Topological Distance Games

Martin Bullinger[1], **Warut Suksompong**[2]

[1]School of Computation, Information and Technology, Technical University of Munich
[2]School of Computing, National University of Singapore

Abstract

We introduce a class of strategic games in which agents are assigned to nodes of a topology graph and the utility of an agent depends on both the agent's inherent utilities for other agents as well as her distance from these agents on the topology graph. This model of *topological distance games (TDGs)* offers an appealing combination of important aspects of several prominent settings in coalition formation, including (additively separable) hedonic games, social distance games, and Schelling games. We study the existence and complexity of stable outcomes in TDGs—for instance, while a jump stable assignment may not exist in general, we show that the existence is guaranteed in several special cases. We also investigate the dynamics induced by performing beneficial jumps.

1 Introduction

You arrive at a hotel for your organization's annual banquet, and some of the seats at the tables have already been taken. You would like to sit close to your friends who work in the same team or share similar hobbies. On the other hand, you want to stay away from colleagues whom you had unpleasant interactions with lately. Which seat should you take? Once everyone has picked a seat, would you regret not having chosen a different seat? Similar issues arise when assigning faculty members to offices in the department building, students to desks in a classroom, or employees to cottages at a company retreat.

Recently, Bilò, Monaco, and Moscardelli (2022) introduced the model of *hedonic games with fixed-size coalitions*, wherein the agents are to be partitioned into coalitions whose sizes have been determined in advance, for example, by the sizes of the tables at the banquet. They assumed *additively separable* utilities, meaning that the utility of an agent for a coalition is the sum of her utilities for the individual agents in her coalition. While their model partially captures some of the aforementioned scenarios—for instance, each table at the banquet can be considered as one coalition—it neglects an important aspect common in such scenarios: the agents are typically assigned to specific locations, and agents prefer to be located close to their friends and far from their enemies. In our banquet scenario, a person sitting next to you has a higher influence on your utility than someone at the opposite end of the table, as you are much more likely to engage in a conversation with the former person than the latter. Similarly, you will in all likelihood run into your office neighbor more frequently than you encounter your colleague at the other end of the corridor.

With these motivating examples in mind, we introduce a class of games that we call *topological distance games (TDGs)*. An instance of TDG contains a topology graph, which is an undirected graph that specifies the locations to which the agents can be assigned. The influence that an agent i has on another agent j is j's inherent utility for i scaled by a factor depending on the distance between the two agents on the topology graph; if the two agents are not connected on the graph, they have no influence on each other. TDGs combine important aspects of several well-studied coalition formation settings, including hedonic games, social distance games, and Schelling games—we discuss these connections in detail in Section 1.2. We sometimes assume that the scaling factor is the reciprocal of the distance between the two agents, but most of our results also hold for arbitrary (strictly decreasing) distance factor functions. Following additively separable hedonic games (ASHGs), we then take the utility of an agent for an assignment to be the sum of all other agents' influences on the agent in question. Our formal model is described in Section 2.

1.1 Our Results

We study a fundamental notion of stability in our setting—*jump stability*—which requires that no agent would rather jump to some empty node than stay at her current node. In Section 3, we warm up by considering the case where agents' utilities are symmetric, that is, for any pair of agents i and j, i's inherent utility for j is the same as j's inherent utility for i. We show that for any distance factor function, there exists a jump stable assignment; on the other hand, finding such an assignment is a PLS-complete problem.

In Section 4, we investigate the more general setting where the utilities are not necessarily symmetric. We observe that a jump stable assignment may no longer exist, even when there are only two agents. On the other hand, if utilities are non-negative and the friendship graph[1] is

Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

[1] That is, the directed graph indicating (ordered) pairs of agents i, j such that i's inherent utility for j is positive.

acyclic, then existence is guaranteed. We then focus on the case where the topology graph is a cycle and every vertex in the friendship graph has out-degree at most 1. In this case, we characterize the friendship graphs for which the resulting instance admits a jump stable assignment, and present an efficient algorithm for computing such an assignment for those graphs. We also provide existence and non-existence results when the topology graph is a path or an (extended) star, and show that deciding the existence is NP-hard for the *reciprocal distance factor function*, whereby the scaling factor is the reciprocal of the distance between the two agents.

Lastly, in Section 5, we explore dynamical aspects of TDGs. If utilities are non-negative and the friendship graph is acyclic, we show that the jump dynamics is guaranteed to converge; however, even under these restrictions, the dynamics may run for an exponential number of steps. In addition, we establish the NP-hardness of deciding if the dynamics can possibly converge, or if it necessarily converges.

1.2 Related Work

The model of TDGs shares certain similarities with a number of existing models, and therefore offers an appealing combination of important aspects of several prominent settings in coalition formation.

Firstly, TDGs are similar to the aforementioned *hedonic games with fixed-size coalitions* (Bilò, Monaco, and Moscardelli 2022) in that one could view each connected component of the topology graph as a coalition of fixed size. In formal terms, hedonic games with fixed-size coalitions form a subclass of TDGs where every connected component is a clique. The main difference between TDGs and hedonic games in general (Aziz and Savani 2016) is that hedonic games do not come with a topology graph, so only the partition of the agents into coalitions matters, whereas in TDGs the distances resulting from the assignment of agents to the topology graph can affect the level of influence that the agents have on one another. Note that additively separable utilities are commonly studied in hedonic games (Bogomolnaia and Jackson 2002; Aziz, Brandt, and Seedig 2013). In fact, ASHGs can be viewed as a special case of TDGs where the topology graph consists of n cliques of size n each (n denotes the number of agents).

