Embeddability of Weighted Graphs in k-Space is Strongly NP-Hard

(Extended Summary)[1]

JAMES B. SAXE
Computer Science Department
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

Abstract--In this paper we investigate the complexity of embedding edge-weighted graphs into Euclidean spaces: Given an (incomplete) edge-weighted graph, G, can the vertices of G be mapped to points in Euclidean k-space in such a way that any two vertices connected by an edge are mapped to points whose distance is equal to the weight of the edge? We prove that the preceding problem is NP-Hard (by reduction from 3-Satisfiability), even when k=1 and the edge weights are restricted to take on the values 1 and 2. Related results are shown for the problem of testing the uniqueness of a known embedding and for variations involving inexact edge weights.

1. Introduction

In many applications of distributed sensor networks[2] there arises the problem of determining the locations of sensors from incomplete (and possibly errorful) information about their distances from each other and from fixed landmarks. This prompts us to ask the following geometric questions:

-Given an incompletely specified distance matrix for a set of points in k-space,[3] when is the complete distance matrix uniquely determined?

-Assuming the distance matrix to be uniquely determined, what is the computational complexity of actually finding the unspecified distances?

In this paper we consider the closely related problem of embeddability:

-Given a (purported) incompletely specified distance matrix for a set of points in k-space, determine whether there can actually exist a set of points satisfying that matrix.

In Section 2 we introduce definitions that will allow us to phrase several forms of the embeddability problem in terms of edge-weighted graphs. In Section 3, we give a simple proof that a 1-dimensional version of the embeddability problem is NP-Complete. In Section 4, we show the more difficult and surprising result that this same 1-dimensional problem is strongly NP-Complete in the sense of Garey and Johnson [1979] and extend

---

[1] In order to meet the space constraints of these *Proceedings*, a number of details have been omitted from this paper, particularly in the proofs of the results presented in Sections 5 and 6. The complete paper will be available as a Carnegie-Mellon University Computer Science Department report of this same title. This research was supported in part by the Office of Naval Research under Contract N00014-76-C-0370.

[2] See, for example, *Distributed Sensor Nets* [1978].

[3] For practical purposes the most interesting cases are k=2 and k=3.

this result to higher dimensions. In Section 5 we address some naturally arising questions concerning the suitability of the Turing Machine model for a problem that inherently involves real numbers, and show that the proofs used in Section 4 have relevance to an "approximate embeddability" problem on the reals. In Section 6 we discuss versions of the problem in which one way to complete an incompletely specified distance matrix is known and it is desired to determine whether a second solution exists. We show that these versions are no easier than corresponding versions studied earlier in the paper. Finally, the contributions of the paper are summarized in Section 7.

2. Fundamental Concepts

We begin by introducing the concepts of weighted graph and embedding:

Definitions:

A weighted graph, $G = \langle V,E,W\rangle$, is an ordered triple such that each element of E is an unordered pair of distinct elements of V and W is a function mapping E into $[0,\infty)$. The elements of V are called the vertices of G. The elements of E are called the edges of G. For each edge, e, of G, the real number W(e) is called the weight of e in G (or simply the weight of e).

Definitions:

Let $G = \langle V,E,W\rangle$ be a weighted graph, and let k be a positive integer. Then an embedding of G in k-space is a function, f, mapping V into the k-dimensional Euclidean space, $\mathbb{R}^k$, such that, for each edge, $e = \{v,w\}$, of G, $|f(v)-f(w)| = W(e)$. G is said to be embeddable in k-space, or k-embeddable, iff there exists an embedding of G in k-space.

For any positive integer, k, the problem of k-embeddability may now be stated as follows:

Problem (k-Embeddability):

Given an arbitrary weighted graph, G, determine whether G is k-embeddable.

In Sections 3 and 4 we will wish to restrict the class of weighted graphs under consideration, so that the notion of NP-Completeness (which is defined in terms of Turing machines) will make sense in relation to Embeddability. We therefore introduce the following definition.

Definition:

Let S be any subset of $[0,\infty)$. Then, an S-weighted graph is a weighted graph, G, such that the weight of each edge of G is an element of S. We will generally refer to $\mathbb{Z}^+$-weighted graphs as integer-weighted graphs.

In Section 5 we will return to the question of graphs with real edge weights.

