Some Complexity Questions Related to Distributive Computing[*]

(Preliminary Report)

Andrew Chi-Chih Yao
Stanford University and Xerox Parc
Palo Alto, California

1. Introduction.

Let $M = \{0, 1, 2, ..., m-1\}$, $N = \{0, 1, 2, ..., n-1\}$, and $f{:}M \times N \to \{0, 1\}$ a Boolean-valued function. We will be interested in the following problem and its related questions. Let $i \in M, j \in N$ be integers known only to two persons P_1 and P_2, respectively. For P_1 and P_2 to determine cooperatively the value $f(i, j)$, they send information to each other alternately, one bit at a time, according to some algorithm. The quantity of interest, which measures the information exchange necessary for computing f, is the minimum number of bits exchanged in any algorithm. For example, if $f(i, j) = (i + j) \bmod 2$, then 1 bit of information (conveying whether i is odd) sent from P_1 to P_2 will enable P_2 to determine $f(i, j)$, and this is clearly the best possible.

The above problem is a variation of a model of Abelson [1] concerning information transfer in distributive computions. In Abelson's model, a "smooth" real-valued function $f(x_1, x_2, ..., x_s; y_1, y_2, ..., y_t)$ is to be computed with processor P_1 knowing the values of $x_1, x_2, ..., x_s$ and P_2 knowing $y_1, y_2, ..., y_t$. Assuming that P_1 and P_2 can send values that are smooth functions to each other, Abelson obtained bounds on the minimum number of such values exchanged to compute f. Our model corresponds to a situation when f is a Boolean function, the x's and y's are Boolean variables ($m = 2^s$, $n = 2^t$), and the values exchanged are bits. In contrast to the analytic flavor of Abelson's model, the present framework addresses computations which are combinatorial in nature.

2. The Deterministic Model.

Consider deterministic algorithms that control the bits exchanged between P_1 and P_2 in deciding the value of f. Initially the value i is known to P_1, and j to P_2. The computation proceeds in the following way: P_1 first sends $a_1 \in \{0, 1\}$ to P_2; seeing a_1, P_2 sends b_1 to P_1; seeing b_1, P_1 sends a_2 to P_2,.... The choice of a_k (or b_k) can depend on all the bits communicated so far. Precisely, an algorithm A specifies the Boolean functions $\{h_k(i; u_1, u_2, ..., u_{k-1}), l_k(j; v_1, v_2, ..., v_k) \mid k = 1, 2, ...\}$ and the bits a_k, b_k are determined by $a_k = h_k(i; b_1, b_2, ..., b_{k-1})$, $b_k = l_k(j; a_1, a_2, ..., a_k)$. The computation ends when either P_1 or P_2 has enough information to determine $f(i, j)$, and sends a special symbol "halt" to the other processor. The cost $\alpha(A)$ is defined to be the maximum number of bits (0 and 1) exchanged for any $i \in M, j \in N$. The *two-way complexity* of f is defined as

$$C(f; 1 \leftrightarrow 2) = min\{\alpha(A) \mid A \text{ computes } f\}.$$

To study the quantity C, we first develop a more convenient way to view the computation. It will not be difficult to see that any algorithm as described above can be represented this way. We illustrate this with an example. Let f be the function defined in Figure 1, an algorithm A for computing f is given as a decision tree in Figure 2.

Each internal node in the decision tree has up to 4 children (two of them leaves and two internal nodes). Branches leading to leaves are labeled H (halt), and branches to internal nodes labeled X or Y. (We use X, Y instead of 0, 1 to avoid confusion with function values.) Moves are made by P_1 and P_2 alternately, starting at the root. Thus, label 1, 3, 5,... along a path are the signals sent by P_1, while label 2, 4, 6,... by P_2. The bit attached to a leaf gives the desired function value $f(i, j)$. We now describe the rules for selecting the branching at an internal node.

To each node v is associated a matrix $S(v) \times T(v) \subseteq M \times N$. At the root r, we have $S(r) \times T(r) = M \times N$. For an internal node v on an odd level (the root being on level 1) with children $v_1, ..., v_l$ ($l \leq 4$), we have $T(v) = T(v_k)$ for $1 \leq k \leq l$, and $\{S(v_k) \mid 1 \leq k \leq l\}$ form a disjoint partition of $S(v)$. On the even levels, similar conditions hold with the roles of S and T reversed. In P_1's moves from a node v (on an odd level), it selects the unique child v_k with $i \in S(v_k)$. A similar statemtent can be said of P_2. A node v is a leaf if and only if f is constant in the block $S(v) \times T(v)$. Note that $\alpha(A) = height(A) - 1$.

