On the NP-Completeness of Finding an Optimal Strategy in Games with Common Payoffs

Francis Chu* Joseph Halpern*

Department of Computer Science
Upson Hall, Cornell University
Ithaca, NY 14853-7501, USA

{fcc,halpern}@cs.cornell.edu

November 14, 2000

Keywords: computational complexity, distributed systems, game theory

Abstract

Given a finite two player game with common payoffs (i.e., the players have completely common interests), we show that the problem of determining whether there exists a joint strategy whose expected joint payoff is at least $(1, 1)$ is NP-complete.

1 Introduction

There is a great deal of overlap between game theory and distributed computing. In particular, both are concerned with dealing with uncertainty in multi-agent systems. However, there are differences as well. In distributed computing we are concerned with, among other things, fault tolerance and scaling effects (what happens as the number of agents in the system gets large). On the other hand, distributed computing tends to ignore strategic concerns, which are perhaps the primary focus of game theory. Issue like utilities (payoffs) are rarely discussed in distributed computing. In this paper, we take a game-theoretic viewpoint to a significant distributed computing problem: Free Flight.

In recent years, the Federal Aviation Administration (FAA) has introduced the concept of *Free Flight*, which will decentralize the National Airspace System (NAS). Currently, the NAS is a centralized command-and-control system between pilots and air traffic controllers. Free Flight will change it to a distributed system that allows pilots, whenever practical, to choose their own route, instead of following pre-assigned routes. One reason for this move on the part of the FAA is because the annual air traffic rate is expected to grow by 3–5% for at least the

*Work supported in part by NSF under grant IRI-96-25901.

next 15 years, and the current airspace architecture and management will not be able to scale. See `http://www.faa.gov/freeflight/ff_ov.htm` for more details on Free Flight.

Since a pilot may plot his own course instead of taking a pre-assigned route under Free Flight, the pilot might well want to optimize the route for his payoff function. For example, he might want to minimize flight time (or fuel consumption, or some more complicated quantity). Note that the actual payoff might also depend on Nature (e.g., the flight time depends on wind speed); since Nature is not static, each choice has an expected payoff (where the probabilities are over the possible behaviors of Nature). Thus the pilots can optimize only for expected payoffs, due to the uncertainty about Nature. Furthermore, in general, there are other planes in the sky, the expected payoff for a pilot depends not only on his own choices, but also on the choices of other pilots. Thus we may view the pilots participating in Free Flight as players in a game, since the payoff of each pilot is determined, in general, by the joint action of all the pilots in conjunction with Nature.

While pilots are no longer obligated to follow routes pre-assigned by the FAA during Free Flight, they might not want to act completely independently, since there might be incentives to cooperate. For example, pilots from the same organization (e.g., airline, shipping company, etc.) might well want to coordinate their flights so as to optimize the payoff function of the organization itself, since what is optimal for a particular flight might not be the best thing to do for the organization as a whole. (Indeed, in the FAA documentation, the term "user" refers not only to individual pilots but organizations as well.) In this case, we may see all the pilots in a organization as having the same payoff function (i.e., their payoff functions are simply the payoff for the organization itself).

This type of of situation, where the agents have a common payoff because they are members of the same organization (or because their payoff can be taken to be the organization's payoff) arises frequently in distributed systems applications. In games of this sort, it turns out that there is an optimal joint strategy—one that gives all the agents at least as high a payoff (individually) as any other joint strategy. This optimal joint strategy is deterministic and is both Pareto optimal and a Nash equilibrium.

An obvious goal is thus to compute the optimal joint strategy. Here, unfortunately, is where the scalabilitiy scalability concern bites. We show that the problem of computing the optimal joint strategy is NP-complete. (More precisely, we show that it is NP-complete to determine whether there is a joint strategy that nets the players a payoff of at least k for a fixed k.)

The rest of the paper is organized as follows. In Section 2 we give a more formal definition of what we mean by a game and review some standard terminology of game theory [1]. In Section 3, we prove the main NP-completeness result of the paper. In Section 4 we offer some concluding remarks.

