Multi-Player Flow Games

Shibashis Guha
Université Libre de Bruxelles
Brussels, Belgium
shibashis.guha@ulb.ac.be

Orna Kupferman
The Hebrew University
Jerusalem, Israel
orna@cs.huji.ac.il

Gal Vardi
The Hebrew University
Jerusalem, Israel
gal.vardi@mail.huji.ac.il

ABSTRACT

In the traditional maximum-flow problem, the goal is to transfer maximum flow in a network by directing, in each vertex in the network, incoming flow to outgoing edges. The problem corresponds to settings in which a central authority has control on all vertices of the network. Today's computing environment, however, involves systems with no central authority. In particular, in many applications of flow networks, the vertices correspond to decision-points controlled by different and selfish entities. For example, in communication networks, routers may belong to different companies, with different destination objectives. This suggests that the maximum-flow problem should be revisited, and examined from a game-theoretic perspective.

We introduce and study *multi-player flow games* (MFGs, for short). Essentially, the vertices of an MFG are partitioned among the players, and a player that owns a vertex directs the flow that reaches it. Each player has a different target vertex, and the objective of each player is to maximize the flow that reaches her target vertex. We study the stability of MFGs and show that, unfortunately, an MFG need not have a Nash Equilibrium. Moreover, the Price of Anarchy and even the Price of Stability of MFGs are unbounded. That is, the reduction in the flow due to selfish behavior is unbounded. We study the problem of deciding whether a given MFG has a Nash Equilibrium and show that it is Σ_2^P-complete, as well as the problem of finding optimal strategies for the players (that is, best-response moves), which we show to be NP-complete. We continue with some good news and consider a variant of MFGs in which flow may be swallowed. For example, when routers in a communication network may drop messages. We show that, surprisingly, while this model seems to incentivize selfish behavior, a Nash Equilibrium that achieves the maximum flow always exists, and can be found in polynomial time. Finally, we consider MFGs in which the strategies of the players may use non-integral flows, which we show to be stronger.

KEYWORDS

Game Theory for practical applications; Noncooperative games: computation; Methodologies for agent-based systems; Noncooperative games: theory & analysis

ACM Reference Format:
Shibashis Guha, Orna Kupferman, and Gal Vardi. 2018. Multi-Player Flow Games. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), Stockholm, Sweden, July 2018,* IFAAMAS, 9 pages.

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), , July 2018, Stockholm, Sweden.

1 INTRODUCTION

A *flow network* is a directed graph in which each edge has a capacity, bounding the amount of flow that can go through it. The amount of flow that enters a vertex equals the amount of flow that leaves it, unless the vertex is a *source*, which has only outgoing flow, or a *target*, which has only incoming flow. The fundamental *maximum-flow problem* gets as input a flow network with a source vertex and a target vertex and searches for a maximum flow from the source to the target [5, 12]. The problem was first formulated and solved in the 1950's [10, 11]. It has attracted much research on improved algorithms [6, 7, 13] and variant settings [8, 22], and has been applied in many application domains, including traffic in road or rail systems, fluids in pipes, currents in an electrical circuit, packets in a communication network, and many more [2].

All studies of flow networks so far assume that all the vertices in the network are controlled by a central authority. Indeed, the maximum-flow algorithm finds a flow that directs the flow in all vertices of the network. In many applications of flow networks, however, the vertices correspond to decision-points controlled by different entities. For example, in *communication networks*, routers may belong to different companies, with different destination objectives, and in *software defined networks* (SDNs), vertices may be SDN switches, programmed by different entities [1, 23]. Likewise, *hostile* entities may try to direct the flow to alternative targets or to locations where flow gets stuck. The above examples suggest that the maximum-flow problem should be revisited, and examined from a game-theoretic perspective. Beyond the applications, such a study is interesting from a theoretical point of view. Indeed, both the maximum-flow problem and algorithmic game theory are fundamental topics in theoretical computer science, and their combination involves interesting ideas and tools from both topics.

We introduce and study *multi-player flow games* (MFGs, for short).[1] Essentially, the vertices of an MFG are partitioned among the players, and a player that owns a vertex directs the flow that reaches it. Each player has a different target vertex, and the objective of the players is to maximize the flow that reaches their target vertices. A strategy for a player advises her how to direct flow that enters vertices under her control. Formally, for each vertex u, let $E^{u\to}$ denote the set of edges outgoing from u. Also, for each edge e, let $c(e) \in \mathbb{N}$ denote its capacity. Then, for each vertex u controlled by the player, a strategy for the player includes a policy $f_u : \mathbb{N} \to \mathbb{N}^{E^{u\to}}$ that maps every incoming flow $x \in \mathbb{N}$ to a function describing how x is partitioned among the edges outgoing from u. For each incoming flow $x \in \mathbb{N}$ and edge $e \in E^{u\to}$, we require that $f_u(x)(e) \leq c(e)$ and $\sum_{e\in E^{u\to}} f_u(x)(e) = \min\{x, \sum_{e\in E^{u\to}} c(e)\}$. Thus, $f_u(x)$ assigns to each edge outgoing from u a flow that is

[1] Not to confuse with games in which players *cooperate* in order to construct a subgraph that maximizes the flow in the traditional setting, which are also termed flow games (c.f., [14]).

bounded by its capacity. Also, when the incoming flow is larger than the capacity of the outgoing edges (which bounds the outgoing flow), then flow is lost and the outgoing flow is lower than the incoming flow. In addition, an initial-flow function assigns an initial flow to some of the vertices. The game is played among k players. Each Player $i \in \{1, \ldots, k\}$ has a target vertex t_i, and the goal of Player i is to maximize the flow that enters t_i. Note that the definition of flow in an MFG is different from the traditional definition of maximum flow, which corresponds to the case where all vertices belong to a single player, and in which the "flow conservation" property is respected in all vertices. Indeed, in the game setting, flow may get lost when it reaches vertices whose outgoing capacity is smaller than the incoming flow. We assume that the network is *acyclic*. Then, given strategies for the players, it is possible to calculate the flow by following a topological ordering of the vertices.

Example 1.1. Consider the MFG $\mathcal{G}$ appearing in Figure 1. The game is played between two players. The vertices of Player 1 are circles, and those of Player 2 are squares. An initial flow of 2 arrives to vertex s, and the targets of the players are t_1 and t_2. We can view $\mathcal{G}$ as a communication network with routers operated by companies with different targets. Unless outgoing channels are filled, a router does not drop packets that reach it, and it can direct the packets however it chooses.

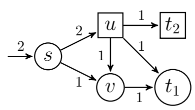

Figure 1: The MFG $\mathcal{G}$

No matter how Player 1 directs the flow in vertex s, a flow of at least 1 reaches t_1. Indeed, if Player 1 directs 1 to v, then it continues from there to t_1. Also, if Player 1 directs 2 to u, then Player 2 directs at most 1 to t_2, and directs the rest, namely at least 1, to v (from where it is directed to t_1) or to t_1. Note that Player 2 may direct no flow to t_2, in which case a flow of 2 reaches t_1, yet Player 2 has no incentive to do so. Moreover, if Player 1 directs 1 to v and 1 to u, and Player 2 directs 1 from u to v, then flow gets lost in v, as the capacity of edges outgoing from v is only 1. □

In [16], the authors introduced and studied *flow games*. Flow games are played on flow networks with a single source and a single target. The vertices in the network are partitioned between two players, MAX and MIN. Player MAX corresponds to the network authority, whose goal is to maximize the flow from the source to the target, while MIN corresponds to a hostile environment, whose goal is to minimize this flow. The authors studied the problem of finding a strategy for MAX that maximizes the flow against every strategy of MIN, and showed that the problem is Σ_2^P-complete. They also studied some theoretical properties of flow games, in particular a restriction to strategies that ensures no loss of flow, and an extension to strategies that allows non-integral flows, which were proved to be stronger. While flow games are strongly related to two player MFGs, they cannot model two-player MFGs, as MIN does not have a target vertex, and her only goal is to minimize the flow that MAX directs to the target. Dually, MFGs cannot model flow games. In particular, adding a target vertex for MIN to which flow may be directed does not work, as, by the definition of flow in MFGs, flow may be directed to this target only after outgoing edges to other vertices are saturated. More importantly, as we elaborate below, the questions on MFGs that we study here originate from its game-theoretic nature, and are very different from those studied in [16].