By induction on the value of mn, it can be shown that any algorithm can be represented in the above fashion. We shall now prove some results.

Definition. Let f be a Boolean-valued function on $M \times N$. We shall call a Cartesian product $S \times T$ (where $S \subseteq M$, $T \subseteq N$) an *f-monochromatic rectangle* if f is constant over $S \times T$. A *k-decomposition* of f is a family $\mathcal{T} = \{S_1 \times T_1, S_2 \times T_2, ..., S_k \times T_k\}$

[*] Part of this research was done while the author was visiting Bell Laboratories, Murray Hill, New Jersey.

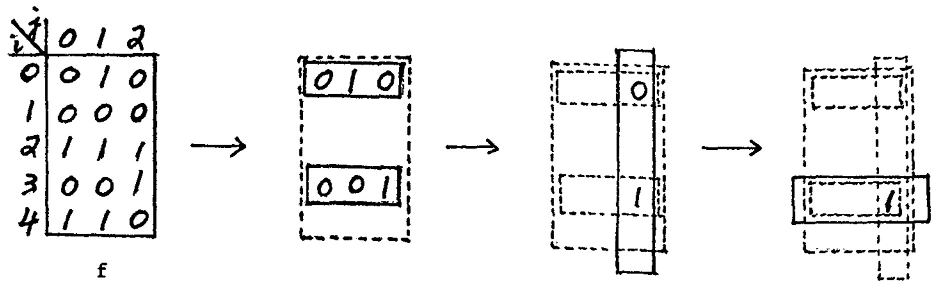

Figure 1 A function f, and the successive steps taken by the algorithm below in finding f(3,2).

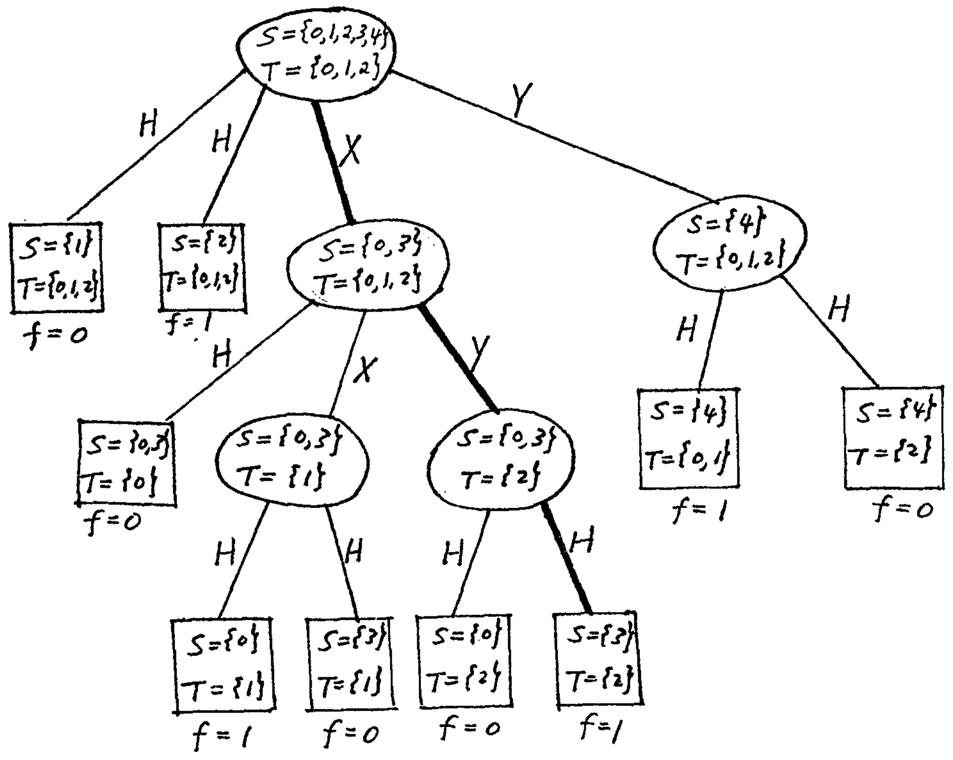

Figure 2. An algorithm for computing f. The sequence of signals exchanged for input i=3, j=2 is XYH.