2 The Formal Model

There are many ways to model games; we will model games in a way that is tailored to our intended application, which means it will differ from some standard textbook approaches. For our purposes, an n-player game, G_n, is a $2n+3$-tuple $(W, A_1, \ldots, A_n, O_1, \ldots, O_n, \Pr, \text{OBS}, \text{PAY})$ such that

- W is the set of possible worlds,
- A_i is the set of possible actions of agent i,
- O_i is the set of possible observations of agent i,
- Pr is the probability distribution on W,
- $\text{OBS}(w, i) \in O_i$ is the observation agent i makes in world w, and
- $\text{PAY}(w, (a_1, \ldots, a_n)) \in \mathbf{R}^n$ is the joint payoff of the joint action $(a_1, \ldots, a_n)$ in world w.

Note that each world $w \in W$ determines the observation each agent makes (via OBS) and the joint payoff given any joint action (via PAY). We assume that the game (i.e., the tuple) is common knowledge among the agents. Intuitively, if $\text{PAY}(w, (a_1, \ldots, a_n)) = (b_1, \ldots, b_n)$, then b_i is the payoff to agent i if the world is w and agent j performs action a_j, $j = 1, \ldots, n$. A *game with common payoffs* is one where all joint payoffs have the form $(b, \ldots, b)$: i.e., each agent gets the same payoff.

Each agent i decides what action to take based on his observation and his strategy. Formally, a *strategy* for agent i is a function from O_i to A_i. That is, for each observation, the strategy prescribes an action. (Note that this is a *deterministic* strategy. In general, *randomized* strategies are also possible. However, as we show shortly, in the games with common payoffs that we are interested in, there is no loss of generality in considering only on deterministic strategies.) A *joint strategy* is an n-tuple $(S_1, \ldots, S_n)$ such that S_i is the strategy for agent i. (For brevity, we will sometimes write $(x_1, \ldots, x_k)$ as $\vec{x}$.) Given a joint strategy $\vec{S}$, we can compute its expected joint payoff

$$\mathbf{E}^{\text{PAY}}(\vec{S}) = \sum_{w \in W} \Pr(w) \cdot \text{PAY}(w, (S_1(\text{OBS}(w, 1)), \ldots, S_n(\text{OBS}(w, n)))).$$

Note that $S_i(\text{OBS}(w, i))$ is the action of agent i in world w. The above is a slight abuse of notation, since random variables and their expected values are typically real numbers. What we really have is n random variables and we are computing each of their expected values, and we are expressing the computation in terms of vector operations. (Given the way scalar multiplication and vector addition work, we get exactly what we want.) If the expected joint payoff of $\vec{S} = (S_1, \ldots, S_n)$ is $(p_1, \ldots, p_n)$, we say that the expected payoff of agent i is p_i with respect to $\vec{S}$, which we denote by $\mathbf{E}_i^{\text{PAY}}(\vec{S})$. We can define a partial ordering $\preceq$ on the joint strategies based on their expected joint payoffs as follows: $\vec{S} \preceq \vec{S}'$ iff $\mathbf{E}_i^{\text{PAY}}(\vec{S}) \leq \mathbf{E}_i^{\text{PAY}}(\vec{S}')$ for all i. Note that if $\vec{S} \preceq \vec{S}'$, then $\vec{S}'$ is at least as good as $\vec{S}$ for all the agents, since no one would be worse off by switching to $\vec{S}'$. A joint strategy is *Pareto optimal* if its expected joint payoff is maximal with respect to $\preceq$.

As is well known, finite games always have Pareto optimal joint strategies and always have strategies that are in Nash equilibrium (if we allow randomized strategies). However, in general, the set of Pareto optimal strategies may be disjoint from the strategies in Nash equilibrium. This is not the case with common payoffs. Indeed, with common payoffs, $\preceq$ is a linear (pre)ordering, so there is a joint strategy with the highest expected (joint) payoff. (There may be more than one, since there could be ties.) Thus, there is always a deterministic joint strategy that dominates every joint strategy (with respect to the $\preceq$ ordering); such a joint

strategy is *a fortiori* Pareto optimal and a Nash equilibrium, since no agent can do (strictly) better with any other joint strategy. Moreover, since the payoff of a randomized strategy is a convex combination of the payoffs of deterministic strategies, there can be no randomized strategy with a higher payoff. Thus, in games with common payoffs, a (deterministic) joint strategy with highest expected payoff is both a Nash equilibrium and Pareto optimal.