Secondly, Brânzei and Larson (2011) proposed the class of *social distance games*, wherein there is a social network that captures the connections among agents. The agents are again partitioned into coalitions, but now the utility of an agent for a coalition is the average, over all agents in the coalition, of the reciprocals of the distance to each agent in the coalition. Here, the distance is taken with respect to the subgraph of the social network induced by the coalition, and the distance of an agent to herself is not considered in this calculation. Like in hedonic games, there is no topology graph in social distance games. Flammini et al. (2021) introduced *distance hedonic games*, which generalize social distance games by allowing the distance function to be arbitrary rather than specifically the reciprocal function. Rey and Rey (2022) considered a distance-based approach for extending the agents' preferences over neighbors to preferences over coalitions in a subclass of hedonic games.

Thirdly, Massand and Simon (2019) studied *graphical one-sided markets*, which assume the existence of both an agent graph and a topology graph. Agents are placed on the topology graph, and the utility of an agent for the placement is the sum of the agent's utility for her assigned node and her utilities for her neighbors on the topology graph, where the latter utilities are taken from the agent graph. While our TDG model is more restrictive from the point of view that it does not allow agents to derive utilities from nodes in the topology graph, it is more general in that an agent's utility depends not only on neighboring agents on the topology graph, but also on agents further away. Elkind et al. (2020) investigated a similar model as Massand and Simon (2019) from the truthfulness perspective, while Bodlaender et al. (2020) considered the special case where an agent's utility depends on her neighbors but not on her assigned node.

Finally, a recent stream of work on *Schelling games* also deals with a topology graph, and an agent's utility depends on the neighboring agents on the graph (Chauhan, Lenzner, and Molitor 2018; Echzell et al. 2019; Agarwal et al. 2021; Bullinger, Suksompong, and Voudouris 2021). However, in that model, the agents have predetermined types and the utility of an agent is defined as the *fraction* of neighboring agents of the same type. Note that jump stability is commonly studied in Schelling games as well.

2 Preliminaries

Let $N = [n]$ be the set of *agents*, where $[k] := \{1, 2, \ldots, k\}$ for each positive integer k. Each agent $i \in N$ is endowed with an *(inherent) utility function* $u_i \colon N \to \mathbb{R}$, which specifies the inherent utility that i has for every other agent; we assume that $u_i(i) = 0$ for all i. A utility function u_i is

- *symmetric* if $u_i(j) = u_j(i)$ for all $i, j \in N$, and
- *binary* if $u_i(j) \in \{0, 1\}$ for all $i, j \in N$.

We say that agent j is a *friend* of agent i if $u_i(j) > 0$; the *friendship graph*[2] is a directed graph with the set of nodes N such that there is an edge from i to j if and only if $u_i(j) > 0$.

There is a *topology graph* $G = (V, E)$, which is a simple (not necessarily connected) undirected graph with at least n nodes. An *assignment* $\lambda \colon N \to V$ is an injective mapping that assigns each agent to a node in V, i.e., each node can be occupied by at most one agent. For $N' \subseteq N$, let $\lambda(N') := \{\lambda(i) \mid i \in N'\}$. A node $v \in V$ is called *empty* with respect to an assignment λ if $v \notin \lambda(N)$.

The *distance factor function* $f \colon \mathbb{Z}_{\geq 1} \to \mathbb{R}_{>0}$ is a strictly decreasing function which determines the level of influence that an agent has on another agent depending on the distance between them, where this distance is the length of the shortest path between their assigned nodes in G. If the two agents are assigned to different connected components of G, then the distance factor between them is taken to be 0, meaning that they have no influence on each other's utilities. The

[2] Friendship graphs have been studied in several papers on hedonic games (Igarashi et al. 2019; Kerkmann et al. 2020; Bullinger and Kober 2021).

reciprocal distance factor function refers to the function[3] $f(k) = 1/k$. We abuse notation slightly by extending utility functions u_i to assignments. In particular, the utility of agent i for assignment λ is

$$u_i(\lambda) := \sum_{j \in N \setminus \{i\}} f(d_G(\lambda(i), \lambda(j))) \cdot u_i(j),$$

where $d_G(v, v')$ denotes the length of the shortest path between v and v' in G. A *topological distance game (TDG)* consists of the agents and their utility functions, the topology graph, and the distance factor function.

Given an assignment λ, an empty node $v \in V$, and an agent $i \in N$, denote by $\lambda^{i \to v}$ the assignment that results when i jumps from her assigned node $\lambda(i)$ to v. We now define the main stability notion that we study in this paper.

Definition 2.1. Given an instance of TDG and an assignment λ, a jump by agent i to an empty node v is a *beneficial jump* in λ if $u_i(\lambda) < u_i(\lambda^{i \to v})$.

The assignment λ is said to be *jump stable* if no agent has a beneficial jump, that is, for each agent $i \in N$ and each empty node $v \in V$, it holds that $u_i(\lambda) \geq u_i(\lambda^{i \to v})$.

Since we consider jump stability, we assume without loss of generality that $|V| > n$, as every assignment is trivially jump stable if $|V| = n$.

All omitted proofs can be found in the full version of our paper (Bullinger and Suksompong 2022).

3 Warm-Up: Symmetric Utilities

We begin by deriving preliminary results for the case of symmetric utilities. First, by a standard potential function argument (Bogomolnaia and Jackson 2002; Suksompong 2015; Bilò, Monaco, and Moscardelli 2022), we can show the existence of a jump stable assignment in this case.

