CMU-CS-86-157

A Strong Separation Between k and k-1 Round Communication Complexity For a Constructive Language

Lyle A. McGeoch*

October 23, 1986

Abstract

For every $k \geq 2$, a language is described that can be recognized by two processors exchanging $O(k \log n)$ bits using a simple k-round communication protocol, but that requires $\Omega(n/k^2 \log n)$ bits if only $k-1$ rounds are used. The language is constructive and easily-described. It is a modification of the constructive language that was shown by Duris, Galil, and Schnitger to require $\Omega(n^{1/2}/k^4 \log^3 n)$ bits of communication in $k-1$ rounds.

*Research supported by NSF Grant DCR-8352081 and an NSF Graduate Fellowship.

1 Introduction

The amount of communication that cooperating processors require in order to decide language acceptance is related to complexity measures for VLSI and branching programs. Communication complexity and its applications have been studied by Aho, Ullman, and Yannakakis [1], Chandra, Furst, and Lipton [2], Lipton and Sedgewick [4], and Yao [6]. In this paper we consider a model of communication described by Papadimitriou and Sipser [5] and further explored by Duris, Galil, and Schnitger [3].

Suppose a language $L \subseteq \{0,1\}^*$ must be accepted by two processors, A and B, which each receive a separate half of the input bits. Based on a communication protocol, they take turns transmitting bit strings until one of the processors accepts or rejects. The problem is to determine (as a function of n) the *communication complexity* of L, the number of bits that must be passed between the processors in order to correctly accept or reject any string in $L \cap \{0,1\}^{2n}$.

A *protocol* on $2n$ inputs is a pair $D_n = (\Pi, \Phi)$, where

(a) Π is a partition of $\{1, \ldots, 2n\}$ into two sets S_A and S_B of equal size. This corresponds to a partition of the input into two halves for the two processors.

(b) Φ is a function from $\{0,1\}^n \times \{0,1\}^*$ to $\{0,1\}^* \cup \{\text{accept}, \text{reject}\}$. The two processors, starting with A, take turns applying this function. Each computes the next communication as a function of its n bits of input and the *broadcast history*, the communication so far. The bits generated by a single application of Φ are called a *round* of communication. (The final "accept" or "reject" is not counted as a round.)

Assume that for no broadcast history $c \in \{0,1\}^*$ are there two strings y, y' such that $\Phi(y,c)$ is a prefix of $\Phi(y',c)$. This *prefix-freeness* property implies that each processor can recognize the completion a round of the other's communication based only on the content of the message, without using an explicit end of transmission symbol.

A protocol D_n *computes* $L \subseteq \{0,1\}^{2n}$ if it correctly accepts or rejects each string of length $2n$. Such a protocol has communication complexity $f(n)$ if every string is accepted or rejected with at most $f(n)$ communication. The communication complexity of L, $I(L)$, is $f(n)$ if $f(n)$ is the minimum complexity over all protocols computing L.

A sequence of protocols $\Delta = \langle D_n \rangle$ computes $L \subseteq \{0,1\}^*$ if, for all n, D_n correctly

accepts or rejects each string of length $2n$. The sequence has communication complexity $f(n)$ if every string is accepted or rejected with at most $f(n)$ communication, and $I(L) = f(n)$ if $f(n)$ is the minimum complexity over all sequences of protocols computing L.

This paper will consider protocols using a constant number of rounds. A sequence of protocols has k-round complexity $I_k(L)$ equal to $f(n)$ if $f(n)$ is the minimum complexity over all sequences of protocols computing L, where each protocol uses no more than k rounds of communication on any input.

The assumption of prefix-freeness is included only for simplicity and is not critical to the proof in this paper. We are considering only asymptotic complexity, and communication with an explicit end of transmission symbol can easily be simulated by prefix-free communication with twice as many bits. Prefix-freeness is essential in proofs considering exact communication requirements, such as in [3] and [5].

