SIAM J. CONTROL AND OPTIMIZATION
Vol. 24, No. 4, July 1986

INTRACTABLE PROBLEMS IN CONTROL THEORY*

CHRISTOS H. PAPADIMITRIOU† AND JOHN TSITSIKLIS†

Abstract. This paper is an attempt to understand the apparent intractability of problems in decentralized decision-making, using the concepts and methods of computational complexity. We first establish that the discrete version of an important paradigm for this area, proposed by Witsenhausen, is NP-complete, thus explaining the failures reported in the literature to attack it computationally. In the rest of the paper we show that the computational intractability of the discrete version of a control problem (the team decision problem in our particular example) can imply that there is no satisfactory (continuous) algorithm for the continuous version. To this end, we develop a theory of continuous algorithms and their complexity, and a quite general proof technique, which can prove interesting by themselves.

Key words. complexity, team theory, decentralized control

1. Introduction.

Most classical problems arising in the fields of optimization and control are, in a very real sense, "easy to solve". By this we mean that there are computational procedures with satisfactory performance, which can be used to compute the solution of such problems. Naturally, a lot of effort is being devoted to finding more and more efficient algorithms which exploit any special structure present, but usually there is *no fundamental intractability* to be overcome. For example, in a nonlinear optimal control problem (under some smoothness assumptions) a solution can always be obtained by discretizing the problem with a dense enough grid and then using the discrete dynamic programming algorithm. Roughly speaking, the accuracy ε of the solution so obtained is inversely proportional to the number of points in the grid and such algorithms require time which is a polynomial function of $1/\varepsilon$. The situation is similar in many other classical problems such as nonlinear optimization or numerical integration of partial differential equations. In fact, in some extremely favorable cases (when, for example, the problem can be reduced to the evaluation of some analytic function), the computation time is polynomial in the *logarithm* of $1/\varepsilon$, or, even better, the solution can be expressed *in closed form.*

On the other hand, certain problems that arise in the field of decentralized decision making and control have defied all attempts for the development of realistic algorithms or representations of their solution. (It has been customary to refer to such problems as *nonclassical* control problems.) Witsenhausen's counterexample in decentralized control [Wi] is a paradigm. This problem can be viewed as a simple two-stage stochastic optimal control problem without perfect recall of the measurements. In contrast to related control problems with perfect recall, for which optimal decision rules are linear and easy to compute, the optimal decision rules for Witsenhausen's problem are provably nonlinear, and it is nontrivial to even show that they exist [Wi]. Despite persistent efforts, a representation of the optimal solution to this problem or an efficient algorithm to compute its solution has never been found. Ho and Chang [HC] took a closer look at the *discrete version* of this problem. They considered the "most reasonable" approaches to the construction of an efficient algorithm, and provided a discussion explaining why such approaches fail. However, this could not rule out the possibility that some other approach might lead to an efficient algorithm, or, more importantly, that an efficient solution for the continuous problem is possible.

* Received by the editors September 7, 1984, and in revised form April 12, 1985. This research was supported in part by the National Science Foundation, and by an IBM Faculty Development Award.

† Department of Computer Science, Stanford University, Stanford, California 94305.

This increase in difficulty in going from the centralized to the distributed problem is usually attributed to a loss of convexity; however, no formal explanation of this phenomenon had been attempted. On the other hand, some recent work has related the complexity of decentralized control, somewhat loosely, with the Theory of Computational Complexity [GJ], [PS]. These results indicate that the discrete versions of some seemingly simple problems in decentralized decision making (unfortunately, so far excluding Witsenhausen's counterexample) are computationally intractable (NP-complete or worse) [PT], [TA], [GJW], [Pa], [Ts], thus providing objective measures for the difficulty of the *discrete* problems. Nevertheless, the above research left open the issue of the intractability of the (more interesting) *original continuous problems.* In general, it is not automatically true that if a discrete version of a problem is hard, then the continuous problem is also hard. A classical example here could be linear programming, which can be solved in polynomial time, despite the fact that its discrete version—integer programming—is much harder.

In this paper we address and in many ways settle the issues raised above. In § 2 we discuss the few available results on the complexity of discrete nonclassical control problems. More importantly, we prove that the discrete version of Witsenhausen's counterexample is NP-complete, thus explaining the lack of progress on it, and the failures reported in [HC]. The goal of the remaining sections is to relate the complexity of discrete and continuous problems. In particular, we show that complexity results for a discrete problem can be used to prove the nonexistence of realistic (i.e., polynomial in the desired accuracy) algorithms for classes of continuous problems. We chose to proceed in terms of a specific example, the static team decision problem [MR], [Ra]; however, our proofs define a methodology by which similar results can be proved for other problems as well. In § 3 we make precise the notion of an algorithm that solves a continuous problem. We observe that there are several possible such notions. We also describe the main construction used in the rest of the paper, whereby from any instance of the discrete version of a decision problem we construct an instance of its continuous counterpart, which is provably closely related to the discrete one. In § 4 we show our main results, linking the difficulty of nonclassical control problems (the team problem in particular) to the theory of computational complexity. For three different notions of "efficiently solvable continuous problem" we present evidence that the team problem is not. These negative results depend on $P \neq NP$ and some related conjectures from Complexity Theory. Finally, in § 5 we discuss our results; we also place them into perspective by contrasting them to other theories of complexity for continuous problems [TW], [TWW], [YN], [Ko].

2. The complexity of discrete nonclassical problems. In this section we consider the computational complexity of the discrete versions of some representative nonclassical control problems: the static team decision problem [MT], [Ra], the discrete version of Witsenhausen's counterexample in stochastic control [Wi], as well as some nonclassical control problems in Markov chains. The main new result is that the discrete version of Witsenhausen's problem is NP-complete. For convenience, we restrict to problems involving two agents only; problems with more agents are bound to be at least as hard.

The discrete static team decision problem. We define below the discrete version of the team decision problem of Marschak and Radner, called DTEAM. The problem is the following: Each one of two agents observes a separate integer random variable k_i, $i = 1, 2$, $1 \leqq k_i \leqq N$ and makes a decision $u_i = \gamma_i(k_i)$, $u_i \in \{1, \cdots, M\}$ based on his own information only. Then a cost $c(k_1, k_2, \gamma_1(k_1), \gamma_2(k_2))$ is incurred. The problem consists of finding decision rules that minimize the expected cost. For simplicity, we take all pairs (k_1, k_2) in the given range to be equiprobable.

An instance $I=(N, M, c, K)$ of DTEAM consists of positive integers N, M (the cardinalities of the observation and decision sets), a nonnegative integer K, and an integer valued cost function $c:\{1, \cdots, N\}^2\times\{1, \cdots, M\}^2\rightarrow Z$. For any pair γ_1, γ_2 of functions $\gamma_i:\{1, \cdots, N\}\rightarrow\{1, \cdots, M\}$, define their cost to be

$$J(\gamma_1, \gamma_2)=\sum_{k_1=1}^{N}\sum_{k_2=1}^{N} c(k_1, k_2, \gamma_1(k_1), \gamma_2(k_2)).$$

The optimal cost is defined to be

$$J^*(I)=\min_{\gamma_1,\gamma_2} J(\gamma_1, \gamma_2).$$

By "solving" this instance, we mean deciding whether $J^*(I)\leqq K$, or not.

We let SDTEAM (for *Simple* DTEAM) be the special case of DTEAM restricted to instances for which $K=0$, $M=4$ and the range of c is $\{0, 1\}$.