3. The Weak NP-Completeness of 1-Embeddability

In this section, we demonstrate the weak NP-Completeness of the problem of 1-Embeddability of integer-weighted graphs. To do this, we first show constructively that 1-Embeddability is in NP. We then use a reduction from Partition[4] to show completeness.

Theorem 3.1:

1-Embeddability of integer-weighted graphs is in NP.

Proof:

To check the 1-embeddability of any integer-weighted graph, a NDTM need only

1. Partition the graph into disjoint connected subgraphs,
2. Guess the direction of each edge of the graph, and
3. Check the consistency of each disjoint connected subgraph.

These operations can clearly be carried out in (nondederministic) polynomial time. □

Theorem 3.2:

1-Embeddability of integer-weighted graphs is NP-Complete.

Proof:[5]

We will show the NP-Completeness of 1-Embeddability by reduction from Partition. Let $S = \{a_1, a_2, \ldots, a_n\}$ be a multiset of positive integers. In polynomial time we may construct from S a description of the cyclic graph $G = \langle V,E,W \rangle$ whose edge weights are the a_i, that is

$$V = \{v_0, \ldots, v_{n-1}\},$$
$$E = \{\{v_i, v_{(i+1 \bmod n)}\} \mid 0 \le i < n\}, \text{ and}$$
$$W = \{(\{v_i, v_{(i+1 \bmod n)}\}, a_i) \mid 0 \le i < n\}.$$

If f is an embedding of G in the line, then the multisets

$$S_1 = \{a_i \mid f(v_i) < f(v_{(i+1 \bmod n)})\} \text{ and}$$
$$S_2 = \{a_i \mid f(v_i) > f(v_{(i+1 \bmod n)})\}$$

constitute a partition of S into two pieces whose sums are equal. Similarly, any such partition of S yields a 1-Embedding of G. □

[4] The Partition problem calls for partitioning a (multi-)set of integers into two subsets with equal sums, and is known to be NP-Complete; see Garey and Johnson [1979].

[5] The construction used in this theorem and that used in the proof of Lemma 4.4 were independenty developed by Yemini [1978], who used them to show the (weak) NP-Completeness of 2-Embeddability of integer-weighted graphs.

:duction from Partition[4] to show

NP.

hted graph, a NDTM need only

:l subgraphs,
raph, and
nnected subgraph.

ondederministic) polynomial time.

Complete.

·ility by reduction from Partition.
integers. In polynomial time we
graph G = <V,E,W> whose edge

:ts

sums are equal. Similarly, any

o two subsets with equal sums, and is

emma 4.4 were independenty developed
of 2-Embeddability of integer-weighted

4. The Strong NP-Completeness of 1-Embeddability

We now come to our key theorem, which asserts that the problem of determining whether an integer-weighted graph is embeddable in the line remains NP-Complete even if the edge weights are restricted to be no greater than four.

Theorem 4.1
1-Embeddability of $\{1,2,3,4\}$-weighted graphs is NP-Complete.

Proof:[6]
Our proof consists of a reduction from 3-Satisfiability (which was shown to be NP-Complete by Cook [1971]) to 1-Embeddability of $\{1,2,3,4\}$-weighted graphs. Let E be any Boolean expression in conjunctive normal form with three literals in each clause. Our goal will be to construct a $\{1,2,3,4\}$-weighted graph, G, which is embeddable iff E is satisfiable. We let n be the number of variables occurring in E and m be the number of clauses in E. Throughout this proof, we will use the convention that the variables of E will be indexed by "i" (which will therefore range from 1 through n), the clauses of E will be indexed by "j" (ranging from 1 through m), and the literals within each clause will be indexed by "k" (ranging from 1 through 3). Thus E has the form

$$E = \prod_{1\le j\le m} C_j,$$

where each clause, C_j, has the form

$$C_j = \sum_{1\le k\le 3} L_{j,k},$$

and each literal, $L_{j,k}$, has the form

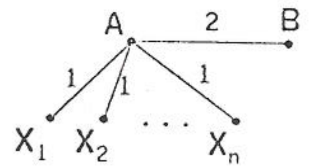
$$L_{j,k} = X_i \quad \text{or} \quad L_{j,k} = \bar{X}_i$$

for some i, $1\le i\le n$. We will also use throughout the proof the convention that "f" represents a hypothetical 1-embedding of G (or of the part of G we have constructed so far).