Papadimitriou and Sipser [5] introduced k-round communication complexity and described a sequence of constructive languages L_k for which $I_k(L_k) = O(k \log n)$ but that they conjectured to have high $(k-1)$-round complexity. They proved that $I_1(L_2) = \Omega(n^{1/2}/\log n)$. Duris, Galil, and Schnitger [3] later showed that $I_{k-1}(L_k) = \Omega(n^{1/2}/k^4 \log^3 n)$. They also proved the existence of languages for all $k \geq 2$ with the same upper bound for k rounds, but a lower bound of $\Omega(n/k)$ for $k-1$ rounds. This is a very strong result, since there is always an $O(n)$ upper bound for one-round computations. They proposed as an open question the problem of finding (for each k) a constructive language with such a strong lower bound. In this work we modify the languages used in Duris, Galil, and Schnitger's construction, and prove an $\Omega(n/k^2 \log n)$ lower bound on their $(k-1)$-round communication complexity.

2 Construction of the Languages

Papadimitriou and Sipser described an encoding of directed graphs of out-degree one, containing 2^m vertices (numbered 0 through $2^m - 1$), for any m, into strings of $m2^m$ bits. A graph can be represented concisely by listing, for each vertex, the number of the vertex reached by its out-edge. The first m bits correspond to vertex 0, the next m bits to vertex 1, and so forth. The strings in our constructive language are similar, but the m-bit name

of each next vertex is hidden in a $14m$-bit field, in such a way that an adversary will know nothing about a vertex's hidden number unless he knows at least $8m$ of the bits.

We use the following method of encoding m bits in a string of $14m$ bits. Call the first $7m$ bits the "mask", and call the remaining bits the "data". The first mask bit corresponds to the first data bit and so on. Some m bits of the mask are set to one; the rest are set to zeroes. The m bits to be encoded are placed in the $7m$ data bits in the positions corresponding to ones in the mask. The other data bits are set arbitrarily. The mask bits can be said to *select* the hidden bits. (Lipton and Sedgewick [4] describe another lower bound on communication complexity involving selection.)

Define L_k as the language of strings representing (in $14m2^m$ bits) those directed graphs with m vertices and out-degree one that have a path of length $k+1$ from vertex 0 to vertex $2^m - 1$. Duris, Galil, and Schnitger used similar languages in their constructive proof, but without the encoding. Without encoding, knowledge of half the bits of a vertex restricts the possible successors to a set of $2^{m/2} < n^{1/2}$ vertices (permitting an upper bound of $I_{k-1}(L_k) = O(k \log n + n^{1/2})$). With encoding, knowledge of even 4/7 of the bits of a vertex can yield no information about the encoded number.

We first prove that L_k can be accepted with a small amount of communication in k rounds. This easy upper bound was implicit in previous work.

Theorem 2.1 *$I_k(L_k) = O(k \log n)$, for all positive k*

Proof. Partition the input so that A receives the first half of the input and B receives the second half of the input. This means that A gets all the input bits associated with vertex 0, and that all the input bits for any vertex are seen by a single processor. The protocol is for A to name, in order, as many vertices on the path from vertex 0 as possible, stopping only when he has named a vertex for which B knows the input. Processor B then continues. The processors alternate until one processor can check if the k^{th} vertex points to vertex $2^m - 1$. A total of k vertices are announced, so at most k rounds are used, with total communication of $km = O(k \log n)$ bits. □

3 Lower Bound on Communication Complexity

Language L_k seems to be well-tuned for acceptance in k rounds of computation. Our intuition is that if there were a bad split of vertices between the two processors, a similar protocol using only $k-1$ rounds would be one round short at the end. At some point an enumeration of all possible next vertices would be necessary, in an attempt to skip an alternation of vertex ownership along the path from source to sink. However, any attempt to prove a lower bound is complicated by the fact that a protocol can split the bits of a vertex between the two processors.

The lower-bound proof is based on showing that under any partition, each processor has many vertices of which it has at most 4/7 of the input bits. We will show that a processor has no real knowledge of a vertex if it has that few of its bits. This eliminates the problem of split vertices; each vertex can be treated as being seen entirely by one processor or the other. We then restrict our attention to inputs representing graphs with an unfavorable alternation of vertex ownership. We assume that they can be correctly accepted or rejected with a certain amount of communication in $k-1$ rounds, and find a contradiction. This establishes a lower bound on $I_{k-1}(L_k)$. This proof follows closely the proof in [3], although the use of encoded vertices permits some simplification, while yielding a stronger bound.

Theorem 3.1 $I_{k-1}(L_k) = \Omega(n/k^2 \log n)$, *for all* $k > 2$

Proof. We show, for sufficiently large n, that $\lfloor n/224k^2 \log n \rfloor$ bits of communication in each of $k-1$ rounds do not suffice to determine membership in L_k. The first lemma relates to the method of hiding the m bits of a vertex's pointer, its out-edge, in a $14m$-bit field. No matter how the bits of a vertex are partitioned between the two processors, a processor receiving even 4/7 of the vertex may know nothing about the encoded value.