In order to describe our contribution, we first need some notations. A *profile* in an MFG is a tuple of strategies, one for each player. Primary questions about games in traditional game-theory applications concern their stability. The most common criterion for stability is the existence of a *Nash equilibrium* (NE, for short) [18]: a profile in which no (single) player can benefit from unilaterally changing her strategy.[2] It is well known that decentralized decision-making may lead to stable profiles that are sub-optimal from the point of view of society as a whole. Formally, a profile is a *social optimum* (SO, for short) if it maximizes the flow to all target vertices together. An SO thus corresponds to a maximum flow in a network obtained from the MFG by adding a source vertex in which the initial flow is generated, and a target vertex to which all target vertices are connected. The inefficiency incurred due to selfish behavior of the players is measured by the *price of anarchy* (PoA) [15, 20] and *price of stability* (PoS) [4] measures. The PoA is the worst-case inefficiency of an NE (that is, the ratio between the flow in an SO and in a worst NE, namely one in which minimum flow reaches all targets). The PoS is the best-case inefficiency of a Nash equilibrium (that is, the ratio between the flow in an SO and a best NE). Another important question in game-theory applications is that of finding a *best-response* move, namely a strategy that maximizes the utility of a given player (that is, the flow to her target, in the case of MFGs), given the strategies of the other players. The absence of regulation by some central authority is a driving theme of *algorithmic game theory*, cf. [19], inspired by the open nature of today's computing environments.[3]

We start with some bad news about the stability of MFGs. We show that there are simple (in fact, two-player) MFGs in which no NE exists. Moreover, the PoA and even the PoS of MFGs are unbounded. That is, for every threshold $x \geq 1$, there is an MFG $\mathcal{G}_x$ such that the SO in $\mathcal{G}_x$ is x (that is, when cooperating, the players can direct x units of flow to their targets), whereas a best NE in $\mathcal{G}_x$ is 1 (that is, in all stable profiles, only 1 flow unit reaches a target vertex). Also, the problem of deciding whether a given MFG has an NE is Σ_2^P-complete, which essentially suggests that we have to go over all possible profiles and deviations from them. We continue with the best-response problem and show that it is NP-complete. The high complexity is not surprising, and corresponds to the known computational price when moving from a nondeterministic setting to a game-based one, for example the increase from PSPACE to 2EXPTIME when moving from temporal satisfiability [17] to temporal realizability [21].

We continue with some good news and consider a variant of MFGs in which flow may be dropped (MFGD, for short). Thus, an

[2]Throughout this paper, we consider *pure* strategies. Unlike mixed strategies, pure strategies may not be random or drawn from a distribution.

[3]Different aspects of networks have already been extensively studied from the perspectives of algorithmic game theory. This includes, for example, network formation games [4] or incentive issues in interdomain routing and the BGP protocol [9]. We are the first, however, to consider the maximum-flow problem from this perspective.

owner of a vertex may choose not to direct some of the incoming flow. In particular, when Player i owns a vertex from which her target cannot be reached, then she has no incentive not to drop the flow. We show that, surprisingly, while this model seems to incentivize the above selfish behavior, it is actually stable, and with no stability inefficiency. Thus, MFGDs always have an NE, and their PoS is 1. Moreover, such an NE that is also an SO can be found in polynomial time. Our algorithm is based on a careful choice of *augmenting paths* in the Ford-Fulkenson method [11], chosen in a way that guarantees that no player has an incentive to deviate from the profile that induces the maximum flow found by the algorithm. We show that this careful choice is acute, as the PoA of MFGs with drops is unbounded.

Recall that the capacities in an MFG are integral and the strategies of the players can assign only integral flows. Integral-flow MFGs arise naturally in settings in which the objects we transfer along the network cannot be partitioned into fractions, as is the case with cars, packets, and more. Sometimes, however, as in the case of liquids, flow can be partitioned arbitrarily. In the traditional maximum-flow problem, it is well known that when the capacities are integral, then there exists an integral maximum flow. We study an extension of MFGs to *non-integral strategies*. We show that, interestingly, non-integral strategies are stronger, in the sense they can guarantee strictly greater outcomes. Despite the richness of non-integral strategies, we can show that our results are carried over to the non-integral case.

2 PRELIMINARIES

For $k \geq 1$, let $[k] = \{1, \ldots, k\}$. A *multi-player flow game* (MFG) is $\mathcal{G} = \langle k, V, E, c, (t_i)_{i \in [k]}, init, owns \rangle$, where k is the number of players, V is a set of vertices, $E \subseteq V \times V$ is a set of directed edges, and $c : E \to \mathbb{N}$ is a capacity function, assigning to each edge an integral amount of flow that the edge can transfer. For a vertex $u \in V$, let $E^{\to u}$ and $E^{u \to}$ be the sets of incoming and outgoing edges to and from u, respectively. That is, $E^{\to u} = (V \times \{u\}) \cap E$ and $E^{u \to} = (\{u\} \times V) \cap E$. A *sink* is a vertex u with no outgoing edges, thus $E^{u \to} = \emptyset$. For each $i \in [k]$, the vertex $t_i \in V$ is a target vertex for Player i. We assume that the targets t_i are distinct, i.e., $t_i \neq t_j$ for all $i \neq j$, and that t_i is a sink for all $i \in [k]$. Let $T = \{t_1, \ldots, t_k\}$. The function $init : V \to \mathbb{N}$ is an initial-flow function, assigning to each vertex an initial flow. Finally, the function $owns : V \to [k]$ assigns to each vertex a player that owns it. We assume that for all $i \in [k]$, we have that $owns(t_i) = i$, and we use V_i to denote the set of vertices owned by player i, thus $V_i = \{v : owns(v) = i\}$. We assume that the capacities and the initial flows are given in unary.

When drawing two-player MFGs, we use circles and squares to describe the vertices of Player 1 and 2, respectively, and use dark filled circles to describe sinks (ownership of sinks is not important). The function *init* is described by edges entering vertices, each labeled with the corresponding initial flow.

A *policy* for a vertex $u \in V \setminus T$ is a function that distributes an incoming flow to the outgoing edges. Formally, a policy for u is a function $f_u : \mathbb{N} \to \mathbb{N}^{E^{u \to}}$ such that for every flow $x \in \mathbb{N}$ and edge $e \in E^{u \to}$, we have $f_u(x)(e) \leq c(e)$ and $\sum_{e \in E^{u \to}} f_u(x)(e) = \min\{x, \sum_{e \in E^{u \to}} c(e)\}$. Thus, $f_u(x)$ assigns to each edge outgoing from u a flow that is bounded by its capacity. Also, when the incoming flow is larger than the sum of the capacities of the outgoing edges (which bounds the outgoing flow), then flow *leaks* and the outgoing flow is lower than the incoming flow. In practice, leaks correspond to either actual leaks – fluid in a pipe system that is lost when the system is overflowed, or to packets that are dropped by routers all of whose outgoing channels are filled. Note that this is different from the traditional definition of flow in a network, which corresponds to the case where all vertices belong to a single player, and in which the "flow conservation" property is respected. Note that as the capacities and initial flows are given in unary, a policy is polynomial in the size of the MFG.

