Characterizing the Decidability of Finite State Automata Team Games with Communication

Michael Coulombe
Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA
mcoulomb@mit.edu

Jayson Lynch
Cheriton School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada
jayson.lynch@uwaterloo.ca

In this paper we define a new model of limited communication for multiplayer team games of imperfect information. We prove that the Team DFA Game and Team Formula Game, which have bounded state, remain undecidable when players have a rate of communication which is less than the rate at which they make moves in the game. We also show that meeting this communication threshold causes these games to be decidable.

1 Introduction

Deciding optimal play in multiplayer games of incomplete information is known to be an undecidable problem [11, 10]. This includes games where the state space is bounded, a surprising result first shown of a collection of abstract computation games [6] that has been extended to generalized versions of real games, like Team Fortress 2 and Super Smash Bros [5]. However, past work has relied on the complete inability of teammates to communicate during the game, which is often not a realistic assumption. In this paper we study deterministic models of communication between players in two of these computation games, the Team DFA Game and the Team Formula Game, and show a sharp change in computational complexity based on whether players are able to eventually communicate all of their moves or only able to communicate a constant fraction.

One motivation for this model is a better understanding of real world games. Many team games played in-person naturally permit free form communication between teammates to coordinate their actions, and online multiplayer video games often provide communication channels such as voice-chat, text, and emotes to simulate this in-person environment. These include many FPS games such as Team Fortress and Left4Dead, MOBAs such as DOTA2 and League of Legends, and RTS games such as Starcraft and Age of Empires. Some of these examples have drawn research interest in AI/ML [14, 3] as well as computational complexity[13, 5]. The real-time nature of these games ensures that communication channels are bounded; however, modeling free form communication, as well as efficiently implementing meaningful player choices in a real-time setting, makes it difficult to analyze these games with these communication features enabled.

Outside of the team setting, communication is a central aspect of many other games. For example, in the cooperative card game Hanabi players are unable to see their own hands of cards, but this information is visible to everyone else. In addition, players are not allowed to communicate except through actions in the game, one of which allows players to reveal partial information about what is in another players hand. A perfect information version of Hanabi was shown to be NP-complete [1]. The Crew: Quest for

D. Della Monica and P. Ganty (Eds.): 13th International Symposium on Games, Automata, Logics and Formal Verification (GandALF 22)
EPTCS 370, 2022, pp. 213–228, doi:10.4204/EPTCS.370.14

Planet Nine is a cooperative trick taking card game which uses limited communication between players as a core mechanic. The complexity of this game was also studied in the perfect information setting [12]. Under the limited information setting, containment in NP for both of these games is not obvious, and we see a need for models of player communication in games. Other examples of cooperative boardgames with limited communication channels between players include Mysterium, The Mind, and Magic Maze.

Other games may limit communication simply with time pressure in the game. Examples of fully cooperative games with imperfect information that use time pressure to limit coordination and communication include Space Alert, 5-Minute Dungeon, Keep Talking and Nobody Explodes, and Spaceteam.

Multi-agent, imperfect information games are also a topic of interest in Reinforcement Learning. In [4] algorithms are developed to address team extensive form games of imperfect information where communication is allowed at certain points during gameplay, with Bridge and collusion among some players between hands in poker being the motivating examples. Other work considers Sequential Social Dilemmas, a type of iterated economic game where in any given instant a player is incentivized towards non-cooperative behavior, but cooperative strategies can obtain higher payoff over the game as a whole. Learning algorithms for these models both with and without explicit bounded communication channels were studied in [8]. Purely cooperative settings have also been of interest [7].

One major achievement was human level performance on a limited version of DOTA2, a MOBA-style video game [3]. These are real-time, team games with partially observable state. Although both text and voice chat are typically allowed in professional play, the AI system did not utilize these explicit communication mechanisms. The board game Diplomacy, while not explicitly a team game, features coordination and temporary alliances between players as a core game dynamic. This game has also seen interest as a new challenge in the AI community, but focusing on No-Press Diplomacy which does not allow explicit communication between players [9, 2].

These examples of both AI and computational complexity research which considers games with cooperation and communication motivate, but frequently ignore the important role of communication in these games, motivates this paper.

Paper Organization. In Section 1.2 we formally define our model of communication for the Team DFA Game. In Section 2 we prove undecidability for a few simple communication patterns to help build intuition for the techniques used in the next section. In Section 3 we prove our main undecidability result for Team DFA Games with Communication. In Section 4 we prove the game becomes decidable when either both players can communicate all information about their moves, or one player receives no information but can communicate all of their moves to their teammate. In Section 5 we show that analogous algorithms and undecidability results hold for Team Formula Games.

1.1 Team DFA Game

The Team DFA Game is a bounded state, two-vs-one multiplayer game, defined by [5] as a simplification of the undecidable Team Computation Game [6]. It involves a team of two players, the $\exists$ players, and an adversary, the $\forall$ player, who take turns sending a bit to a deterministic finite automaton (DFA). Each team's goal is to put the DFA into any of a set of winning final states for their team. The $\forall$ player knows the moves of the $\exists$ players and thus the state of the DFA; however the $\exists$ players do not know each other's moves, which they must make after learning only one of $\forall$'s moves each turn.

Definition 1. The Team DFA Game (TDG) is a two-versus-one team game. An instance of the game is a DFA $D = (\{0,1\}, Q, q_0, \delta, F = F_\exists \cup F_\forall)$. The existential team $\{\exists_0, \exists_1\}$ competes against the universal team $\{\forall\}$. The game starts with D in state q_0 and each round proceeds as follows:

1. If D's state $q \in F_\exists$ then team existential wins. If $q \in F_\forall$ then team universal wins.
2. $\forall$ learns the state q of D then inputs two bits b_0, b_1 into D.
3. $\exists_0$ learns b_0 then inputs one bit m_0 into D. $\forall$ learns m_0.
4. $\exists_1$ learns b_1 then inputs one bit m_1 into D. $\forall$ learns m_1.

The problem we consider in this paper is: given an instance of the game, does a particular team have a *forced win*? More formally, does there exist a strategy function s_i for each player i on the team, specifying on each turn which move to make based on any information they have learned so far, that when followed will guarantee that this team will win? In this paper, we define the complexity of a game as the complexity of whether a specified team has a forced win in the game, such as in the following:

Theorem 1 (previous work [5])**.** *The Team DFA Game is undecidable.*

A number of variations of this game, all undecidable in the general case, exist. These include Team Computation Game where players give inputs to a Turing machine on a bounded tape [11], Team Constraint Logic Game where players make moves in a partially observable constraint logic graph [6]; and Team Formula Game where players flip the values of Boolean variables trying to satisfy different formulas [10, 6].

1.2 Communication Model

We model communication in the Team DFA Game with a policy that specifies the bandwidth of a dynamic information channel, as one might have due to natural factors (e.g. playing a real-time game with voice chat) or intentional game design (e.g. Hanabi's card-revealing moves) allowing a predictable but bounded amount of player-to-player communication between moves. Specifically, a policy P is a DFA over a unary alphabet with functions $P_{\text{MID}}, P_{\text{END}}$ over its states. In a round of the game in policy state p, $P_{\text{MID}}(p)$ is the number of bits which are exchanged simultaneously between $\exists_0$ and $\exists_1$ after (b_0, b_1) are revealed but before (m_0, m_1) must be determined, and similarly $P_{\text{END}}(p)$ is the number of bits to exchange after (m_0, m_1) are submitted but before the next round starts.