Complexity theory. At this point is seems appropriate to introduce some basic notions from complexity theory. See [GJ], [HU], [PS] for more complete and formal treatments.

Most of the discrete problems that we deal with in this paper will be of the *language recognition* kind, that is, problems of deciding whether a given string (encoding some combinatorial object) belongs to a fixed set of strings or not. In DTEAM, for example, the string encodes the integers M, N, and K, and the table of the cost function. The question is whether this string is in the set of strings (language) that encode instances of DTEAM in which the optimum cost is below K.

Our precise choice of a model of computation is not very critical. We could choose any variant of the Turing machine, or the random access machine models which appear to be much closer to actual computers [AHU]. All such choices are essentially equivalent (modulo a polynomial), as long as they are basically *realistic.* This latter clause excludes models which, for example, assume real arithmetic with infinite precision at unit cost per operation. Any model, whose units of computation can be achieved within a constant amount of time with constant hardware, is "realistic" in the above sense.

In the interest of differentiating between "easy" and "hard" problems, let us define P to be the class of all such problems that can be solved by an algorithm in a number of steps which is a polynomial in the length of the input string. Some well-known "hard" problems, including the satisfiability problem for Boolean formulas and the traveling salesman problem (with a limit on the cost of the tour, as in the definition of DTEAM), are not known, neither believed, to be in P; they belong, however, in another class, called NP (for nondeterministic polynomial). A problem is in NP if, whenever a string encodes a "yes" instance, there is a polynomially short and polynomially easy to check "certificate" that testifies to this. A "no" instance has no such certificate. For example, in the traveling salesman problem, the certificate is the shortest tour, of cost less than the set limit; in DTEAM the optimum decision rule that achieves cost K or less; and so on. Another, equivalent way to define NP is in terms of problems that can be solved in polynomial time by *nondeterministic* Turing machines (hence the name NP).

Is P = NP? This turns out to be the central open question in Complexity Theory today. It is widely believed that P ≠ NP, that is, that P is a proper subset of NP, but no proof exists (or is in sight). However, even in the absence of a definite answer to this question, for certain problems in NP we have quite convincing evidence that they are indeed intractable. What has been shown is that these problems are NP-*complete.* This means that all problems in NP reduce in polynomial time to these. Hence, NP-complete problems are "the hardest problems in NP", in the sense that, if P is not NP, then the NP-complete problems will be the first to be intractable, of nonpolynomial complexity. A great variety of some of the hardest and most stubborn computational problems from combinatorics, optimization, logic, number theory and graph theory have been shown to be NP-complete (including the traveling salesman problem and the satisfiability of Boolean formulas; see [GJ] for a complete census, circa 1979). The usual way that a new problem is shown NP-complete is to reduce a known NP-complete problem to it. We shall see a rather involved example shortly.

Problem SDTEAM is known to be NP-complete [PT]. In fact, it follows easily from our proof that SDTEAM remains NP-complete even if the instances are restricted so that we know that the optimum cost is either zero or one, and we must decide which of the two. (This is done by taking any instance with $M=3$—a case which is already NP-complete [PT]—and adding to each pair of observations a choice which can guarantee an overall cost of one). We shall use this fact in our proofs.

In our analysis of the complexity of nonclassical control problems, we shall also refer to complexity classes *above* P and NP. In analogy to polynomial-time computation, one can study the *exponential-time* analog, that is, problems solvable within a number of steps that grows as 2^{cn}, for some constant c. We let EXP and NEXP denote the corresponding deterministic and nondeterministic complexity classes. Also, we let DEXP and NDEXP be the analogous classes for *doubly exponential* complexities, that is, growths of the form 2^{c2^n}. These complexity classes are not, of course, nearly as practically important as P and NP, but they too are unresolved puzzles: It is not known whether EXP = NEXP or DEXP = NDEXP (although we expect that inequality holds). What *is* known, however, is that P = NP implies EXP = NEXP, which in turn implies DEXP = NDEXP (see [HU] for the standard arguments needed to show this).

Witsenhausen's counterexample revisited. Witsenhausen's counterexample is the following problem [Wi]:

$$\text{minimize } E[K(\gamma(x))^2+(\delta(x+\gamma(x)+v)+x+\gamma(x))^2],$$

with respect to all measurable real valued functions γ, δ of a single variable, where x, v are independent, normal, zero mean random variables (with given variance) and K a nonnegative constant. (Notice that this is not a (discrete) computational problem of the kind we introduced in the previous subsections. For more formal treatment of continuous computational problems, see the next section.) As was pointed out in the introduction, a representation of an optimal solution to this problem or an efficient algorithm has never been found. Of course, an algorithm can always be constructed as follows: discretize the densities of the random variables x, v and constrain the decision rules γ, δ to have finite range; then solve the discretized problem by exhaustive enumeration. However, this is unsatisfactory because the number of decision rules that have to be enumerated is exponential in the cardinality of the allowed range of the decision rules. It is this discrete problem that was studied by Ho and Chang [HC] with very little success. We explain this persistent record of failures by proving below that the discretized version of Witsenhausen's problem, as defined by Ho and Chang, is NP-complete.

Let us now define formally the discrete problem of interest:

Problem WITSENHAUSEN: Given probability mass functions f, $g: Z \to Q$ for integer variables x, v and integer constants K, B are there functions γ, $\delta: Z \to Z$ such that

$$J(\gamma, \delta) = E[\gamma^2(x) + K(x+\gamma(x)+\delta(x+\gamma(x)+v))^2] \leqq B?$$

THEOREM 1. WITSENHAUSEN *is* NP-*complete.*

Proof. We first introduce a variation of the problem of three-dimensional matching (3DM) [GJ]:

3DM: Given a set S and a family F of subsets of S—of cardinality three—can we subdivide F into three subfamilies C_0, C_1, C_2 such that a) subsets in each family are disjoint; b) the union of the subsets in C_0 equals S?

LEMMA 1. 3DM *is* NP-*complete.*

Sketch. We basically use the construction in the standard proof that the (less restricted) version of 3DM, in which the sets in C_1, C_2 are not required to be disjoint, is NP-complete [GJ], [PS]. In that proof we construct, for each Boolean formula with three literals per clause, an instance of 3DM, such that there is a subfamily C_0 as described in 3DM iff the formula was satisfiable. It is not hard to observe, however, that, once a subfamily C_0 exists, the remaining sets of the instance can be subdivided into two subfamilies of disjoint sets. □

To prove Theorem 1, we reduce 3DM to WITSENHAUSEN. Suppose that we are given an instance S, F of 3DM, where $S=\{1, \cdots, m\}$, $F=\{S_1, \cdots, S_n\}$. Without loss of generality, assume that $n \leqq m$. We now construct an instance of WITSENHAUSEN. There will be $3n$ values of the random variable x with nonzero probability and $M=1+4n\nu+3n$ such values for v, where $\nu=\lceil\sqrt{3n-m}+1\rceil$. All these values will be taken equiprobable. To complete the construction, we need to specify the sets $X=\{x_1, \cdots, x_{3n}\}$, $V=\{v_1, \cdots, v_M\}$ of values with nonzero probability. Concerning the constants B, K, we let $B=(3n-m)/3nM$, $K=3nM(B+1)$. To define the actual integers with nonzero probabilities, we need a lemma:

LEMMA 2. *There are n distinct integers* $0 \leqq z_1 \leqq \cdots \leqq z_n \leqq 3n^4$ *such that*

(a) *All the differences* $z_p - z_q$ *are distinct.*

(b) *Any difference* $(z_{i+1}-z_i)-(z_{j+1}-z_j)$ *is distinct from any difference in* (a).