Theorem 3.1. *For any distance factor function and symmetric utilities, there exists a jump stable assignment.*

Proof. Consider an assignment maximizing the potential function $\Phi(\lambda) := \sum_{i \in N} u_i(\lambda)$; such an assignment must exist because the number of possible assignments is finite. Assume for contradiction that some agent i^* prefers to jump from her current node v to an empty node w. We have

$$u_{i^*}(\lambda) < u_{i^*}(\lambda^{i^* \to w}), \tag{1}$$

where we know that

$$u_{i^*}(\lambda) = \sum_{j \in N \setminus \{i^*\}} f(d_G(v, \lambda(j))) \cdot u_{i^*}(j)$$

and

$$u_{i^*}(\lambda^{i^* \to w}) = \sum_{j \in N \setminus \{i^*\}} f(d_G(w, \lambda(j))) \cdot u_{i^*}(j).$$

Now, if i^* jumps to w, the potential function changes by

$$\Phi(\lambda^{i^* \to w}) - \Phi(\lambda) = \sum_{i \in N} u_i(\lambda^{i^* \to w}) - \sum_{i \in N} u_i(\lambda)$$

$$= \left[u_{i^*}(\lambda^{i^* \to w}) + \sum_{j \in N \setminus \{i^*\}} (f(d_G(\lambda(j), w)) \cdot u_j(i^*)) \right]$$
$$- \left[u_{i^*}(\lambda) + \sum_{j \in N \setminus \{i^*\}} (f(d_G(\lambda(j), v)) \cdot u_j(i^*)) \right]$$
$$= \left[u_{i^*}(\lambda^{i^* \to w}) + \sum_{j \in N \setminus \{i^*\}} (f(d_G(w, \lambda(j))) \cdot u_{i^*}(j)) \right]$$
$$- \left[u_{i^*}(\lambda) + \sum_{j \in N \setminus \{i^*\}} (f(d_G(v, \lambda(j))) \cdot u_{i^*}(j)) \right]$$
$$= 2 \cdot \Big(u_{i^*}(\lambda^{i^* \to w}) - u_{i^*}(\lambda) \Big) > 0,$$

where we use the symmetry of the utilities for the second equality and (1) for the inequality. This contradicts the assumption that λ maximizes the potential function Φ. □

Despite its guaranteed existence, a jump stable assignment can be difficult to compute. Our reduction is from a local variant of MAX CUT.

Theorem 3.2. *For any distance factor function and symmetric utilities, finding a jump stable assignment is* PLS-*complete.*

4 Asymmetric Utilities

We now consider the more general setting where the agents' inherent utilities for each other are not necessarily symmetric. First, we observe that a jump stable assignment may no longer exist, even when there are only two agents.

Proposition 4.1. *Let G be a connected graph of diameter at least* 3. *For any distance factor function, there exists an instance with topology graph G and two agents such that no jump stable allocation exists.*

Proof. Let $n = 2$, $u_1(2) = 1$, and $u_2(1) = -1$, and consider any assignment λ. If $d_G(\lambda(1), \lambda(2)) \geq 2$, then agent 1 would jump to a neighboring node of agent 2. Else, $d_G(\lambda(1), \lambda(2)) = 1$. In this case, consider two nodes $v, w \in V$ with $d_G(v, w) \geq 3$. It must be that $d_G(\lambda(1), v) \geq 2$ or $d_G(\lambda(1), w) \geq 2$. If the former holds, then $v \neq \lambda(2)$, and agent 2 has a beneficial jump to v; an analogous argument holds in the latter case with w instead of v. □

Proposition 4.1 does not hold if we lower the diameter threshold to 2: for a star graph and any number of agents, one can check that there is always a jump stable assignment.

As we will see later, a jump stable assignment may not exist even if utilities are non-negative and the friendship graph is a cycle. We show next that the existence of such an assignment is guaranteed under non-negative utilities when the friendship graph is acyclic; this result will also be useful for our characterization in the case of a cycle friendship graph (Theorem 4.3). We remark that acyclic friendship graphs can model situations in which there is a hierarchy among the

[3] As discussed in Section 1.2, a similar idea involving reciprocals of the distance has been used in social distance games (Brânzei and Larson 2011).

agents: for instance, at research conferences, younger researchers may be keen to interact with certain older ones who might help their career, but not vice versa.

Theorem 4.2. *For any distance factor function, if the friendship graph is acyclic and utilities are non-negative, then a jump stable assignment exists and can be computed in polynomial time.*

Proof. Assume that the friendship graph is acyclic and utilities are non-negative. There exists a *topological order* π of the agents in N (Kahn 1962), i.e., a function $\pi\colon N \to [n]$ such that $\pi(i) > \pi(j)$ whenever $u_i(j) > 0$. Based on this order, we provide a polynomial-time algorithm that computes a jump stable assignment. Given a subset of agents $M \subseteq N$, a partial assignment $\lambda\colon M \to V$, an agent $i \in N \setminus M$, and a node $v \in V \setminus \lambda(M)$, let $\lambda^{[i\to v]}\colon M \cup \{i\} \to V$ be the partial assignment extended from λ by assigning agent i to v. Also, let $\lambda_\emptyset\colon \emptyset \to V$ be the empty partial assignment.