Definition 2. The Team DFA Game with Communication (TDGC) is a game of the existential team $\{\exists_0, \exists_1\}$ versus the universal team $\{\forall\}$, extending the Team DFA Game. An instance of the game is a pair of a game DFA $D = (\{0,1\}, Q, q_0, \delta, F_\exists \cup F_\forall)$ and a policy P, which consists of a policy DFA $(\{1\}, \Pi, p_0, \pi, \emptyset)$ and functions $P_{\text{MID}}, P_{\text{END}} : \Pi \to \mathbb{N} \times \mathbb{N}$. The game starts with D in state q_0 and the policy DFA in state p_0, and each round proceeds with added communication steps as illustrated in Figure 1.

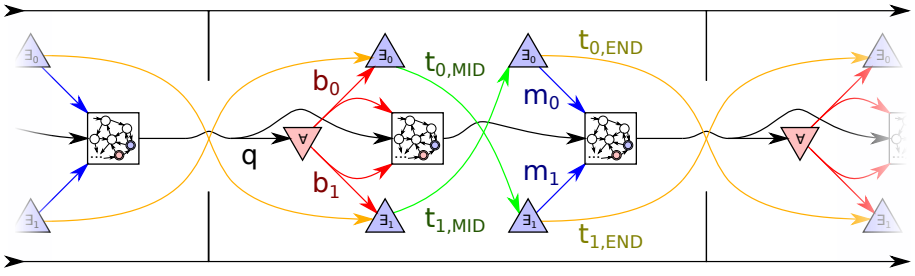

Figure 1: Information flow graph of one round of the Team DFA Game with Communication, including from the previous round and into the next round. New to this game are the mid-round transmissions, $t_{0,\text{MID}}$ and $t_{1,\text{MID}}$, and the end-of-round transmissions, $t_{0,\text{END}}$ and $t_{1,\text{END}}$, which have sizes determined by P_{MID} and P_{END} applied to the policy state.

There are two beneficial aspects of studying policies as DFAs on unary alphabets: bounding the state space allows for the policy to be computable by the mechanics of a bounded-state game (such as the DFA in the Team DFA Game), and giving every state exactly one next state (for the next round of the game) means the bandwidth every round will be known in advance when building our constructions, rather than being dependent on player actions. As a result of this choice, it is important to note that the shape of its state transition graph will always have the form: from the start state, there is an initial chain of unique states (possibly of length zero) that leads to a cycle of periodically-repeating states. Also shown in Figure 2.

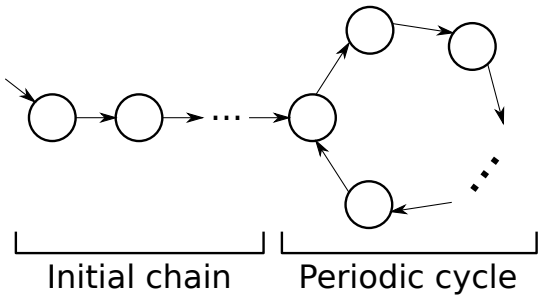

Figure 2: General form of a policy DFA: an initial chain followed by a cycle.

2 Undecidability of Simple Communication Games

In this section, we will explore some basic classes of policies that preserve the undecidability of the Team DFA Game with Communication. Our proof technique is to reduce from the zero-communication Team DFA Game, where we compensate for the message passing by "clogging the channel" with the forced transmission of garbage bits that do not facilitate information sharing. Section 3 builds upon these examples to obtain more general results, however proving the special cases in this section allows us to introduce ideas needed in the full proof and discuss some of the techniques more concretely.

For each class of policies $\mathscr{P}$ below, we will show that given any policy $P \in \mathscr{P}$ and DFA D for playing TDG, we can produce a DFA D' for playing TDGC under P such that the $\exists$ team has a forced win on D with no channel iff they have a forced win on D' given a channel following policy P. As TDG is undecidable, so will be TDGC under any policy $P \in \mathscr{P}$. For simplicity, we consider policies with DFA C_r, a length-r cycle of states $\Pi = \{0, 1, \ldots, r-1\}$ with no initial chain, for arbitrary values of r.

Theorem 2. *TDGC is undecidable with a 1 bit mid-round exchange every $r \geq 2$ rounds: policies P where $P_{\text{MID}}(p) = (1,1)$ if $p \equiv 0 \mod r$, $P_{\text{MID}}(p) = (0,0)$ otherwise, and $P_{\text{END}}(p) = (0,0)$.*

Proof. We construct a DFA D' by first augmenting the state q of D with the state p of C_r. When $p \not\equiv 0 \mod r$, D' simply simulates D for one round. However, when $p \equiv 0 \mod r$, D' instead takes the inputs (b_0, b_1, m_0, m_1) and tests $b_0 = m_1 \wedge b_1 = m_0$. If the test fails, then D' enters a final state for $\forall$.

How D' clogs the channel is diagrammed in Figure 3. By tracking the round number, it knows exactly when $\exists_0$ and $\exists_1$ will exchange bits, and in that round D' expects $\exists_0$ to guess b_1, a bit that $\exists_0$ does not learn by the game procedure, and vice-versa. $\exists_0$ and $\exists_1$ are forced to spend their single bit of communication on exchanging b_0 and b_1 to their teammate, in order to guarantee survival against any $\forall$ strategy for choosing b_0 and b_1.

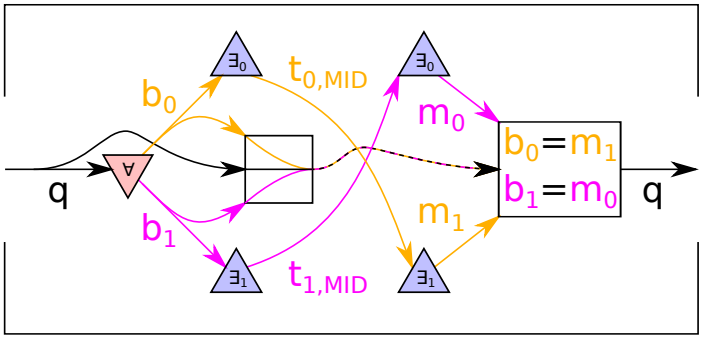

Figure 3: Mid-round 1-bit channel clogging technique. Values with the same color must be equal, namely $t_i = b_i = m_{1-i}$, or else the DFA permanently enters $F_\forall$.

Since $\exists_0$ and $\exists_1$ do not learn anything from each other or alter the simulated D's state in the rounds with communication, they have a winning strategy on D' playing TDGC under P if and only if they have a strategy for the non-exchanging rounds (which happen infinitely-often since $r \geq 2$) that would give a winning strategy on D playing TDG. □

Theorem 3. *TDGC is undecidable with n rounds of 1-bit mid-round exchanges across $r > n$ rounds: policies P where $P_{\text{MID}}(p) \in \{(0,0),(1,1)\}$ with pre-image size $|P_{\text{MID}}^{-1}((1,1))| = n$ and $P_{\text{END}}(p) = (0,0)$.*

Proof. We generalize Theorem 2 by constructing a DFA D' which clogs the channel on any round $p \pmod r$ in which $P_{\text{MID}}(p) = (1,1)$ and simulates D in the other $r - n > 0$ out of r rounds. By the same argument, this prevents communication between $\exists_0$ and $\exists_1$ while playing TDGC beyond the corresponding play of TDG taking place during non-exchanging rounds, and thus preserves forced win-ability. □

3 Undecidability of General Communication Games

This section proves our main result: that a broad class of policies with sufficiently low communication rate remain undecidable for the Team DFA game. We now define this more general notion.