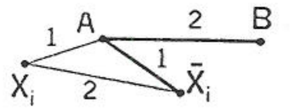
To construct G, we will use the "building blocks" shown in Figure 4.1. We begin with the subgraph shown in Figure 4.1(a). We assume without loss of generality that $f(A) = 0$ and $f(B) = 2$. This assumption constrains f to assign each of the X_i (which we identify with the variables of E) to 1 or -1 (which we identify with the Boolean values TRUE and FALSE, respectively). Note that each possible mapping of the X_i into $\{1,-1\}$ corresponds to some assignment of truth values to the X_i. In the remaining steps of the construction, we will add edges which have precisely the effect of constraining f to map the X_i to $\{1,-1\}$ in such a way that the corresponding assignment of the X_i satisfies E.

[6]Another NP-Complete problem involving a form of graph embedding is the Bandwidth Minimization Problem (see Papadimitriou [1976]). In the full paper we will also exhibit a reduction from Bandwidth Minimization to Embeddability. This reduction will suffice to show the strong NP-Completeness of 1-Embeddability of integer-weighted Graphs, although it is somewhat less economical than the construction given here.

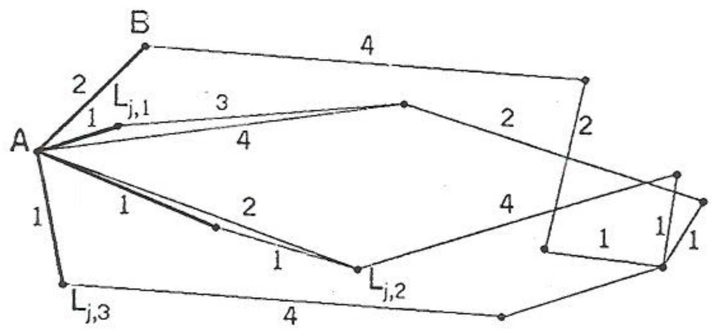
The next step in our construction is to augment G by adding the edges shown in Figure 4.1(b) for each i, $1 \leq i \leq n$. The heavy lines in that figure represent already-existing edges. We now have vertices $\bar{X}_i$ such that for each variable, X_i, f maps X_i to 1 (TRUE) iff it maps $\bar{X}_i$ to -1 (FALSE), and vice-versa. The possible mappings from $\{X_i\} \cup \{\bar{X}_i\}$ to $\{1,-1\}$ under f now correspond precisely to the possible (consistent) truth assignments of the X_i and $\bar{X}_i$, but still without regard to whether those assignments satisfy E.



(a) Implementation of variables.



(b) Implementation of a negative literal.



(c) Implementation of a disjunctive clause.

Figure 4.1 Building blocks for transforming an expression in 3-DNF to a graph.

For the final step of our construction, we add the edges indicated in Figure 4.1(c) for each j, $1 \leq j \leq m$. The vertices $L_{j,k}$ are identified with the X_i and $\bar{X}_i$ precisely as the corresponding literals, $L_{j,k}$, are formally identical with the X_i and $\bar{X}_i$. Once again, the heavy lines indicate edges which were present at earlier stages of the construction. Careful study of the graph in Figure 4.1(c) will reveal that it is impossible to embed it in the line in such a way that A is sent to 0, B is sent to 2, and all three of the $L_{j,k}$ are sent to -1 (FALSE), but if one or more of the $L_{j,k}$ are to be sent to 1 (TRUE), then an embedding is possible (in fact, exactly one such embedding is possible). Thus, for each j, $1 \leq j \leq m$, the effect of the edges in Figure 4.1(c) is precisely to constrain f to map the X_i to $\{1,-1\}$ in such a way that the corresponding truth assignment for the X_i satisfies clause C_j.

The effect of all the edges of G is therefore to constrain f to map the X_i to $\{1,-1\}$ in such a way that the corresponding assignment of truth values to the X_i satisfies E. If there is no such assignment then G is not 1-embeddable. If there are any assignments satisfying E, then for each such assignment G can be (uniquely)

1-embedded by a function sending A to 0 and B to 2 and mapping the X_i to $\{1,-1\}$ in accordance with that assignment. Finally, it is clear that the preceding construction can be carried out in polynomial time. This completes the proof. □

For future reference, we note that the construction used in the preceding proof is such that the 1-embeddings of G are in one-to-one correspondence (up to translation and reflection) with the truth assignments that satisfy E. We note also that the preceding theorem immediately yields the following result:

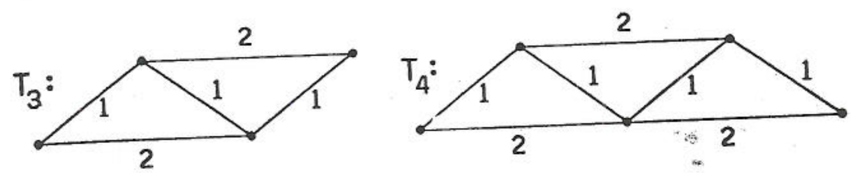
<u>Corollary 4.2</u>:
1-Embeddability of integer-weighted graphs is strongly NP-Complete.

<u>Proof</u>:
It suffices to note that translation of a sequence of numbers in $\{1,2,3,4\}$ from binary to unary can be accomplished in linear time and causes only a constant factor increase in the length of the input. □

We may also immediately derive:

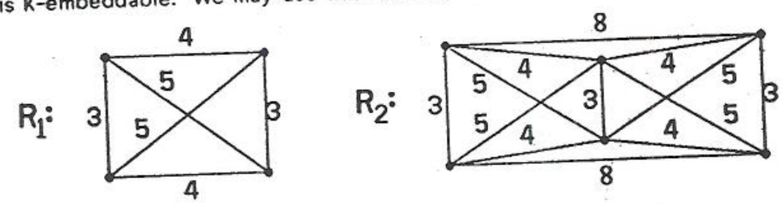
<u>Corollary 4.3</u>:
1-Embeddability of $\{1,2\}$-weighted graphs is NP-Complete.

<u>Proof</u>:
Consider the graphs shown in Figure 4.2. By replacing single edges of weights 3 and 4 with configurations T_3 and T_4, respectively, we can reduce any $\{1,2,3,4\}$-weighted graph, G, to a $\{1,2\}$-weighted graph, H, that is 1-embeddable iff G is 1-embeddable. □



<u>Figure 4.2.</u> Building long "edges" from short edges.

In fact, for any positive integer, k, the graph H so constructed will be k-embeddable iff G is k-embeddable. We may use this fact to prove our next lemma.



<u>Figure 4.3.</u> Gadgets for adding a dimension.

<u>Lemma 4.4</u>:
For every positive integer, k, k-Embeddability of $\{1,2\}$-weighted graphs is NP-Hard.

Proof:

Consider the graphs shown in Figure 4.3. Given any {1,2}-weighted graph, G, each edge of G having weight 1 may be replaced by the R_1 and each edge of weight 2 by R_2, yeilding a graph, H, which, for any positive integer, k, is embeddable in (k+1)-space iff G is embeddable in k-space. By the methods of Theorem 4.3, H may be transformed into a {1,2}-weighted graph, J, that is embeddable in precisely those spaces in which H is embeddable. The transformation from G to J involves only a constant factor increase in the length of a specification of the graph and can clearly be accomplished in polynomial time. It follows by mathematical induction that, for any positive integer, k, 1-Embeddability is polynomial-time reducible to k-Embeddability for {1,2}-weighted graphs. □

Once again, we note that the (k+1)-embeddings of J will be in one-to-one correspondence (up to rotation and reflection) with the k-embeddings of G. Finally, Theorem 4.4 gives us the following result.

Corollary 4.5:

Let k be any positive integer. Then k-Embeddability of integer-weighted graphs is strongly NP-Hard.

Proof:

This result follows from Lemma 4.4 and the same reasoning used in the proof of Corollary 4.2. □

5. Graphs with Real-Valued Edge Weights

We will now discuss the applicability of NP-Completeness to problems whose inputs are real numbers in general, and to embedding problems in particular. A number of reasons for doubting the relevance of the Turing Machine model seem naturally to present themselves.

- NP-Completeness is defined for language recognition problems on Turing Machines, which inherently can deal only with integers and not with arbitrary reals.

- Given a "random" embedding of an unweighted graph into a Euclidean space, any two of the edge weights induced by the embedding will be incommensurable with probability 1. Moreover, if the graph is overconstrained and the dimension of the space is at least two, then rounding the induced edge-weights to multiples of some small distance will almost always produce a weighted graph that is not embeddable in the space.

In order to deal with these issues, we introduce the notion of approximate embeddings.

Definitions.