As suggested in the introduction, we can view Free Flight as a game, with the pilots as the players. Each pilot acts according to his strategy and his observation. We can assume that the observation includes his position, velocity, wind speed, and other such relevant factors (which, after all, the pilot learns by observing his cockpit instruments). Of course, the space of possible observations is enormous, so computing the optimal strategy will be nontrivial. Indeed, as we show in the next section, it is NP-complete.

3 The NP-Completeness Result

We now show the problem of determining the optimal strategy in a game with common payoffs is NP-complete. For simplicity, we restrict to 2-player games. It will be clear from the proof that the result holds for arbitrary n-player games. For technical reasons, we further restrict to games whose probabilities and payoffs are rational numbers, and that the rational number $\frac{p}{q}$ is represented by pairs of integers (p, q), where $q > 0$. (This ensures that the games are finitely *described*, so that we can finish reading their descriptions in finite amount of time. If we are truly pedantic, we should actually fix an encoding of the games, but we will not get into such tedious details. Readers not familiar with NP-completeness results and the techniques by which they are established may wish to consult [2] for an introduction.) If the game is a common payoff game, we write $\text{PAY}(w, (a_1, \ldots, a_n)) = p$, where p is the common payoff. We then take $\mathbf{E}^{\text{PAY}}(\vec{S})$ to be a single number, rather than a tuple (all of whose components are identical).

The problem of finding the optimal strategy is an optimization problem. So that it fits into the standard framework of complexity classes, we convert it to a decision problem:

UTIL: Given a two player game (with common payoff), is there a joint strategy $\vec{S}$ such that $\mathbf{E}^{\text{PAY}}(\vec{S}) \geq 1$?

The following theorem shows that UTIL is NP-complete.

Theorem 3.1: *UTIL is NP-complete.*

Proof: UTIL is clearly in NP, since it suffices to guess a joint strategy and verify that the expected joint payoff is indeed at least 1. To show that UTIL is NP-hard, we reduce 3SAT to UTIL. Recall that an *instance* of 3SAT consists of a Boolean formula which is a conjunction of *clauses*, each of which is a disjunction of three *literals*, each of which is either a (Boolean) variable or the negation of a (Boolean) variable. An instance of UTIL is simply a (finite) two player game (with common payoff). A *positive instance of 3SAT* is a formula (of 3SAT) which is satisfiable (i.e., there is a truth assignment that satisfies all the clauses) while a *positive instance of UTIL* is a (finite) 2-player game (with common payoff) for which there exists a joint strategy whose expected payoff is at least 1. Recall that to show that 3SAT reduces to UTIL,

we need to give a polynomial time transformation f that maps instances of 3SAT to instances of UTIL such that φ is a positive instance of 3SAT iff $f(\varphi)$ is a positive instance of UTIL.

Let φ be an instance of 3SAT; let n be the number of clauses in φ and let m be the number of variables in φ. (Note that $m \leq 3n$, since each clause contains at most three variables.) Let $z_1, \ldots, z_m$ be the variables and $C_1, \ldots, C_n$ be the clauses. Thus $\varphi = C_1 \wedge \cdots \wedge C_n$, where $C_i = (\ell_{i,1} \vee \ell_{i,2} \vee \ell_{i,3})$ is a clause, where $\ell_{i,j}$ is a literal. If $\ell_{i,j} = z_k$ or $\ell_{i,j} = \neg z_k$, we say that z_k is the variable *associated with* $\ell_{i,j}$; we denote the variable associated with a literal ℓ by $v(\ell)$. Let $f(\varphi)$ be the following game:

- $W = \{w_{i,j} : 1 \leq i \leq n \text{ and } 1 \leq j \leq 3\}$ (the world $w_{i,j}$ corresponds to the literal $\ell_{i,j}$),
- $A_1 = \{\mathrm{T}, \mathrm{F}\}$,
- $A_2 = \{1, 2, 3\}$,
- $O_1 = \{z_1, \ldots, z_m\}$,
- $O_2 = \{C_1, \ldots, C_n\}$, and
- $\Pr(w_{i,j}) = \frac{1}{3n}$ for all i, j.
- $\textsc{obs}(w_{i,j}, 1) = v(\ell_{i,j})$ and $\textsc{obs}(w_{i,j}, 2) = C_i$; that is, in world $w_{i,j}$, the observation of agent 1 is $v(\ell_{i,j})$ and the observation of agent 2 is C_i.
- PAY is defined as follows:

$$\textsc{pay}(w_{i,j}, (\mathrm{T}, i)) = \begin{cases} 3 & \text{if } \ell_{i,j} = v(\ell_{i,j}) \\ 0 & \text{if } \ell_{i,j} = \neg v(\ell_{i,j}), \end{cases} \quad \textsc{pay}(w_{i,j}, (\mathrm{F}, i)) = \begin{cases} 0 & \text{if } \ell_{i,j} = v(\ell_{i,j}) \\ 3 & \text{if } \ell_{i,j} = \neg v(\ell_{i,j}). \end{cases}$$

Note that the size of $f(\varphi)$ is linear in the number of clauses, so it is easy to implement f in polynomial time. We now turn to an example.

> Let $\varphi = (z_1 \vee \neg z_2 \vee z_3) \wedge (z_2 \vee z_4 \vee \neg z_1)$. Then $f(\varphi) = (W, A_1, A_2, O_1, O_2, \Pr, \textsc{obs}, \textsc{pay})$, where
>
> - $W = \{w_{1,1}, w_{1,2}, w_{1,3}, w_{2,1}, w_{2,2}, w_{2,3}\}$;
> - $A_1 = \{\mathrm{T}, \mathrm{F}\}$;
> - $A_2 = \{1, 2, 3\}$;
> - $O_1 = \{z_1, z_2, z_3, z_4\}$;
> - $O_2 = \{(z_1 \vee \neg z_2 \vee z_3), (z_2 \vee z_4 \vee \neg z_1)\}$;
> - $\Pr(w_{i,j}) = \frac{1}{6}$;
> - OBS is defined as follows:
>   - $\textsc{obs}(w_{1,1}, 1) = z_1$, $\textsc{obs}(w_{1,2}, 1) = z_2$, $\textsc{obs}(w_{1,3}, 1) = z_3$, $\textsc{obs}(w_{2,1}, 1) = z_2$, $\textsc{obs}(w_{2,2}, 1) = z_4$, $\textsc{obs}(w_{2,3}, 1) = z_1$
>   - $\textsc{obs}(w_{1,*}, 2) = (z_1 \vee \neg z_2 \vee z_3)$, $\textsc{obs}(w_{2,*}, 2) = (z_2 \vee z_4 \vee \neg z_1)$

- PAY is defined as follows:

	1	2	3
T	3	0	0
F	0	0	0

$w_{1,1}$

	1	2	3
T	0	0	0
F	0	3	0

$w_{1,2}$

	1	2	3
T	0	0	3
F	0	0	0

$w_{1,3}$

	1	2	3
T	3	0	0
F	0	0	0

$w_{2,1}$

	1	2	3
T	0	3	0
F	0	0	0

$w_{2,2}$

	1	2	3
T	0	0	0
F	0	0	3

$w_{2,3}$

The intuition behind f is easy to explain. Since we want to map positive instances of 3SAT to positive instances of UTIL, we want to establish a correspondence between satisfying truth assignments and joint strategies whose expected payoff is at least 1, so that there exists a satisfying assignment iff there exists a joint strategy with the desired expected payoff. The observations of agent 1 are variables and the actions of agent 1 are truth values; thus, the strategies of agent 1 are truth assignments. Intuitively, we can think of agent 1 as a "guesser" (who guesses a truth assignment) and agent 2 as a "verifier" (who tries to verify that the guess indeed satisfies the formula). To try to verify that a guess satisfies the formula, agent 2 picks a literal from each clause and sees if that literal is satisfied by the truth assignment. If so, the (joint) payoff in the particular world corresponding to the literal is 3 (recall that the world $w_{i,j}$ corresponds to the literal $\ell_{i,j}$); otherwise, the joint payoff is 0. It is easy to check that if every literal agent 2 picks is indeed satisfied by the truth assignment, then the expected payoff is exactly 1; otherwise the expected payoff is less than 1.