Algorithm 1: Jump stable assignment for acyclic friendship graph and non-negative utilities.

```
Input: Topology graph $G = (V, E)$, topological order $\pi\colon N \to [n]$
Output: Jump stable assignment $\lambda\colon N \to V$
    $V^e \leftarrow V$
    $\lambda_0 \leftarrow \lambda_\emptyset$
    for $k = 1, \dots, n$ do
        $i \leftarrow \pi^{-1}(k)$
        Select $v^* \in \arg\max_{v \in V^e}\{u_i(\lambda_{k-1}^{[i\to v]})\}$
        $\lambda_k \leftarrow \lambda_{k-1}^{[i\to v^*]}$
        $V^e \leftarrow V^e \setminus \{v^*\}$
    end for
    return $\lambda \leftarrow \lambda_n$
```

Algorithm 1 describes our procedure for computing a jump stable assignment. In each iteration of the for-loop, we determine the position of some agent. We place agents according to the topological order π—this ensures that whenever an agent is placed, all of her friends have already been placed, so her utility is not influenced by later agents.

Clearly, the algorithm runs in polynomial time. It remains to show that the returned assignment λ is jump stable. Consider an arbitrary agent $i \in N$; it suffices to show that i cannot perform a beneficial jump. Notice that all nodes that i could potentially jump to were available at the moment when i was assigned during the execution of Algorithm 1. Moreover, since $u_i(j) = 0$ for all agents j with $\pi(j) > \pi(i)$, we have $u_i(\lambda^{i\to v}) = u_i(\lambda_{\pi(i)}^{i\to v}) \le u_i(\lambda_{\pi(i)}) = u_i(\lambda)$ for any node $v \in V \setminus \lambda(N)$; here, the inequality follows from the maximization in the algorithm. Hence, agent i cannot perform a beneficial jump. □

If the topology graph is a cycle (e.g., a party table) and every vertex in the friendship graph has out-degree at most 1, we completely characterize the friendship graphs and topology graphs for which the resulting instance admits a jump stable assignment.

Theorem 4.3. *Suppose that the topology graph G is a cycle, and each agent has at most one friend and utility 0 for the remaining agents. For any distance factor function, a jump stable assignment exists if and only if neither of the following cases occurs:*

- *The friendship graph is a 3-cycle;*
- *The friendship graph is a 5-cycle.*

If a jump stable assignment exists, it can be computed in polynomial time.

The proof of Theorem 4.3 is involved and follows from a detailed analysis of stable assignments in this setup.

We have seen that, with non-negative utilities, a jump stable assignment always exists if the friendship graph is acyclic, but may not exist if both the friendship graph and the topology graph are cycles. Is existence guaranteed when the friendship graph is a cycle but the topology graph is acyclic? If the topology graph is a path, the answer is positive as long as each agent has at most two friends.

Theorem 4.4. *For any distance factor function, if the topology graph G is a path and each agent has at most two friends and utility 0 for the remaining agents, then a jump stable assignment exists and can be computed in polynomial time.*

Proof. We assign agents to nodes along the path from left to right. First, assign an arbitrary agent to the leftmost node. For each subsequent node, among the unassigned friends j of the agent i occupying the previous node, assign one with the highest $u_i(j)$; if all of i's friends have been assigned (or if i has no friend), then assign an arbitrary agent. Clearly, this procedure runs in polynomial time.

We show that the final assignment λ is jump stable. Consider any agent i. Since all utilities are non-negative, it suffices to show that the jump to the $(n+1)$th node from the left, denoted by v, is not beneficial for i. If all of i's friends are to her left, this is obvious. Else, if i has one friend j to her right, then j is directly next to i, so by jumping to the $(n+1)$th node, i gets closer to neither j nor i's other friend (if i has a friend other than j). Otherwise, i has both friends j and k to her right. Assume without loss of generality that $u_i(j) \ge u_i(k)$ and that the algorithm assigns j next to i. Suppose that i and k are on the d_i-th and d_k-th nodes from the left, respectively (so j occupies the $(d_i + 1)$th node). Then we have

$$\begin{aligned} u_i(\lambda) &= f(1) \cdot u_i(j) + f(d_k - d_i) \cdot u_i(k) \\ &\ge f(1) \cdot u_i(k) + f(d_k - d_i) \cdot u_i(j) \\ &\ge f(n + 1 - d_k) \cdot u_i(k) + f(n - d_i) \cdot u_i(j) \\ &= u_i(\lambda^{i\to v}), \end{aligned}$$

where the second inequality holds because $d_k \le n$ and f is a decreasing function. □

On the other hand, if the topology graph is a tree, existence is no longer guaranteed even when the friendship graph is a cycle.

Theorem 4.5. *For any distance factor function, a jump stable assignment may not exist even if the topology graph is a tree, the agents have binary utilities, and the friendship graph is a cycle.*

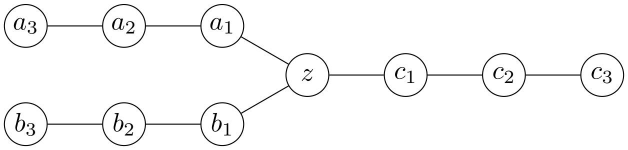

Figure 1: Topology graph in the proof of Theorem 4.5.

Proof. Consider an instance with $N = [6]$ and the topology graph $G = (V, E)$ depicted in Figure 1. Assume that the utilities are given by $u_i(i+1) = 1$ for $i \in [6]$, where we take agent indices modulo 6, and $u_i(j) = 0$ for all other pairs $i, j \in N$. Suppose for contradiction that there exists a jump stable assignment $\lambda \colon N \to V$. Let $W := \lambda(N) \subseteq V$ be the set of occupied nodes.

