Temporal Team Formation Games with Dynamic Preferences

Cameron Egbert[a], Aaron Lin[a,*], Judy Goldsmith[a,], Pearson Garner[a] and Ruby Harris[a]**

[a]University of Kentucky
ORCID (Cameron Egbert): https://orcid.org/0009-0008-3823-9339, ORCID (Aaron Lin): https://orcid.org/0009-0009-3317-4545, ORCID (Judy Goldsmith): https://orcid.org/0000-0002-8383-5390, ORCID (Pearson Garner): https://orcid.org/0009-0007-0375-1650, ORCID (Ruby Harris): https://orcid.org/0009-0006-0915-8084

Abstract. In the professional world, it is imperative for management entities to allocate their human resources to a work schedule, and such models of coalition formation games are well-studied. However, most existing literature only considers coalition formation in the context of a single snapshot in time, without accounting for changing preferences of individuals as they work together. Our primary contribution for this paper is a new team formation game that incorporates skill-based team formation plus a dynamic variant of Additively Separable Hedonic Games. The Temporal Team Formation Game with Dynamic Preferences allows for two psychologically common preference dynamics: variety preference and loyalty preference. We show that local search can plan team assignments for a series of known, synchronous tasks to model for dynamically-changing preferences over time.

1 Introduction

In an increasingly competitive job market where employers potentially invest multiple times into an employee's salary [28], the need to reduce employee turnover is becoming vital. Studies have shown that a large contributing factor to an employee's decision to either stay or quit is company culture [29]. Although company culture is an abstract concept, a large part of its effect can be attributed to reducing interpersonal conflict [19]. Models of interpersonal relations in the business literature are sometimes derived from personality types rather than individual interactions [15], or by links in a social network [30].

We define an assignment optimization problem that includes aspects of task assignment [34] or skill based coalition formation [5], as well as additively separable hedonic games [8]. These have occurred together in the ComSoc literature as well as the business literature (further citations in the next section).

In our model, each worker has one or more skills and opinions about some or all of their co-workers. These interpersonal evaluations can change after individuals work together. There is a well-studied phenomenon in psychology of *satiation,* whereby people get overwhelmed by a stimulus — often discussed as food — and it loses its appeal for a while, until variety is re-introduced [17, 33]. For this work, we consider that each worker has a latent or *baseline* valuation (utility) for each other. However, over time, workers' baseline utilities for their teams can be affected by their recent team assignments. Some workers prefer *variety,* meaning that their utility for an individual is temporarily depressed by each recent task they did with that individual, but increases again with each recent task they did without that individual. Workers that prefer *loyalty* have their utility for a co-worker increased with each recent task they did together, and decreased when they didn't work together. In fact, in the psychology literature, research shows that the satiation depends not only on recent consumption, but on the extent of memory of other things consumed [16]. We interpret this in a social setting as individuals having limited memory of interactions. We model this with finite lookback for assessing recent variety or loyalty of co-workers.

It may be the case that it is impossible to staff all of the tasks. We will therefore consider each time period a success if the maximum number of workers are assigned to full-staffed tasks. Given this constraint, we attempt to optimize the workers' total (undiscounted) utility. We call this model a *Temporal Team Formation Game with Dynamic Preferences (TTFG-DP)*. In this paper we are interested in finding utilitarian optimal assignments. Bachrach et al. [6] proved that finding optimal coalition structures for even single-task Coalition Skills Games is NP-hard; Gerkey and Mataric [18] showed a reduction from an NP-hard scheduling problem. In addition, Aziz et al. [3] showed that finding a (utilitarian or egalitarian) partition in an ASHG is NP-hard. Each of these variants is a special case of the one-time-step optimal TTFG-DP problem. Thus, the problem of finding optimal assignments in our setting is also NP-hard.