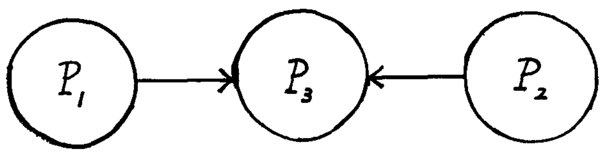

Figure 3 Illustration for $C'_\varepsilon(f;1\rightarrow 3\leftarrow 2)$ in Section 4.

of k disjoint f-monochromatic rectangles that partition $M \times N$. Let $d(f)$ be the minimum k such that a k-decomposition of f exists.

Theorem 1. $C(f; 1 \leftrightarrow 2) \geq \log_2 d(f) - 2$.

Corollary. Let $\mathcal{F}_n$ be the set of all Boolean-valued function on $N \times N$. Then, with probability $1 - O(2^{-n^2/2})$, a random $f \in \mathcal{F}_n$ satisfies

$$C(f; 1 \leftrightarrow 2) \geq \log_2 n - 4.$$

Proof. Let A be an optimal algorithm for computing f, represented in the form of a tree as described earlier. As each internal node has at most two children that are leaves, it follows that

$$\begin{aligned} d(f) &\leq \text{number of leaves} \\ &\leq 2 \times I(f), \end{aligned} \tag{1}$$

where $I(f)$ is the number of internal nodes of f. As the internal nodes form a tree with height $\alpha(A)$ and branching at most 2, we have

$$\frac{(I(f)+1)}{2} \leq 2^{\alpha(A)}. \tag{2}$$

From (1) and (2), we obtain

$$2^{\alpha(A)} \geq \frac{d(f)}{4}.$$

This implies

$$C(f; 1 \leftrightarrow 2) \geq \log_2 d(f) - 2.$$

This proves Theorem 1. ∎

proof of Corollary. Since there are at most 2^{2n} sets of the form $S \times T$ with $S \subseteq N$, $T \subseteq N$ (S, T may be empty), the number of $f \in \mathcal{F}_n$ with $d(f) \leq k$ is at most 2^{2kn}. Now there are 2^{n^2} distinct functions in $\mathcal{F}_n$. The fraction of f satisfying $d(f) > n/4$ (hence with $C(f; 1 \leftrightarrow 2) > log_2 n - 4$) is, therefore, at least

$$(2^{n^2} - 2^{2nn/4})/2^{n^2} = 1 - O(2^{-n^2/2}).$$

This proves the corollary. ∎

As $C(f; 1 \leftrightarrow 2) \leq 2\lceil \log_2 n \rceil$ (P_1 can simply sends the binary representation of i to P_2), the above corollary determines up to a factor of two the complexity of almost all Boolean-valued functions for large n.

Below we give some specific functions whose $C(f; 1 \leftrightarrow 2)$ are of the order $\log n$, because of Theorem 1 (proofs omitted).

Example 1. The identification function: $f(i,j) = \delta_{ij}$ for $i,j \in \{0, 1, ..., n-1\}$.

Example 2. The relatively-prime testing function: $f(i,j) = 1$ iff i and j are relatively prime, where $i,j \in \{0, 1, ..., n-1\}$.

Example 3. The ordering function: $f(i,j) = 1$ iff $0 \leq i \leq j < n$.

Example 4. The set-intersection function: $f(i,j) = 1$ iff the k-th bits of i and j are both 1 for some k, where $i,j \in \{0, 1, ..., n-1\}$.

How good is the bound given in Theorem, as a function of k? The following theorem states that it can be achieved (up to a constant factor) if f has a "planar" $d(f)$-decomposition.

Definition. A k-decomposition of f, $\mathcal{I} = \{S_1 \times T_1, S_2 \times T_2, ... S_k \times T_k\}$, is *planar* if each S_i (and T_i) consists of a block of consecutive integers.

Theorem 2. If f has a planar k-decomposition ($k \geq 1$), then

$$C(f; 1 \leftrightarrow 2) \leq \frac{2\log_2 k}{\log_2(4/3)} + 6.$$

Proof. It is easy to see that $C(f; 1 \leftrightarrow 2) \leq k$. Therefore, the theorem is true for $k \leq 4$. We shall prove by induction that, for all $k \geq 5$,

$$C(f; 1 \leftrightarrow 2) \leq \frac{2\log_2(k-4)}{\log_2(4/3)} + 6. \tag{3}$$

The inequality is clearly true for $k = 5, 6$. Now suppose that $k \geq 7$. We shall show that, for some functions f_1 and f_2, each with a planar $(\lfloor 3k/4 \rfloor + 1)$-decomposition,

$$C(f; 1 \leftrightarrow 2) \leq 2 + max\{C(f_a; 1 \leftrightarrow 2) \mid a = 1, 2\}. \tag{4}$$

By the induction hypothesis, this would imply

$$\begin{aligned} C(f; 1 \leftrightarrow 2) &\leq 2 + 2(\log_2(\lfloor 3k/4 \rfloor + 1 - 4))/\log_2(4/3) + 6 \\ &\leq 2(\log_2(k-4))/\log_2(4/3) + 6, \end{aligned}$$

hence completing the induction.