A *flow* in an MFG is a function $f \in \mathbb{N}^E$ that assigns to each edge the flow that travels in it. We require that for every edge $e \in E^{u \to}$, we have $f(e) \leq c(e)$, and for every vertex $u \in V \setminus T$, we have $\sum_{e \in E^{u \to}} f(e) = \min\{init(u) + \sum_{e \in E^{\to u}} f(e), \sum_{e \in E^{u \to}} c(e)\}$. That is, the flow in each edge is bounded by its capacity, and the flow that leaves each vertex is the minimum of the flow that enters the vertex, by the initial flow or from its neighbors, and the sum of the capacities of edges outgoing from it. We focus on the case where the graph $\langle V, E \rangle$ is acyclic. Then, given policies f_u for all vertices in $u \in V \setminus T$, we can calculate the flow in the game as follows. First, we order the vertices in a topological ordering. If a vertex v_2 can be reached from a vertex v_1 along some path, then v_2 appears after v_1 in the topological ordering. We start from the first vertex u in the topological ordering, and use f_u to assign a flow to each edge in $E^{u \to}$. Now, we continue to the next vertex in the topological ordering. Whenever we reach a vertex v, the incoming flow to v, denoted x, has already been calculated. We then use $f_v(x)$ to assign a flow for each edge in $E^{v \to}$, and continue along the topological ordering until we reach all targets in T. Since the flow that enters a vertex u depends only on the sub-game that reaches u, it is easy to see that the calculation above is independent of the topological ordering. Indeed, if u_1 and u_2 are not ordered, then flow that leaves u_1 does not reach u_2, and vice versa.

A strategy of Player i is a collection of policies, one for each vertex in $V_i \setminus \{t_i\}$. A *profile* $P = \langle \pi_1, \ldots, \pi_k \rangle$ is a vector of strategies, one for each player. For a profile P and a strategy π of Player $i \in [k]$, let $P[i \leftarrow \pi]$ denote the profile obtained from P by replacing the strategy of Player i in P by π. Given a profile P, the flow in which the players follow their strategies in P is denoted f^P and can be calculated as described above. Given a profile P, the *outcome of Player i*, denoted $outcome_i(P)$, is the amount of flow that reaches her target t_i, thus $outcome_i(P) = \sum_{e \in E^{\to t_i}} f^P(e)$. The *outcome* of a game for profile P is then $outcome(P) = \sum_{i=1}^{k} outcome_i(P)$, namely the flow that reaches all the targets in T.

A profile of strategies is a *Nash equilibrium* (NE, for short) if no (single) player can increase her outcome by unilaterally changing her strategy. Given an MFG $\mathcal{G}$, the set of NEs of $\mathcal{G}$ is denoted by NE($\mathcal{G}$). A *social optimum* (SO, for short) is a profile in which the outcome of $\mathcal{G}$ is maximized. An NE need not be an SO. The standard measures to quantify the inefficiency caused due to the selfish behavior of the players is to compare the outcome of the NEs with that of the SO. Specifically, the *price of stability* (PoS) is the ratio between the SO and the outcome of a best NE; formally, $\text{PoS}(\mathcal{G}) = \min_{P \in \text{NE}(\mathcal{G})} outcome(SO)/outcome(P)$, and the *price of* *anarchy* (PoA) is the ratio between the SO and the outcome of a worst NE; formally, $\text{PoA}(\mathcal{G}) = \max_{P \in \text{NE}(\mathcal{G})} \mathit{outcome}(SO)/\mathit{outcome}(P)$. We note that since the objective in MFGs is to maximize the outcome, the PoS and PoA ratios have the outcome of the SO in the numerator, as opposed to games in which the outcome is associated with costs and the objective is to minimize it.

3 EQUILIBRIA IN MFGS

In this section we study equilibria and its inefficiency in MFGs. Our results are negative: An MFG need not have a Nash Equilibrium, and deciding the existence of an NE in a given MFG is Σ_2^P-complete. Moreover, the Price of Anarchy and even the Price of Stability of MFGs are unbounded.

Theorem 3.1. *There exists an MFG with no NE.*

Proof. Consider the MFG $\mathcal{G} = \langle 2, V, E, c, (t_1, t_2), \mathit{init}, \mathit{owns} \rangle$ appearing in Figure 2. The edges for which the capacity is not specified have capacity 1.

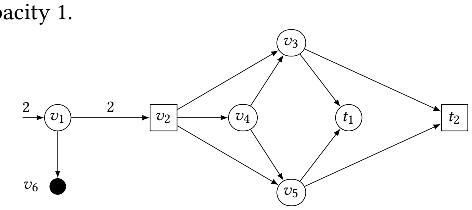

Figure 2: An MFG with no NE

We claim that there is no NE in $\mathcal{G}$. Consider a profile P. First, if $\mathit{outcome}_1(P) < 2$, then we claim that Player 1 has a beneficial deviation that increases her outcome to 2. Indeed, let f_{v_2} be the policy of Player 2 in v_2, and let x_3, x_4, and x_5 be the flow that Player 2 directs to v_3, v_4, and v_5, respectively, when a flow of 2 reaches v_2. Formally, $x_3 = f_{v_2}(2)(\langle v_2, v_3 \rangle)$, $x_4 = f_{v_2}(2)(\langle v_2, v_4 \rangle)$, and $x_5 = f_{v_2}(2)(\langle v_2, v_5 \rangle)$. A strategy of Player 1 that ensures an outcome of 2 then directs all the initial flow into v_2, thus $f_{v_1}(2)(\langle v_1, v_2 \rangle) = 2$, and directs the flow in v_4 so that the incoming flow into both v_3 and v_5 is 1. Since $x_3, x_4, x_5 \in \{0, 1\}$ and $x_3 + x_4 + x_5 = 2$, this is possible. Specifically, if $x_3 = x_4 = 1$, then the flow in v_4 is directed to v_5; dually, if $x_5 = x_4 = 1$, then the flow in v_4 is directed to v_3, and if $x_3 = x_5 = 1$, then the policy in v_4 is irrelevant. Now, when Player 1 directs the flow in v_3 and v_5 to t_1, then each of them contributes 1 to the flow, thus the flow reaching t_1 is 2, and we are done.

Now, if $\mathit{outcome}_1(P) = 2$, then we claim that Player 2 has a beneficial deviation that increases her outcome from 0 to 1. First, note that in order for $\mathit{outcome}_1(P)$ to be 2, it must be that Player 1 directs all the initial flow to v_2. Also, since $\mathit{outcome}_1(P) = 2$, it must be that $\mathit{outcome}_2(P) = 0$. Moreover, the flow of 2 that gets to t_1 must arrive from v_3 and v_5. Let f_{v_4} be the policy of Player 1 in v_4, and let x_3 and x_5 be the flow that Player 1 directs to v_3 and v_5, respectively, when a flow of 1 arrives to v_4. Formally, $x_3 = f_{v_4}(1)(\langle v_4, v_3 \rangle)$ and $x_5 = f_{v_4}(1)(\langle v_4, v_5 \rangle)$. Consider a strategy for Player 2 in which the policy at v_2 is such that when an incoming flow of 2 arrives, then 1 is directed to v_4, and in addition, if $x_3 \geq x_5$, then 1 unit is directed to x_3, and if $x_3 < x_5$, then 1 is directed to x_5. The above policy ensures that one of the vertices v_3 or v_5 has an incoming flow of 2. Accordingly, even a policy of Player 1 that first saturates the edges to t_1 has to direct 1 into t_2, and we are done.