Proof. We define z_k, $1 \leqq k \leqq n$, recursively. Let $z_1 = 0$ and assume that $z_1, \cdots, z_k$, $k<n$, have been constructed and $z_k \leqq 3k^4$. In order to pick a value for z_{k+1}, notice that it has to obey only the following constraints: (i) $z_{k+1} > z_k$, (ii) $z_{k+1} - z_p \neq z_i - z_j$, $(1 \leqq j, p, q \leqq k)$, (iii) $(z_{k+1}-z_k)-(z_{j+1}-z_j) \neq z_p - z_q$, $(1 \leqq j, p, q \leqq k+1)$. (Some of the constraints in (iii) hold automatically.) So, z_{k+1} has to avoid at most $3k^4+k^3+(k+1)^3 < 3k^4+9k^3<3(k+1)^4$ values. Therefore, there exists an integer less than or equal to $3(k+1)^4$ whose value can be assigned to z_{k+1}. □

Notice that, given n, the integers $z_1, \cdots, z_n$ can be constructed recursively in polynomial time by means of the procedure suggested by the proof of Lemma 2. Let us assume that such a sequence $z_1, \cdots, z_{3n}$ has been constructed. Moreover, let us multiply each element of the sequence by 4, so that the expressions which are distinct by Lemma 2 are different by at least 4.

We now complete the construction of the sets X, V. The set X contains $3n$ elements; each element $x \in X$ is associated to a set $S_i \in F$ and an element $j_{ik} \in S_i$, where j_{ik} $(i=1, \cdots, n; 1, 2, 3)$ denotes the kth element of S_i. We then let

$$x_{3(i-1)+k} = 3mz_{3(i-1)+k} + 3j_{ik}. \tag{2.1}$$

The set V contains the element 0; also for any consecutive elements x_i, x_{i+1} of X *corresponding to the same set* (that is, $i=1$, $2 \pmod 3$)), V contains the numbers $x_{i+1}-x_i+\rho$, $\rho \in U=\{-\nu, -\nu+1, \cdots, -2, -1, 1, 2, \cdots, \nu-1, \nu\}$. Finally, V contains the numbers $3m(A+z_i)$, $i=1, 2, \cdots, 3n$, where $A=3z_{3n}$; this completes the construction.

Let us put together a few facts, for future reference:

LEMMA 3. (a) *If for some γ, δ, we have $J(\gamma,\delta)\leqq B$, then $|\gamma(x)|\leqq \nu$, $\forall x\in X$.*

(b) *The expressions $|x_i-x_j|$, $|x_i-x_j+x_p-x_q|$, $|x_{i+1}-x_i-x_{j+1}+x_j+x_p-x_q|$, $(1\leqq i-1, j-1, p, q\leqq n)$ are either zero or no smaller than $3m$. For large enough m, $3m>4\nu+2$.*

(c) *$|x_i-x_j|\leqq 3mA-2-\nu$, for large enough m.*

Proof. (a) If for some $x\in X$ we have $|\gamma(x)|>\nu$, then

$$J(\gamma,\delta)>\frac{\nu^2}{3nM}\geqq\frac{3n-m}{3nM}=B.$$

(b) We use (2.1), the inequality $j_{ik}\leqq m$ and the fact that the magnitudes of the corresponding expressions involving the z's instead of the x's are either zero or at least four; we obtain in the second case, for example, $|x_{i+1}-x_i-x_{j+1}+x_j+x_p-x_q|\geqq 12m-9m=3m$. Finally notice that ν increases only as the square root of m, which also proves part (c). □

The following lemma completes the proof of the theorem.

LEMMA 4. *(S,F) is a "yes" instance of* 3DM *if and only if there exist γ, δ such that $J(\gamma,\delta)\leqq B$ for the above constructed instance of* WITSENHAUSEN.

Proof. If. Suppose that there exist γ, δ such that $J(\gamma,\delta)\leqq B$. Then, in particular, $K(x+\gamma(x)+\delta(x+\gamma(x)+v))^2/3Mn\leqq B$, $\forall x\in X$, $\forall v\in V$. Therefore, $|x+\gamma(x)+\delta(x+\gamma(x)+v)|^2\leqq B/(B+1)<1$, which implies that $x+\gamma(x)+\delta(x+\gamma(x)+v)=0$, $\forall x\in X$, $\forall v\in V$. Let $x_i\neq x_j$. Then, using Lemma 3(a, b), we have

$$\begin{aligned}|\delta(x_i+\gamma(x_i)+v)-\delta(x_j+\gamma(x_j)+v')|&=|x_i+\gamma(x_i)-x_j-\gamma(x_j)|\\&\geqq|x_i-x_j|-|\gamma(x_i)-\gamma(x_j)|\\&\geqq 3m-2\nu>0,\end{aligned}$$

which shows that

$$x_i+\gamma(x_i)+v\neq x_j+\gamma(x_j)+v' \quad \forall v, v'\in V,\quad \forall x_i, x_j\in X,\quad x_i\neq x_j. \tag{2.2}$$

Let x_i, x_{i+1} be two consecutive elements of X corresponding to the same set S_j. Inequality (2.2) must hold for $v'=0$ and $v=x_{i+1}-x_i+\rho$, $\forall\rho\in U$. Thus, $\gamma(x_i)+\rho\neq\gamma(x_{i+1})$, $\forall\rho\in U$. Consequently, either $\gamma(x_i)=\gamma(x_{i+1})$, or $|\gamma(x_i)-\gamma(x_{i+1})|>\nu$, which would contradict Lemma 3(a). Therefore γ takes the same value on those elements of X corresponding to the same set S_j. We denote this value by $\gamma(S_j)$.

Inequality (2.2) must also hold when x_i, x_j correspond to the same element $k\in S$ belonging to different subsets S_p, S_q; that is, $x_i=3mz_i+3k$, $x_j=3mz_j+3k$. Let $v=3m(A+z_j)$, $v'=3m(A+z_i)$. Inequality (2.2) becomes $\gamma(x_i)\neq\gamma(x_j)$, which implies $\gamma(S_p)\neq\gamma(S_q)$, whenever $S_p\neq S_q$, $S_p\cap S_q\neq\emptyset$.

Notice that (by our choice of B), $\gamma(x)$ can be nonzero for at most $3n-m$ elements of X. Moreover, at most one $\gamma(x)$ per element of $S=\{1,\cdots,m\}$ can be zero; thus, $\gamma(x)$ must be nonzero for exactly $3n-m$ elements; for those elements, $|\gamma(x)|=1$. Let C_0 (respectively, C_1, C_2) be the family of subsets of F for which $\gamma(S_p)=0$ (respectively, $\gamma(S_p)=1$, $\gamma(S_p)=-1$). By the discussion in the last paragraph, subsets within the same family have to be disjoint. Moreover, $\gamma(x)=0$ for exactly m elements, which shows that C_0 covers S exactly and we have a "yes" instance of 3DM.