Lemma 3.2 *Any* $8m$ *bits of a* $14m$*-bit field can be fixed in such a way that any* m*-bit number can be encoded with some setting of the remaining bits.*

Proof. The $7m$ data/mask pairs (*positions*) fall into four classes, depending on the bits that must be fixed:

A. Mask and data bits both fixed.

B. Mask bit fixed, data bit free.

C. Mask bit free, data bit fixed.

D. Mask and data bits both free.

Let a, b, c, and d represent the number of positions in each class. Since there are $7m$ positions $a + b + c + d$ must equal $7m$.

The following procedure for setting the fixed bits permits any m-bit number to be encoding by some setting of the free bits.

Suppose $b+d \geq m$. Fix all mask bits to zero, except m of the B and D mask bits, which are fixed to ones. The m (free) data bits selected by the ones of the mask can represent any m-bit number.

Otherwise, $a + c > 6m$. There are at most $4m$ type A positions, so $c > 2m$. Fix the data bits of the first $2m$ C positions with alternating zeroes and ones, and set all the mask bits in the A, B, and D positions to zero. Now each pair of consecutive C positions may be used to encode either a one or zero by setting the appropriate (free) mask bit. So any m-bit number can be represented. □

Now define the following constants:

$a = \lfloor n/224k^2 \log n \rfloor$

$s = \lceil 2^m/32k \rceil$

$b = 2s$

These constants ensure that the following lemma holds.

Lemma 3.3 *For positive m, $s > ka$.*

Proof. Because $2n = 14m2^m$, it follows that $m < \log n$ and $2^m > n/7 \log n$ for positive m. Therefore, $s/k \geq 2^m/32k^2 > n/224k^2 \log n \geq a$, and the lemma holds. □

If, under a particular partition, a processor receives at most 4/7 of the bits of a vertex, say that the vertex is *hard* for that processor. It is easy to show that disjoint sets of at least 1/8 of the vertices are hard for each processor.

Identify $k+1$ disjoint sets (*levels*) of vertices:

$B_0 = \{0\}$

B_1 = a set consisting of some single vertex hard for A.

$B_i (1 < i \leq k, i \text{ even})$ = a set of some b vertices hard for B.

$B_i (1 < i \leq k, i \text{ odd})$ = a set of some b vertices hard for A.

This requires finding, for each processor, at most $bk/2$ hard vertices among vertices 1 through $2^m - 1$ or, alternatively, $(bk/2) + 1$ hard vertices among the whole set. So if $(bk/2) + 1 < 2^m/8$, the B_i can be found. This holds if $2^m > 32k$.

Now consider a restriction of the domain $\{0,1\}^{2n}$ for which the protocol must function correctly. We want to show that if the amount of communication permitted is too low, then under any protocol there will be two strings in the restricted set, one in L_{k+1} and one not, that result in the same communication. Therefore one of them will be incorrectly accepted or rejected. This gives a lower bound on the communication complexity of L_k that applies both to the restricted domain and to $\{0,1\}^{2n}$.

Let the restricted set of strings, S, represent graphs in which each vertex points to a vertex in the next level. Vertices in level k point either to vertex 0 or vertex $2^m - 1$. This implies that if there is a path from vertex 0 to vertex $2^m - 1$ of length $k+1$, the i^{th} vertex is in B_i. More formally, let S consist of all strings in $\{0,1\}$ such that:

1. At any particular vertex v, there is a fixed way of encoding a pointer to any other particular vertex w. This is equivalent to saying that no two strings in S represent the same graph.

2. Vertex 0 points to the vertex in B_1.

3. Each vertex in B_i, i odd, has all bits received by A fixed in such a way that the vertex may still point to any other vertex. This means that A has no information about those vertices. Similarly, each vertex in B_i, i even, has bits fixed so that B has no knowledge of the encoded pointer.