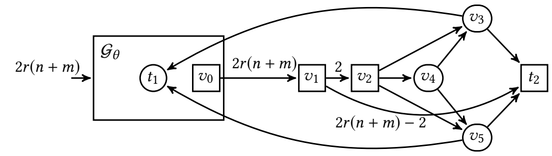

Figure 3: An NE exists iff θ is satisfiable

It follows that in each profile at least one player has an incentive to change her strategy, thus no profile is an NE. □

Theorem 3.1 gives rise to the *exists-NE problem*, namely deciding, given an MFG $\mathcal{G}$, whether $\mathcal{G}$ has an NE.

Theorem 3.2. *The exists-NE problem for MFGs is Σ_2^P-complete.*

Proof. We start with the upper bound. Recall that a strategy for Player i is a collection of policies $f_u : \mathbb{N} \to \mathbb{N}^{E^{u\to}}$, for all $u \in V_i$. Clearly, the policy has to refer only to incoming flow that is smaller or equal to the sum of the capacities of the edges in $E^{\to u}$ and the initial flow assigned to u by *init*. Thus, since we assume that capacities are given in unary, the description of strategies is polynomial in the input. Given a profile P, checking whether there exists a beneficial deviation for some player is in NP. Consequently, deciding whether there exists a profile P from which no player has a beneficial deviation can be solved by a nondeterministic polynomial-time Turing machine with an NP oracle.

We continue to the lower bound and describe a reduction from QBF_2: satisfiability for quantified Boolean formulas with 2 alternations of quantifiers, where the most external quantifier is "exists". Let ψ be a Boolean propositional formula over the variables $x_1, \ldots, x_n, y_1, \ldots, y_m$ and let $\theta = \exists x_1 \ldots \exists x_n \forall y_1 \ldots \forall y_m \psi$. We assume that ψ is in positive normal form in which every literal appears r times.

We are going to use as a black box the following reduction, a variant of which is proven in [16].

Lemma 3.3. *Given a QBF_2 formula $\theta = \exists x_1 ... \exists x_n \forall y_1 ... \forall y_m \psi$ in which every literal appears r times, we can construct a two-player MFG $\mathcal{G}_\theta$ with targets t_1 and v_0 and an initial flow of $2r(n+m)$, such that if θ is satisfiable, then Player 1 has a strategy that ensures that a flow of 1 reaches t_1 and a flow of $2r(n+m)-1$ reaches v_0, and if θ is not satisfiable, then Player 2 has a strategy that ensures that no flow reaches t_1 and a flow of $2r(n+m)$ reaches v_0.*

Consider the MFG $\mathcal{G}$ appearing in Figure 3. Note that $\mathcal{G}$ combines the MFG $\mathcal{G}_\theta$ from Lemma 3.3 with the "no-NE" example from Theorem 3.1. We prove that $\mathcal{G}$ has an NE iff θ is satisfiable. Assume first that θ is satisfiable. Then, by Lemma 3.3, Player 1 can ensure that a flow of 1 reaches t_1 and a flow of $2r(n+m)-1$ reaches v_1. Consider a strategy of Player 2 in which she directs a flow of $2r(n+m)-2$ from v_1 to t_2 and the remaining flow of 1 to v_2. Arguing, in the same way as in Theorem 3.1, we can see that Player 1 has a strategy such that now a total flow of 2 units reaches t_1 and the remaining flow of $2r(n+m)-2$ units reaches t_2. We claim that this profile is an NE. In the game $\mathcal{G}_\theta$, Player 1 can ensure a maximum flow of 1 to t_1 while the remaining flow of $2r(n+m)-1$ reaches v_0 which is forwarded to v_1. If Player 2, now forwards a flow of 1 to v_2 from v_1, Player 1 can ensure that this 1 unit of flow reaches t_1 and thus a total flow of 2 units reaches t_1. Hence given the strategy of

Player 2, Player 1 does not have a strategy to ensure that more flow reaches t_1. Now we show that Player 2 also cannot deviate from her current strategy and increase her flow. In particular, Player 2 can change her policy in v_1 and send 2 units of flow to v_2 while the remaining flow of $2r(n+m)-3$ is sent to t_2. Even in this case, out of the 2 units of flow reaching v_2, no more than a flow of 1 can reach t_2 and hence Player 2 does not have a deviation from her strategy that increases her flow.

Assume now that θ is not satisfiable. Then, by Lemma 3.3, Player 2 can ensure that a flow of $2r(n+m)$ reaches v_1. The only policy of Player 2 at v_1 is to send a flow of $2r(n+m)-2$ units to t_2 and the remaining flow of 2 units to v_2. The same arguments used in the proof of Theorem 3.1 imply that the profile we have is not an NE. Further, when θ is unsatisfiable, we note that for every strategy of Player 1, Player 2 has a strategy such that a flow of $2r(n+m)-1$ reaches t_2 while the remaining flow of 1 reaches t_1 and for every strategy of Player 2, Player 1 has a strategy such that a flow of 2 units reaches t_1, and hence an NE cannot exist. □

We continue to study the PoA and PoS, for the cases where an NE exists, and show that they are both unbounded.

Theorem 3.4. *The PoA in MFGs is unbounded.*

Proof. Consider the two-player MFG $\mathcal{G}_x$ appearing in Figure 4.

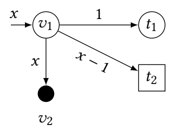

Figure 4: An MFG with unbounded PoA

The outcome of an SO of $\mathcal{G}_x$, namely the maximum flow to t_1 and t_2 together, is x, obtained when Player 1 directs all the initial flow to t_1 and t_2. On the other hand, consider a profile in which Player 1 directs to t_1 only a flow of 1 and directs to the sink v_2 a flow of $x-1$. The profile is an NE, yet its outcome is only 1. Thus, $PoA(\mathcal{G}_x) \geq x$. Since x can be unbounded, we are done. □

Theorem 3.4 is not too surprising, as the PoA considers worst NEs. We now show that the PoS is unbounded too. Here, we have to bound the best NE, which is technically much more challenging.

Theorem 3.5. *The PoS in MFGs is unbounded.*

Proof. Consider the MFG $\mathcal{G}_x$ appearing in Figure 5. For simplicity, we assume that x is even.

It is easy to see that the maximum flow to t_1 and t_2 together, namely the outcome of an SO, is x. We show that $\mathcal{G}_x$ has an NE, yet no NE has an outcome of more than 1. Consider the profile P in which Player 1 directs a flow of 1 from a to b and the policy of Player 2 in b for an incoming flow of $y \geq 1$ is as follows: a flow of 1 is directed to c_1 and a flow of $y-1$ is directed to v_2. It is not hard to see that the profile P is an NE.