Definition 3. A policy is (r, x_0, x_1, N)**-rate-limited** if, after a fixed number of rounds N, the rate of transmission from player $\exists_i$ to $\exists_{1-i}$ is x_i during every period of r rounds. Specifically, it must satisfy $x_i = \sum_{k=k_0}^{k_0+r-1} P_{\text{MID}}(p_k)[i] + P_{\text{END}}(p_k)[i]$ for $k_0 = N + \ell r$, where $\ell \in \mathbb{N}$ and p_k is the policy state on round k.

This now allows us to state the main theorem.

Theorem 4. *TDGC is undecidable under all (r, x_0, x_1, N)-rate-limited policies where $x_0, x_1 < r$.*

3.1 Properties of Rate-Limited Policies

Before proceeding to the proof of Theorem 4, we will establish useful lemmas about rate-limited policies.

First, we have the following two simple observations:

Lemma 1. *Any policy implemented as a unary-alphabet DFA with $n > 1$ states is (r, x_0, x_1, N)-rate-limited for some $1 < r \leq n$ and some initial segment of length $N \leq n$.*

Lemma 2. *Any (r, x_0, x_1, N)-rate-limited policy is also $(2r, 2x_0, 2x_1, N)$-rate-limited if $r > 1$.*

Next, we will need the following property bounding the partial sums of certain repeated finite sequences for analyzing the transmission rates in each part of a round across a period.

Definition 4. Let a be any sequence of $2n$ natural numbers $(a_0, a_1, \ldots, a_{2n-1})$ with sum at most $n-1$, and let i be an index into a. ROTATE-BOUNDED(a,i) is the predicate that holds when the infinite sequence $b_j^{(i)} = a_{(i+j \mod 2n)}$ with partial sums $B_j^{(i)} = \sum_{k=0}^{j-1} b_k^{(i)}$ satisfies $\forall j > 0.\ B_j^{(i)} < \frac{j}{2}$.

Lemma 3. *For any such a,* ROTATE-BOUNDED(a,i) *holds for some even index i.*

Proof. Let $C_j^{(i)} = B_j^{(i)} - \frac{j}{2}$. Let us find an even index i such that $C_j^{(i)} < 0$ for all $j > 0$, and consider the largest length $j^* < 2n$ which maximizes $C_{j^*}^{(0)}$. If $C_{j^*}^{(0)} < 0$ then $i = 0$ satisfies the claim, so suppose $C_{j^*}^{(0)} \geq 0$.

Because the sum of a is at most $n-1$, notice that for any j, $j+2n$ in the next period satisfies $C_{j+2n}^{(0)} = B_{j+2n}^{(i)} - \frac{j+2n}{2} < \left(B_j^{(i)} + n\right) - \left(\frac{j}{2} + n\right) = C_j^{(0)}$. Since j^* is the largest maximizer in the first period, for all $j > j^*$:

$$0 > C_j^{(0)} - C_{j^*}^{(0)} = \left(B_j^{(0)} - \frac{j}{2}\right) - \left(B_{j^*}^{(0)} - \frac{j^*}{2}\right) = B_{j-j^*}^{(j^*)} - \frac{j-j^*}{2} = C_{j-j^*}^{(j^*)}$$

and therefore $\forall j > 0.\ C_j^{(j^*)} < 0$, thus $i = j^*$ is an index for which a is rotate bounded. If j^* is even we are done. If j^* is odd, then we know that $j^* < 2n-2$ because we supposed that $C_{j^*}^{(0)} \geq 0$ whereas $C_{2n-1}^{(0)} < 0$:

$$C_{2n-1}^{(0)} = B_{2n-1}^{(0)} - \frac{2n-1}{2} \leq (n-1-a_{2n-1}) - \left(n - \frac{1}{2}\right) = -a_{2n-1} - \frac{1}{2} < 0$$

Thus, consider the even j^*+1 and let $j' \in [j^*+1, 2n)$ be the maximum length such that $C_{j'}^{(0)} = C_{j^*+1}^{(0)}$. $0 > C_{j^*+1}^{(0)} - C_{j^*}^{(0)} = C_1^{(j^*)} = b_0^{(j^*)} - \frac{1}{2}$ therefore $b_0^{(j^*)} = 0$ and $C_{j'}^{(0)} = C_{j^*+1}^{(0)} = C_{j^*}^{(0)} - \frac{1}{2}$. Since $B_{j^*}^{(0)}$ is an integer and j^* is odd, j' must also be even, and by similar arguments $\forall j > j'.\ C_{j'}^{(0)} > C_j^{(0)}$, thus $\forall j > 0.\ C_j^{(j')} < 0$, so $i = j'$ satisfies the claim. □

Corollary 1. *For any such a,* ROTATE-BOUNDED(a,i) *holds for some odd index i.*

Proof. Consider $a' = (a_1, a_2, \ldots, a_{2n-1}, a_0)$. By Lemma 3, there is an even index i' which satisfies ROTATE-BOUNDED(a', i'). Thus $i \equiv i' + 1$ is an odd index such that ROTATE-BOUNDED(a,i). □

3.2 Construction Outline

First, we introduce our reduction from the Team DFA Game to the Team DFA Game with Communication. Given an (r, x_0, x_1, N)-rate-limited policy P and an underlying DFA D, we create a DFA D' for playing the TDGC under P which simulates playing the TDG on D while completely clogging the communication between the $\exists$ team to nullify any advantage such communication could bring. This lets us conclude that a winning strategy for TDGC on D' exists exactly when a winning strategy exists for TDG on D.

The reduction applies when each $x_i < r$, meaning the communication rate defined by P is eventually below an average of one bit per round. We also assume $r > 1$ and each $x_i > 0$: if there is no communication at all then TDGC is identical to TDG, and if communication only occurs in one direction then the

aspects of the construction that deal with the silent direction may be omitted. Lastly, by Lemma 2, we take period length r to be even without loss of generality.

The code for D' is fully shown in Algorithm 1. The behavior of D' is designed around what we call the *honest* strategy for the $\exists$ team. We will show that it is the only strategy that guarantees the $\exists$ team will pass validation checks by D', but also that it requires using all available transmission bits, resulting in no information transfer between players for their additional benefit in the simulated TDG.

Along with the current state of D in the TDG, D' maintains two queues X_0, X_1 of clogging bits that have been given to each $\exists$ player by the adversary $\forall$ in specific rounds. These bits are expected to be submitted by the opposite $\exists$ player to D' for validation in later rounds in order to avoid losing the game, so the players are forced to use transmission bandwidth to exchange this information. The honest $\exists_i$ player sends these bits directly and as soon as possible to $\exists_{1-i}$, who maintains a "knowledge" queue K_{1-i} of all bits sent from $\exists_i$ but not yet validated by D'. We note that $X_i \setminus K_{1-i}$ is thus the set of yet-to-be-transmitted private bits known only to $\exists_i$.

3.3 Build-up Phase

D' begins the build-up phase after N rounds, once P has started to repeat its policy states. This phase lasts for exactly r^2 rounds, or r periods of P's cycle. D' starts with empty X_0 and X_1, and every round D' simply enqueues b_0 and b_1 into the appropriate queues.

