $z \equiv xy \pmod N$ and let $c_x, c_y, c_z \in \{0,1\}^k$ be vectors associated with $x, y, z \in Z_N^*$, respectively. Then $c_z \equiv c_x + c_y \pmod 2$.*

Lemma 11. *Let N be the product of k distinct odd primes and let $c_x \in \{0,1\}^k$ be a vector associated with $x \in Z_N^*$. For any integer $e \ge 0$, let $c_y \in \{0,1\}^k$ be a vector associated with $y \equiv x^e \pmod N$. Then $c_y \equiv e c_x \pmod 2$.*

The following notion of "dominant" is one of the most important ones in our main result here. It plays a role similar to a basis in a linear space over $GF(2)$.

Definition 12. Let N be the product of k distinct odd primes. A vector $y = (y_1, y_2, \ldots, y_k)$ is said to be dominant of Z_N^* if vectors $c_{y_1}, c_{y_2}, \ldots, c_{y_k} \in \{0,1\}^k$, each of which is associated with $y_i \in Z_N^*$ $(1 \le i \le k)$, are linearly independent over $GF(2)$.

Hereafter, we use d_i for a dominant vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$ of Z_N^* to denote a vector associated with $y_i \in Z_N^*$ instead of c_{y_i} $(1 \leq i \leq k)$.

Let $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$ be a vector over Z_N^* and let $e = (e_1, e_2, \ldots, e_k)$ be a vector over $GF(2)$. For simplicity, here we use $\boldsymbol{y} \uparrow e$ to denote $\boldsymbol{y} \uparrow e \equiv y_1^{e_1} y_2^{e_2} \cdots y_k^{e_k} \pmod N$. In the following lemma, we show that if $k = O(\log |N|)$, then a dominant vector $\boldsymbol{y}$ of Z_N^* can be efficiently sampled by a probabilistic polynomial time oracle Turing machine with an access to the promise oracle for promised QR $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$.

Lemma 13. *If $k = O(\log |N|)$, then there exists a probabilistic polynomial time oracle Turing machine D with an access to the promise oracle for $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$ that on input $\langle x, N \rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$ outputs either a dominant vector $\boldsymbol{y}$ of Z_N^* with probability at least $3/4$ or $\perp$ with probability at most $1/4$.*

Proof. Let $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$ be a vector over Z_N^* and let $c_i \in \{0,1\}^k$ be a vector associated with $y_i \in Z_N^*$ for each i $(1 \leq i \leq k)$. The probability P_{ind} that the vectors $c_1, c_2, \ldots, c_k$ are linearly independent over $GF(2)$ is bounded by

$$P_{ind} = \frac{1}{\|Z_N^*\|^k} \prod_{j=0}^{k-1} \left(\|Z_N^*\| - 2^j \cdot \frac{\|Z_N^*\|}{2^k} \right) = \prod_{i=1}^{k} (1 - 2^{-i})$$

$$\geq \prod_{i=1}^{\infty} (1 - 2^{-i}) = 1 + \sum_{i=1}^{\infty} \frac{(-1)^i}{(2-1)(2^2-1)\cdots(2^i-1)} > \frac{2}{7},$$

where $\|A\|$ denotes the cardinality of a (finite) set A. Then the machine D randomly chooses m vectors $\boldsymbol{y}_j = (y_{1j}, y_{2j}, \ldots, y_{kj})$ over Z_N^* $(1 \leq j \leq m)$. For each $\boldsymbol{y}_j$ $(1 \leq j \leq m)$, the machine D computes $q_j^\ell \equiv \boldsymbol{y}_j \uparrow \mathrm{bin}(\ell) \pmod N$ for each ℓ $(1 \leq \ell \leq 2^k - 1)$, and queries q_j^ℓ to the promise oracle for $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$ to get the answer $a_j^\ell \in \{0,1\}$, where $\mathrm{bin}(\ell)$ is the binary representation of an integer ℓ $(1 \leq \ell \leq 2^k - 1)$. If there exists an index j $(1 \leq j \leq 2^k - 1)$ such that $a_j^\ell = 0$ for each ℓ $(1 \leq \ell \leq 2^k - 1)$, then the machine D outputs $\boldsymbol{y} = \boldsymbol{y}_j$ as a dominant vector of Z_N^*; otherwise the machine D outputs $\perp$.