Now we show that no NE has an outcome of more than 1 in $\mathcal{G}$. Consider a profile P such that $outcome(P) > 1$. Then, there must be a flow of $y > 1$ from b. If $outcome_2(P) > 0$, in which case $outcome_1(P) < y$, then Player 1 can change her strategy and ensure that the entire flow of y reaches t_1. Indeed, no matter how Player 2

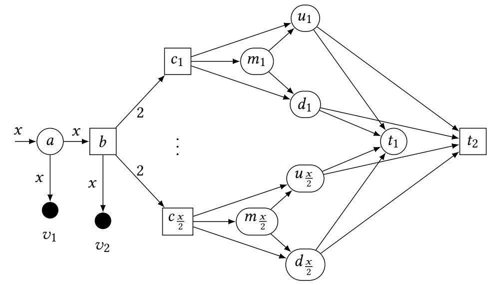

Figure 5: An MFG with unbounded PoS

directs the flow from b and from the c_j vertices, Player 1 can direct all of it to t_1. On the other hand, if $outcome_2(P) = 0$, then Player 2 has the following beneficial deviation. In b, she directs a flow greater than 1 to some c_j, for $j \in [\frac{x}{2}]$, and in c_j, she directs the flow so that either u_j or d_j have incoming flow greater than 1. Thus, if the policy of Player 1 in m_j is to direct the flow to u_j, then Player 2 directs a flow of 1 to u_j and a positive flow to m_j. Now, since from u_j Player 1 can direct only a flow of 1 to t_1, then a positive flow reaches t_2. It follows that no NE with an outcome greater than 1 exists.

Since x, and hence the SO, is unbounded, so is the PoS. □

4 THE BEST-RESPONSE PROBLEM

Given an MFG $\mathcal{G}$ with k players, a profile P, and an index $i \in [k]$, a strategy π_i of Player i is a *best response* with respect to P if $outcome_i(P[i \leftarrow \pi_i]) \geq outcome_i(P[i \leftarrow \pi_i'])$ for all strategies π_i' of Player i. That is, π_i is a strategy that maximizes the outcome of Player i assuming the other players do not change their strategies in P. In this section we study the computational complexity of the best-response problem, namely the question of deciding, given P, i, and a threshold $\lambda \in \mathbb{N}$, whether there exists a strategy π_i of Player i such that $outcome_i(P[i \leftarrow \pi_i]) \geq \lambda$.

Theorem 4.1. *The BR problem for MFGs is NP-complete.*

Proof. Membership in NP is easy. Given a profile P in an MFG, a strategy π_i of Player i, and a threshold λ, it can be checked in polynomial time whether $outcome_i(P[i \leftarrow \pi_i]) \geq \lambda$.

We prove NP-hardness by a reduction from CNF-SAT. We consider a normal form for propositional formulas in which all literals appear the same number of times. Consider a propositional formula ψ over n variables $x_1, \ldots, x_n$. We assume that ψ has m clauses and every literal appears in exactly r clauses. It is not hard to see that every CNF formula can be translated in polynomial time to one that satisfies the above assumption. We construct, in polynomial time, a two-player MFG $\mathcal{G}_\psi$ and a strategy π_2 of Player 2 such that Player 1 has a best response π_1 with $outcome_1(\langle \pi_1, \pi_2 \rangle) \geq (m-r) \cdot n$ iff ψ is satisfiable.

A scheme of the MFG $\mathcal{G}_\psi$ appears in Figure 6. The MFG has three types of vertices: (1) Variable vertices $v_1, \ldots, v_n$, (2) Literal vertices $x_1, \overline{x}_1, \ldots, x_n, \overline{x}_n$, and (3) Clause vertices $C_1, \ldots, C_m$. Since we examine the best response of Player 1, the target vertex of Player 2 is not important, and can be one of the sinks.

For a literal l and a clause C_j, we have an edge $\langle l, C_j \rangle$ iff the literal l does not appear in the clause C_j in ψ. Since there are m clauses and each literal appears in exactly r clauses, then each literal vertex

l has outgoing edges to $m - r$ clause vertices and an outgoing edge to a sink. Consider the strategy π_2 of Player 2 in which for each literal vertex, if the incoming flow is exactly $m - r$, then the entire flow is directed to the clause vertices, and otherwise the flow is directed to the sink. We claim there is a best response π_1 such that $outcome_1(\langle \pi_1, \pi_2 \rangle) = (m - r) \cdot n$ iff ψ is satisfiable.

Assume first that ψ is satisfiable. Consider a satisfying assignment to ψ. Let π_1 be the strategy for Player 1 that directs the initial flow of $m - r$ in each variable vertex to the corresponding literal vertex that is assigned true. Thus, n of the $2n$ literal vertices have an incoming flow of $m - r$ each, while the remaining n literal vertices have an incoming flow of 0. Since every clause C_j is satisfied in the assignment, at least one of the literals appearing in C_j is assigned true. Let l be such a literal. By the construction, the edge $\langle l, C_j \rangle$ does not appear in $\mathcal{G}_\psi$. Thus, the maximum incoming flow to C_j is $n - 1$. The strategy π_1 directs all the incoming flow to C_j to the target vertex t_1, and thus no flow is lost. Hence, if ψ is satisfiable, then Player 1 has a strategy π_1 such that $outcome_1(\langle \pi_1, \pi_2 \rangle) = (m - r) \cdot n$.

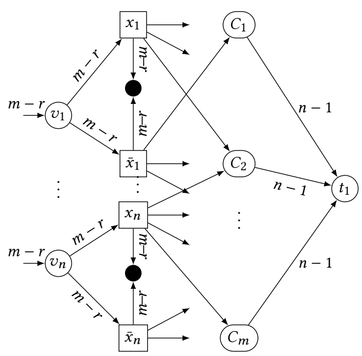

Figure 6: NP hardness of the BR problem

Assume now that ψ is not satisfiable. Then, for every assignment of the variables, there is a clause C_j that is not satisfied. Thus, none of the literals that appear in C_j is assigned true. Recall that corresponding to a literal l not appearing in C_j, there is an edge $\langle l, C_j \rangle$ in $\mathcal{G}_\psi$. Thus, each literal l' that is assigned true has an edge $\langle l', C_j \rangle$, implying that the incoming flow to C_j is n, whereas the capacity of the outgoing edge from C_j is $n - 1$. Hence, there is a flow loss in C_j, implying that $outcome_1(\langle \pi_1, \pi_2 \rangle) < (m - r) \cdot n$. □

5 MFGS WITH DROPS

Recall that in MFGs, flow is lost only if it reaches a vertex whose outgoing capacity is lower than its incoming flow. In this section we study multiplayer flow games *with drops* (MFGD, for short), where incoming flow is allowed to be dropped whenever the player chooses, even if the outgoing capacity is not full. Thus, the players have full control on their vertices and they may drop flow if they wish. This setting is useful, for example, in switched networks in which routers can choose to drop packets. Formally, a policy for a vertex u is a function $f_u : \mathbb{N} \to \mathbb{N}^{E^{u\to}}$ such that for every flow $x \in \mathbb{N}$ and edge $e \in E^{u\to}$, we have $f_u(x)(e) \leq c(e)$ and $\sum_{e \in E^{u\to}} f_u(x)(e) \leq x$.

Note that when Player i owns a vertex from which her target cannot be reached, then she has no incentive not to drop the flow. Thus, the MFGD model seems to be less optimal for the society as a whole. We show that, surprisingly, it is actually stable, and with no stability inefficiency.