During these rounds, $\forall$ can send one bit per round to $\exists_i$, who can transmit those bits to $\exists_{1-i}$, for each $i \in \{0,1\}$. Because the rate of transmission can vary above or below one bit per round, there is some maximum amount $x'_i \leq x_i$ out of r bits that can reach $\exists_{1-i}$ in the first r rounds. Each subsequent r rounds, x_i out of the r new bits can be sent (by the rate-limitedness of P), thus after r^2 rounds at most $x'_i + (r-1)x_i \leq rx_i < r(r-1)$ bits in X_i can be sent to $\exists_{1-i}$ and thus at least r are not known. By this argument, at the end of the build-up phase we can say that an honest player's knowledge queue has size $|K_i| = x'_{1-i} + (r-1)x_{1-i} \in [r-1, r(r-1)]$, since we assume $x_{1-i} \geq 1$.

3.4 Clogging Phase

In the clogging phase, D' simulates playing TDG on D while clogging the transmissions between $\exists$ players at a steady rate to keep $|X_i|$ and $|K_i|$ constant on period boundaries. In the last round of every period of r rounds, D' alternates between (1) having $\forall, \exists_0, \exists_1$ play one round of TDG, and (2) forcing $\forall$ to tell the $\exists$ team if they have won in the TDG yet and therefore if D' is going to start the next phase: the tear-down phase.

In the first $r-1$ rounds of each period in this phase, D' clogs the transmissions between $\exists$ players by requiring that bits given to $\exists_i$ by $\forall$ (placed into queue X_i) are sent to $\exists_{1-i}$. This is done by dequeuing the oldest bit b from X_i and checking for $\exists_{1-i}$ to submit $m_{1-i} = b$, otherwise they will lose the game. Specifically, to preserve the size of X_i and keep up with the rates at which the $\exists$ players can transmit information to each other, D' will do $\text{ENQ}(X_i, b_i)$ then validate $\text{DEQ}(X_i) = m_{1-i}$ for the first x_i rounds of each period.

Across the whole period, K_i will gain x_{1-i} new bits transmitted from $\exists_{1-i}$ (by the rate-limitedness of P). New available bits always exist because the number of private bits available to be sent is $|X_{1-i} \setminus K_i| \geq r > x_{1-i}$ at the start of the period. Additionally, across the first x_{1-i} rounds of the period, K_i will lose x_{1-i} bits submitted by $\exists_i$ to D', which are always known because $|K_i| \geq (r-1)x_{1-i} \geq x_{1-i}$ at the start. Overall, this means $|K_i|$ is preserved on period boundaries and honest players will always be able to submit the correct bit and pass the validation.

Labeling the first clogging period with index 0, at the end of every odd-indexed period, D' will simulate TDG by forwarding the inputs of all players directly to D. However, at the end of even-indexed periods, D' will ignore $\exists_0, \exists_1$ and expect both $\forall$ bits to state whether or not the $\exists$ team has won in TDG, specifically requiring that $b_0 = b_1 = [q \in F_\exists]$. If this validation fails, then D' will halt with a $\exists$ team victory, so the $\forall$ player must give the correct information to both $\exists$ players to avoid losing, which it is always able to do.

Assuming validation never fails, which is achieved by the honest strategy, the clogging phase continues until the simulated Team DFA Game ends. If the $\exists$ players lose in the simulation, they lose immediately, otherwise after the even-indexed period when the $\exists$ players learn they have won, D' moves onto the tear-down phase to perform the final validation checks.

3.5 Tear-down Phase

The tear-down phase starts at the beginning of a period, so by the previous arguments for queue size preservation, it starts with $|X_i| = r^2$ and $|K_{1-i}| = x'_i + (r-1)x_i$. In order to ensure the $\exists$ team's transmissions have been completely clogged all the way until the simulated victory, D' must validate that the remaining bits in K_i have actually been sent by this point.

This phase is split into two parts, with a boosting sub-phase to adjust the size of X_i and K_{1-i} for the following draining sub-phase that empties them. Once each queue has been drained and all validation checks have been passed, then D' will halt with an $\exists$ team victory. We will need the following fact:

Lemma 4. *There exists a k_{end} such that, in every round up to the k^{th}_{end} round of a period, the cumulative number of bits $\exists_0$ will transmit to $\exists_1$ before $\exists_1$ submits a bit to D' in the k^{th}_{end} round is always upper-bounded by the cumulative number of bits $\exists_1$ will submit to D' in that time (from round N onwards).*

Proof. Say the period begins in round $m \geq N$, and recall that we can assume the period length r is even. Consider the sequence a_k of the number of bits transmitted from $\exists_0$ to $\exists_1$ in the k half-rounds starting in round m, so $a_k = P_{\text{MID}}(p_{m+k/2})[0]$ when k is even and $a_k = P_{\text{END}}(p_{m+(k-1)/2})[0]$ when k is odd. Since policy states repeat, $\forall k \geq 0$. $a_{k+r} = a_k$, and $a_0 + \ldots + a_{r-1} = x_0 < r$, so we can apply Corollary 1 to the reversed sequence $(a_{r-1}, \ldots, a_0)$ to get an odd index i such that $\forall j > 0$. $B^{(i)}_j < \frac{j}{2}$.

Since $B^{(i)}_j$ is the cumulative number of bits transmitted from $\exists_0$ to $\exists_1$ across the j half-rounds ending when $\exists_1$ submits a bit to D' in round $m + \frac{r-1-i}{2}$, and $\lceil \frac{j+1}{2} \rceil \geq \frac{j}{2}$ is the cumulative number of bits $\exists_1$ submits to D' across the same set of j half-rounds, then round offset $k_{end} = \frac{r-i-1}{2}$ satisfies Lemma 4. □

Draining Sub-Phase Given k_{end} from Lemma 4 (by symmetry, the lemma applies in both directions), let $t_{end} \leq k_{end}$ be the total number of bits transmitted from the beginning of a period until the bit submission in the k^{th}_{end} round. If a period starts with $|X_i| \leq k_{end}$ and $|X_i \setminus K_{1-i}| = t_{end}$, then we can have D' validate bits in the $|X_i|$ rounds before the k^{th}_{end} round and reach $|X_i| = |K_{1-i}| = 0$ where each of the t_{end} transmitted bits are clogging bits from $X_i \setminus K_{1-i}$ with no room for extra communication from $\exists_i$ to $\exists_{1-i}$.

In order to ensure some period starts with $|X_i| \leq k_{end}$ and $|X_i \setminus K_{1-i}| = t_{end}$ we use some n_i periods beforehand to drain each queue appropriately. Since in each period there are x_i transmission bits (fixed) and up to r validated bits (based on D'), it suffices to have $|X_i| \leq n_i r + k_{end}$ and $|X_i \setminus K_{1-i}| = n_i x_i + t_{end}$.

Boosting Sub-Phase The tear-down phase must start with $|X_i \setminus K_{1-i}| = r^2 - (x'_i + (r-1)x_i) \geq r-1$, but this may not be $n_i x_i + t_{end}$ for any n_i, so before $n_i + 1$ draining periods, we will have additional periods

to increase the number of private bits by $\delta_i = (n_i x_i + t_{end}) - (r^2 - (x'_i + (r-1)x_i))$. So for $\delta_i \geq 0$, we can choose any sufficiently-large n_i.