The vector $\boldsymbol{y}$ sampled by the machine D is always dominant of Z_N^*. We show this by contradiction. We assume that the vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$ sampled by D is not dominant of Z_N^*. Then for a vector c_i associated with $y_i \in Z_N^*$ $(1 \leq i \leq k)$, there exists a nonzero vector $e = (e_1, e_2, \ldots, e_k)$ over $GF(2)$ such that $e_1 c_1 + e_2 c_2 + \cdots + e_k c_k \equiv 0 \pmod 2$. This implies that $z \equiv \boldsymbol{y} \uparrow e \pmod N$ is a square modulo N and this contradicts the fact that $q^\ell \equiv \boldsymbol{y} \uparrow \mathrm{bin}(\ell) \pmod N$ is not a square modulo N for each ℓ $(1 \leq \ell \leq 2^k - 1)$. The probability P_{dom} that the machine D samples a dominant vector $\boldsymbol{y}$ of Z_N^* is bounded by $P_{dom} = 1 - (1 - P_{ind})^m > 1 - (1 - 2/7)^m$. Then letting $m \geq 5$, $P_{dom} \geq 3/4$. Since the machine D queries to the promise oracle for $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$ at most $m2^k$ times, it runs in probabilistic polynomial (in $|N|$) time.

Thus on input $\langle x, N \rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$, the machine D with an access to the promise oracle for promised QR $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$ outputs either a dominant vector $\boldsymbol{y}$ of Z_N^* with probability at least $3/4$ or $\perp$ with probability at most $1/4$. ∎

The lemma below is the essential to reduce the communication complexity of a competitive interactive proof system for promised QR $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$.

Lemma 14. *Let N be the product of k distinct odd primes and let a vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$ be dominant of Z_N^*. Then for any $x \in Z_N^*$, there exists a* unique *vector $e = (e_1, e_2, \ldots, e_k)$ over $GF(2)$ such that $x \simeq \boldsymbol{y} \uparrow e$.*

Proof. We assume that $N = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$, where $p_1, p_2, \ldots, p_k$ are distinct odd primes and $\alpha_i \geq 1$ for each i $(1 \leq i \leq k)$. Let $\mathbf{c}_x$ be a vector associated with $x \in Z_N^*$ and let $\boldsymbol{d}_i$ be a vector associated with $y_i \in Z_N^*$ for each i $(1 \leq i \leq k)$. Since $\boldsymbol{y}$ is dominant of Z_N^*, $\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_k$ are linearly independent over $GF(2)$. Then there exists a unique vector $e = (e_1, e_2, \ldots, e_k)$ over $GF(2)$ such that $\mathbf{c}_x \equiv e_1 \boldsymbol{d}_1 + e_2 \boldsymbol{d}_2 + \cdots + e_k \boldsymbol{d}_k \pmod 2$. Here we define $z \in Z_N^*$ to be $z \equiv \boldsymbol{y} \uparrow e \pmod N$. From the property of the Jacobi symbol, it follows that $Q_{p_i}(x) = Q_{p_i}(z)$ for each i $(1 \leq i \leq k)$ and thus $x \simeq z \simeq \boldsymbol{y} \uparrow e$.