Only If. Conversely, suppose that we have a "yes" instance of 3DM and let C_0, C_1, C_2 be the desired families of subsets. We construct γ by letting $\gamma(x_i)=0$ (respectively, 1, -1) if x_i corresponds to an element of a subset $S_p\in C_0$ (respectively, C_1, C_2). Since C_0 is a cover to S, $\gamma(x)=0$ for exactly m elements $x\in X$. Consequently, $E[\gamma^2(x)]=1/(3nM)(3n-m)=B$. It remains to show that δ can be chosen so that

$x+\gamma(x)+\delta(x+\gamma(x)+v)=0$, $\forall x\in X$, $\forall v\in V$. For this it is sufficient to prove that $x_i+\gamma(x_i)+v\neq x_j+\gamma(x_j)+v'$, whenever $x_i\neq x_j$ and for all v, $v'\in V$. So, suppose that the desired inequality does not hold for some x_i, x_j, v, v'. We will derive a contradiction, but we will have to consider the various possible cases for v and v'.

(i) $v=v'=0$. If $x_i+\gamma(x_i)=x_j+\gamma(x_j)$, then $|x_i-x_j|\leqq 2$, which contradicts Lemma 3(b).

(ii) $v=0$, $v'=x_{l+1}-x_l+\rho$, $\rho\in U$. Then

$$|(x_i-x_j)-(x_{l+1}-x_l)|=|\gamma(x_j)-\gamma(x_i)+\rho|\leqq 2\nu+2<3m, \tag{2.3}$$

which implies, by Lemma 3(b), that $x_i-x_j=x_{l+1}-x_l$. It follows that $i=l+1$, $j=l$; therefore, $\gamma(x_i)=\gamma(x_j)$ and, using (2.3), $\rho=0$, which is a contradiction.

(iii) $v=x_{p+1}-x_p+\rho$, $v'=x_{l+1}-x_l+\rho'$, ρ, $\rho'\in U$. Then

$$|x_i-x_j+x_{p+1}-x_p-x_{l+1}+x_l|=|\gamma(x_j)-\gamma(x_i)-\rho+\rho'|<2\nu+2<3m,$$

which implies that $x_i-x_j+x_{p+1}-x_p-x_{l+1}+x_l=0$. So, one of the following must hold: $x_i=x_p$, $x_j=x_l$ or $x_p=x_l$. If $x_i=x_p$, it follows that $x_j=x_l$ (and conversely); in either case, we obtain $x_{p+1}=x_{l+1}$ and $x_p=x_l$; therefore, $x_i=x_j$, which is a contradiction.

(iv) $v=0$, $v'=3m(A+z_l)$. Then, $|x_i-x_j|=|\gamma(x_j)-\gamma(x_i)+3m(A+z_l)|\geqq 3mA-2$, which contradicts Lemma 3(c).

(v) $v=x_{l+1}-x_l+\rho$, $\rho\in U$, $v'=3m(A+z_p)$. Then,

$$|x_i-x_j+x_{l+1}-x_l|=|\gamma(x_j)-\gamma(x_i)-\rho'+3m(A+z_p)|\geqq 3mA-2-\nu,$$

which contradicts Lemma 3(c).

(vi) $v=3m(A+z_p)$, $v'=3m(A+z_q)$. Let $x_i=3mz_i+k$, $x_j=3mz_j+k'$. Then, $3m|z_i-z_j+z_p-z_q|=|3(k'-k)+\gamma(x_j)-\gamma(x_i)|$. If $z_i-z_j+z_p-z_q=0$, then $2\geqq|\gamma(x_i)-\gamma(x_j)|=3|k-k'|$, which implies $k=k'$. Therefore, $\gamma(x_i)=\gamma(x_j)$, which is a contradiction because γ takes different values when x_i, x_j correspond to the same element of S. Therefore, $12m\leqq|z_i-z_j+z_p-z_q|\leqq 3m+2$, which is also a contradiction. This completes the proof of the lemma and the theorem. □

Decentralized and output control of imperfectly observed Markov chains. By simply observing that Witsenhausen's counterexample and the static team decision problem are at the root of several problems in decentralized control, we obtain some interesting Corollaries of Theorem 1. For example, one might be interested in formulating and studying problems of decentralized control of Markov chains. However, a single stage of such a problem would require the solution of a static team decision problem and NP-completeness (or worse) follows.

One could also formulate a problem of output control of a Markov chain, similar to the problem studied in [LA] under linear quadratic Gaussian assumptions: that is, the decision at time k would be constrained to be a function only of the observation made at time k (no recall). In fact, problem WITSENHAUSEN is a two-stage output control problem for a Markov chain and NP-completeness follows. The two-stage output control problem can be also easily seen to contain as a special case the problem of minimum distortion quantization which is also NP-complete [GJW]. Infinite horizon average cost versions of that problem can be also easily shown to be NP-complete. Finally, problems of causal coding and control of Markov chains, as defined in [WV], are also NP-complete for the same reasons.

3. Continuous problems and algorithms.

Continuous problems and their complexity. Our final aim is to derive complexity results for continuous problems. Unfortunately, there is no standard model of computation—or complexity measure—for such problems. In this subsection we shall discuss various notions of computation and complexity pertaining to continuous problems. A comparison of our framework and other existing work on the complexity of continuous problems appears in the last section.

In an instance of a typical continuous problem, we are given a finite set $F = \{f_1, \cdots, f_n\}$ of real functions (without loss of generality, with domains some power of the unit interval) and we are asked to evaluate (usually approximately) a functional $G(F) \in \mathfrak{R}$ of these functions. For example, $f_1, \cdots, f_n$ may be the boundary conditions for a partial differential equation and $G(f_1, \cdots, f_n)$ the value of the corresponding solution at a specific point. Closer to our concerns in this paper, we can define the continuous counterpart of the DTEAM problem mentioned in the previous section. In an instance of this problem, we are given a function $c: [0, 1]^4 \rightarrow [0, 1]$, assigning a cost to each combination of observations $y_1, y_2 \in [0, 1]$ and decisions $\gamma_1(y_1), \gamma_2(y_2) \in [0, 1]$. (Notice that we are assuming, for simplicity, that the probability distribution is uniform over $[0, 1]^2$.) The goal is to compute the functional $J^*(c)$ defined by

$$J^*(c) = \inf_{\gamma_1, \gamma_2} \int_0^1 \int_0^1 c(y_1, y_2, \gamma_1(y_1)\gamma_2(y_2)) \, dy_1 \, dy_2.$$

We shall be interested in the special case of this problem in which the function c is *Lipschitz continuous* with Lipschitz constant 1 (with respect to the max norm on $\mathfrak{R}^4$), as a representative of those special cases that we can hope to solve efficiently. Without such "smoothness" conditions, no realistic solution of continuous problems is possible, for simple information-theoretic considerations. We call the continuous version of the DTEAM problem with the Lipschitz condition the *Lipschitz continuous team problem*, or LCTEAM. That is, LCTEAM is the set of all instances, as described and restricted above. It should be obvious that a host of problems of continuous nature are amenable to similar formalization.

There are several possible notions of what it means for an algorithm to solve such a problem, and, equally important, the complexity of its operation. The subtle part is defining the sense in which the continuous functions f_i are "given". We examine a number of such approaches below.

Oracle algorithms. Continuous problems of the type defined above can often be solved by an algorithm which operates as follows: The input of the algorithm is a positive real ε, and the output is an approximation of the functional with error at most ε. Every time that the algorithm needs the value of a function f_i at some point x, this is done as follows: The algorithm submits x (a rational point), and an integer k to an *oracle* for f_i, and the oracle gives back the k most significant digits of the answer $f_i(x)$. The algorithm is "charged" for this service k steps, plus of course the time it took to construct x up to the desired precision. We say that an oracle algorithm solves a continuous problem Π in polynomial time if, for every instance I of Π there is a polynomial p_I such that the algorithm solves I within accuracy ε in time $p_I(1/\varepsilon)$.