Let G be a weighted graph and ϵ be a positive real number. Then an ϵ-approximate k-embedding of G is a function, f, that maps the vertices of G into Euclidean k-space such that for every edge, $\{u,v\}$, of G, $1-\epsilon < |f(u)-f(v)|/W(\{u,v\}) < 1+\epsilon$. If such an embedding exists, then G is said to be <u>ϵ-approximately</u> <u>k-embeddable</u>.

Given a positive integer, k, and two reals, ϵ_1 and ϵ_2, such that $0 < \epsilon_1 < \epsilon_2$, we may now define the following more "robust" embeddability problem:

<u>Problem</u> (ϵ_1,ϵ_2-Approximate k-Embeddability):

Given a weighted graph, G, assert correctly either (1) that G <u>is</u> ϵ_2-approximately k-embeddable (this is called <u>accepting</u> G) or (2) that G <u>is</u> <u>not</u> ϵ_1-approximately k-embeddable (this is called <u>rejecting</u> G).

Note that if the least ϵ for which G is ϵ-approximately k-Embeddable lies in the interval $[\epsilon_1,\epsilon_2]$, then it is permissible either to accept or to reject G. In this problem definition, we have attempted to capture, without introducing inordinately many complexities of detail, the essential problem of embedding as it would apply to real computers given inexact data.

We now wish to investigate the computational complexity of ϵ_1,ϵ_2-Approximate Embeddability problems. Is it possible, for example, to solve all such problems in time polynomial in the size of a specification of G (where the degree of the polynomial, or even just the "constant" factor, depends on $(\epsilon_2-\epsilon_1)^{-1}$)?

It turns out that such polynomial solutions are not possible in the general case (assuming that $P \neq NP$). In particular, we have the following result.

<u>Theorem</u> <u>5.1</u>:

1/18,1/9-Approximate 1-Embeddability of integer-weighted graphs is NP-Complete.

<u>Sketch</u> <u>of</u> <u>Proof</u>:

We note that the embeddability properties of the graphs used in the proof of Theorem 4.1 depend only on cycles of length no greater than 16 having edges whose lengths are multiples of 1. It follows from this that for such graphs ϵ-Approximate 1-Embeddability is equivalent to ordinary 1-embedability for any $\epsilon <$ 1/8, and the desired result is at hand. □

It is interesting to note that Approximate 1-Embeddability problems restricted to graphs consisting of a single cycle (such as were used in the proof of Theorem 3.2) are always solvable in polynomial time.[7] This shows that the weak NP-Completeness result given in Section 3 does not say all there was to say about the difficulty of the practical (*i.e.*, with inexact data, etc.) form of the problem. Loosely speaking, we could say that we have shown the notion of strong *vs.* weak NP-Completeness to be significant even for problems that naturally involve reals rather than integers. It should be noted, however, that Theorem 5.1 followed not from Theorem 4.4 but rather from the particular construction used in the proof of Theorem 4.4.

---

[7]This follows from the existence of fast approximation algorithms for Partition. See, for example, Lawler [1977].

6. Ambiguous Embedding Problems

Another variation on the embeddability problem that may arise in practical applications is that of "ambiguity of solution." Given an incomplete weighted graph and some embedding of that graph into a Euclidean space, we may wish to know whether the given into a Euclidean space, we may wish to know whether the given embedding is unique.[8] To pose the problem more precisely, we introduce the following definitions.

<u>Definitions</u>:

Let G be a weighted graph and k be a positive integer. Then two k-embeddings, f and g, of G are said to be <u>congruent</u> iff for each two vertices, u and v, of G, $|f(u)-f(v)| = |g(u)-g(v)|$. A k-embedding, f, of G is said to be <u>unique</u> (up to congruence) iff every k-embedding of G is congruent to f, and in this case G is said to be <u>uniquely</u> <u>k-embeddable</u>. If G has two or more non-congruent k-embeddings, then G is <u>ambiguously</u> k-embeddable.

For any positive integer, k, we may now define the problem of Ambiguous k-Embedding as follows:

<u>Problem</u> (Ambiguous k-Embedding):

Given a weighted graph, G, and a k-embedding, f, of G, determine whether G is ambiguously k-embeddable (*i.e.*, whether there exists a k-embedding of G which is not congruent to f).