The uniqueness of a vector $e = (e_1, e_2, \ldots, e_k)$ can be shown by contradiction. Here we assume that there exist distinct vectors $e = (e_1, e_2, \ldots, e_k)$ and $\boldsymbol{f} = (f_1, f_2, \ldots, f_k)$ over $GF(2)$ such that $x \simeq \boldsymbol{y} \uparrow e \simeq \boldsymbol{y} \uparrow \boldsymbol{f}$. Then for a vector $\mathbf{c}_x$ associated with $x \in Z_N^*$, $\mathbf{c}_x \equiv e_1 \boldsymbol{d}_1 + e_2 \boldsymbol{d}_2 + \cdots + e_k \boldsymbol{d}_k \equiv f_1 \boldsymbol{d}_1 + f_2 \boldsymbol{d}_2 + \cdots + f_k \boldsymbol{d}_k \pmod 2$. This implies that there exists a nonzero vector $\boldsymbol{g} = (g_1, g_2, \ldots, g_k)$ over $GF(2)$ such that $g_1 \boldsymbol{d}_1 + g_2 \boldsymbol{d}_2 + \cdots + g_k \boldsymbol{d}_k \equiv 0 \pmod 2$, where $g_i \equiv e_i + f_i \pmod 2$ for each i $(1 \leq i \leq k)$, and this contradicts the assumption that $\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_k$ are linearly independent over $GF(2)$. ■

The following lemma shows that if $k = O(\log |N|)$, then there exists an efficient algorithm that for a dominant vector $\boldsymbol{y}$ of Z_N^* and any $z \in Z_N^*$, finds a (unique) vector $\boldsymbol{f} \in \{0,1\}^k$ satisfying $z \simeq \boldsymbol{y} \uparrow \boldsymbol{f}$.

Lemma 15. *Let N be the product of k distinct odd primes. Let $\boldsymbol{y}$ be dominant of Z_N^* and let $z \in Z_N^*$. If $k = O(\log |N|)$, then there exists a deterministic polynomial time algorithm* FIND *with an access to the promise oracle for $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$ that on input $\langle \boldsymbol{y}, z \rangle$ always outputs a (unique) vector $\boldsymbol{f} \in \{0,1\}^k$ such that $z \simeq \boldsymbol{y} \uparrow \boldsymbol{f}$.*

Proof. The following is an algorithm with an access to the promise oracle for $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$ that on input $\langle \boldsymbol{y}, z \rangle$ outputs $\boldsymbol{f} \in \{0,1\}^k$ such that $z \simeq \boldsymbol{y} \uparrow \boldsymbol{f}$.

Algorithm FIND:

Input: $\langle \boldsymbol{y}, z \rangle$, where $\boldsymbol{y}$ is dominant of Z_N^* and $z \in Z_N^*$.
Step 1: Compute $q_\ell \equiv (\boldsymbol{y} \uparrow \mathrm{bin}(\ell)) \times z \pmod N$ for each ℓ $(0 \leq \ell \leq 2^k - 1)$.
Step 2: Query q_ℓ to the promise oracle for $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$ to get the answer $a_\ell \in \{0,1\}$ for each ℓ $(0 \leq \ell \leq 2^k - 1)$.
Step 3: If $a_\ell = 1$ for some ℓ $(0 \leq \ell \leq 2^k - 1)$, then outputs $\boldsymbol{f} = \mathrm{bin}(\ell)$; otherwise output $\perp$.
Output: $\boldsymbol{f} \in \{0,1\}^k$ such that $z \simeq \boldsymbol{y} \uparrow \boldsymbol{f}$.

It follows from Lemma 14 that if y is dominant of Z_N^*, the algorithm FIND always finds a *unique* $f \in \{0,1\}^k$ such that $z \simeq y \uparrow f$. Since $k = O(\log |N|)$ and the algorithm FIND queries to the promise oracle for $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$ at most 2^k times, the algorithm FIND runs in deterministic polynomial (in $|N|$) time. ■

4.2 A Low Communication Competitive IP for Promised QR

We now describe the whole protocol of a competitive interactive proof system for promised QR $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$ with considerably low communication complexity.

Protocol PQR-2: A Competitive IP for Promised QR

common input: $\langle x, N \rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$, where $k = O(\log |N|)$.

P1: P runs the machine D to sample a vector $y = (y_1, y_2, \ldots, y_k)$ as a candidate of dominant of Z_N^*.

$P \to V$: $y = (y_1, y_2, \ldots, y_k)$.

V1-1: If V receives $\perp$ from P, then V halts and rejects $\langle x, N \rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$; otherwise V continues.