Uniformly polynomial oracle algorithms. There is a stronger notion of efficiency, which requires that the polynomial be independent of the instance I. We call algorithms with this property *uniformly polynomial.* Notice that is a much stronger notion than that of plain polynomial-time oracle algorithms.

Note: The distinction between polynomial and uniformly polynomial algorithms has no counterpart in the context of combinatorial (discrete) problems, since in discrete problems the instance plays the role of both ε and I in the above definitions. It is, however, meaningful for continuous problems. For example, consider the problem in which we are given one Lipschitz continuous function f over $[0, 1]$, and we are asked to compute $G(f) = \inf_{x\in[0,1]} f(x)$. A straightforward discretization leads to an algorithm with time requirements $O(K_f/\varepsilon)$, where K_f is the Lipschitz constant of f; so, this is a polynomial algorithm. On the other hand, $O(K_f/\varepsilon)$ is also a lower bound and since this problem contains instances with arbitrarily large Lipschitz constants, no uniformly polynomial algorithm exists.

Uniformly hard instances. One way to show that a continuous problem has no polynomial-time oracle algorithm at all is to exhibit an instance for which no polynomial-time oracle algorithm exists; such instances are called *uniformly hard.* Naturally, for discrete problems there are no hard single instances.

Instance-specific algorithms. We obtain an interesting variant of the concept of oracle algorithms by considering single instances of the problem Π. In each instance I, we just wish to compute a number, namely $G(F)$. We could ask the question, is this number *polynomial-time computable*, in the sense that we can compute it within accuracy ε in time polynomial in $1/\varepsilon$ by an ordinary algorithm (involving no oracles). This is a meaningful question only if the functions f_i are themselves polynomial-time computable, in that the value $f_i(x)$ can be computed in time which is polynomial in the accuracy in which x is given, and the desired accuracy.

Iterative algorithms. In numerical analysis or mathematical programming we are often interested in convergent *iterative algorithms.* These differ from the class of algorithms we introduced above in that they do not take ε as an input, and they never halt. Rather, from time to time they produce output values $G_i(F)$, $i = 1, 2, \cdots$ which are increasingly accurate approximations of $G(F)$. We may call an iterative algorithm *polynomial* if there is a polynomial p such that, for every instance, at time $p(1/\varepsilon)$, the most recent output value is accurate, within ε. It is clear that if a polynomial iterative algorithm exists, there also exists a uniformly polynomial algorithm for the problem. In fact the converse also holds [9]: take a uniformly polynomial algorithm and run it with $\varepsilon = 2^{-k}$, $k = 1, 2, \cdots$. The resulting algorithm is a polynomial iterative algorithm. For this reason, we shall not consider iterative algorithms any further.

3.7. The basic construction. Our method of connecting the complexity of the continuous version of the TEAM problem to the (much better understood) complexity of the discrete one, is based on the following lemma and construction:

LEMMA 5. *For each instance I of the* SDTEAM *problem we can define a function* $c_I:[0,1]^4 \to [0,1]$ *such that*:

(i) *Function c_I is Lipschitz continuous* (*with Lipschitz constant* 1), *and thus it defines an instance of* LCTEAM.

(ii) *The optimum $J^*(c_I)$ equals $1/20N^4$ if the optimum of I was* 1, *and* 0 *if it was* 0 (*recall for that instances of* SDTEAM *these are the only possibilities*; *N is the number of possible observations in I*).

(iii) *For any I and k-bit numbers y_1, y_2, u_1, u_2, and any $l > 0$, the l most significant bits of $c_I(y_1, y_2, u_1, u_2)$ can be computed in time polynomial in k, l, and the size of I.*

Proof. Let us first define a function $\alpha:[0, N] \to [0, 1/N]$ as follows:

$$\alpha(x) = \begin{cases} x - \lfloor x \rfloor & \text{if } x - \lfloor x \rfloor \leqq \dfrac{1}{N}, \\[2ex] \lceil x \rceil - x & \text{if } \lceil x \rceil - x \leqq \dfrac{1}{N}, \\[2ex] \dfrac{1}{N} & \text{otherwise} \end{cases}$$

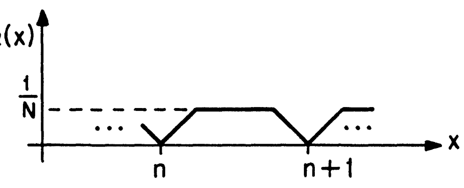

FIG. 1

(see Fig. 1), and define $q(y_1, y_2) = (1/(1-1/N)^2)\alpha(y_1)\alpha(y_2)$. Notice that q has Lipschitz constant $4/N$, and that its integral over $[0, N]^2$ is one.

Let us now recall the cost function of the discrete instance I, call it $d:\{1, 2, \cdots, N\}\times\{1, 2, 3, 4\}\to\{0, 1\}$. For $1 \leqq x_1, x_2 \leqq N$, integers, and $v_1, v_2 \in [1, 4]$, let $h(x_1, x_2, v_1, v_2)$ be the smallest $\delta \leqq 1$ such that there are u_1, u_2 with $|u_1 - v_1|, |u_2 - v_2| \leqq \delta$ and $d(x_1, x_2, u_1, u_2) = 0$, or 1 if no such δ exists. Then, define, for $0 \leqq y_1, y_2, u_1, u_2 \leqq 1$ the cost function $c_I(y_1, y_2, u_1, u_2)$ to be

$$\frac{q(Ny_1, Ny_2)}{20}[h(\lceil Ny_1\rceil, \lceil Ny_2\rceil, 3u_1+1, 3u_2+1) + 2p(3u_1+1) + 2p(3u_2+1)],$$

where $p(x)$ is the distance between x and its closest integer.

Let us verify the properties of c_I. To verify (i), function h has discontinuities at integral values of its first two arguments, but q is zero there, so c_I is continuous. To check that the Lipschitz constant is 1, recall that if the functions f_i have Lipschitz constants L_i and maxima m_i, $i = 1, 2$, then the function $f_1 f_2$ has Lipschitz constant $L_1 m_2 + L_2 m_1$. Within each "rectangle" of constant $\lceil Ny_1\rceil$, $\lceil Ny_2\rceil$ h has maximum 1 and Lipschitz constant 3, and $p(3u_1+1)+p(3u_2+1)$ has Lipschitz constant 6 and maximum 2, so their sum has maximum 3 and Lipschitz constant 9. Also, q has maximum $4/N^2$ and Lipschitz constant at most 8, and so their product has Lipschitz constant at most 20. It follows that c_I has indeed Lipschitz constant at most 1, as required.

For (ii), let us denote by $\mathcal{D}$ the set of all functions $\gamma:[0, 1]\to[0, 1]$ which are piecewise constant, with discontinuities at points i/N, and taking the values $\{0, \frac{1}{3}, \frac{2}{3}, 1\}$. We claim that, for fixed γ_2, the decision function $\gamma_1(y_1)$ which minimizes $J(\gamma_1, \gamma_2)$ is in $\mathcal{D}$. Notice that $\inf_\gamma J(\gamma_1, \gamma_2)$ is equal to

$$\frac{1}{(1-1/N)^2}\int_0^1 \alpha(Ny_1)\cdot\min_{u_1}\int_0^1 \alpha(Ny_2)[h + 2p(3u_1+1) + 2p(3\gamma_2(y_2)+1)]\,dy_2\,dy_1.$$

To carry out this minimization, it is sufficient to minimize, with respect to u_1,

$$\int_0^1 \alpha(Ny_2)[h(\lceil Ny_1\rceil, \lceil Ny_2\rceil, 3u_1+1, 3\gamma_2(y_2)+1)]\,dy_2 + \int_0^1 2p(3u_1+1)\,dy_2$$

for each y_1. The first term does not depend on y_1 within the interval $[i/N, (i+1)/N]$, and thus the optimum u_1 is indeed constant within this interval. Secondly, notice that the second term, together with the Lipschitz condition, ensures that the minimum is achieved at integer values of $3u_1+1$, that is, at values of γ_1 in $\{0, \frac{1}{3}, \frac{2}{3}, 1\}$.