First, observe that W must be a connected set of nodes. Indeed, if there is a proper subset of agents $C \subsetneq N$ that occupy a connected component of the subgraph of G induced by W, then there exists an agent $i \in C$ such that $i+1 \notin C$. Moreover, there is an empty node on the (unique) path in G connecting $\lambda(i)$ with $\lambda(i+1)$, and therefore i has an incentive to jump to this node.

The observation above implies that $z \in W$, and for every $x \in \{a, b, c\}$, there exists $r_x \in \{0, 1, 2, 3\}$ such that $\{x_1, x_2, x_3\} \cap W = \{x_i \colon i \in [r_x]\}$. Assume without loss of generality that $r_a \geq r_b \geq r_c$. We perform a case analysis based on the vector (r_a, r_b, r_c).

- $(r_a, r_b, r_c) = (3, 2, 0)$. Consider the agent $i = \lambda^{-1}(b_2)$. The agent that has i as a friend must be placed at b_1; otherwise, she would benefit by jumping to b_3. But since c_1 is closer to every node except b_1 than b_2 is, i can benefit by jumping to c_1, a contradiction.
- $(r_a, r_b, r_c) = (3, 1, 1)$. The unique agent that has $\lambda^{-1}(c_1)$ as a friend must be placed at z; otherwise, she would benefit by jumping to c_2. However, the same must hold for the unique agent that has $\lambda^{-1}(b_1)$ as a friend, which is impossible.
- $(r_a, r_b, r_c) = (2, 2, 1)$. Assume without loss of generality that $\lambda(1) = b_2$. Then, $\lambda(6) = b_1$, as otherwise agent 6 would have a beneficial jump to b_3. This in turn implies $\lambda(2) = z$, because otherwise agent 1 can jump to either a_3 or c_2 to improve her utility. Let $s \in \{3, 4, 5\}$ be such that $\lambda(s) = a_2$. The same chain of arguments as for agent 1 yields that $\lambda(s+1) = z$, which is impossible since $\lambda(2) = z$.

We have reached a contradiction in all cases, so λ cannot be jump stable. □

Interestingly, if the topology graph is an "extended star" with three branches and the friendship graph is a cycle, as in the instance used in the proof of Theorem 4.5, then a jump stable assignment always exists whenever there are at least 16 agents. In fact, we show a more general result with an arbitrary number of branches. To this end, we define an *extended star* as a tree in which only one vertex, called the *center*, has degree at least 3. A *branch* of an extended star is a path of maximal length such that one endpoint is the center. The *size* of a branch is the number of nodes on the branch, not counting the center of the star.

Theorem 4.6. *Let $k \geq 3$ be an integer, and assume that G is an extended star with k branches. Suppose that there are $n \geq 5k+1$ agents with non-negative utilities, and the friendship graph is a cycle. For any distance factor function, there exists a jump stable assignment.*

Proof. Assume that the agents are ordered $1, 2, \dots, n$ according to the cycle in the friendship graph. Call each branch of size at most 4 a *type-1 branch* and each branch of size at least 5 a *type-2 branch*. We assign the agents following their order branch by branch, starting with type-1 branches, then type-2 branches, and finally the center node. For each type-1 branch, we fill the entire branch, whereas for each type-2 branch, we assign at least 5 agents to the branch. Since there are at least $5k+1$ agents and only k branches, we have enough agents for type-2 branches.

For each type-1 branch, we fill in the agents from the leaf towards the center. For each type-2 branch, let r be the number of agents that we want to assign to the branch. We divide into two cases depending on the parity of r.

- If $r = 2s$ is even, we assign the first s agents starting from the node closest to the center and leaving one empty node after assigning each agent (except the s-th agent). We then fill in the other s agents starting from the subsequent node and moving back towards the center.
- If $r = 2s+1$ is odd, we proceed similarly except that we start with the *second* closest node to the center.

Finally, we assign the last agent to the center node. An example with $k = 3$ branches of size $4, 6, 7$ is shown in Figure 2.

Since $n \geq 5k+1$, there is at least one type-2 branch. We now show that the resulting assignment is jump stable.

- Consider agent n assigned to the center node. If there is a type-1 branch, agent n cannot get closer to her friend, agent 1, because every type-1 branch is completely filled. Otherwise, agent n is at distance at most 2 from agent 1, and cannot get closer to agent 1 because agent 1's branch contains at least five agents.
- For each type-1 branch, every agent is already next to her friend except the agent next to the center node. For this latter agent, her friend is in either a type-1 branch—in which case she cannot get closer because the branch is already filled—or a type-2 branch—in which case she is at distance at most 3 from her friend and every empty node has distance at least 4 from this friend.
- Consider a type-2 branch with an even number of agents $2s$. The s-th agent (in order of agent number) is already next to her friend. The $(2s)$-th agent is at distance at most 4 from her friend and every empty node has distance at least 4 from this friend. Every other agent on this branch is at distance 2 from her friend, and no node adjacent to this friend is empty. A similar argument applies when the branch contains an odd number of agents.