Because the problem of finding even approximately optimal assignments is hard, we investigate the efficacy of local search to find good assignments. In Section 2, we provide a very brief survey of task assignment and skill based assignment problems. Section 3 gives formal definitions of our problems and the computational complexity of the problem. Section 4 describes heuristic algorithms used to validate computation on our model. Subsection 4.1 describes a greedy algorithm for finding assignments and Subsection 4.2 describes a local search algorithm for this problem. In Subsection 4.3 we describe our instance generation algorithm, and Subsection Section 5 describes the instances used to benchmark the algorithm, our experiments, and the results of those experiments.

* Corresponding Author. Email: aaron.lin399@gmail.com.
** Corresponding Author. Email: goldsmit@cs.uky.edu

2 Related Works

The literature of team formation includes both one-time and iterative team formation. We begin with different approaches to one-time team formation, starting with three application areas.

We are all familiar with the formation of student project teams. This has been studied repeatedly with respect to inter-student preferences (e.g., [12]) or student preferences over team roles (e.g., [14]). Andrejczuk, et al. [2] considered student competences (skill levels), team diversity, and the balance of group skill levels across the class, but only for one time step or semester.

The problem of team formation in the workplace is extensively studied in business and management fields, as was recently surveyed [20, 35]. Juárez et al. provide a taxonomy of team formation problems; ours falls into multi-team assignment with positions. Stavrou et al. write, "It was interesting to discover that the majority of solutions try to maximize the project requirements. Finding a team whose members' talents align with the needs of the intended project is the goal here." But they also found a wide focus on personality traits and cooperativeness. There is a reasonable map from these criteria to our skills and interpersonal preferences. One can make an argument that a broader measure of interpersonal harmony might arise from personality types, in that pairwise preferences need neither be formed nor elicited. Surprisingly, of the many works consulted and the 30 used for that survey, a large number used evolutionary computing to come up with assignments. They do not address issues that arise from iterative team formation.

In the robotics community, team formation typically focuses on distributed team formation; in military or emergency settings, such work is often done with limited communication [13]. While this body of work overlaps with ours due to having tasks with specific skill or competency needs and with tasks occurring over time (going back to Shehory and Kraus [34]), the major differences with our work are that firstly, these are distributed algorithms, whereas ours is centralized, and secondly, barring science fiction settings, robots do not have preferences over other individual robots, though [32] mentions potential synergies between *types of* robots. There have been multiple taxonomies of multi-robot task allocation, e.g., Nunes et al. [27] and Gerkey and Matarić [18]. Both of these look at decentralized team formation, where robots use potentially limited communication to negotiate roles [13] for themselves, typically without explicit rewards, except from fulfilled tasks [36].

The computational social choice literature contains two particularly relevant models. The first is Additively Separable Hedonic Games, where an agent's utility for their coalition is based on the utility they have for each other agent. There are other, less close individual-utility-based hedonic coalition formation games, such as Friends and Enemies Hedonic Games [], Friends, Enemies, and Neutrals Hedonic Games [22], and general graph-based hedonic games, which severely limit the possible inter-agent utility values, and Fractional Hedonic Games [4], in which an agent's utility for a coalition is their average utility of other agents in the coalition. The second highly-related family is the Skill-Based Coalition Formation Games. There are a number of related but distinct formulations, for instance, Tran-Than, et al. [37]'s coalition skill vector model, in which agents may have fractional skills, and the sum over all agents assigned to a given task of their skill levels matches the level of that skill required by the task. There are many variants of skill-vector based games. Andrejczuk, et al. [1] also use fractional skills, but add to their model utility of some central agent for skill-level balance between the coalitions (making this a non-hedonic game), as well as workload balance within teams, based on assignments of individuals to sets of skills they will deploy. Finally, they consider extraneous factors such as diversity of personality types and genders, things we do not consider in this paper.

All of the aforementioned coalition formation games are, however, static games, whereas we are interested in repeated coalition formation. Merida-Campos and Willmott [23, 24] present skill-based iterative coalition formation problems one task at a time. The assignments are closer to skill vector games than to our formulation. Agents seek to maximize their own share of utilities over time, sometimes entering or leaving a task if something better comes along. This captures some of our model without the dynamic utilities, and in a distributed rather than centralized setting.