V1-2: V chooses $a_j \in_{\mathrm{R}} \{0,1\}^k$ and $r_j \in_{\mathrm{R}} Z_N^*$ and computes $z_j \equiv (y \uparrow a_j) \times r_j^2 \pmod N$ for each j $(0 \le j \le 1)$.

$V \to P$: $z_0, z_1 \in Z_N^*$.

P2: P computes $\alpha_j \in \{0,1\}^k$ such that $z_j \simeq y \uparrow \alpha_j$ for each j $(0 \le j \le 1)$.

$P \to V$: $\alpha_0, \alpha_1 \in \{0,1\}^k$.

V2-1: V checks that $\alpha_j = a_j$ for each j $(0 \le j \le 1)$.

V2-2: If either $\alpha_0 \neq a_0$ or $\alpha_1 \neq a_1$, then V halts and rejects $\langle x, N \rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$; otherwise V continues.

V2-3: V chooses $e_j \in_{\mathrm{R}} \{0,1\}$, $b_j \in_{\mathrm{R}} \{0,1\}^k$, $s_j \in_{\mathrm{R}} Z_N^*$ for each j $(0 \le j \le 1)$.

V2-4: V computes $w_j \equiv x^{e_j} \times (y \uparrow b_j) \times s_j^2 \pmod N$ for each j $(0 \le j \le 1)$.

$V \to P$: $w_0, w_1 \in Z_N^*$.

P3: P computes $\beta_j \in \{0,1\}^k$ such that $w_j \simeq y \uparrow \beta_j$ for each j $(0 \le j \le 1)$.

$P \to V$: $\beta_0, \beta_1 \in \{0,1\}^k$.

V3-1: V checks that $\beta_j = b_j$ for each j $(0 \le j \le 1)$.

V3-2: If either $\beta_0 \neq b_0$ or $\beta_1 \neq b_1$, then V halts and rejects $\langle x, N \rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$; otherwise V halts and accepts $\langle x, N \rangle \in \mathrm{QR}_k \cup \mathrm{QNR}_k$.

Correctness of PQR-2: We show that even when $k = O(\log |N|)$, Protocol PQR-2 is a competitive interactive proof system for promised QR $\langle \mathrm{QR}_k, \mathrm{QNR}_k \rangle$.

(*Completeness*) Assume that $\langle x, N \rangle \in \mathrm{QR}_k$. It follows from Lemma 13 that in step V1-1, V receives a dominant vector $y = (y_1, y_2, \ldots, y_k)$ of Z_N^* from P with probability at least 3/4.

Assume that y is dominant of Z_N^*. Then it follows from Lemma 14 that there exists a unique vector $\alpha_j \in \{0,1\}^k$ such that $z_j \simeq y \uparrow \alpha_j$ for each j $(0 \le j \le 1)$. To find such a vector $\alpha_j \in \{0,1\}^k$, P executes the algorithm FIND on input $\langle y, z_j \rangle$ for each j $(0 \le j \le 1)$. Since $k = O(\log |N|)$, P runs in deterministic polynomial (in $|N|$) time in step P2 (see Lemma 15). From the assumption that

y is dominant of Z_N^*, it follows that $\alpha_j = a_j$ for each j $(0 \leq j \leq 1)$. This implies that if y is dominant of Z_N^*, then V never rejects $\langle x, N\rangle \in \mathrm{QR}_k$ in step V2-2.

For any $r \in Z_N^*$, let $z \equiv xr \pmod N$. From the fundamental property of the Jacobi symbol, it is easy to see that $z \simeq r$ if $\langle x, N\rangle \in \mathrm{QR}_k$. This implies that $w_j \simeq y \uparrow \beta_j$ regardless of the value of $e_j \in \{0,1\}$ for each j $(0 \leq j \leq 1)$. Then it follows from Lemma 14 that P can find a *unique* vector $\beta_j \in \{0,1\}^k$ by running the algorithm FIND on input $\langle y, w_j\rangle$ for each j $(0 \leq j \leq 1)$. Since $k = O(\log|N|)$, P runs in deterministic polynomial (in $|N|$) time in step P3 (see Lemma 15). From the assumption that y is dominant of Z_N^*, it follows that $\beta_j = b_j$ for each j $(0 \leq j \leq 1)$. This implies that if y is dominant of Z_N^*, then V always accepts $\langle x, N\rangle \in \mathrm{QR}_k$ in step V3-2.

