CMSC 858F: Algorithmic Game Theory
Fall 2010
Achieving Byzantine Agreement and Broadcast against Rational Adversaries

Instructor: Mohammad T. Hajiaghayi
Scribe: Adam Groce, Aishwarya Thiruvengadam, Ateeq Sharfuddin

December 8, 2010

1 Overview

In this presentation, we describe byzantine agreement and broadcast with respect to rational adversaries with particular preferences on the output of the honest players.

2 Introduction

The Byzantine Generals Problem was introduced in 1982 by Leslie Lamport, Robert Shostak, and Marshall Pease in 1982 [3] as a way to describe, in abstract terms, their earlier work on distributed computation among fallible processors [4]. Lamport, Shostak, and Pease introduced two protocols in their paper [3]: Byzantine Agreement (BA) and Broadcast. Let

- n be the number of generals in the Byzantine army,
- t be the number of traitors,
- $v(i)$ be the input of player i.
- and $w(i)$ be the output of player i.

Each player uses some mechanism to combine $v(1), v(2), v(3), \ldots, v(n)$ into a single plan of action.

Definition 1 (Broadcast) *Assume player* i *acting as the* sender *and sending his input* $v(i)$ *to the remaining* $n-1$ receivers. *A protocol is a* broadcast protocol tolerating t traitors *if the following two conditions hold for any adversary controlling at most* t *traitors:*

IC1: *All honest players receive the same message.*

IC2: *If the sender is loyal, then every honest player outputs the sender's input.*

Definition 2 (Byzantine Agreement) *Assume each player* $\mathtt{i}$ *initially has input* $v(\mathtt{i})$*. A protocol is a* Byzantine agreement protocol tolerating $\mathtt{t}$ traitors *if the following conditions hold for any adversary controlling at most* $\mathtt{t}$ *traitors:*

IC1': *All honest players act on the same order. That is, if* $\mathtt{i}$ *and* $\mathtt{j}$ *are honest players, then* $w(\mathtt{i}) = w(\mathtt{j})$*.*

IC2': *If all honest players begin with the same input value, i.e.* $v(\mathtt{i}) = \mathtt{c}\ \forall \mathtt{i}$*, then all honest players have that same value as output, i.e.* $w(\mathtt{i}) = \mathtt{c}\ \forall \mathtt{i}$*.*

While we only consider cases where all pairwise communication is possible, the Byzantine agreement problem can be defined on graphs lacking full connectivity.

3 Rational adversaries

3.1 Prior consideration

In traditional cryptography, the goal is always to prove that the security conditions are satisfied no matter what algorithm the adversary chooses to use. However, in some cases it might be desirable to weaken this definition. One such weakening is to borrow from game theory and introduce the idea of rational adversaries. In this case, it is assumed that the adversary has some particular goal rather than just to arbitrarily violate security conditions. The goal is to show that in this case protocols can be developed that are superior to those that are possible against generic malicious adversaries.

3.2 Byzantine agreement and broadcast

We now apply the idea of rational adversaries to Byzantine agreement and broadcast. We assume that the adversary has some goal that it is trying to achieve. Once a utility function has been chosen, a definition of security is immediately implied:

Definition 3 (Security against rational adversaries) *A protocol for Byzantine agreement or broadcast is secure against rational adversaries with a particular utility function if for any adversary* A_1 *against which the protocol does not satisfy the security conditions there is another adversary* A_2 *such that:*

- *When the protocol is run with* A_2 *as the adversary, all security conditions are satisfied.*
- *The utility achieved by* A_2 *is greater than that achieved by* A_1*.*

3.3 Utility definitions

We limit ourselves to protocols that are attempting to broadcast or agree on a single bit. In our first definitions, we will consider all output to be one of the following types:

- All honest players output 1.
- All honest players output 0.
- Honest players have disagreeing output.