Chalkiadakas et al. [11] model repeated coalition formation with uncertain inter-agent utilities; as agents experience each other, they refine their beliefs about others' types, and thus about their utilities for those others. The innate utilities don't change, but experience reduces the uncertainty about those utilities. Bullinger and Romen consider team formation made dynamic by agents arriving or leaving [10].

There are other approaches to situation-driven preference changes in coalition games. Nguyen et al. and others have looked at altruistic preferences, which depend not only on agent-to-agent utilities, but the utilities of agents' connections in a social network [26], or agents reachable in a social network [31]. Thus, an agent's utility depends on the current coalition structure, rather than on a history of coalition structures. Monaco et al. [25] offer a similar formulation.

The closest model to ours is Boehmer et al.'s [7]. They consider history dependent preferences, where an agent grows more fond of agents when they are in the same team, and less fond of them when the other agents abandon their team. This corresponds to our notion of loyalty preference, but does not capture preferences for variety. Bullinger and Kober [9] introduce a hybrid of Boehmer et al.'s approach and altruistic utilities, which is situational rather than history-based.

3 The Dynamic Team Assignment Problem

For any productive organization, corporation, or coalition, there exists a finite set of skills needed for an entity to achieve their goals: a painting company needs painters, drivers, managers, etc. Let Σ be the set of skills needed to complete all of the tasks. Each agent has a subset of those skills, defined by a skill vector. Additionally, we model an agent's preferences through time with a preference value to represent an agent's response to changing team compositions and additively separable preferences for teams. We assume that every agent in any set of agents N has a complete set of attributes.

Definition 1. *We define a set of* workers *(or* agents*)* N *in our model with the following attributes for agent* i*:*

- $u_i : N \to \mathbb{R}$ *is the* baseline utility *agent* i *holds for each other agent; we limit values of* $u_i[j] \in [-1, 1]$*. Let* $u_i[i] = 0$*.*
- $\epsilon_i \in \mathbb{R}$ *is* i*'s* preference*. Negative values represent a preference for variety, and positive values represent a preference for loyalty of work groups over time.*
- $s_i \in \{0, 1\}^{|\Sigma|}$ *is* i*'s skill vector, where 0 in index* j *represents agent* i*'s lack of skill* j *and 1 represents possession of skill* j*.*

Definition 2. *At each time step* t*, the set of tasks is* $\mathcal{T}^t = \{T_1^t, T_2^t, ..., T_{last_t}^t\}$*, where each task* $T_i^t \in \mathbb{Z}_{\geq 0}^{|\Sigma|}$ *specifies how many agents with each skill are required. That is,* $T_i^t[j]$ *is the number of agents with skill* j *needed to complete task* i *at time* t*.*

Definition 3. *Let $\pi^t : \mathcal{T}^t \to 2^N$ delegate tasks to groups of agents at time t. We call π* valid *for a given schedule $\mathcal{T}$ if and only if both hold for every time t:*

1. *Skill coverage: For every task T_i^t, the assigned team $\pi^t(T_i^t)$ can be partitioned by skill into disjoint groups, where exactly $T_i^t[j]$ agents have skill j, and each agent possesses the required skill for their group.*
2. *No agent overload: No agent is assigned to more than one task at the same time. That is, for any $i \neq j$, $\pi^t(T_i^t) \cap \pi^t(T_j^t) = \emptyset$.*

We define the set of all agents assigned to tasks at time t as $\pi^t(\mathcal{T}) = \bigcup_{T_i^t \in \mathcal{T}^t} \pi^t(T_i)$.