Thus, if there exists a satisfying truth assignment, then there exists a joint strategy with the desired expected payoff: let the strategy of agent 1 be a satisfying truth assignment and the strategy of agent 2 be a function that picks a literal from each clause that is satisfied by the truth assignment. On the other hand, if there is no satisfying truth assignment, then no matter which joint strategy is chosen, the expected payoff is (strictly) less than 1. Thus φ is a positive instance of 3SAT iff $f(\varphi)$ is a positive instance of UTIL, so we are done. ∎

Although we restricted to 2-player games in Theorem 3.1, it should be clear that the analogous problem is also NP-complete for n-player games. The upper bound is clear, since it suffices to guess a joint strategy just as before. And it is easy to modify our lower bound proof to deal with n-player games; we leave details to the reader.

There is one technical point worth observing. Say that player i *considers* w_j *possible in* w_k iff $\text{OBS}(w_k, i) = \text{OBS}(w_j, i)$ (i.e., player i makes the same observation in both worlds). Intuitively, if a player considers many worlds possible, he has a lot of uncertainty. Note that in the game constructed in the proof of Theorem 3.1, player 2 considers only three worlds possible in any given world (since there are only three literals in each clause) while player 1 may consider many worlds possible in some worlds (since a variable may appear in many clauses). Is it necessary for one of the players to have much uncertainty for our result to hold? It turns out that the problem remains NP-complete even if player 1 considers at most three worlds possible in each world as well. The reason that player 1 may consider many worlds possible is because a variable may occur in many clauses. It is easy to convert a formula φ in which a variable may occur many times to a formula φ' in which each variable occurs in at most three clauses such that

φ' is satisfiable iff φ is satisfiable and φ' can be constructed in polynomial time. (This can be done via a technique very similar to the reduction of SAT to 3SAT.) Thus the problem remains NP-complete even if both players consider at most three worlds possible in each world. While we know that 2SAT is in P and that SAT restricted to formulas in which each variable occurs at most twice is in P, we have not investigated whether UTIL remains NP-complete if both players consider at most two worlds possible in each world. Clearly if both players consider only one world possible (i.e., they have perfect information), then we can find the optimal joint strategy in linear time.

4 Conclusion

We have shown that the problem of determining the optimal strategy in a common payoff game is NP-complete. (More precisely, it follows from our results that we can find a strategy within ϵ of optimal, for a fixed ϵ, using a finite number of calls to an NP-oracle.) Essentially the same argument shows that it is NP-complete to find a strategy which is within a fixed fraction of optimal. Thus, we cannot even find approximately optimal strategies in polynomial time. What does this say about problems such as Free Flight? Should we necessarily give up on finding optimal strategies? Recent successes in finding solutions to NP-complete problems [3, 4] suggest that there may be some reason to hope; NP-complete problems may not be so infeasible in practice. Of course, further research needs to be done to see if problems such as Free Flight can in fact be represented in a reasonable way as a game that is in practice soluble.

Acknowledgements

We would like to thank Ken Birman for bringing Free Flight to our attention.

References

[1] D. Fudenberg and J. Tirole. *Game Theory.* MIT Press, Cambridge, Mass., 1991.

[2] M. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* W. Freeman and Co., San Francisco, Calif., 1979.

[3] T. Hogg, B.A. Huberman, and C.P. Williams, editors. *Artificial Intelligence*, volume 81. Elsevier, 1996. Special Issue on *Phase Transitions and Complexity.*

[4] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky. Typical-case complexity results from a new type of phase transition. *Nature*, 400(8):133–137, 1999.