It remains to prove (4); we distinguish two cases.

Case A. There exists an $s \in M$ such that there are $h \geq \lceil k/2 \rceil$ distinct $S_l \times T_l$ with $i \in S_l$. Without loss of generality, we can assume that they are $S_1 \times T_1, S_2 \times T_2, ..., S_h \times T_h$, and that $T_1, T_2, ..., T_h$ are consecutive intervals covering the set $N = \{0, 1, ..., n-1\}$. Let $N_1 = T_1 \bigcup T_2 \bigcup \cdots \bigcup T_{\lceil h/2 \rceil}$ and $N_2 = N - N_1$. It is easy to see that each $M \times N_a$ ($a = 1, 2$) intersects at most $k - \lfloor h/2 \rfloor \leq (3k/4) + 1$ sets $S_l \times T_l$ for $j \in \{1, 2, ..., k\}$. Thus each f_a ($a = 1, 2$), the restriction of f to $M \times N_a$, has a planar $(\lfloor 3k/4 \rfloor + 1)$-decomposition. Since P_2 can communicateto P_1 in 1 bit whether the variable j (owned by P_2) is in N_1 or N_2, we obtain formula (4).

Case B. Suppose the condition in *Case A* is not satisfied. Let $s \in S$ be the smallest s such that there are at least $k/4$ $S_l \times T_l \subseteq M_1 \times N$, where $M_1 = \{0, 1, ..., s\}$. Denote $M - M_1$ by M_2. As each S_l is an interval, any $S_l \times T_l$ that intersects $M_1 \times N$ must satisfy either $s \in S_l$ or $S_l \times T_l \subseteq (M_1 - \{s\}) \times N$. This implies that at most $\lceil k/2 \rceil + k/4 \leq 3k/4 + 1$ sets $S_l \times T_l$ may intersect $M_1 \times N$. Thus, each $M_a \times N$ intersects at most $\lfloor 3k/4 \rfloor + 1$ sets $S_l \times T_l$, and hence each f_a, the restriction of f to $M_a \times N$, has a planar $(\lfloor 3k/4 \rfloor + 1)$-decomposition. As only 1 bit sent from P_1 to P_2 suffices to express whether the variable i owned by P_1 is in M_1, formula (4) follows.

This completes the proof of Theorem 2. ∎

For general non-planar decompositions, we have the following theorem.

Theorem 3. There exists a constant $\lambda > 0$ such that, for all f,

$$C(f; 1 \leftrightarrow 2) \leq \lambda \sqrt{d(f) \log_2 d(f)}.$$

Skech of Proof. Let $k = d(f)$. We shall prove the theorem by induction on k. For each $s \in M$, let $pat_s = \{l \mid s \in S_l\}$. Either there exists an s with $|pat_s| \geq 2\sqrt{k/\log_2 k}$ or there are altogether no more

than
$$\binom{k}{\lceil 2\sqrt{k/\log_2 k}\rceil} \leq e^{\lambda_1\sqrt{k\log_2 k}}$$
distinct pat_s, where $\lambda_1 > 0$ is a constant. In the former case, by an argument similar to the one used in the last theorem, one can show that there exit f_α ($\alpha = 1, 2$) with $d(f_\alpha) \leq k - \lfloor\sqrt{k/\log_2 k}\rfloor$ such that

$$C(f; 1 \leftrightarrow 2) \leq 2 + max\{\, C(f_\alpha; 1 \leftrightarrow 2) \mid \alpha = 1, 2\,\}. \qquad (5)$$

In the latter case, $O(\lambda_1\sqrt{k\log_2 k})$ bits sent from P_1 to P_2 would enable P_2 to decide the value of f immediately. The induction step can be carried out in either case.

We have proved Theorem 3. ∎

3. The Probabilistic Models.

An interesting recent innovation in algorithm design is the inclusion of stochastic moves and the allowance for an ϵ probability of error (See e.g. Rabin[2]). If each processor P_i is given a random number generator, can they determine the value of f with much less information exchanged? We discuss two models.