This is, of course, a substantial simplification of possible outcomes. For example, some sorts of disagreement could be preferred over any unanimous output while other types are disliked. Nevertheless, these outcomes capture a meaningful portion of potential outcome variation.

Definition 4 (Strict preferences) *An adversary with strict preferences is one that will maximize the likelihood of its first-choice outcome at all costs. For a particular strategy, let* $\mathfrak{a}_1$ *be the probability of the adversary achieving its first-choice outcome (of the three listed above) and* $\mathfrak{a}_2$ *be the probability of achieving the second choice outcome. Let* $\mathfrak{b}_1$ *and* $\mathfrak{b}_2$ *be the same probabilities for a second potential strategy. We say that the first strategy is preferred if and only if* $\mathfrak{a}_1 > \mathfrak{b}_1$ *or* $\mathfrak{a}_1 = \mathfrak{b}_1$ *and* $\mathfrak{a}_2 > \mathfrak{b}_2$.

Definition 5 (Linear utility) *An adversary with linear utilities has its utility defined by*

$$\mathfrak{utility} = \mathfrak{u}_1 \Pr[\text{players output 0}] + \mathfrak{u}_2 \Pr[\text{players output 1}] + \mathfrak{u}_3 \Pr[\text{players disagree}] \quad (1)$$

where $\mathfrak{u}_1$, $\mathfrak{u}_2$, *and* $\mathfrak{u}_3$ *are arbitrary parameters. One strategy is preferred over another if the resulting utility is higher.*

In the next definition, we stop considering all disagreements to be equal:

Definition 6 (0-preferring) *A 0-preferring adversary is one for which*

$$\mathfrak{utility} = \mathfrak{E}[\text{number of honest players outputting 0}]. \quad (2)$$

4 Results

4.1 Equivalence of Broadcast and Byzantine Agreement

The equivalence between Broadcast and Byzantine agreement is not as straightforward in the rational adversary setting. Assume that we have a broadcast protocol that is secure against adversaries with strict preferences in the 0s>1s>disagreement order. To construct a Byzantine agreement protocol, we tell each player to use this broadcast protocol to send their values to each other

and then take the majority of values received. If the broadcast protocol's security is not violated, this reduction is sound. However, we can no longer assume that its security wouldn't be violated.

Consider the case where there are five players, P_1 and P_2 with input 0 and P_3, P_4 and P_5 with input 1.

However, imagine that the adversary could break the broadcast protocols. It is possible that they could break the protocols in the particular way.

Theorem 1 *Assume that a Byzantine agreement protocol exists that is secure against rational adversaries with a particular set of preferences on the probability distribution of possible outcomes. (We note that this preference set must be independent of the input honest players have.) A protocol can be constructed for broadcast that is secure against the same type of rational adversary.*

Proof: The sender simply sends his input to all players. Then they all (including the sender) execute the Byzantine agreement protocol using those received values as input. The players use the output of this Byzantine agreement protocol as the output of the broadcast protocol. Because the output of the broadcast protocol will be identical to that of the Byzantine agreement protocol, the adversary will never benefit by breaking the Byzantine agreement protocol in a way that is contrary to his preferences on the larger broadcast protocol. Because of this, we can assume that the Byzantine agreement protocol works correctly, and the reduction then proceeds without problem. ■

4.2 A simple protocol

We begin with the case that the adversary has strict preferences with ordering 0s>1s>disagreement. Protocol 1 is effective in this case.

Protocol 1

1. Each player sends his input to every other player.
2. Each player outputs the majority of all the inputs that he has received. If there is no strict majority, output 0.

Theorem 2 *Protocol 1 achieves Byzantine agreement in the presence of a rational adversary with strict preferences and ordering 0s>1s>disagreement when that adversary controls* t *players, where* $t < n/2$.