Thus for any $\langle x, N\rangle \in \mathrm{QR}_k$, the (probabilistic polynomial time bounded) honest prover P with an access to the promise oracle for $\langle \mathrm{QR}_k, \mathrm{QNR}_k\rangle$ can cause the honest verifier V to accept $\langle x, N\rangle \in \mathrm{QR}_k$ with probability at least 3/4.

(*Soundness*) Assume that $\langle x, N\rangle \in \mathrm{QNR}_k$. If V receives $\perp$ from P in step V1-1, then V halts and rejects $\langle x, N\rangle \in \mathrm{QNR}_k$. Then any dishonest prover P^* needs to send to V a vector $y = (y_1, y_2, \ldots, y_k)$ over Z_N^*. Assume that y is not dominant of Z_N^*. For each $z_j \in Z_N^*$ $(0 \leq j \leq 1)$ in step V1-2, there are 2^t $(1 \leq t \leq k)$ possible $\alpha_j \in \{0,1\}^k$ such that $z_j \simeq y \uparrow \alpha_j$ for each j $(0 \leq j \leq 1)$. This implies that if y is not dominant of Z_N^*, then with probability at most $2^{-2t} \leq 1/4$, any all powerful P^* can find a vector $\alpha_j \in \{0,1\}^k$ such that $\alpha_j = a_j$ for each j $(0 \leq j \leq 1)$ in step P2. Thus if y is not dominant of Z_N^*, then V halts and rejects $\langle x, N\rangle \in \mathrm{QNR}_k$ in step V2-2 with probability at least 3/4.

Assume that y is dominant of Z_N^*. Since $\langle x, N\rangle \in \mathrm{QNR}_k$, there exists a unique vector $e \in \{0,1\}^k$ such that $x \simeq y \uparrow e$ and $e \neq 0$. For each j $(0 \leq j \leq 1)$, there exist $\beta_j^0, \beta_j^1 \in \{0,1\}^k$ such that $w_j \simeq y \uparrow \beta_j^0$ and $w_j \simeq x \times (y \uparrow \beta_j^1)$. Indeed, for i, j $(0 \leq i, j \leq 1)$, $\beta_j^i \equiv b_j + \{(e_j + i) \times e\} \pmod 2$. This implies that any dishonest prover P^* cannot guess better at random the value of $e_j \in \{0,1\}$ for each j $(0 \leq j \leq 1)$ even if it is infinitely powerful. Thus if y is dominant of Z_N^*, then with probability at most 1/4, any all powerful P^* can find a vector $\beta_j \in \{0,1\}^k$ such that $\beta_j = b_j$ for each j $(0 \leq j \leq 1)$ in step P3. Then V halts and rejects $\langle x, N\rangle \in \mathrm{QNR}_k$ in step V3-2 with probability at least 3/4.

Thus for any $\langle x, N\rangle \in \mathrm{QNR}_k$, any all powerful dishonest prover P^* can cause the honest verifier V to accept $\langle x, N\rangle \in \mathrm{QNR}_k$ with probability at most 1/4. ■

5 Discussions

Let $CC_P(\mathrm{A})$ (resp. $CC_V(\mathrm{A})$) be the total number of bits sent by the prover P (resp. the verifier V) to the verifier V (resp. the prover P) in the protocol A.