3.1 One-Way Probabilistic Communications.

To determine $f(i,j)$, processor P_1 sends stochastically a string to P_2, based on which P_2 stochastically decides the value of $f(i,j)$ with chance of error $\leq \epsilon$. There are several different ways to define the complexity. In the simplest case, let all the strings transmitted be of uniform length. The *1-way prbabilistic complexity* of f, $C'_\epsilon(f; 1 \leftrightarrow 2)$, is then defined to be $\lceil\log_2 k\rceil$, where k is the minimun integer such that the following system of inequalities has a solution in $\vec{p}_i = (p_{i1}, p_{i2}, ..., p_{ik})$, $1 \leq i \leq m$, and $\vec{q}_j = (q_{j1}, q_{j2}, ..., q_{jk})$, $1 \leq j \leq n$.

$$\begin{cases} \sum_\ell p_{i\ell} = 1 & \text{for all } \; i, j, \\ p_{i\ell} \geq 0; \quad 1 \geq q_{i\ell} \geq 0 & \text{for all } \; i, j, \ell, \\ \vec{p}_i \cdot \vec{q}_j \geq 1 - \epsilon & \text{if } \; f(i,j) = 1, \\ \vec{p}_i \cdot \vec{q}_j < \epsilon & \text{if } \; f(i,j) = 0. \end{cases} \qquad (6)$$

Remark. Regard $(f(i,j))$ as a 0–1 matrix, and denote by $nrow(f)$, $ncol(f)$ the numbers of distinct rows and columns of f, respectively. The 1-way complexity for the deterministic situation is easily seen to be $\lceil\log_2(nrow(f))\rceil$.

The complexity for a special case, the identification function defined in Section 2, was studied in Rabin and Yao[3], and shown to be of order $\log\log n$. The following result determines C'_ϵ for a random f. However, the determination of C'_ϵ for specific functions in general seems very difficult.

Theorem 4. Let $0 < \epsilon < 1/2$ be any fixed number. Denote by $\mathfrak{F}_n$ the set of all Boolean-valued functions on $N \times N$. Then, with probability $1 - O(2^{-n^2/2})$, a random $f \in \mathfrak{F}_n$ satisfies

$$\lceil\log_2 n\rceil \geq C'_\epsilon(f; 1 \rightarrow 2) \geq \lceil\log_2 n\rceil - \log_2\log_2 n - 2.$$

*Proof.*The first inequality is clearly true for all f, so we need only prove the second inequality.

Let $\mathfrak{F}_{n,k} \subseteq \mathfrak{F}_n$ denote the set of those f, for which k-component vectors $\{\,\vec{p}_i, \vec{q}_j$ exist that satisfy (6). For each $f \in \mathfrak{F}_n$ we choose a solution and define a (2kn)-tuple

$$\gamma(f) = (\lceil 4p_{ij}k/(1-2\epsilon)\rceil, \quad \lceil 4q_{ij}k/(1-2\epsilon)\rceil \mid 1 \leq i \leq n, 1 \leq j \leq k)$$

Call $\gamma(f)$ the *representative* of f. Clearly, there are at most

$$(1 + \lceil 4k/(1-2\epsilon)\rceil)^{2kn} = \exp(2kn\ln k + O(1))$$

distinct representatives.

Now we assert that no two distinct f and f' in $\mathfrak{F}_{n,k}$ may have the same representative. Otherwise, let $\vec{p}_i, \vec{q}_j$ and $\vec{p}'_i, \vec{q}'_j$ be associated with f and f', respectively. Choose s, t so that $f(s,t) \neq f'(s,t)$. According to (6), we have

$$|\vec{p}_s \cdot \vec{q}_t - \vec{p}'_s \cdot \vec{q}'_t| \geq 1 - 2\epsilon. \qquad (7)$$

On the other hand, as f and f' have the same representative, we have

$$\begin{aligned} |\vec{p}_s \cdot \vec{q}_t - \vec{p}'_s \cdot \vec{q}'_t| &\leq \sum_\ell p_{s\ell}|q_{t\ell} - q'_{t\ell}| + q_{t\ell}|p_{s\ell} - p'_{s\ell}| \\ &\leq 2k\frac{1-2\epsilon}{4k} \\ &< 1 - 2\epsilon, \end{aligned}$$

contradicting (7). This proves the assertion.