Hence, in all cases, there is no beneficial jump. □

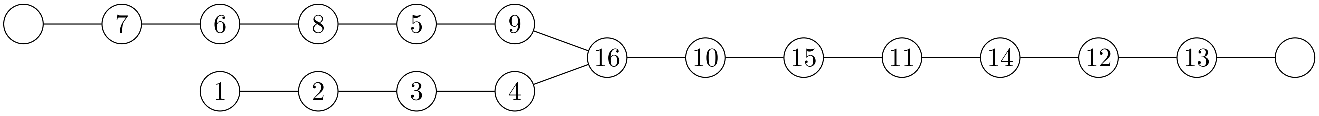

Figure 2: Topology graph in the proof of Theorem 4.6 when $n = 16$, $k = 3$, and the three branches have size 4, 6, and 7.

Since jump stable assignments are not guaranteed to exist in TDGs, a natural question is whether we can efficiently decide if such an assignment exists. As it is known that determining whether a *Nash stable partition* exists in ASHGs is NP-hard (Sung and Dimitrov 2010), the connection between ASHGs and TDGs that we outlined in Section 1.2 implies that the jump stability question for TDGs is also NP-hard. However, using reductions between the two classes of games results in TDG instances where the number of connected components is linear in n, the number of nodes is quadratic in n, and the distance factor function does not play any role (because every connected component is a clique). We therefore show that the hardness still holds even for instances that better reflect the essence of TDGs.

Theorem 4.7. *For the reciprocal distance factor function, it is* NP*-complete to decide whether there exists a jump stable assignment in a given TDG. This holds even if the topology graph G consists of a constant number of connected components and* $|V| = \Theta(n)$*.*

Proof sketch. For the hardness, we provide a reduction from the NP-complete problem EXACT 3-COVER (Karp 1972). Given an instance (R, S) of this EXACT 3-COVER, where R is a ground set and S is a collection of 3-element subsets, we construct a TDG instance as follows. There are three types of agents. The first two types represent the elements of R and the sets in S. The last type, which consists of only one agent, is a "disturber" strictly avoided by agents representing sets in S. The topology consists of four connected components. One component serves to hold all agents representing R and agents representing an exact 3-cover from S. The edges of this component are designed in such a way that it is possible to assign agents to all nodes of this component if and only if the original instance is a Yes-instance. The other three components are cliques of carefully chosen sizes which either allow all other agents representing sets in S to flee the disturber (in case of a Yes-instance), or provide sufficient space to allow for a run-and-chase dynamics that prevents stability (in case of a No-instance). □

5 Dynamics

In this section, we investigate the dynamics induced by performing beneficial jumps. Dynamics offer an interesting distributed perspective on stability and have been examined, for instance, in hedonic games (Brandt, Bullinger, and Wilczynski 2021; Fanelli, Monaco, and Moscardelli 2021; Brandt, Bullinger, and Tappe 2022; Boehmer, Bullinger, and Kerkmann 2023). We are interested in the following questions:

- Given an initial assignment, is it *possible* that the dynamics converges, that is, there exists a sequence of beneficial jumps that results in a jump stable assignment?
- Given an initial assignment, is it *necessary* that the dynamics converges, that is, all sequences of beneficial jumps result in a jump stable assignment?

First, observe that dynamics are guaranteed to converge for symmetric utilities, because the potential function in the proof of Theorem 3.1 increases with every beneficial jump. In this sense, the proof yields more than merely the existence of a jump stable assignment. For asymmetric utilities, since a jump stable assignment may not exist, convergence of the dynamics is also no longer guaranteed. Nevertheless, if the friendship graph is acyclic and utilities are non-negative, the convergence is retained.

Theorem 5.1. *For any distance factor function, if the friendship graph is acyclic and utilities are non-negative, then the jump dynamics is guaranteed to converge.*

Proof. Consider an instance of TDG such that the friendship graph is acyclic and utilities are non-negative. As in the proof of Theorem 4.2, there exists a topological order π of the agents in N, i.e., a function $\pi\colon N \to [n]$ such that $\pi(i) > \pi(j)$ whenever $u_i(j) > 0$. We shall define a potential function based on this order.

Consider a sequence of assignments $(\lambda_k)_{k\geq 1}$, where for each $k \geq 1$, $\lambda_{k+1} = \lambda_k^{d_k \to v_k}$ for some agent d_k and node v_k which is empty in λ_k. Associate any assignment λ with the vector

$$\Lambda(\lambda) = (u_{\pi^{-1}(1)}(\lambda), u_{\pi^{-1}(2)}(\lambda), \dots, u_{\pi^{-1}(n)}(\lambda)).$$

For two vectors $\mathbf{x} = (x_1, \dots, x_n)$ and $\mathbf{y} = (y_1, \dots, y_n)$, we say that $\mathbf{x}$ *lexicographically dominates* $\mathbf{y}$, denoted by $\mathbf{x} >_{\text{lex}} \mathbf{y}$, if there exists $i \in \{1, \dots, n\}$ such that $x_j = y_j$ for all $j \in \{1, \dots, i-1\}$ and $x_i > y_i$.