In the full paper, we will show that Ambiguous k-Embeddability is (strongly) NP-Hard for all positive k and that it is NP-Complete for k=1. Here we have space only for a brief overview of the steps of the proof, which go as follows:

1. We define Ambiguous 3-Satisfiability and Ambiguous 4-Satisfiability in a manner analogous to the definition of Ambiguous k-Embedding.
2. We show the NP-Completeness of Ambiguous 4-Satisfiability by reduction from 3-Satisfiability.
3. We show the NP-Completeness of Ambiguous 3-Satisfiability by reduction from Ambiguous 4-Satisfiability.
4. We show the (strong) NP-Completeness of Ambiguous 1-Embedding by reductiion for Ambiguous 3-Satisfiability, using the fact that the construction used in the proof of Theorem 4.1 "preserves uniqueness" (up to congruence).
5. We show the (strong) NP-Hardness of Ambiguous k-Embedding for k>1 by reduction from Ambiguous (k-1)-Embedding.

7. Conclusions

The results of this paper fall into two classes, those of interest to persons concerned with embedding problems (such as the sensor positioning problem) and those that are of more general theoretical interest. To those concerned with finding efficient solutions to the Embedding problem (given a weighted graph, find "the" embedding), these results

---

[8]For example, are the nodes of our sensor network really where we think they are, or might they be in some very different configuration?

say what all NP-Completeness results say: "You are trying to solve the wrong problem." Rather than looking for an efficient worst-case algorithm, it would be more promising to seek an algorithm that gives good performance in cases which arise in practice (for example, cases in which the graph is highly overconstrained). Pursuing this topic, in the full paper we will discuss a linear-time algorithm due to Shamos [1978] for determining whether a given *complete* graph is k-embeddable (for any fixed k). His algorithm assumes a model in which real arithmetic operations can be performed in constant time.

The most specific result of theoretical interest is our discovery of some new *strongly* NP-Hard geometric problems, and our use of some interesting gadgets to carry out the proofs of NP-Hardness. Of more general interest are the two new classes of problems introduced in Sections 5 and 6. The "ϵ_1,ϵ_2-approximate" problems introduced in Section 5 offer a new way of looking at the notion of NP-Completeness in the context of problems involving continuous variables. As we have seen, weak NP-Completeness may not say all there is to say in this context. "Ambiguous solution" problems address the question of determining whether a known solution to a problem is in fact the unique solution. In Section 6, we exhibited a fundamental NP-Complete problem, 3-Satisfiabilty, whose ambiguous version is also NP-Complete, and exhibited a method for obtaining new NP-Completeness results for such ambiguous solution problems, namely the use of reductions that preserve uniquenes of solution.

Acknowledgements

The author would like to thank Steven E. Saunders for leading him to study the problem of Embeddability, Michael I. Shamos for helpful discussions, and Jon L. Bentley for his encouragement and advice during the preparation of this document.

References

[1] Cook, S. A. "The complexity of theorem proving procedures." *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing.* Association for Computing Machinery, New York (1971). pp. 151-158.

[2] *Distributed Sensor Nets.* Proceedings of a conference sponsored by the Information Processing Techniques Office, Defense Advanced Research Projects Agency and hosted by Carnegie-Mellon University, December, 1978.

[3] Garey, M. R. and D. S. Johnson. *Computers, Complexity, and Intractability.* Freeman, San Francisco (1979).

[4] Lawler, E. L. "Fast approximation algortihms for knapsack problems." *Proceedings of the 18th Annual Symposium on Foundations of Computer Science.* IEEE Computer Society, Long Beach, CA (1977). pp. 206-213.

[5] Papadimitriou, C. H. "The NP-Completeness of the bandwidth minimization problem." *Computing* 16 (1976). pp. 263-270.

[6] Shamos, M. I. Personal communication (1978).

[7] Yemini, Y. "The positioning problem--A draft of an intermediate summary." In *Distributed Sensor Nets* [2].

PROCEEDINGS

SEVENTEENTH ANNUAL ALLERTON CONFERENCE

ON COMMUNICATION, CONTROL, AND COMPUTING

J. B. Cruz, Jr.
F. P. Preparata
Conference Co-Chairmen

Conference held

October 10-12, 1979
Allerton House
Monticello, Illinois

Sponsored by

The Department of Electrical Engineering
and
The Coordinated Science Laboratory
of the
UNIVERSITY OF ILLINOIS
at
Urbana-Champaign