Theorem 5.1. *Every MFGD has an NE. Furthermore, The PoS in MFGD is 1, and an NE that is also an SO can be found in polynomial time.*

Proof. Consider an MFGD $\mathcal{G} = \langle k, V, E, c, (t_i)_{i\in[k]}, init, owns \rangle$. We show an algorithm for finding an SO that is also an NE. Consider the flow network $\mathcal{G}'$ obtained from $\mathcal{G}$ by adding a source vertex s from which the initial flow is directed, and a target vertex t to which all target vertices may direct their flow. Formally, $\mathcal{G}' = \langle V', E', c', s, t \rangle$, where $V' = V \cup \{s, t\}$, $E' = E \cup (\{s\} \times V) \cup (T \times \{t\})$, and the capacities are $c'(e) = c(e)$, for $e \in E$, $c'(\langle s, u \rangle) = init(u)$, for $u \in V$, and $c'(\langle t_i, t \rangle) = C$, for some large C, for $t_i \in T$. Our algorithm follows the Ford-Fulkerson method [5, 11] (FF method, for short) for finding a maximum flow from s to t in $\mathcal{G}'$, where the augmenting paths are chosen in a way that would guarantee the stability of the induced profile.

Before we describe the algorithm, let us briefly review the FF method. We start with a flow function f for which $f(e) = 0$ for all edges $e \in E$, giving an initial flow value of 0. At each iteration, we improve the flow f in $\mathcal{G}'$ by finding an augmenting path in an associated residual network $\mathcal{G}'_f$. The residual network $\mathcal{G}'_f$ consists of edges with capacities that represent how we can change the flow f in $\mathcal{G}'$. Essentially, these are either edges of $\mathcal{G}'$ that are not saturated in f, in which case their capacity in $\mathcal{G}'_f$ is the difference between their capacity and the flow that f assigns to them, indicating it can be increased in this amount, or reverse of edges to which f assigns a positive flow, in which case their capacity in $\mathcal{G}'_f$ is this flow, indicating that it can be decreased in this amount. Once we find an augmenting path from s to t in $\mathcal{G}'_f$, we can identify the edges in $\mathcal{G}'$ for which we can change f and obtain an improved flow. We repeat this process until the residual network has no augmenting path, which implies we have reached a maximum flow.

We use a variant of the FF method in which after an augmenting path is found, the improved flow is obtained by transferring a flow of 1 in it. Thus, in each iteration, the value of the flow increases by 1. In addition, the residual path is found as follows: For a subset $H \subseteq V'$ and two vertices $u, v \in V$, we say that v is *H-reachable* from u if there is a path from u to v that visits only vertices in H (in particular, $u, v \in H$).

In each iteration of our algorithm we start with a current flow f in $\mathcal{G}'$ and find a simple path ρ from s to t in $\mathcal{G}'_f$. Then, we check for every vertex $u \in \rho$, by the order of ρ, whether the player that owns u can "take control of" the path. That is, if $u \in V_i$, then we check whether t_i is V_i-reachable from u in $\mathcal{G}'_f$. If the answer is yes, then we change the path ρ to a path ρ' that is the concatenation of the subpath of ρ from s to u with a simple path from u to t through t_i that visits only vertices in V_i. We use ρ' as the augmenting path. If no player can take control of ρ, then we use ρ as the augmenting path. Clearly, this algorithm follows the FF method and thus it gives a maximum flow from s to t in $\mathcal{G}'$. We denote this flow by $f : E' \to \mathbb{N}$.

Let P be the profile of strategies induced from f as follows. Consider a vertex $u \in V$. Let x_u be the incoming flow to u in f. The policy for u is then $f_u(y)(e) = f(e)$, for every $y \geq x_u$ and

$e \in E^{u\rightarrow}$, and $f_u(y)(e) = 0$, for every $y < x_u$ and $e \in E^{u\rightarrow}$. Thus, if the incoming flow to u is at least the flow that enters u in f, then the flow in $E^{u\rightarrow}$ according to f_u agrees with f. Otherwise, namely if the incoming flow to u is strictly smaller than the flow that enters u in f, then all the incoming flow is dropped. Note that $outcome(P)$ is the value of f, and thus P is an SO. We now show it is an NE.

Consider a player $i \in [k]$. We show that Player i has no beneficial deviation from P. Let $\mathcal{G}'_i = \langle V'_i \cup \{s'_i\}, E'_i, c'_i, s'_i, t_i\rangle$ be a flow network induced by $\mathcal{G}'$, where V'_i is the vertices $u \in V_i$ such that t_i is V_i-reachable in $\mathcal{G}'$ from u, and s'_i is a new source vertex. Note that $t_i \in V'_i$. The flow network $\mathcal{G}'_i$ contains the edges $e \in E' \cap (V'_i \times V'_i)$ with capacities $c'_i(e) = c'(e)$, and edges from s'_i to V'_i. For every vertex $v \in V'_i$, we define $c'_i(\langle s'_i, v\rangle) = \sum_{u \in V' \setminus (\{s\} \cup V'_i)} f(\langle u, v\rangle) + init(v)$. Note that by changing her strategy in the MFGD $\mathcal{G}$, Player i cannot direct to a vertex $v \in V'_i$ an incoming flow of more than $c'_i(\langle s'_i, v\rangle)$ from outside of V'_i, since the incoming flow to v from a vertex $u \in V_j$ for $j \neq i$ is bounded by $f(\langle u, v\rangle)$ according to the strategy of Player j, and the initial incoming flow to v is $init(v)$. Let $|f_i|$ be maximum flow in $\mathcal{G}'_i$. In order to prove that Player i has no beneficial deviation from P, we prove that $|f_i| = outcome_i(P)$. Thus, Player i cannot increase the incoming flow to a vertex $u \in V'_i$ from outside of V'_i beyond $c'_i(\langle s'_i, u\rangle)$, and under this restriction she cannot increase the flow that reaches t_i.

In order to prove that $|f_i| = outcome_i(P)$, we describe a run of the FF method on $\mathcal{G}'_i$ that is induced by the run on $\mathcal{G}'$ described above. Intuitively, if an augmenting path in the residual graph of $\mathcal{G}'$ visits V'_i, then each time it enters V'_i, it either reaches t_i, in which case it induces a path from s'_i to t_i in the residual graph of $\mathcal{G}'_i$, or it leaves V'_i, in which case it induces a path in the residual graph of $\mathcal{G}'_i$ from s'_i to some other vertex. According to the way we chose the augmenting paths in the residual graphs of $\mathcal{G}'$, if an augmenting path leaves V'_i then t_i is not V'_i-reachable from the vertices in this augmenting path. We partition the set V'_i in the residual graph of $\mathcal{G}'$ to a subset $H \subseteq V'_i$ that contains the vertices from which t_i is V'_i-reachable, and a subset $H' = V'_i \setminus H$. Note that there are no edges from H' to H in the residual graph. Also, according to the way we choose the augmenting paths, it is not possible that a path visits a vertex in H and moves to a vertex in H'. Thus, t_i is not V'_i-reachable from the vertices in H' in the residual graphs of $\mathcal{G}'$ also in the subsequent iterations of the algorithm since an edge from H' to H in a subsequent residual graph may appear only if we use an augmenting path that traverses from H to H'. Consider a run of the FF method on $\mathcal{G}'_i$ where the augmenting paths are induced by the augmenting paths in the run of the FF method on $\mathcal{G}'$ that reach t_i. We ignore augmenting paths in the run on $\mathcal{G}'$ that do not reach V'_i, and also ignore subpaths that enter and then leave V'_i. Recall that we use a variant of the FF method in which after an augmenting path is found, the next residual graph is obtained by transferring a flow of 1 in the augmenting path, even if the residual capacity of this path is greater than 1.