We claim that for every $k \geq 1$, it holds that $\Lambda(\lambda_{k+1}) >_{\text{lex}} \Lambda(\lambda_k)$. Let $k \geq 1$ and consider the deviator d_k. By definition of the topological order, for each $j < \pi(d_k)$, we have $u_{\pi^{-1}(j)}(d_k) = 0$. Hence, for every $j < \pi(d_k)$, it holds that $u_{\pi^{-1}(j)}(\lambda_{k+1}) = u_{\pi^{-1}(j)}(\lambda_k)$, because this utility is not affected by d_k's jump. It follows that the first $\pi(d_k) - 1$ entries of $\Lambda(\lambda_k)$ and $\Lambda(\lambda_{k+1})$ are identical. Moreover, as d_k improves her utility with the beneficial jump, the $\pi(d_k)$-th entry of λ_k increases, and therefore $\Lambda(\lambda_{k+1}) >_{\text{lex}} \Lambda(\lambda_k)$. □

Theorem 5.1 also shows that there exists a potential function with respect to which the jump dynamics is increasing. This implies that the problem of finding jump stable outcomes in instances with an acyclic friendship graph and nonnegative utilities is contained in the complexity class PLS. However, it is unlikely that this problem is PLS-complete, because we can compute jump stable assignments for this class of games in polynomial time (cf. Theorem 4.2). In fact, PLS-completeness for this problem would imply that

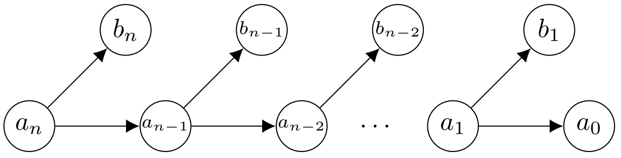

Figure 3: Friendship graph of the game in the proof of Theorem 5.2.

PLS = P. Still, similarly to PLS-complete problems, the jump dynamics, which is the natural local search dynamics for our problem, can run in exponential time.

Theorem 5.2. *For any distance factor function, the jump dynamics may run for an exponential number of steps, even if the friendship graph is acyclic and utilities are non-negative.*

Proof. We prove that, for each $k \geq 1$, there exists an additively separable hedonic game H_k with $n = 2k + 1$ agents, non-negative utilities, and an acyclic friendship graph such that the jump dynamics runs for at least 2^k steps. Recall from Section 1.2 that ASHGs form a subclass of TDGs.

Let $k \geq 1$ and define an ASHG (N, u), where $N = \{a_j, b_j \colon j \in [k]\} \cup \{a_0\}$ is a set of $2k + 1$ agents and the utilities are given as follows:

- For each $j \in [k]$, $u_{a_j}(b_j) = 1$ and $u_{a_j}(a_{j-1}) = 2$.
- All other utilities are set to 0.

Clearly, the utilities are non-negative and the friendship graph is acyclic. An illustration of the friendship graph is given in Figure 3.

We show by induction on k that there exists a sequence of beneficial jumps in H_k such that agent a_k performs at least 2^k jumps. For the base case $k = 1$, start with a partition of the agents into singletons, and let a_1 first join b_1 and then a_0.

Assume that we have constructed the dynamics for some $k \geq 1$. Observe that H_{k+1} is an extension of H_k with the addition of agents a_{k+1} and b_{k+1}. Given the dynamics constructed for H_k, we extend it to one for H_{k+1} as follows. Agents a_{k+1} and b_{k+1} are initially in singleton coalitions. After every jump by agent a_k in the original dynamics, we insert the following jumps: a_{k+1} joins b_{k+1}, then a_{k+1} joins the coalition containing a_k. This leads to a total of $2 \cdot 2^k = 2^{k+1}$ jumps by a_{k+1}. Notice that all jumps in this extended sequence (whether by a_{k+1} or not) are beneficial jumps. Indeed, original jumps are not affected by the new agents a_{k+1} and b_{k+1}, because these two agents do not influence the utilities of original agents. Moreover, whenever a_k jumps, she always leaves the coalition containing a_{k+1}. The only exception is a_k's first jump, when a_{k+1} is still in a singleton coalition; however, even for this jump, a_k does not join a_{k+1}. Hence, agent a_{k+1} has a utility of 0 after each jump by a_k, so all jumps by a_{k+1} are beneficial jumps. □

Since we make use of the connection between ASHGs and TDGs in Theorem 5.2, the topology graph of the constructed game has a non-constant number of connected components. We conjecture that exponential running time is possible even when the number of components is constant.

Finally, we prove that the questions of whether the jump dynamics starting from an initial assignment possibly or necessarily converges are both computationally hard.

Theorem 5.3. *For any distance factor function, deciding whether the jump dynamics possibly converges is* NP*-hard, even if utilities are restricted to be non-negative.*

Proof sketch. As in the proof of Theorem 4.7, we reduce from EXACT 3-COVER. Given an instance (R, S) of EXACT 3-COVER, we construct a TDG containing, among other agents, three agents whose friendship graph forms a cycle. The jump dynamics will run into a cycle due to these three agents; the only way to prevent this is through a jump by a special agent δ initially assigned to a connected component containing agents representing the sets in S. The agent δ can disrupt the cycle if and only if agents representing an exact 3-cover of R jump to a connected component containing agents representing the elements of R. □

The proof for deciding whether the dynamics necessarily converges is similar. In this case, the special agent δ initially blocks cycling, and can initiate cycling only after the jumps by a set of agents representing an exact 3-cover.

Theorem 5.4. *For any distance factor function, deciding whether the jump dynamics necessarily converges is* coNP*-hard, even if utilities are restricted to be non-negative.*

6 Discussion

In this work, we have introduced the model of topological distance games (TDGs), which aim to capture scenarios in which the utility of an agent depends on both her inherent utilities for other agents and her distance from them. We presented results on the existence, computational, and dynamical properties of jump stable assignments in our model. While such assignments may not exist even under weak assumptions, it turns out that existence guarantees can be obtained for symmetric utilities as well as in the presence of structured friendship relations.