The same argument shows that γ_2 may be constrained to be in $\mathcal{D}$ as well. Once we have shown that the optimizing decision functions are in $\mathcal{D}$, we have essentially shown that the continuous LCTEAM problem defined by c_I is in fact "isomorphic" to the discrete one I, and it has optimum which is $J^*(c_I) = (1/20N^4)J^*(I)$, where $J^*(I)$ is the optimum of I, either 0 or 1. Part (iii) is trivial. □

4. The main results. In this section we present our evidence that the Lipschitz continuous team problem is indeed intractable. We prove *three* such theorems, corresponding to three different notions of complexity of continuous problems introduced in the previous section, namely uniformly polynomial oracle algorithms, polynomial instance-specific algorithms, and uniformly hard instances. In all three cases, we show the intractability of LCTEAM, assuming that a very likely conjecture in Complexity Theory is true. Naturally, the stronger our notion of intractability of LCTEAM, the stronger the complexity-theoretic conjecture needed.

4.1. Nonexistence of uniformly polynomial algorithms.

THEOREM 2. *There is a uniformly polynomial algorithm for* LCTEAM *if and only if* P = NP.

Proof. If. Suppose that P = NP. We shall describe a uniformly polynomial algorithm for LCTEAM. The algorithm works by discretizing the problem, and obtaining appropriately approximate solutions by solving discrete instances of DTEAM (which is possible, once P = NP).

Let R correspond to a *truncation* operation: given some $x \in [0, 1]$ and some $\varepsilon > 0$, $R(x, \varepsilon)$ retains the $\lceil \log(1/\varepsilon) \rceil$ most significant bits of x; consequently, $|x - R(x, \varepsilon)| \leqq \varepsilon$ and if $\log(1/\varepsilon)$ is an integer, then $(1/\varepsilon)R(x, \varepsilon)$ is also an integer.

Given the cost function c of an instance of LCTEAM and some $\varepsilon > 0$ such that $\log(1/\varepsilon)$ is an integer, we construct an instance I of DTEAM by letting $\Delta = \varepsilon/8$, $N = 1/\Delta$, $M = 1/\Delta$ and cost function

$$d(i, j, k, l) = \frac{1}{\Delta} R(c(i\Delta, j\Delta, k\Delta, l\Delta), \Delta), \qquad (i, j, k, l) \in \left\{1, \cdots, \frac{1}{\Delta}\right\}^4. \tag{4.1}$$

Clearly, d is integer-valued and $C = \max d \leqq 1/\Delta$. Let $J^c(\gamma_1, \gamma_2)$, $J^d(\delta_1, \delta_2)$ denote the costs of pairs of decision rules for the continuous (c) and discrete (d) instances, respectively. J^{*c}, J^{*d} are the corresponding optimal costs.

LEMMA 6. $\|J^{*c} - \Delta^3 J^{*d}| \leqq \varepsilon$.

Proof. Let δ_1, δ_2 be the optimal for d. Let $\gamma_i(y_i) = \Delta\delta_i(k_i)$ for $y_i \in [(k_i - 1)\Delta, k_i\Delta]$, $k_i = 1, \cdots, N$, $i = 1, 2$. Using the definition of d and the Lipschitz continuity of c, we obtain

$$\begin{aligned} J^{*c} \leqq J(\gamma_1, \gamma_2) &= \sum_{k=1}^{N} \sum_{m=1}^{N} \int_{(k-1)\Delta}^{k\Delta} \int_{(m-1)\Delta}^{m\Delta} c(y_1, y_2, \gamma_1(y_1), \gamma_2(y_2))\, dy_1\, dy_2 \\ &\leqq \Delta^2 \sum_{k=1}^{N} \sum_{m=1}^{N} (\Delta d(k, m, \delta(k), \delta(m)) + 2\Delta) \\ &= \Delta^3 J^d(\delta_1, \delta_2) + 2\Delta = \Delta^3 J^{*d} + 2\Delta \leqq \Delta^3 J^{*d} + \varepsilon. \end{aligned} \tag{4.2}$$

In order to prove the converse inequality, suppose that $\bar{\gamma}_1, \bar{\gamma}_2 : [0, 1] \to [0, 1]$, are such that $J^c(\bar{\gamma}_1, \bar{\gamma}_2) \leqq J^{*c} + \Delta$. Let $f(y_1, y_2) = \int_0^1 c(y_1, y_2, u_1, \bar{\gamma}_2(y_2))\, dy_2$. Then f is also Lipschitz continuous with Lipschitz constant 1. It follows that

$$|\inf_u f(y, u) - \inf_u f(y', y)| \leqq 2|y - y'| \quad \forall y, y' \in [0, 1].$$

Let $\hat{\gamma}_1(y_1) = \operatorname{argmin}_{u \in [0,1]} f(k\Delta, u)$, for $y_1 \in ((k-1)\Delta, k\Delta)$. Then,

$$\begin{aligned} J^c(\hat{\gamma}_1, \bar{\gamma}_2) &= \int_0^1 f(y_1, \hat{\gamma}_1(y_1))\, dy_1 \leqq \int_0^1 \left(\inf_{u \in [0,1]} f(y_1, u_1) + 2\Delta \right) dy_1 \\ &= \inf_{\gamma_1} \int_0^1 f(y_1, \gamma_1(y_1))\, dy_1 + 2\Delta = \inf_{\gamma_1} J^c(\gamma_1, \bar{\gamma}_2) + 2\Delta \\ &\leqq J^c(\bar{\gamma}_1, \bar{\gamma}_2) + 2\Delta \leqq J^{c*} + 3\Delta. \end{aligned}$$

In a similar way, we may construct a piecewise constant function $\hat{\gamma}_2$: $[0, 1] \to [0, 1]$ such that

$$J^c(\hat{\gamma}_1, \hat{\gamma}_2) \leqq J^c(\hat{\gamma}_1, \bar{\gamma}_2) + 2\Delta \leqq J^{c*} + 5\Delta.$$

The decision rules $\hat{\gamma}_i$, $i = 1, 2$, being piecewise constant determine corresponding decision rules $\hat{\delta}_i$, $i = 1, 2$, for the discrete cost function. Then, a chain of inequalities similar to (4.2) leads to

$$\Delta^3 J^{d*} \leqq \Delta^3 J^d(\hat{\delta}_1, \hat{\delta}_2) \leqq J^c(\hat{\gamma}_1, \hat{\gamma}_2) + 2\Delta \leqq J^{c*} + 7\Delta \leqq J^{c*} + \varepsilon. \qquad \square$$

Since P = NP there exists an algorithm for the problem of computing the optimal cost of any instance of DTEAM (based on the algorithm for DTEAM and binary search), which is polynomial in M, N, $\log C$, where C is the largest integer appearing in the cost function. Consider then the following algorithm for LCTEAM:

(i) Decrease ε (at most by a factor of 2) so that $\log(1/\varepsilon)$ is an integer.