After one period where $\forall$ gives c_i new clogging bits to $\exists_i$ and D' validates v_{1-i} bits from $\exists_{1-i}$, we would have $\Delta|X_i| = c_i - v_{1-i}$ and $\Delta|K_{1-i}| = x_i - v_{1-i}$ (given that $\exists_i$ initially has $|X_i \setminus K_{1-i}| \geq x_i$ private bits to transmit to $\exists_{1-i}$), thus $\Delta|X_i \setminus K_{1-i}| = c_i - x_i$. Therefore, if we set $c_i = x_i + 1 \leq r$ and $v_{1-i} = x_i$, then we get $\Delta|X_i| = +1$, $\Delta|K_{1-i}| = 0$, and $\Delta|X_i \setminus K_{1-i}| = +1$. If $\delta_i < \delta_{1-i}$, then to delay we also need "filler" rounds with no change to the sizes of any queues, which can be achieved by setting $c_i = v_{1-i} = x_i$.

To ensure δ_i is positive and $|X_i| \leq n_i r + k_{end}$ at the *end* of this sub-phase, we need to choose an n_i that satisfies the following constraints at the *start* of the tear-down phase:

$$\begin{aligned} 0 \leq \delta_i &= (n_i x_i + t_{end}) - |X_i \setminus K_{1-i}| \\ n_i &\geq (|X_i \setminus K_{1-i}| - t_{end}) / x_i \end{aligned}$$

and

$$\begin{aligned} n_i r + k_{end} &\geq r^2 + \delta_i \\ n_i r + k_{end} &\geq |X_i| + (n_i x_i + t_{end}) - |X_i \setminus K_{1-i}| = |K_{1-i}| + (n_i x_i + t_{end}) \\ n_i &\geq (|K_{1-i}| + t_{end} - k_{end}) / (r - x_i) \end{aligned}$$

We pick n_i to be the smallest natural number satisfying both lower bounds:

$$\begin{aligned} n_i &= \left\lceil \max\left\{ \frac{|X_i \setminus K_{1-i}| - t_{end}}{x_i}, \frac{|K_{1-i}| + t_{end} - k_{end}}{r - x_i} \right\} \right\rceil \\ &= \left\lceil \max\left\{ \frac{r^2 - (x'_i + (r-1)x_i) - t_{end}}{x_i}, \frac{(x'_i + (r-1)x_i) + t_{end} - k_{end}}{r - x_i} \right\} \right\rceil \end{aligned}$$

Since $0 \leq x'_i \leq x_i < r$ and $0 \leq t_{end} \leq k_{end} < r$, we can upper bound $n_i = O(r^2)$.

Putting it all together At the beginning of the tear-down period, D' will run a set of δ_i periods where $\forall$ produces $x_i + 1$ new bits and D' validates x_i bits, followed by $\max\{\delta_0, \delta_1\} - \delta_i$ periods of x_i new and validated bits. After $\delta_{\max} = \max\{\delta_0, \delta_1\}$ rounds, we will have $|X_i| = r^2 + \delta_i$ and $|X_i \setminus K_{1-i}| = n_i x_i + t_{end}$, preserving $|K_{1-i}| = x'_i + (r-1)x_i$. D' will then run n_i periods plus k_{end} rounds ignoring $\forall$ and validating the remainder of X_i (starting $|X_i|$ rounds before the end).

3.6 Proof of Undecidability

Theorem 4. *TDGC is undecidable under all (r, x_0, x_1, N)-rate-limited policies where $x_0, x_1 < r$.*

Proof. We reduce from the Team DFA Game. For any (r, x_0, x_1, N)-rate-limited policy P where $x_0, x_1 < r$, given an input DFA D for playing the TDG, we construct the DFA D' described in Algorithm 1 for playing the TDGC under policy P. Since determining whether or not the $\exists$ team has a forced win in the TDG is undecidable, this reduction will show that the same question of the TDGC under policy P is undecidable as well.

Given the analysis of D' from the previous sections, we first note that D' is indeed a finite automaton: the waiting counter takes on N values; each queue X_i has maximum size $r^2 + \delta_i$ bits, where $n_i = O(r^2)$ so $\delta_i = O(r^3)$; the state q of D has $|Q|$ possible values; and the various other counters require $O(\log r)$ bits each. From beginning to end, the maximum memory requirement is $O\left(\max\left\{\log N, r^2 + \log|Q|, r^3\right\}\right)$ bits, summarizing Table 3.6.

State Category	**Space Needed (bits)**
HALT($winner$)	$\Theta(1)$
WAITING(w)	$\Theta(\log N)$
BUILD-UP(X_0, X_1)	$\Theta(r^2)$
CLOG(X_0, X_1, q, p, k, c_{01}, c_{10})	$\Theta(r^2 + \log \lvert Q \rvert)$
BOOST(X_0, X_1, d_{01}, d_{10}, k, c_{01}, c_{10})	$\Theta(r^2 + \delta_{\max})$
DRAIN(X_0, X_1, c_{01}, c_{10})	$\Theta(r^2 + \delta_{\max})$

Table 1: Memory Requirements of D' over the course of the TDGC.

If there is a winning strategy S for the $\exists$ team on D in the TDG, then the corresponding honest strategy described above that plays the simulated TDG using S will be a winning strategy for the $\exists$ team on D' in the TDGC under policy P.

If there is a winning strategy for the $\exists$ team on D' in the TDGC under policy P, then consider any winning execution γ. Since winning requires termination, let C be the number of periods in the clogging phase.

If γ reaches HALT($\exists$) in the clogging phase because $\forall$ did not correctly tell the $\exists$ team whether or not $q \in F_\exists$, then $\forall$ did not play optimally. Since $\forall$ has perfect information and is allowed to give either 0 or 1 by the game rules, there is an alternate execution γ' where $\forall$ gives the correct answer instead and the game continues, so no $\exists$ team strategy can force a win in this way.

The only other way for the $\exists$ team to win is for γ to reach HALT($\exists$) at the end of the draining phase, which means they must pass all of the validation checks by D'.

Phase	ENQ(X_i) Count	DEQ(X_i) Count	Information $\exists_i \to \exists_{1-i}$
Build-up	r^2	0	$x'_i + (r-1) \times x_i$
Clogging	$C \times x_i$	$C \times x_i$	$C \times x_i$
Boosting	$\delta_{\max} \times x_i + \delta_i$	$\delta_{\max} \times x_i$	$\delta_{\max} \times x_i$
Draining	0	$r^2 + \delta_i$	$n_i \times x_i + t_{end_i}$

Table 2: Accounting of ENQ(X_i), DEQ(X_i), and Information Transfer between players in each phase

Table 2 details the value of three quantities in each phase of the game: the number of bits enqueued into X_i, the number of bits dequeued from X_i, and the amount of meaningful bits of information that can be transmitted from $\exists_i$ to $\exists_{1-i}$. By the definition of δ_i and some algebra, it can be seen that each column has the same sum; let I be this total quantity of bits.