From the above discussions, we have

$$\frac{|\mathfrak{F}_{n,k}|}{|\mathfrak{F}_n|} \leq \frac{\exp(2kn\ln k + O(1))}{2^{n^2}}.$$

Take $k = n/(4\ln n)$. This leads to the conclusion that the probability is at most $O(2^{-n^2/2})$, for $C'_\epsilon(f; 1 \rightarrow 2) \leq \lceil\log_2 n\rceil - \log_2\ln n - 2$. Theorem 4 follows easily. ∎

3.2 Two-Way Probabilistic Communications.

In the basic 2-way model of Section 2, one can also allow stochastic moves in both processors. Let ϵ (where $0 < \epsilon < 1/2$) be the error probaibility allowed. Define $\alpha'(A)$ to be the expected number of bits transferred for the worst input under algorithm A, and let $C'_\epsilon(f; 1 \leftrightarrow 2) = inf\{\, \alpha'(A) \mid A \;$ an algorithm with error probability at most $\; \epsilon\,\}$. The following result provides a limit to the power of stochastic algorithms. For example, it means that the 2-way probabilistic complexity for the identification function is also of order $\log\log n$.

Theorem 5. Let $0 < \epsilon < 1/2$ be fixed. Then there exists a constant $\lambda' > 0$ such that, for any f,

$$C'_\epsilon(f; 1 \leftrightarrow 2) \geq \lambda'(\log_2\log_2(nrow(f)) + \log_2\log_2(ncol(f))).$$

Proof. We omit the proof here because of its complexity. ∎

4. Concluding Remarks.

In this paper we have studied the information exchange needed, when two processors cooperate to compute Boolean-valued functions. The deterministic 1-way model is mathematically trivial and is completely understood. Below we shall comment on our understanding of the other three models and propose open questions.

A. The deterministic 2-way complexity. It is relatively well understood. Almost all functions in $\mathfrak{F}_n$ have complexity $\approx \log n$, and for many familiar functions, the complexity is determined up to a constant factor. One basic question remains unanswered, however. Let $a_k = max\{C(f; 1 \leftrightarrow 2) \mid d(f) = k\}$. It is known that $c \log k \leq a_k \leq c' \sqrt{k \log k}$. What is a_k ?

B. The Probabilistic 1-way complexity. This is a most interesting subject for future research. It is known that almost all functions in $\mathcal{F}_n$ have complexity $\approx \log n$, yet no specific one is known to have this complexity. We conjecture that the ordering function (Section 2) has complexity $\approx \log n$.

C. The Probabilistic 2-way complexity. It is poorly understood, despite that there is a somewhat difficult lower bound result (Theorem 5). We do not even know the complexity of a random function in $\mathcal{F}_n$. Note that Theorem 5 impies that the probabilistic 2-way communication improves over deterministic 1-way communication at most logarithmically.

D. More than two processors. Most of the results can be extended to communications between more than two processors in some form. We mention one situation that deserves special attention. Suppose three processors P_1, P_2, P_3 cooperate to compute a function $f(i, j)$, with P_1 knowing i and P_2 knowing j initially. Instead of processors P_1, P_2 communicating to each other, suppose that P_1, P_2 both have an 1-way stochastic communication channel to processor P_3 (see Figure 3), who then computes the value of $f(i, j)$ with error probability less than ϵ. Suppose that P_3 uses a Boolean-valued function g on some domain $M' \times N'$, and computes f to be $g(i', j')$ where i' and j' are the integers received from P_1 and P_2 respectively. The complexity $C'_\epsilon(f; 1 \rightarrow 3 \leftarrow 2)$ is then the minimum of $\log |M'| + \log |M'|$ over all possible choice of g, M', N'. An interesting alternative way to look at it is to regard $C'_\epsilon(f; 1 \rightarrow 3 \leftarrow 2)$ as the log of the minimun *table size* $|M'||N'|$ needed, when we compute $f(i, j)$ by "probabilistic hashing". What is the complexity of, say, the identification function?

E. NP-Completeness. Is the computing of the complexity $C(f; 1 \leftrightarrow 2)$ NP-complete?

References

[1] H. Abelson, *Lower Bounds on Information Transfer in Distributed Computations*, Proc. IEEE 19-th Annual Symp. on Foundations of Computer Science, Ann Arbor, 1978, pp. 151-158.

[2] M. O. Rabin, *Probabilistic Algorithms*, in *Algorithms and Complexity: Recent Results and New Directions*, edited by J. F. Traub, Academic Press, 1976, pp. 21-40.

[3] M. O. Rabin and A. C. Yao, in preparation.