4. Each vertex in B_i $(i \leq k-1)$ points to some vertex in B_{i+1}.

5. Each vertex in B_k points to either vertex 0 or vertex $2^m - 1$.

6. Each vertex not in some B_i is fixed to some arbitrary value.

Now consider equivalence classes of input strings with respect to the vertices of a particular level. Two strings are equivalent with respect to a level of vertices if, for each vertex in the level, the pointers are the same in both strings. We will let P_i be the set of equivalence classes with respect to level i. It is evident that $|P_1| = b$, $|P_i| = b^b$ (for $1 < i \leq k-1$) and $|P_k| = 2^b$.

There are also equivalence classes of input strings with respect to all of the vertices on even levels. Call the set of such equivalence classes S_0. Likewise, call the set of equivalence classes with respect to all of the vertices on odd levels S_1. Note that $|S_0| = |P_2||P_4| \cdots |P_{\ell(0)}|$ and $|S_1| = |P_1||P_3| \cdots |P_{\ell(1)}|$, where $\ell(i)$ is the greatest integer $\leq k$ such that $\ell(i) \equiv i \bmod 2$. Set S_0 corresponds to the set of possible inputs to A and S_1 corresponds to the set of possible inputs to B. Use the notation $V \times W$, where $V \subseteq S_0$ and $W \subseteq S_1$, to refer to all strings formed by A receiving an input from V and B receiving an input from W. More formally, $V \times W \equiv (\bigcup_{s \in V} s) \cap (\bigcup_{t \in W} t)$. Note that $S_0 \times S_1 = S$. Define ρ_i to be the number of inputs (to processor ($i \bmod 2$)) in $S_{i \bmod 2}$ that have the bits of levels below i fixed to some particular values. That is, for $i \leq k$,

$$\rho_i = \prod_{\substack{j=i \\ (j \equiv i \bmod 2)}}^{k} |P_j|.$$

The value of ρ_{k+1} is 1, because there is only one possible input when all bits are fixed.

We now prove Duris, Galil, and Schnitger's main lemma. A communication by a processor always permits an observer (such as the other processor) to refine its knowledge of the set of inputs that A could have received. The lemma states that when only a bits of communication are permitted in each round, the processors can not describe a refinement of their inputs sets well: for any i there is an i-round communication consistent with s large sets of inputs that have a common path through the first i levels of their graphs and then continue on to distinct vertices in level $i+1$. Figure 1 illustrates the situation that the lemma proves must exist after i rounds of communication.

Lemma 3.4 (Duris, Galil, and Schnitger [3]) *For any i from 1 to $k-1$, there is a communication C_i resulting from i rounds of computation, such that:*

1. *Each round of communication is limited to a bits.*

2. *There is a set $V_i \subseteq S_{(i+1) \bmod 2}$ of inputs to the processor receiving the bits of level $i+1$. Call this processor, which made the i^{th} round of communication,* LAST.

3. *There is a set $W_i \subseteq S_{i \bmod 2}$ of inputs to the other processor. This processor will make the next communication, so call it* NEXT.

4. *The inputs in $V_i \times W_i$ are consistent with the partial communication C_i and are identical with respect to all vertices in levels 0 through i of their graphs, except some one vertex g_i on level i. Each input represents a graph with an identical path from vertex 0 through the successive levels to vertex g_i.*

5. *There are s subsets $W_i^1, \ldots, W_i^s$ of W_i each of which corresponds to a different assignment to the bits of vertex g_i. For each j, the inputs in $V_i \times W_i^j$ have an edge from g_i to a distinct vertex g_{i+1}^j of level $i+1$.*

6. *$|V_i| \geq \rho_{i+1}/2^{ai}$. This means that the set of remaining inputs for processor* LAST *remains large, even after the input bits in levels 0 through $i-1$ are fixed by the communication. At least $1/2^{ai}$ of its inputs (with those levels fixed) are consistent with the communication.*

7. *For each j, $|W_i^j| \geq \rho_{i+2}/2^{ai}$. This means that the set of inputs to processor* NEXT *is large, even if the input bits of vertex g_i are fixed in any one of s different ways. At least $1/2^{ai}$ of its inputs (with levels 1 through i fixed) are consistent with the communication.*

Proof.

BASE CASE ($i = 1$): Choose the level-one vertex as g_1. The first communication is made by A; the choice of C_1 implies a partition of the set of its possible inputs. Because the communication is limited to a bits, for some C_1 the set V_1 must contain at least $|S_0|/2^a = \rho_2/2^a$ inputs. This is large enough to satisfy the lemma. Any assignment to the bits of the level-one vertex is possible; let $W_1^1, \ldots, W_1^s$ be sets of inputs each of which gives g_1 an edge to a distinct vertex g_1^j of level 2. Any assignment to the other bits that B receives (vertices in levels 3, 5, ...) remains possible, so each W_1^j has size ρ_3, which is sufficiently large.