Because D' validates the value of each dequeued bit, in order for $\exists_i$ to guarantee they pass all validation checks, they must send I bits of information to $\exists_{1-i}$. However, because I is the maximum amount of information $\exists_i$ can send to $\exists_{1-i}$, no further information can be sent, which means that in every round in which D' simulates the TDG on D', $\exists_i$ has the same amount of information about the state q of D as it would when actually playing TDG on D. Therefore, if the $\exists$ team has a winning strategy for playing TDGC on D' under policy P, within it is a winning sub-strategy for them to play the TDG on D. □

Algorithm 1 Pseudocode for the D' internal update function per round

$q' \leftarrow$ WAITING(1) ▷ *Initial state*
function DFA-ROUND-UPDATE(q', b_0, b_1, m_0, m_1)
  switch q'
    case HALT(*winner*) ▷ *Game is over, with* $q' \in F'_{winner}$
      return HALT(*winner*)
    case WAITING(w) ▷ *Waiting Phase, delaying until policy starts repeating*
      if $w < N$ **then return** WAITING($w+1$)
      return BUILD-UP([], [])
    case BUILD-UP(X_0, X_1) ▷ *Build-up Phase, filling up* X_i *queues*
      ENQ(X_0, b_0)
      ENQ(X_1, b_1)
      if LENGTH(X_0) $< r^2$ **then return** BUILD-UP(X_0, X_1)
      return CLOG(X_0, X_1, q_0, EVEN, r, x_0, x_1)
    case CLOG(X_0, X_1, q, p, k, c_{01}, c_{10}) **given** $k > 1$ ▷ *Clogging Phase, boosting*
      for all $i \in \{0,1\}$
        if $c_{i,1-i} > 0$ **then**
          ENQ(X_i, b_i)
          if DEQ(X_i) $\neq m_{1-i}$ **then return** HALT($\forall$)
          $c_{i,1-i} \leftarrow c_{i,1-i} - 1$
      return CLOG(X_0, X_1, q, p, $k-1$, c_{01}, c_{10})
    case CLOG(X_0, X_1, q, ODD, 1, 0, 0) ▷ *Clogging Phase, simulating D*
      $q \leftarrow \delta(\delta(\delta(\delta(q,b_0),b_1),m_0),m_1)$
      return CLOG(X_0, X_1, q, EVEN, r, x_0, x_1)
    case CLOG(X_0, X_1, q, EVEN, 1, 0, 0) ▷ *Clogging Phase, testing for* $\exists$ *win*
      if $\neg(b_0 = b_1 = [q \in F_\exists])$ **then return** HALT($\exists$)
      if $q \in F_\exists$ **then return** BOOST(X_0, X_1, δ_0, δ_1, r, x_0, x_1)
      if $q \in F_\forall$ **then return** HALT($\forall$)
      return CLOG(X_0, X_1, q, ODD, r, x_0, x_1)
    case BOOST(X_0, X_1, d_{01}, d_{10}, k, c_{01}, c_{10}) **given** $k > 1$ ▷ *Boost Phase, clogging*
      for all $i \in \{0,1\}$
        if $c_{i,1-i} > 0$ **then**
          ENQ(X_i, b_i)
          if DEQ(X_i) $\neq m_{1-i}$ **then return** HALT($\forall$) ▷ *Return from caller*
          $c_{i,1-i} \leftarrow c_{i,1-i} - 1$
      return BOOST(X_0, X_1, d_{01}, d_{10}, $k-1$, c_{01}, c_{10})
    case BOOST(X_0, X_1, d_{01}, d_{10}, 1, 0, 0) ▷ *Boost Phase, new boost bits*
      for all $i \in \{0,1\}$
        if $d_{i,1-i} > 0$ **then**
          ENQ(X_i, b_i)
          $d_{i,1-i} \leftarrow d_{i,1-i} - 1$
      if $d_{01} + d_{10} > 0$ **then return** BOOST(X_0, X_1, d_{01}, d_{10}, r, x_0, x_1)
      return DRAIN(X_0, X_1, $r \times n_0 + k_{end_0}$, $r \times n_1 + k_{end_1}$)
    case DRAIN(X_0, X_1, c_{01}, c_{10}) given $c_{01} + c_{10} > 0$ ▷ *Drain Phase, emptying queues*
      for all $i \in \{0,1\}$
        if $c_{i,1-i1} > 0$ **then**
          if $|X_i| = c_{i,1-i} \wedge$ DEQ(X_i) $\neq m_{1-i}$ **then return** HALT($\forall$)
          $c_{i,1-i1} \leftarrow c_{i,1-i1} - 1$
      return DRAIN(X_0, X_1, c_{01}, c_{10})
    case DRAIN([], [], 0, 0) ▷ *Drain Phase, finished!*
      return HALT($\exists$)

4 Decidability

We show that our general construction from the previous section is tight with respect to the transmission rate between $\exists$ players.

For our precise bounds, we assume the straightforward encoding of the input DFA D with n states as a table for δ containing $2n$ states, a state q_0, and the states in $F_\exists$ and $F_\forall$, thus the input size is $\Theta(|Q|)$.

First, we demonstrate $(r,r,r,0)$-rate-limited policies under which the Team DFA Game with Communication is not only decidable but in PSPACE. Later we will show more restrictive communication patterns are in EXPSPACE. Recall $(r,r,r,0)$-rate-limited policies are the case where both players are allowed to exchange r bits over the course of a period of length r.

Theorem 5. *TDGC is decidable in PSPACE with a 1-bit, mid-round exchange in both directions every round: policies P with $P_{\text{MID}}(p) = (1,1)$ and $P_{\text{END}}(p) = (0,0)$ for all $p \in \Pi$.*

Proof. Under such a policy, TDGC becomes a perfect information game. In each round of the game, the optimal play for $\exists_i$ is to send b_i to $\exists_{1-i}$ immediately after receiving it, meaning $\exists_{1-i}$ will know both b_0 and b_1 before it chooses m_{1-i}. Since the $\exists$ team knows the initial state q_0 of D, we can consider strategy functions $s : (q,b_0,b_1) \mapsto (m_0,m_1)$, which both players can use to decide their own next move and know what move their teammate will perform as well, letting them use δ to learn the state q of D in the next round and beyond.

Note that it suffices for the $\exists$ team to have a memoryless strategy because the policy P is constant per round, DFA transitions do not depend on the history of the game, and the adversarial $\forall$ player's choices are not bound by the history either. It also suffices to have a deterministic strategy: if there exists a non-deterministic winning strategy s', then we can fix $s(q,b_0,b_0)$ to be some (m_0,m_1) with $\Pr[s'(q,b_0,b_1) = (m_0,m_1)] > 0$ because all game executions in which the $\exists$ team plays with deterministic strategy s are possible executions when playing with strategy s', thus must also be winning.

We show that deciding whether or not the $\exists$ team has a forced win in TDGC under policy P is in PSPACE by giving a brute-force search algorithm. For every strategy s among the $4^{4|Q|}$ possible strategy functions, we construct a game graph G_s where each state $q \in Q \setminus F_\exists$ is a vertex and for all $b_0,b_1 \in \{0,1\}$, q has an edge to $q' = \delta(\delta(\delta(\delta(q,b_0),b_1),m_0),m_1)$ where $(m_0,m_1) = s(q,b_0,b_1)$ as long as $q' \notin F_\exists$. This means s is a winning strategy if and only if all $q \in F_\forall$ and all cycles are not reachable from q_0 in G_s, since otherwise the traversal corresponds to a losing execution or the start of a potentially non-terminating execution of the game that the $\forall$ player can force to occur. We can thus perform an exhaustive depth-first search from q_0 for a counterexample (of length at most $|Q|$) to decide whether or not s is a winning strategy. Since we only need $\Theta(|Q|)$ space to store the current s, G_s, and depth-first search stack, this algorithm runs in PSPACE. □

Since it is sufficient to send only one bit of useful information mid-round, we can extend Theorem 5 to higher transmission rates.