(ii) Use the oracle to read the $\log(8/\varepsilon)$ most significant bits of $c(i\Delta, j\Delta, k\Delta, m\Delta)$, $1 \leqq i, j, k, m \leqq N$, where $N\Delta = 1$, $\Delta = \varepsilon/8$.

(iii) Run the assumed algorithm on the resulting instance of DTEAM, as defined by (4.1). Multiply the output by Δ^3 and return it.

This is clearly a uniformly polynomial algorithm, and the proof of the *if* part is complete.

Only If. If we had a uniformly polynomial algorithm for LCTEAM, we could solve any instance I of SDTEAM of size N as follows:

(i) Construct the corresponding instance c of LCTEAM, as in Lemma 5.

(ii) Simulate the assumed algorithm for LCTEAM on it, with desired accuracy $\varepsilon = 1/40N^4$. The time required is polynomial in N, including the computations of c, which, by Lemma 5(iii), are polynomially related to the "charges" for oracle calls of the corresponding computation of the algorithm.

(iii) If the result is less than $\varepsilon = 1/40N^4$, then the optimum cost of SDTEAM was 0, otherwise 1.

Since this is a polynomial-time algorithm for SDTEAM, an NP-complete problem, it follows that P = NP. $\square$

4.2. A hard instance with efficiently computable cost.

THEOREM 3. *If* DEXP ≠ NDEXP *then there exists an instance c of* LCTEAM *such that*

(i) *The cost* $c(y_1, y_1, u_1, u_2)$, *where* y_1, y_2, u_1, u_2 *are k-bit numbers between* 0 *and* 1 *can be computed with accuracy* ε *in time polynomial in* 2^k *and* $1/\varepsilon$, *whereas*

(ii) *The optimum value* J^* *is not polynomially computable*; *that is, it cannot be computed within accuracy* ε *in time polynomial in* $1/\varepsilon$.

Proof. We first need a lemma concerning the existence of certain "hard" sequences of instances of NP-complete problems, in the spirit of [HSI].

LEMMA 7. *If* DEXP ≠ NDEXP *then there is a sequence* $I_1, I_2, \cdots$ *of instances of* SDTEAM *such that*:

(a) *Instance* I_i *has size* (*that is,* N) *equal to* 2^i.

(b) *There is an algorithm which, given i, constructs* I_i *in time polynomial in* 2^i (*the size of the instance produced*).

(c) *There is no polynomial-time algorithm that solves all instances* I_i.

Sketch. Consider a problem L in NDEXP − DEXP. Without loss of generality, instances of L are encoded in binary, and therefore an instance i also represents an integer, in binary. For each instance i of L, let $f(i)$ be the string of length 2^i which starts with the string i and has 0's in all other positions. The language $f(L) = \{f(i): i \in L\}$

is in NP (since L is in NDEXP), and thus there is a polynomial-time transformation that transforms each string $f(i)$ to an instance of SDTEAM such that the instance of SDTEAM is a "yes" instance if and only if $f(i) \in f(L)$. By "padding" these instances of SDTEAM to make them of size a power of two, and filling in the gaps with "null" instances, we obtain the sequence of the lemma. □

To show the theorem, consider a sequence of instances as constructed in the lemma. For each such instance I_i, we construct a continuous function $c_i: [0,1]^4 \to [0,1]$, as in Lemma 5. Consider now a *scaled*, *shifted* version of c_i, call it c_i', with support $[1-2^{-i}, 1-2^{-(i+1)}]^2 \times [0,1]^2$ (see Fig. 2), defined in this range as

$$c_i'(y_1, y_2, u_1, u_2) = \frac{1}{2^{i+1}} c_i(2^{i+1}(y_1 - 1 + 2^{-i}), (2^{i+1}(y_2 - 1 + 2^{-i}), u_1, u_2)).$$

c_i is zero outside this domain. Finally, define the function c to be

$$c(y_1, y_2, u_1, u_2) = \sum_{i=1}^{\infty} c_i'(y_1, y_2, u_1, u_2).$$

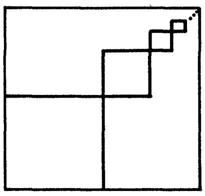

FIG. 2

This function is Lipschitz continuous with constant 1 (due to the scaling), as required by the theorem, and it is easy to see that it satisfies condition (i) (compare with (iii) of Lemma 5). To show (ii), notice that the optimum value J^* corresponding to c can be expressed in terms of the optima J_i^{*d} of the instances I_i that comprise it, as follows:

$$J^* = \sum_{i=1}^{\infty} J_i^{*d} \frac{1}{20 \cdot 2^{4i}} \frac{1}{2^{i+1}} \frac{1}{2^{2(i+1)}}.$$

The first term of the addend is the cost of the original discrete instance, known to be either 0 or 1 in SDTEAM. The second term was introduced by the construction of Lemma 5. The third is due to the scaling, whereas the last term represents the area of the support of instance c_i', as defined above. Thus, $J^* = \frac{1}{160} \sum_{i=1}^{\infty} J_i^{*d} 2^{-7i}$. From the form of the sum, it is evident that, if we could compute J^* within accuracy ε in time polynomial in $1/\varepsilon$, then we could compute the optimum cost of I_i in time polynomial in the size of I_i, contrary to Lemma 7. □

Theorem 3 has a weak converse. It can be shown that, if an instance of LCTEAM as described in Theorem 3 exists, then EXP$\neq$NEXP. The argument goes as follows: If such a hard instance exists, then its *discretizations* (that is, the sequence of discrete problems resulting by subdividing the unit interval in 2^i equal intervals, and by defining the cost function on this grid by a restriction of the continuous cost function) are not all solvable by polynomial-time algorithms. Thus, we have a hard exponentially sparse sequence of *optimization* problems of the DTEAM type (in which we are asked to determine the optimum cost). Since each optimization problem in this sequence can be reduced to an exponential number of a *recognition* problems (asking whether the cost of an instance is below some bound), say, by binary search, [PS], we obtain a hard polynomially sparse sequence of instances of this problem, known to be NP-complete. The existence of such hard sequences is known (see [HSI], or the argument above) to be equivalent to EXP$\neq$NEXP.

A uniformly hard instance. If P$\neq$NP we can show something stronger than the nonexistence of uniformly polynomial algorithms proved in Theorem 2. In particular, we can show that there is a uniformly hard instance of LCTEAM, which "fools" all polynomial-time oracle algorithms. Our construction has to use diagonalization arguments which are not polynomially constructive, and as a result the instance constructed is not one that can be computed efficiently. Since a complete proof of this result would require the introduction of machinery in a scale disproportional to the information added, we only present an outline of the proof.

THEOREM 4. *There is a uniformly hard instance of* LCTEAM *if and only if* P$\neq$NP.

Sketch. One direction follows from Theorem 2. For the *if* direction, we first need to define a discrete analog of an oracle algorithm. One way to do this is to consider *sequence algorithms*, that is, algorithms which operate on infinite sequences of instances of a problem. Such an algorithm accepts as its input an infinite tape with the sequence, together with an integer i, and it returns the answer ("yes" or "no") of the ith instance of the sequence. To capture the charges due to precision of the queries and the answers of oracle machines, we require that the algorithm is charged $\lceil \log k \rceil$ to determine the value of the kth bit of its input tape.