INDUCTION STEP: Assume that the lemma holds for $i < k-1$ and show it holds for $i+1$.

We will show that there is some round of communication that processor NEXT can add to the existing communication C_i, while maintaining the conditions of the lemma. Vertex

g_{i+1} will be selected from one of the candidates g^j_{i+1} already specified. Set V_{i+1} will be a refinement of W^j_i, for the same j. Set W_{i+1} will be a refinement of V_i.

The first problem is to determine subsets of V_i that would be suitable as sets W^j_{i+1}. Consider the division of V_i into equivalence classes U_j with respect to the vertices of level $i+1$. Call U_j large if $|U_j| \geq \rho_{i+3}/(2 \cdot 2^{ai})$. Let $q = \lceil |P_{i+1}|/(2 \cdot 2^{ai}) \rceil$. There are at least q large U_j's. If not, there would be $q' < q$ large U_j's and

$$|V_i| \leq q'\rho_{i+3} + \frac{(|P_{i+1}| - q')\rho_{i+3}}{2 \cdot 2^{ai}} < \frac{\rho_{i+1}}{2^{ai}} \leq |V_i|,$$

a contradiction. So let be $U_1, \ldots, U_q$ be some q of the large sets.

Now, for some l from 1 to s there is a set of inputs W^l_i such that s different edges from g^l_{i+1} to level $i+2$ occur in the graphs corresponding to inputs $W^l_i \times \bigcup_{j=1}^{q} U_j$. If not, then for every l, the number of such edges would be less than s. So the number of possible bit assignments to the $(i+1)^{st}$ level, q'', would be less than $s^s b^{b-s}$. But

$$s^s b^{b-s} = \frac{b^b}{(b/s)^s} = \frac{b^b}{2^s} < \frac{b^b}{2^{ka}} < \frac{|P_{i+1}|}{2 \cdot 2^{ai}} \leq q \leq q'',$$

which is a contradiction. (Note the use of the $(s > ka)$ lemma here.)

Set V_{i+1} will be chosen as a restriction of W^l_i for some l for which this holds. The corresponding vertex g^l_{i+1} becomes vertex g_{i+1}. Let the sets W^j_{i+1} be any s large U_j sets that yield edges to different vertices g^j_{i+2} in level i+2, and let $W_{i+1} = \bigcup_{j=1}^{s} W^j_{i+1}$. Each of these sets has size $\rho_{i+3}/(2 \cdot 2^{ai})$, which is large enough.

All that remains is to show how W^l_i is partitioned by the $(i+1)^{st}$ communication. Since at most a bits are transmitted, for some communication the corresponding partition V_{i+1} of W^l_i is sufficiently large, that is,

$$|V_{i+1}| \geq |W^l_i|/2^a \geq \rho_{i+2}/2^{a(i+1)}.$$

This completes the proof of Lemma 3.4. □

To finish the proof of the theorem, assume that a bits of communication in each of $k-1$ rounds suffice to accept L_k and apply the lemma with $i = k-1$.

Consider the set $V_{k-1} \times W_{k-1}$ of inputs consistent with communication C_{k-1}. There are at least s vertices $(g^1_k, \ldots, g^s_k)$ in level k to which there are paths consistent with

the communication. The processor that must accept or reject did not receive the bits of level k as input, so it knows nothing about those vertices except what it learned from the communication. If it correctly accepts each input that passes through g_k^j, it must be because the vertex points to the same next vertex (0 or $2^m - 1$) in *every* input. Input set V_{k-1}, already restricted to fixed assignment to all vertices except those of level k, therefore has s of its level-k vertices fixed. This leaves $b - s$ vertices that can each take one of two assignments. Therefore $|V_{k-1}| \leq 2^{b-s} = |P_k|/2^s < |P_k|/2^{ak} < |P_k|/2^{a(k-1)} \leq |V_{k-1}|$, a contradiction. (Note again the use of the $s > ka$ lemma.) So a bits of communication in each of $k - 1$ rounds do not suffice to recognize L_k. This proves the lower bound. □

References

[1] A. Aho, J. Ullman, and M. Yannakakis. On notions of information transfer in VLSI circuits. In *Proceedings of the 15th ACM Symposium on Theory of Computing*, pages 133–139, Boston, 1983.