Proof: If all the honest players held the same input, the protocol would terminate with the honest players agreeing on that input despite what the adversary says. This is also true of not all honest players agreed, but those with a given input make up a majority of players. On the other hand, if the honest players with

input 1 do not form a majority, it is in the adversary's interests to send 0's to all honest players as this allows him to guarantee his first choice output. Thus, the adversary will never choose to force disagreement. Since disagreement will never happen and unanimous honest parties always receive the correct output, the security conditions are met and the protocol is valid. ■

Theorem 3 *Protocol 1 achieves Byzantine agreement in the presence of a rational adversary who controls* t *players where* $t < n/2$ *as long as the adversary's most preferred outcome is unanimous output of 0s by all players.*

Proof: Let a be the number of honest players with input 1. If $a > n/2$ the adversary cannot influence the outcome, and all honest players will output 1. This assures that when input is unanimously 1, the correct output is given. In all other cases, the adversary can act as if its controlled players were honest with input 0 and force all players to output 0. Since this is its most-preferred outcome, it will not use any strategy that does not guarantee this result. Since these two cases are exhaustive, disagreement will never occur. Since the outcomes that result are also consistent with the security conditions for Byzantine agreement, the protocol achieves Byzantine agreement correctly. ■

4.3 A general solution

The natural next task is to design another simple protocol that will work for the disagreement>0s>1s case. However, we have not succeeded at finding such a simple and efficient protocol secure against this adversary. Instead, we conclude by presenting a very powerful protocol that solves the broadcast problem for this and many other adversary preferences (including many not explicitly defined here). This protocol is based on *detectable broadcast*, which was defined by Fitzi et al. [2]. In addition to the definition, they give a protocol that realizes their definition. This protocol is much less efficient than the other solutions discussed previously. It is, however, a very powerful feasibility result.

We begin by defining detectable broadcast.

Definition 7 (Detectable broadcast) *A protocol for detectable must satisfy the following three conditions:*

- ***Correctness:*** *All honest players either abort or accept and output 0 or 1. If any honest player aborts, so does every other honest player. If no honest players abort then the output satisfies the security conditions of broadcast.*
- ***Completeness:*** *If all players are honest, all players accept (and therefore achieve broadcast without error).*
- ***Fairness:*** *If any honest player aborts then the adversary receives no information about the sender's bit.*

The protocol given by [2] requires $t+5$ rounds and $O(n^8(\log n + k)^3)$ total bits of communication, where k is a security parameter. This compares with one round and n^2 bits for the simple protocols we have already given.

We now use detectable broadcast to achieve broadcast against rational adversaries with strict preferences and a disagreement>0s>1s preference ordering. This is done with Protocol 2:

Protocol 2

1. Run the detectable broadcast protocol with the same sender and input.
2. If no abort occurs, output the result of the detectable broadcast protocol. If abort does occur, output 1.

Protocol 2 is obviously secure. What is most impressive is that the detectable broadcast protocol in [2] and therefore our resulting protocol is actually secure as long as $t < n$. This means that for the broadcast case (but not Byzantine agreement) we can do even better than we did in the previous section (though with less efficiency).

We note also that a similar protocol can be constructed for Byzantine agreement. The protocol that [2] construct has two steps. The first step establishes a public key infrastructure. It is in this step that an abort is possible. Once a public key infrastructure has been established, broadcast (for $t < n$) and Byzantine agreement (for $t < n/2$) are both easily achieved.

References

[1] Matthias Fitzi, Daniel Gottesman, Martin Hirt, Thomas Holenstein, and Adam Smith, *Detectable byzantine agreement secure against faulty majorities*, Proceedings of the twenty-first annual symposium on Principles of distributed computing (New York, NY, USA), PODC '02, ACM, 2002, pp. 118–126.

[2] L. Lamport, R. Shostak, and M. Pease, *The byzantine generals problem*, ACM Trans. Program. Lang. Syst. **4** (1982), no. 1, 382–401.

[3] M. Pease, R. Shostak, and L. Lamport, *Reaching agreement in the presence of faults*, J. ACM **27** (1980), 228–234.