We show that in every iteration, the subgraph induced by the vertices from which t_i is V'_i-reachable in the residual graph of $\mathcal{G}'$ is similar to the subgraph induced by the vertices from which t_i is V'_i-reachable in the residual graph of $\mathcal{G}'_i$. This property follows by an induction as follows. Note that a subpath in V'_i of an augmenting path in $\mathcal{G}'$ that enters and leaves V'_i visits only vertices from which t_i is not V'_i-reachable and does not induce an augmenting path in $\mathcal{G}'_i$. A subpath in V'_i of an augmenting path in the residual graph of $\mathcal{G}'$ that visits vertices from which t_i is V'_i-reachable induces an augmenting path in $\mathcal{G}'_i$ and affects both residual graphs similarly.

In order to show that the FF run that we described on $\mathcal{G}'_i$ is valid, we show that after the last iteration, there is no augmenting path in the residual graph of $\mathcal{G}'_i$. Assume that there is a simple augmenting path τ in the residual graph of $\mathcal{G}'_i$ after the last iteration and let $\langle s'_i, u\rangle$ be the first edge in τ. We denote the residual graphs of $\mathcal{G}'$ and $\mathcal{G}'_i$ after the last iteration by $\mathcal{G}''$ and $\mathcal{G}''_i$ respectively. We denote the flow obtained for $\mathcal{G}'_i$ by $g : E'_i \rightarrow \mathbb{N}$, thus, $\mathcal{G}''_i$ is the residual graph of $\mathcal{G}'_i$ for the flow g. Since t_i is V'_i-reachable from u in $\mathcal{G}''_i$ then it is also V'_i-reachable from u in $\mathcal{G}''$. Hence, t_i is also V'_i-reachable from u in the residual graphs in all the iterations of the runs on $\mathcal{G}'$ and $\mathcal{G}'_i$. Every augmenting path that reaches u in the run on $\mathcal{G}'$ induces an augmenting path in the run on $\mathcal{G}'_i$. Therefore, $g(\langle s'_i, u\rangle) = \sum_{v \in (V' \setminus V'_i)} f(\langle v, u\rangle) = f(\langle s, u\rangle) + \sum_{v \in (V' \setminus (V'_i \cup \{s\}))} f(\langle v, u\rangle)$. Since t_i is reachable from u in $\mathcal{G}''$ and there are no augmenting paths in $\mathcal{G}''$ then $f(\langle s, u\rangle) = c'(\langle s, u\rangle) = init(u)$. Thus, $g(\langle s'_i, u\rangle) = init(u) + \sum_{v \in (V' \setminus (V'_i \cup \{s\}))} f(\langle v, u\rangle) = c'_i(\langle s'_i, u\rangle)$. Therefore, we have reached a contradiction to the assumption that τ is an augmenting path in $\mathcal{G}''_i$ that starts with the edge $\langle s'_i, u\rangle$.

Note that the augmenting paths in the run on $\mathcal{G}'_i$ correspond to the augmenting paths in the run on $\mathcal{G}'$ that reach t_i. Hence, the maximum flow in $\mathcal{G}'_i$ equals the incoming flow to t_i according to f, which equals the flow that reaches t_i in the MFGD $\mathcal{G}$ with the profile P.

We now analyze the complexity of the algorithm. In each iteration of the FF run, finding the augmenting path can be done in linear time by solving reachability problems. The number of iterations is the value of the maximum flow, which is bounded by $\sum_{v \in V} init(v)$. Since $init$ is given in unary, the time complexity of the algorithm is polynomial. □

As in the case of MFGs, the PoA for MFGDs is unbounded, and the BR problem for MFGDs is NP-complete. The proofs are similar to these of Theorems 3.4 and 4.1.

Theorem 5.2. *The PoA in MFGDs is unbounded.*

Theorem 5.3. *The BR problem for MFGDs is NP-complete.*

6 NON-INTEGRAL MFGS

Recall that the capacities in an MFG are integral and that a policy for a vertex can assign only integral flows. As discussed in Section 1, integral-flow MFGs arise naturally in settings in which the objects we transfer along the network cannot be divided into fractions. Moreover, sometimes, as in the cases of messages or other information packages, objects can be split up to a known granularity. It is easy to see that by multiplying all capacities by a factor γ and solving an integer-flow game in the obtained game, we get a solution that involves strategies with fractions of $\frac{1}{\gamma}$ in the original game. In the traditional maximum-flow problem, which corresponds to the SO, it is well known that when the capacities are integral, then there exists an integral maximum flow. In this section we study an extension of MFGs to *non-integral strategies*. Let $\mathbb{R}_+$ denote the set of non-negative real numbers. A *non-integral policy* for a vertex $u \in V$ is a function $f_u : \mathbb{R}_+ \rightarrow \mathbb{R}_+^{E^{u\rightarrow}}$ such that for every flow $x \in \mathbb{R}_+$ and edge $e \in E^{u\rightarrow}$, we have $f_u(x)(e) \le c(e)$ and $\sum_{e \in E^{u\rightarrow}} f_u(x)(e) = \min\{x, \sum_{e \in E^{u\rightarrow}} c(e)\}$. We say that a strategy is a *non-integral strategy* if it contains non-integral policies. We first show that, interestingly, non-integral strategies are stronger, in the sense they can guarantee strictly greater outcomes. Formally, for a strategy π_i of Player i and a threshold $\lambda > 0$, we say that π_i *guarantees outcome* λ if for every profile P in which Player i uses π_i, we have that $outcome_i(P) \ge \lambda$.

Theorem 6.1. *There is an MFG with integral capacities and initial flow, and a threshold λ, such that no integral strategy of Player i guarantees outcome λ, yet Player i has a non-integral strategy that guarantees outcome λ.*

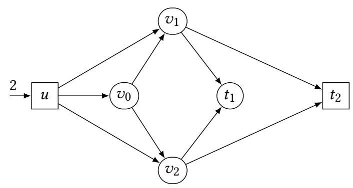

Figure 7: Player 1 can guarantee a flow of 1.5 that cannot be guaranteed using integral strategies

Proof. Consider the MFG $\mathcal{G}$ appearing in Figure 7. Note that $init(u) = 2$, and the capacity of every edge is 1. Consider the following strategy π_1 of Player 1. In vertices v_1 and v_2, if the incoming flow is more than 1, the policy is to direct a flow of 1 to t_1 and the remaining incoming flow to t_2. If the incoming flow is less than or equal to 1, then the policy is to direct the entire flow to t_1. The policy in v_0 is to split an incoming flow equally between v_1 and v_2 Formally, $f_{v_1}(x)(\langle v_1, t_1\rangle) = \min\{1, x\}$ and $f_{v_1}(x)(\langle v_1, t_2\rangle) = \max\{0, x-1\}$, and similarly for v_2. Also, $f_{v_0}(x)(\langle v_0, v_1\rangle) = f_{v_0}(x)(\langle v_0, v_2\rangle) = \frac{x}{2}$. It is not hard to see that π_1 guarantees a flow of 1.5. Also, an integral strategy cannot guarantee a flow of 1.5. To see this, note that for every candidate integral strategy π_1 of Player 1, Player 2 can respond with a strategy that would cause the incoming flow to either v_1 or v_2 to be 2, forcing Player 1 to direct a flow of 1 to t_2.