We first show that, if P$\neq$NP, there is a sequence of instances of the SDTEAM problem, of size exponentially increasing, which cannot be solved by any sequence algorithm. The construction is carried out by enumerating all polynomial-time sequence algorithms, and using the ith non-"null" instance in the sequence to rule out the ith sequence algorithm as a potential solver of the present instance (i.e., sequence). Since P$\neq$NP, an instance on which the ith sequence algorithm does the wrong thing, and which is of size larger than some given bound, must exist. To avoid the possibility in which the ith algorithm takes "advice" from the previous or subsequent instances in solving the current instance, we interject a doubly exponential number of "null" instances between two such consecutive instances.

We finally construct an instance of the LCTEAM problem from the given sequence of SDTEAM problems, exactly as in the proof of Theorem 3. If this instance could be solved by some oracle algorithm, it can be argued that the sequence constructed in the previous paragraph can be solved by a sequence algorithm, which is impossible by its construction. $\square$

5. Discussion. We have shown that the team decision problem with a Lipschitz continuous cost function and uniform probability distribution holds the same place in the continuous world that NP-complete problems do in the discrete world: it possesses an approximate algorithm which is polynomial in the desired accuracy if and only if P$=$NP. A similar result can be also proved if Lipschitz continuity is replaced by some other, possibly stronger, smoothness requirement such as once or twice differentiability, etc. Only the construction in Lemma 5 would have to be a little more elaborate. A similar result is also possible for Witsenhausen's counterexample in stochastic control if the assumption of normality of the underlying random variables is relaxed. Since the team decision problem is a basic component of (generally harder) problems in decentralized stochastic control, such problems (at least in the absence of any more special structure) are qualitatively different from the vast majority of traditional problems in continuous mathematics and classical control. Such problems, including nonlinear optimization, filtering and control, as well as partial differential equations, possess algorithms which are polynomial in the desired accuracy, when some smoothness conditions are satisfied, and are solvable from a realistic point of view.

The proofs of our results are based on the fact that the discrete version of the team problem is NP-complete. In this sense, we demonstrate that NP-completeness results can be exploited to make inferences about the computational complexity of continuous problems. It should be noted, however, that the various notions of intractability used call for conjectures of varying strength from Complexity Theory, all of them implying $P \neq NP$.

The proofs of Theorems 2, 3, and 4 determine a methodology that can be applied to obtain similar negative complexity results concerning other continuous problems as well. Abstracting the main elements of the proofs, we see that the following properties of LCTEAM were heavily used:

(i) The discrete version of the problem of interest should be NP-complete.

(ii) We should be able to take an instance of the discrete problem and construct an instance of the continuous problem as in Lemma 5, while respecting certain smoothness requirements.

(iii) The above construction should be simple enough, so that the corresponding oracle calls can be efficiently simulated by a Turing machine.

(iv) Finally, it should be possible to take a sequence of increasingly large discrete instances and imbed them into a single one, while keeping the dimension of the continuous problem constant.

Finally, let us comment on the relation and some differences of our framework with other theories of complexity for continuous problems. The complexity of any algorithm solving a continuous problem can be roughly divided into two kinds of activities: oracle calls to obtain information about the instance to be solved and computations based on the values returned by the oracle. A lot of past research [TW], [TWW], [NY] has obtained lower bounds on the overall complexity by deriving lower bounds on the number of oracle calls necessary to obtain enough information so that an ε-approximate solution is possible. This is a valid approach for the types of problems emphasized in that research (mainly mathematical programming and numerical integration of partial differential equations) and has produced many interesting results; the main reason is that in such problems the amount of any further computation necessary can be bounded by a polynomial (and some times linear) function of the number of oracle calls. The team problem, however, is different: while $O(1/\varepsilon^4)$ oracle calls provide sufficient information for an ε-approximate solution, we have shown that further computations require time which is exponential in ε (unless $P = NP$). In other words, the structure of the team problem forces us to emphasize its computational complexity rather than its informational requirements.

Much closer to our approach are the very interesting recent results in [Ko]. In that paper, it is shown that there are ordinary differential equations which are given in terms of easily computable functions, but which cannot be integrated efficiently, unless $P = PSPACE$. In this sense, Ko's results are quite similar in spirit to Theorem 3. One of the differences is that Ko's notion of efficiency requires that algorithms operate in time polynomial in the *logarithm* of $1/\varepsilon$.

REFERENCES

[AHU] A. V. Aho, J. E. Hopcroft and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[GJ] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability: A Guide to the Theory of* NP-*Completeness*, W. H. Freeman, San Francisco, CA, 1979.

[GJW] M. R. GAREY, D. S. JOHNSON AND H. S. WITSENHAUSEN, *The complexity of the generalized Lloyd-Max problem*, IEEE Trans. Inform. Theory, IT-28 (1982), pp. 255-256.

[HC] Y. C. HO AND T. S. CHANG, *Another look at the nonclassical information problem*, IEEE Trans. Automat. Control, AC-25 (1980), pp. 537-540.

[HSI] J. HARTMANIS, V. SEWELSON AND N. IMMERMAN, *Sparse Sets in* NP−P: EXPTIME *vs.* NEXPTIME, Proc. 1983 STOC Conference, 1983, pp. 382-391.

[HU] J. E. HOPCROFT AND J. D. ULLMAN, *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley, Reading, MA, 1979.

[Ko] K.-I. KO, *On the computational complexity of ordinary differential equations*, Inform. and Control, 58 (1984), pp. 157-194.

[LA] W. S. LEVINE AND M. ATHANS, *On the determination of optimal output feedback gains for linear multivariable systems*, IEEE Trans. Automat. Control, AC-15 (1970), pp. 44-48.

[MR] J. MARSCHAK AND R. RADNER, *The Economic Theory of Teams*, Yale Univ. Press, New Haven, CT, 1972.

[NY] A. S. NEMIROVSKY AND D. B. YUDIN, *Problem Complexity and Method Efficiency in Optimization*, John Wiley, New York, 1983.

[Pa] C. H. PAPADIMITRIOU, *Games against nature*, Proc. 1983 IEEE Conference on Foundations of Computer Science; J. Comput. System Sci. (1986), to appear.

[PS] C. H. PAPADIMITRIOU AND K. STEIGLITZ, *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, NJ, 1982.

[PT] C. H. PAPADIMITRIOU AND J. N. TSITSIKLIS, *On the complexity of designing distributed protocols*, Inform. and Control, 53 (1982), pp. 211-218.

[Ra] R. RADNER, *Team decision problems*, Ann. Math. Statist., 33 (1962), pp. 857-881.

[TW] J. F. TRAUB AND H. WOZNIAKOWSKI, *A General Theory of Optimal Algorithms*, Academic Press, New York, 1980.

[TWW] J. F. TRAUB, G. W. WASILKOWSKI AND H. WOZNIAKOWSKI, *Information, Uncertainty, Complexity*, Addison-Wesley, Reading, MA, 1983.

[TA] J. N. TSITSIKLIS AND M. ATHANS, *On the complexity of decentralized decision making and detection problems*, IEEE Trans. Automat. Control, AC-30 (1985), pp. 42-50.

[Ts] J. N. TSITSIKLIS, *Problems in decentralized decision making and computation*, Ph.D. Thesis, Dept. Electrical Engineering and Computer Science, Massachusetts Inst. Technology, Cambridge, MA, 1984.

[WV] J. C. WALRAND AND P. VARAIYA, *Optimal causal coding-decoding problems*, IEEE Trans. Inform. Theory, IT-29 (1983), pp. 814-820.

[Wi] H. S. WITSENHAUSEN, *A counterexample in stochastic optimum control*, this Journal, 6 (1968), pp. 138-147.