After workers are assigned to tasks with their coworkers, their utility for other agents may change between time steps. We give the instance *myopia factor* m (which we henceforth refer to as the *look-back*), representing agents' limited memory, and let inter-agent utility change depend only on the previous m assignments. An agent's utility for another agent is updated based on their preference for varying or unchanging team compositions. Agents with positive preference ϵ prefer loyalty and are averse to changing team compositions; their preferences for agents they recently worked with will increase in the next time step and decrease for other agents, reflecting their desire to preserve their team composition. Similarly, agents who prefer variety, having negative ϵ, decrease their utility for teammates they recently worked with and increase their utility for all other agents. Let $f_{i,j,t}$, for a time t, be the frequency that i and j have worked together in the previous m time steps, given π. Let a history H^t at time t contain Boolean values for the past m time steps denoting whether i and j were assigned the same task. Then, if π assigns i and j the same task at time t, the utility of i for j is $u_i^{\pi,t+1}[j] = u_i[j] \cdot (1+\epsilon_i)^{f_{i,j,t}} \cdot (1-\epsilon_i)^{min(m,t)-f_{i,j,t}}$. Since $u_i[i] = 0$, we have $u_i^{\pi,t}[i] = 0$ for all i.

Definition 4. *An instance of a* Temporal Team Formation Game with Dynamic Preferences (TTFG-DP) *consists of a set of agents N, a schedule of tasks $\mathcal{T}$, a myopic factor m, and a penalty for agents without assignments β. The utility-maximizing assignment is defined by:*

$$\text{argmax}_{\pi \text{ is valid}} \sum_{0 \leq t \leq end} \left[\sum_{T_i^t \in \mathcal{T}^t} \sum_{i,j \in \pi^t(T_i^t)} u_i^{\pi,t} - \sum_{N \setminus \pi^t(\mathcal{T})} \beta \right].$$

In the computational social choice setting, there are many approaches to optimization. The most common optimization notions for one-step games are utilitarian, egalitarian, and Pareto optimal. Note that optimization notions for multi-step games may look at total, discounted, or average utilities over time. Because we are looking at finite horizon instances, *our goal is to find an undiscounted utilitarian optimal π*.

The decision version of DTAP is, given an instance I of DTAP and a $b \in \mathcal{N}$, is there an assignment of workers to tasks in I with utility at least b? This is in NP. It is known that finding an optimal ASHG partition is NP-hard by reducing the classic NP-complete problem of TRIPARTITE MATCHING [21]. There is a straightforward mapping from optASHG to DTAP with one skill and one time step (and therefore, unchanging inter-worker utilities). Thus, there is a reduction from TRIPARTITE MATCHING to DTAP with tasks of size 3. Furthermore, the DTAP examples in the reduction have inter-worker utilities in $\{0, 1\}$, and the problem remains NP-hard.

We can see from this reduction that any constant bounds on time steps, skills, or size of tasks (starting from 3, at least) will not change the NP-completeness of the problem. Thus, in terms of parameterized complexity with any of these parameters, the problem is in paraNP.

If we fix the maximum number of workers with each skill to k, then a straightforward algorithm computes the utility for each of the $\mathcal{O}(n^k T)$ assignments for an instance with n workers over T time steps. This problem is in XP.

4 Algorithms

We describe here greedy algorithms and the local search algorithm used to find utilitarian optimal assignments, and an algorithm to generate instances with known, reasonably high utility assignments.

4.1 Greedy Assignments

We used two greedy assignment generation algorithms which repeatedly chose assignments for agents at every timestep. A task, agent pair is chosen repeatedly, according to a heuristic, and then used to generate a valid assignment at every timestep. The second method expands on this largely random assignment generation by interjecting local searches at checkpoints to improve the current partial assignment. Each of these methods

Random Task Generation

For each time step, a task is repeatedly chosen at random and then validly filled, if possible, until no tasks that can be validly filled remain.

Random Tasks with Greedy Local Search Again, for each time step, a task is repeatedly chosen at random and then validly filled, if possible. Once the task is filled, a local search is performed to find a better assignment for that task. The algorithm searches the set of remaining agents for swaps between agents assigned to this task and unassigned agents. In each iteration of the algorithm, it calculates the utility of every swap and performs the swap that has the highest utility. The algorithm continues until either the algorithm reaches a maximum number of swaps of 100 or until there are no improving swaps available.