Note that we could take two copies of $\mathcal{G}$ and obtain an example with a threshold of 3. Thus, the superiority of non-integral strategies applies to both integral and non-integral thresholds. Note also that just multiplying all capacities by 2 is not sufficient for getting an example with threshold 3. □

Theorem 6.1 motivates the study of *Non-Integral MFGs* (NIMFGs, for short), where players may use non-integral strategies. Note that the capacity of the edges and the initial flow assigned by *init* are still integral. We first show that the bad news about the stability of MFGs are carried over to NIMFGs:

Theorem 6.2. *There exists an NIMFG with no NE. The PoA and PoS of NIMFGs are unbounded.*

Proof. We start with the first claim and show that the MFG with no NE described in the proof of Theorem 3.1 does not have an NE even when we allow non-integral strategies. In fact, the proof there stays valid, except that now x_3, x_4, and x_5 are in $[0, 1]$ rather than $\{0, 1\}$. Similarly, the examples for the unbounded PoA and PoS for MFGs, described in the proofs of Theorems 3.4 and 3.5 apply also to NIMFGs. □

On the positive side, since the SO involves integral flows, and the profile described in the proof of Theorem 5.1 is resistant also to deviations by non-integral strategies, the good news about the PoS of MFGDs being 1 stays valid in the non-integral case.

Finally, since the policies in NIMFGs should refer to uncountably many possible incoming flow values, there is no finite representation of strategies. Since the BR problem gets a profile as an input, it is not well defined for NIMFGs. As we elaborate in Section 7, the challenge is to find a finite representation of non-integral strategies to which attention can be restricted.

7 DISCUSSION

Today's computing environment involves systems with no central authority. This calls for a game-theoretic examination of classical algorithmic problems. We introduced and studied MFGs, which capture settings in which the vertices of a flow network are owned by entities with different destination objectives. While the results regarding the stability and efficiency of MFGs are negative, we show that allowing the players to drop flow makes the game much more stable and efficient: an MFGD always has an SO that is an NE, and that can be found in polynomial time. This positive result implies that even networks that are controlled by many different entities can reach a stable SO. Also, when considering networks that are controlled by different entities, allowing them to drop flow is recommended in order to improve stability and efficiency.

Unlike the traditional maximum-flow problem, where an integral maximum flow always exists, in MFGs players can benefit from using non-integral strategies. The need to consider real-valued flows gives rise to the challenge of finite representation of strategies. One way to cope with it is to prove a sufficient-granularity property, bounding the granularity to which unit flows should be divided. Another way is to develop a specification formalism for non-integral strategies, say "saturate the edge to v_1 and divide the remaining flow evenly between v_2 and v_3". A finite representation of strategies would make it possible to reason about a best-response dynamics in NIMFGs, and may simplify the witnesses used in the NP and Σ_2^P algorithms for MFGs. A related future work is an extension of MFGs to a probabilistic setting, where policies in vertices specify for each outgoing edge the probability that an incoming flow would be directed to it. Thus, profiles induce a distribution over possible flows, and the objective of a player is to increase the flow expected to reach her target vertex. While the probabilistic setting may seem more stable, our negative results in the non-integral case may carry over to it, as strategies that break an integral flow to fractions in $[0, 1]$ have a lot in common with strategies that direct this integral flow according to probabilities in $[0, 1]$.

Finally, MFGs motivate problems around network design, where the goal is to design stable networks. In particular, in *MFG repair*, we are given an MFG and we are asked to modify it in order to achieve stability or reduce the PoS or PoA (see [3], for a similar study in the context of repairing multi-agent systems with ω-regular objectives). Allowed modifications may increase or decrease the capacity of edges, change ownership of vertices, possibly assigning some vertices to an authority. Each such modification has a cost, and the goal is to understand the trade-off between the budget we have for repairs and the achieved stability.

REFERENCES

[1] S. Agarwal, M. S. Kodialam, and T. V. Lakshman. 2013. Traffic engineering in software defined networks. In *Proc. 32nd IEEE International Conference on Computer Communications*. 2211–2219.

[2] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. 1993. *Network flows: Theory, algorithms, and applications*. Prentice Hall Englewood Cliffs.

[3] S. Almagor, G. Avni, and O. Kupferman. 2015. Automatic Generation of Quality Specifications. In *Proc. 26th Int. Conf. on Concurrency Theory*, Vol. 42. 325–339.

[4] E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, T. Wexler, and T. Roughgarden. 2008. The Price of Stability for Network Design with Fair Cost Allocation. *SIAM J. Comput.* 38, 4 (2008), 1602–1623.

[5] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. 1990. *Introduction to Algorithms*. MIT Press and McGraw-Hill.

[6] E.A. Dinic. 1970. Algorithm for solution of a problem of maximum flow in a network with power estimation. *Soviet Math. Doll* 11, 5 (1970), 1277–1280. English translation by RF. Rinehart.

[7] J. Edmonds and R.M. Karp. 1972. Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM* 19, 2 (1972), 248–264.

[8] S. Even, A. Itai, and A. Shamir. 1976. On the Complexity of Timetable and Multicommodity Flow Problems. *SIAM J. Comput.* 5, 4 (1976), 691–703.

[9] J. Feigenbaum, C.H. Papadimitriou, R. Sami, and S. Shenker. 2005. A BGP-based mechanism for lowest-cost routing. *Distributed Computing* 18, 1 (2005), 61–72.

[10] L.R. Ford and D.R. Fulkerson. 1956. Maximal flow through a network. *Canadian journal of Mathematics* 8, 3 (1956), 399–404.

[11] L.R. Ford and D.R. Fulkerson. 1962. *Flows in networks*. Princeton Univ. Press, Princeton.

[12] A.V. Goldberg, É. Tardos, and R.E. Tarjan. 1989. *Network flow algorithms*. Technical Report. DTIC Document.

[13] A.V. Goldberg and R.E. Tarjan. 1988. A new approach to the maximum-flow problem. *J. ACM* 35, 4 (1988), 921–940.

[14] E. Kalai and E. Zemel. 1982. Totally balanced games and games of flow. *Mathematics of Operations Research* 7, 3 (1982), 476–478.

[15] E. Koutsoupias and C. Papadimitriou. 2009. Worst-case equilibria. *Computer Science Review* 3, 2 (2009), 65–69.

[16] O. Kupferman, G. Vardi, and M.Y. Vardi. 2017, to appear. Flow Games. In *Proc. 37th Conf. on Foundations of Software Technology and Theoretical Computer Science*.

[17] O. Lichtenstein and A. Pnueli. 1985. Checking that Finite State Concurrent Programs Satisfy their Linear Specification. In *Proc. 12th ACM Symp. on Principles of Programming Languages*. 97–107.

[18] J.F. Nash. 1950. Equilibrium points in n-person games. In *Proceedings of the National Academy of Sciences of the United States of America*.

[19] N. Nisan, T. Roughgarden, E. Tardos, and V.V. Vazirani. 2007. *Algorithmic Game Theory*. Cambridge University Press.

[20] C. H. Papadimitriou. 2001. Algorithms, games, and the Internet. In *Proc. 33rd ACM Symp. on Theory of Computing*. 749–753.

[21] A. Pnueli and R. Rosner. 1989. On the Synthesis of a Reactive Module. In *Proc. 16th ACM Symp. on Principles of Programming Languages*. 179–190.

[22] É. Tardos. 1985. A strongly polynomial minimum cost circulation algorithm. *Combinatorica* 5, 3 (1985), 247–255.

[23] S. Vissicchio, L. Vanbever, and O. Bonaventure. 2014. Opportunities and research challenges of hybrid software defined networks. *Computer Communication Review* 44, 2 (2014), 70–75.