Corollary 2. *TDGC is decidable in PSPACE with at least a 1-bit, mid-round exchange in both directions every round: policies P with $P_{\text{MID}}(p)[i] \geq 1$ for all $p \in \Pi$ and each $i \in \{0,1\}$.*

Next, we consider the decidability of TDGC under $(r,r,0,0)$-rate-limited policies, which is tight given the undecidability of $(r,r-1,0,0)$-rate-limited policies. This shows that only one member of the team needs to have perfect information.

Theorem 6. *TDGC is decidable in EXPSPACE with a 1-bit, mid-round exchange every round from $\exists_0$ to $\exists_1$, but none from $\exists_1$ to $\exists_0$: policies P with $P_{\text{MID}}(p) = (1,0)$ and $P_{\text{END}}(p) = (0,0)$ for all $p \in \Pi$.*

Proof. As described in the proof of Theorem 5, $\exists_0$ can and should send b_0 to $\exists_1$ each round to give $\exists_1$ perfect information, but $\exists_0$ themself can learn nothing about b_1. Using the terminology from [11], this asymmetry makes TDGC under P a hierarchical team game. To decide the existence of a winning strategy, we adapt ideas from the proof of Theorem 4 in the same paper that shows $\text{DTIME}\left(2^{2^{2^{cS(n)}}}\right) \supseteq \text{MPA}_2\text{-SPACE}(S(n))$, the languages decided by hierarchical 2-vs-1 private alternation Turing machines in $S(n)$ space.

Consider the set of all possible mid-round configurations (q, b_0, b_1) of the game, which are fully known to $\forall$ and $\exists_1$. Define C be the set of possible configurations (b_0, u) of $\exists_0$'s mid-round knowledge: the known b_0 and the set $u \in \mathscr{P}(Q \times \{b_0\} \times \{0,1\})$ of possible mid-round configurations given the history of the game thus far. Since two game states with the same $c \in C$ are strategically equivalent from the perspective of $\exists_0$ (and thus $\exists_1$ too), a winning strategy only needs to account for the $|C| = 2^{2|Q|+1}$ knowledge configurations in its decision-making.

Given this, we can do a brute-force search as in Theorem 5 over the space of deterministic $\exists$ team strategies $s : c \in C \mapsto (m_0, m_1)$ of size $4^{|C|}$. For each s, we construct the game graph G_s, where $c \in C$ has an outgoing edge representing the outcome of each b_0, b_1 choice of $\forall$ after the $\exists$ players use s to make their moves and $\exists_0$ updates their knowledge, and then search for counter-example game executions with length up to $|C|$ to decide whether s is a winning strategy. Therefore, TDGC under P is decidable in $\Theta(|C|)$ space, which is exponential in $|Q|$. □

As before, Theorem 5 extends to higher transmission rates from $\exists_0$ to $\exists_1$ (or vice versa), as long as the receiver stays silent.

Corollary 3. *TDGC is decidable in PSPACE with at least a 1-bit, mid-round exchange in one direction every round, but none in the other direction: policies P with $P_{\text{MID}}(p)[i] \geq 1$ and $P_{\text{MID}}(p)[1-i] = P_{\text{END}}(p)[1-i] = 0$ for all $p \in \Pi$ and some $i \in \{0,1\}$.*

5 Team Formula Games with Communication

Formula games model many types of games. The Team Formula Game was defined and proven undecidable in [6]. We define a communication version of this game and prove results analogous to the ones for TDGC.

Definition 5. A *Team Formula Game* (TFG) instance consists of sets of Boolean variables X, X', Y_1, Y_2 and their initial values; variables $h_0, h_1 \in X$; and Boolean formulas $F(X, X', Y_0, Y_1)$, $F'(X, X')$, and $G(X)$ such that F implies $\neg F'$. The TFG problem asks whether $\{W_0, W_1\}$, team White, has a forced win against $\{B\}$, team Black, in the game that repeats the following steps in order ad infinitum:

1. B sets X to any values. If F and G are true, then Black wins. If F is false, White wins.
2. B sets X' to any values. If F' is false, then White wins.
3. W_0 sets Y_1 to any values.
4. W_1 sets Y_2 to any values.

where B has perfect information but W_i can only see the values of Y_i and h_i.

Definition 6. *Team Formula Game with Communication* (TFGC) is TFG along with a policy P which specifies a number of bits to be transmitted between W_0 and W_1 mid-round (before each step 3) and at the end of the round (after each step 4)

Theorem 7. *TFGC is undecidable under all (r, x_0, x_1, N)-rate-limited policies where $x_0, x_1 < r$.*

Proof. For any such policy P, we reduce from the Team DFA Game with Communication under the same policy P, adapting the reduction done in Theorem 8 of [6] from the Team Computation Game to the Team Formula Game. In the reduction, the White team plays as the $\exists$ team and B plays as $\forall$ while also facilitating the simulation of TDGC in TFGC.

Given a DFA D to play TDGC under P, we first augment D so it will be suitable for the simulation. To each state, we add a 3-value counter to eliminate any four-edge cycles in the transition graph ($t \xrightarrow{\delta} (t+1) \xrightarrow{\delta} (t+2) \xrightarrow{\delta} t \xrightarrow{\delta} (t+1) \neq t$). Also, we add four new states in a path $q_0 \xrightarrow{\delta} q_0^{(1)} \xrightarrow{\delta} q_0^{(2)} \xrightarrow{\delta} q_0^{(3)} \xrightarrow{\delta} q_0^{(4)}$ from a new initial state q_0 to the original initial state $q_0^{(4)}$ in order to delay the first meaningful state transitions until the start of the second round, which is when the first set of player inputs are available.

In the instance of TFGC, we will have (1) variables $h_i = b_i \in X$ and $b_i' \in X'$, representing the $\forall$ player's chosen bits in the current and previous round; (2) $Y_i = \{m_i\}$, containing the $\exists_i$ player's message bit each round; (3) sets of $\Theta(\log |Q|)$ variables $\langle q' \rangle \subset X'$ and $\langle q \rangle \subset X$ that encode the previous state q' and current state q; (4) and two parity bits $p \in X$ and $p' \in X'$ which B will be required to flip each round. We also choose the initial value of q' to be q_0 so that in step 1 of the first round B will be forced to set q to $q_0^{(4)}$; other initial values are arbitrary.

In step 1, formula F holds if B sets X so $q = \delta(\delta(\delta(\delta(q', b_0'), b_1'), m_0), m_1)$, $q \notin F_\exists$, and $p' \neq p$. Formula G will be true if the current state $q \in F_\forall$. Thus, when F and G are both true, then in the TDGC the state transition function was correctly implemented and led to a final state where $\forall$ has won, and thus Black wins the TFGC. On the other hand, if F is false, then either B violated the simulation or the TDGC led to a final state where $\exists$ team has won, and thus White wins the TFGC.

In step 2, formula F' will be true if B sets X' such that $q' = q$ and $p' = p$, updating the previous state for the next round to the new state. If F' is false, then B violated the simulation, and thus White wins the TFGC. Additionally, the parity bit checks guarantee that F implies $\neg F'$.

Since this is a faithful simulation where each round of TFGC corresponds exactly to one round of TDGC, and by Theorem 4 it is undecidable whether or not there exists a winning strategy for the $\exists$ team playing TDGC under P, it is also undecidable whether or not there exists a winning strategy for White playing TFGC under the same policy P. □

The strategy for proving decidability results of Team DFA Game with Communication also be used to give the following tight decidability results on the Team Formula Game with Communication.