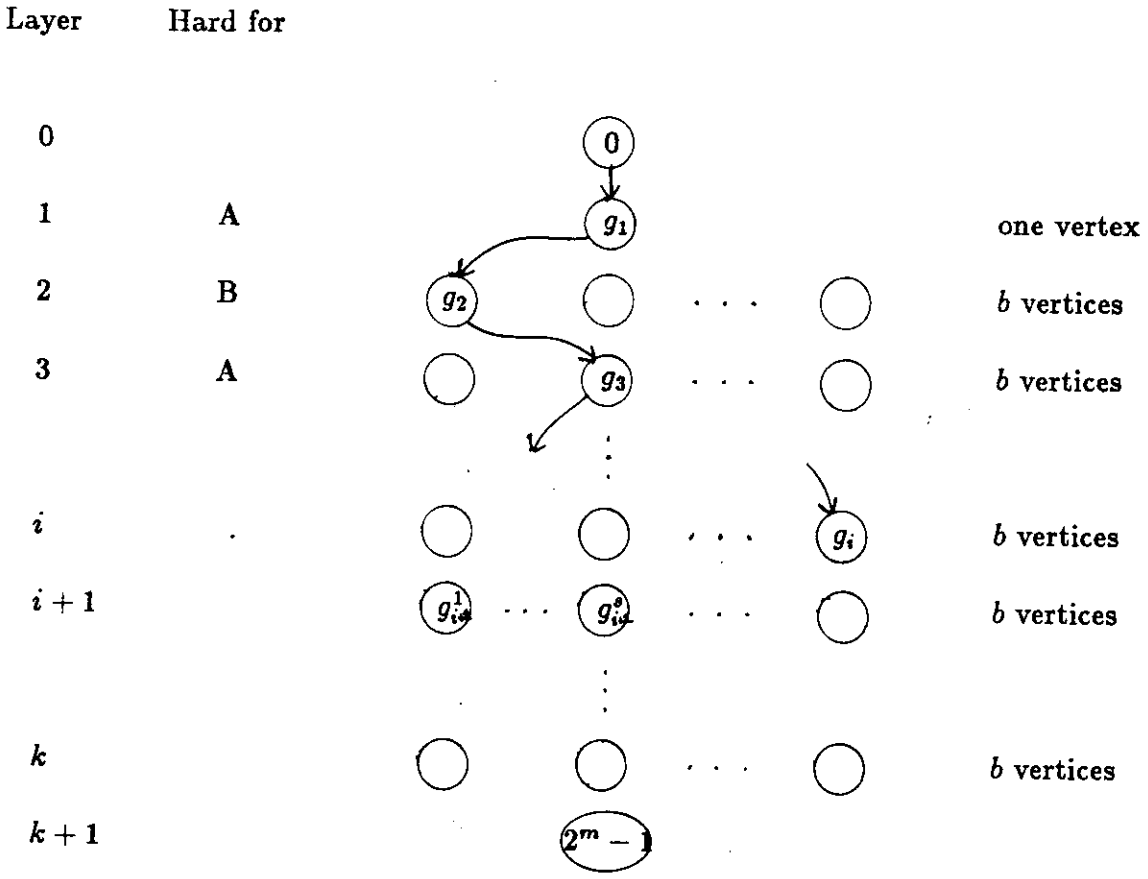
[2] A. Chandra, M. Furst, and R. Lipton. Multi-party protocols. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, pages 94–99, Boston, 1983.

[3] P. Duris, Z. Galil, and G. Schnitger. Lower bounds on communication complexity. In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing*, pages 81–91, Washington, 1984.

[4] R. Lipton and R. Sedgewick. Lower bounds for VLSI. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing*, pages 300–307, 1981.

[5] C. Papadimitriou and M. Sipser. Communication complexity. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing*, pages 196–200, 1982.

[6] A. Yao. Some complexity questions related to distributive computing. In *Proceedings of the 11th Annual ACM Symposium on Theory of Computing*, pages 209–213, 1979.



All inputs in $V_i \times W_i$ correspond to path $0, g_1, \ldots, g_i$.
All inputs in $V_i \times W_i^j$ have an extension of that path to g_{i+1}^j.

Figure 1: Inputs consistent with communication C_i