On one hand, in Protocol PQR-1 (see section 3), we have $CC_P(\mathrm{PQR\text{-}1}) = (2^k - 1)(|N| + 2^k + 2)$ and $CC_V(\mathrm{PQR\text{-}1}) = (2^k - 1)(2^k + 2)\,|N|$. On the other hand, in Protocol PQR-2 (see section 4), we have $CC_P(\mathrm{PQR\text{-}2}) = k\,|N| + 2k + 2k = k(|N| + 4)$ and $CC_V(\mathrm{PQR\text{-}2}) = 2\,|N| + 2\,|N| = 4\,|N|$. From the fact that $k \geq 1$,

it immediately follows that

$$\frac{CC_P(\text{PQR-2})}{CC_P(\text{PQR-1})} = \frac{k(|N|+4)}{(2^k-1)(|N|+2^k+2)} \leq \frac{k(|N|+4)}{(2^k-1)(|N|+2+2)} = \frac{k}{2^k-1};$$
$$\frac{CC_V(\text{PQR-2})}{CC_V(\text{PQR-1})} = \frac{4|N|}{(2^k-1)(2^k+2)|N|} \leq \frac{4|N|}{(2^{2k}+2-2)|N|} = \frac{4}{2^{2k}},$$

and thus Protocol PQR-2 considerably reduces the communication complexity.

In Protocol PQR-1, the constraint that $k = O(\log\log|N|)$ is caused by the task in step P1. In step P1, the prover P queries to the promise oracle for $\langle \text{QR}_k, \text{QNR}_k \rangle$ at most 2^{2^k} times to sample a representative vector $\boldsymbol{y}$ of Z_N^*. Then we must assume that $k = O(\log\log|N|)$ in Protocol PQR-1 to guarantee that P runs in probabilistic polynomial (in $|N|$) time. In Protocol PQR-2, however, the prover P queries to the promise oracle for $\langle \text{QR}_k, \text{QNR}_k \rangle$ to sample a dominant vector $\boldsymbol{y}$ of Z_N^* at most 2^k times. Then we must assume that $k = O(\log|N|)$ in Protocol PQR-2 to guarantee that P runs in probabilistic polynomial (in $|N|$) time. The essential of a dominant vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$ of Z_N^* is that it can generate a representative vector $\boldsymbol{y}' = (y'_1, y'_2 \ldots, y'_{2^k-1})$ of Z_N^* by $y'_\ell \equiv \boldsymbol{y} \uparrow \text{bin}(\ell) \pmod N$ for each ℓ $(1 \leq \ell \leq 2^k - 1)$.

References

1. Beigel, R. and Feigenbaum, J., "On Being Coherent Without Being Very Hard," *Computational Complexity*, Vol.2, No.1, pp.1-17 (1992).
2. Bellare, M. and Goldwasser, S., "The Complexity of Decision versus Search," MIT/LCS/TM-444 (April 1991).
3. Bellare, M. and Goldwasser, S., "The Complexity of Decision versus Search," to appear in *SIAM J. on Comput.*
4. Babai, L. and Moran, S., "Arthur-Merlin Games: A Randomized Proof Systems and a Hierarchy of Complexity Classes," *JCSS*, Vol.36, pp.254-276 (1988).
5. Even, S., Selman, A., and Yacobi, Y., "The Complexity of Promise Problems with Applications to Public-Key Cryptography," *Information and Control*, Vol.61, pp.159-173 (1984).
6. Feige, U., Fiat, A., and Shamir, A., "Zero-Knowledge Proofs of Identity," *J. of Cryptology*, Vol.1, pp.77-94 (1988).
7. Goldwasser, S., Micali, S., and Rackoff, C., The Knowledge Complexity of Interactive Proof Systems," *SIAM J. on Comput.*, Vol.18, No.1, pp.186-208 (1989).
8. Goldreich, O., Micali, S., and Wigderson, A., "Proofs That Yield Nothing But Their Validity or All Languages in $\mathcal{NP}$ Have Zero-Knowledge Interactive Proof Systems," *J. of the ACM*, Vol.38, No.1, pp.691-729 (1991).
9. Grollmann, J. and Selman, A., "Complexity Measures for Public-Key Cryptosystems," *SIAM J. on Comput.*, Vol.17, No.2, pp.309-335 (1988).
10. Tompa, M. and Woll, H., "Random Self-Reducibility and Zero-Knowledge Interactive Proofs of Possession of Information," *Proc. of FOCS*, pp.472-482 (1987).