Theorem 8. *TFGC is decidable in 2-EXPSPACE with a 1-bit, mid-round exchange in both directions every round: policies P with $P_{\text{MID}}(p) = (1,1)$ and $P_{\text{END}}(p) = (0,0)$ for all $p \in \Pi$.*

Theorem 9. *TFGC is decidable in 3-EXPSPACE with a 1-bit, mid-round exchange every round from W_0 to W_1, but none from W_1 to W_0: policies P with $P_{\text{MID}}(p) = (1,0)$ and $P_{\text{END}}(p) = (0,0)$ for all $p \in \Pi$.*

6 Open Problems

One exciting question is whether we can prove computational complexity results about real games with communication. It seems plausible that TDGC may be sufficient for applications to games with highly structured communication. We present a number of questions that we think may help strengthen results to allow their application to more real world scenarios or questions we find particularly interesting for their own sake.

One of the main technical questions left open by this work is the complexity for rate-limited policies with $x_0 \geq r$ and $r > x_1 > 0$. We conjecture this case is decidable but our current arguments rely on both players either having full information or no information.

Looking further, there are many interesting variations and extensions of this model to study. Our arguments rely heavily on communication policies having some bounded period which is useful both for algorithms to bound the uncertainty in the game and for undecidability to allow for constructions that simulate a step in a zero information game after a bounded number of rounds. What happens if our policy is described by something more general than a DFA, such as a sequence recognizable by a pushdown automaton?

Similarly, some of our arguments rely on the fact that the game is played on something with bounded state, such as a DFA or Boolean Formula. What happens with team games on more general systems, such as a pushdown automaton or a bounded space Turing Machine?

Many realistic scenarios have noisy communication channels. How does the computational complexity change under different models of noise? We conjecture that there will again be a cutoff based on whether the information capacity of the channel is sufficiently high. However, it is also possible that the small probability of error will compound over these games of unbounded length resulting in different behavior. It would also be interesting to understand what happens when other sources of inherent randomness are introduced to these games.

It is also often the case that one's ability to communicate depends on the state of the environment and potentially the actions of the people involved. Thus having communication policies that depend on player actions or the game state would be another interesting generalization.

We also only consider two players on the Existential Team. We believe that when more players are added, undecidability will emerge if at least two players have imperfect information. However, this should be verified and the details around more complex communication patterns may lead to richer behavior.

Finally, there is an issue when trying to apply these results to real games or real world problems. Our characterization in some sense relies on communication being high or low compared to critical or meaningful choices in the games. Many natural scenarios have a much larger action space than communication rate, however many of those choices may be essentially equivalent or strategically inadvisable. Undecidability proofs such as those for Team Fortress 2 [5] have very inefficient reductions and require significant numbers of in-game actions to simulate one move in the DFA game. This makes a direct application of our results difficult.

Acknowledgements

We would like to thank Erik Demaine and other participants in the class 6.892 Algorithmic Lower Bounds: Fun with Hardness Proofs (Spring 2019) for useful discussion and the suggestion of potential applications. Thanks to Sophie Monahan for editing assistance.

References

[1] Jean-François Baffier, Man-Kwun Chiu, Yago Diez, Matias Korman, Valia Mitsou, André van Renssen, Marcel Roeloffzen & Yushi Uno (2017): *Hanabi is NP-hard, even for cheaters who look at their cards*. 675, pp. 43–55, doi:10.1016/j.tcs.2017.02.024.

[2] Anton Bakhtin, David J. Wu, Adam Lerer & Noam Brown (2021): *No-Press Diplomacy from Scratch*. *Advances in Neural Information Processing Systems 34: Annual Conference*

on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual, pp. 18063–18074. Available at https://proceedings.neurips.cc/paper/2021/hash/95f2b84de5660ddf45c8a34933a2e66f-Abstract.html.

[3] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse et al. (2019): *Dota 2 with large scale deep reinforcement learning*. arXiv:1912.06680.

[4] Andrea Celli, Marco Ciccone, Raffaele Bongo & Nicola Gatti (2019): *Coordination in adversarial sequential team games via multi-agent deep reinforcement learning*. arXiv:1912.07712.

[5] Michael J. Coulombe & Jayson Lynch (2018): *Cooperating in Video Games? Impossible! Undecidability of Team Multiplayer Games*. *9th International Conference on Fun with Algorithms (FUN 2018)* 100, pp. 14:1–14:16, doi:10.4230/LIPIcs.FUN.2018.14.

[6] Erik D. Demaine & Robert A. Hearn (2008): *Constraint logic: A uniform framework for modeling computation as games*. In: *2008 23rd Annual IEEE Conference on Computational Complexity*, IEEE, College Park, MD, USA, pp. 149–162, doi:10.1109/CCC.2008.35.

[7] Jakob N. Foerster, Yannis M. Assael, Nando de Freitas & Shimon Whiteson (2016): *Learning to Communicate with Deep Multi-Agent Reinforcement Learning*. *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 2137–2145. Available at https://proceedings.neurips.cc/paper/2016/hash/c7635bfd99248a2cdef8249ef7bfbef4-Abstract.html.

[8] Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, Dj Strouse, Joel Z. Leibo & Nando De Freitas (2019): *Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning*. *Proceedings of the 36th International Conference on Machine Learning* 97, pp. 3040–3049. Available at https://proceedings.mlr.press/v97/jaques19a.html.

[9] Philip Paquette, Yuchen Lu, Steven Bocco, Max O. Smith, Satya Ortiz-Gagne, Jonathan K. Kummerfeld, Joelle Pineau, Satinder Singh & Aaron C. Courville (2019): *No-Press Diplomacy: Modeling Multi-Agent Gameplay*. *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 4476–4487. Available at https://proceedings.neurips.cc/paper/2019/hash/84b20b1f5a0d103f5710bb67a043cd78-Abstract.html.

[10] Gary Peterson, John Reif & Salman Azhar (2001): *Lower bounds for multiplayer noncooperative games of incomplete information*. *Computers & Mathematics with Applications* 41(7-8), pp. 957–992, doi:10.1016/S0898-1221(00)00333-3.

[11] Gary L. Peterson & John H. Reif (1979): *Multiple-person alternation*. In: *20th Annual Symposium on Foundations of Computer Science (sfcs 1979)*, IEEE, San Juan, Puerto Rico, pp. 348–363, doi:10.1109/SFCS.1979.25.

[12] Frederick Reiber (2021): *The Crew: The Quest for Planet Nine is NP-Complete*. *CoRR*. arXiv:2110.11758.

[13] Giovanni Viglietta (2014): *Gaming is a hard job, but someone has to do it!* *Theory of Computing Systems* 54(4), pp. 595–621, doi:10.1007/s00224-013-9497-5.

[14] Oriol Vinyals, Igor Babuschkin, Junyoung Chung, Michael Mathieu, Max Jaderberg, Wojtek Czarnecki, Andrew Dudzik, Aja Huang, Petko Georgiev, Richard Powell, Timo Ewalds, Dan Horgan, Manuel Kroiss, Ivo Danihelka, John Agapiou, Junhyuk Oh, Valentin Dalibard, David Choi, Laurent Sifre, Yury Sulsky, Sasha Vezhnevets, James Molloy, Trevor Cai, David Budden, Tom Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Toby Pohlen, Dani Yogatama, Julia Cohen, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Chris Apps, Koray Kavukcuoglu, Demis Hassabis & David Silver (2019): *AlphaStar: Mastering the Real-Time Strategy Game StarCraft II*. https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/.