Given that TDGs combine important aspects of other models in coalition formation, including hedonic games, social distance games, and Schelling games (see Section 1.2), our study may inspire further work from several angles. For instance, while we have shown that a jump stable assignment exists in a number of cases, one could try to obtain a more complete characterization of the topology and friendship graphs that admit such an assignment or even guarantee convergence of the jump dynamics. Understanding precisely when it is possible to efficiently determine whether a jump stable assignment exists is also an interesting direction. Yet another potential avenue is to extend our results to *weighted* graphs, wherein the distance between two agents is the length of the shortest path (in terms of the sum of weights) between their assigned nodes. Finally, in addition to jump stability, other notions such as *swap stability* or *envy-freeness* are worth exploring in TDGs as well—we provide some initial results on swap stability in the full version of our paper (Bullinger and Suksompong 2022).

Acknowledgments

This work was partially supported by the Deutsche Forschungsgemeinschaft under grants BR 2312/11-2 and BR 2312/12-1, by the Singapore Ministry of Education under grant number MOE-T2EP20221-0001, and by an NUS Start-up Grant. We thank the anonymous reviewers for their valuable feedback.

References

Agarwal, A.; Elkind, E.; Gan, J.; Igarashi, A.; Suksompong, W.; and Voudouris, A. A. 2021. Schelling games on graphs. *Artificial Intelligence*, 301: 103576.

Aziz, H.; Brandt, F.; and Seedig, H. G. 2013. Computing desirable partitions in additively separable hedonic games. *Artificial Intelligence*, 195: 316–334.

Aziz, H.; and Savani, R. 2016. Hedonic games. In Brandt, F.; Conitzer, V.; Endriss, U.; Lang, J.; and Procaccia, A. D., eds., *Handbook of Computational Social Choice*, chapter 15, 356–376. Cambridge University Press.

Bilò, V.; Monaco, G.; and Moscardelli, L. 2022. Hedonic games with fixed-size coalitions. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, 9287–9295.

Bodlaender, H. L.; Hanaka, T.; Jaffke, L.; Ono, H.; Otachi, Y.; and van der Zanden, T. C. 2020. Hedonic seat arrangement problems. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 1777–1779.

Boehmer, N.; Bullinger, M.; and Kerkmann, A. M. 2023. Causes of stability in dynamic coalition formation. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI)*. Forthcoming.

Bogomolnaia, A.; and Jackson, M. O. 2002. The stability of hedonic coalition structures. *Games and Economic Behavior*, 38(2): 201–230.

Brandt, F.; Bullinger, M.; and Tappe, L. 2022. Single-agent dynamics in additively separable hedonic games. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, 4867–4874.

Brandt, F.; Bullinger, M.; and Wilczynski, A. 2021. Reaching individually stable coalition structures in hedonic games. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)*, 5211–5218.

Brânzei, S.; and Larson, K. 2011. Social distance games. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, 273–279.

Bullinger, M.; and Kober, S. 2021. Loyalty in cardinal hedonic games. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI)*, 66–72.

Bullinger, M.; and Suksompong, W. 2022. Topological distance games. *CoRR*, abs/2211.11000.

Bullinger, M.; Suksompong, W.; and Voudouris, A. A. 2021. Welfare guarantees in Schelling segregation. *Journal of Artificial Intelligence Research*, 71: 143–174.

Chauhan, A.; Lenzner, P.; and Molitor, L. 2018. Schelling segregation with strategic agents. In *Proceedings of the 11th International Symposium on Algorithmic Game Theory (SAGT)*, 137–149.

Echzell, H.; Friedrich, T.; Lenzner, P.; Molitor, L.; Pappik, M.; Schöne, F.; Sommer, F.; and Stangl, D. 2019. Convergence and hardness of strategic Schelling segregation. In *Proceedings of the 15th International Conference on Web and Internet Economics (WINE)*, 156–170.

Elkind, E.; Patel, N.; Tsang, A.; and Zick, Y. 2020. Keeping your friends close: Land allocation with friends. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI)*, 318–324.

Fanelli, A.; Monaco, G.; and Moscardelli, L. 2021. Relaxed core stability in fractional hedonic games. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI)*, 182–188.

Flammini, M.; Kodric, B.; Olsen, M.; and Varricchio, G. 2021. Distance hedonic games. In *Proceedings of the 47th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, 159–174.

Igarashi, A.; Ota, K.; Sakurai, Y.; and Yokoo, M. 2019. Robustness against agent failure in hedonic games. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, 364–370.

Kahn, A. B. 1962. Topological sorting of large networks. *Communications of the ACM*, 5(11): 558–562.

Karp, R. M. 1972. Reducibility among combinatorial problems. In Miller, R. E.; and Thatcher, J. W., eds., *Complexity of Computer Computations*, 85–103. Plenum Press.

Kerkmann, A. M.; Lang, J.; Rey, A.; Rothe, J.; Schadrack, H.; and Schend, L. 2020. Hedonic games with ordinal preferences and thresholds. *Journal of Artificial Intelligence Research*, 67: 705–756.

Massand, S.; and Simon, S. 2019. Graphical one-sided markets. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, 492–498.

Rey, A.; and Rey, L. 2022. FEN-hedonic games with distance-based preferences. *CoRR*, abs/2201.13158.

Suksompong, W. 2015. Individual and group stability in neutral restrictions of hedonic games. *Mathematical Social Sciences*, 78: 1–5.

Sung, S.-C.; and Dimitrov, D. 2010. Computational complexity in additive hedonic games. *European Journal of Operational Research*, 203(3): 635–639.