4.2 Local Search

We used a standard local search algorithm on assignments to $\mathcal{T}$, the entire task schedule, to find local optima with respect to total utility over all agents. Improvements are made to an assignment at a time step t by either inserting an unassigned agent into an unfilled task, replacing an assigned agent by an unassigned agent, or swapping two assigned agents.

An operation is *legal* if the total utility of the assignment increases and all agents have the necessary skills required for the new assignment. i.e., an insertion is legal if the unassigned agent has the required skills for the task. A substitution is legal if the unassigned agent has the skill the assigned agent is fulfilling. And a swap is legal if each agent has the skill the other agent is fulfilling in their assigned task.

Note that if we swap agents i and j in time step t, this affects i and j's immediate utilities (based on their old and new tasks) and for the next m time steps. It also affects the utilities of members of the two tasks with respect to i and j for the next m steps, because it changes the number of times that other agents have worked on a task with i and with j. The same logic applies to insertions and substitutions, where the utility of a previously unassigned agent i

affects their immediate utility, as well as the utility of all agents in the selected task for the next m steps.

Each iteration of local search generates an assignment π^t for every time step $t \in 0 \leq t \leq last_t$, and makes improvements to an assignment by evaluating all potential assignments made by legal moves for all time steps and choosing an assignment that maximizes the total utility. Local search converges when there are no legal operations in the neighborhood of the current assignment.

We created initial assignments for local search to iterate over using both random and greedy methods. Random assignments are made by selecting tasks in a random order and assigning agents with the required skills to the task in a random order. Tasks are filled one at a time until either a valid assignment is found or no agents have the required skill to be assigned to the last task. We found that using greedy methods to generate initial assignments *reduced* their overall variation, since the random assignments used to initialize LS were not constrained to be valid, and thus offered a wider range of possible starting points in the local search topography. Due to this, we only used random initial assignments in the experiments presented here.

4.3 Instances with Good Assignments

In order to provide the tested algorithms with a reasonable benchmark, we generate instances around an intuitively high-value assignment. This high-value assignment is generated in conjunction with the tasks of an instance to ensure the existence of a valid assignment. Then, based on each agent's pairwise assignment history, preferences are chosen to suit their assignments.

First, a random set of agents is declared without inter-agent utilities, but with instantiated skill-vectors and preferences for loyalty or variety (or neither). Then a number of tasks is chosen in $\{1, \ldots, |A|\}$ for every time step. Initially, the skill vectors for these tasks are all 0s. For time step 0, for each agent $a \in N$ selected in a randomized order, we randomly choose one of a's skills, j. A task T_i^t is then randomly chosen. The position count $T_i^t[j]$ is then incremented and agent a gets assigned to T_i^t or $a \in \pi^t(T_i^t)$. For each agent we initialize *pre-utilities* for each other agent to 0.

For subsequent time steps $t+1$, each agent a_i maintains a vector of frequencies for working with each agent a_j over the last t time steps. Then for each time step $t \in \{1, \ldots, 11\}$, at the beginning of the time step we update pre-utilities for each $a \in N$ for each other agent based on whether the two agents worked together in the previous time step, and a's preference. Specifically, agents who with positive ϵ increment their pre-utilities by 1 for each agent they work with and decrement it by 1 for every other agent. Agents who prefer variety update their pre-utilities analogously. Next (still at time step t) we assign agents, considered in a random order, to tasks. Agent a is assigned to T_i^t if T_i^t is an already partly populated task that has the highest value according to the pre-utilities of a and those agents already assigned to the task. However, if that sum is negative, or if there are no partially filled tasks, then a is randomly assigned to an unpopulated task. Finally, we assign skill counts for tasks, and define ρ, as at time 0.

Next, agents' baseline utilities are defined. To assign $u_i[j]$, we use a_i's final pre-utility, normalized to the range $[-1, 1]$.

We refer to these instances as *high-utility instances*, and the algorithm as the *high-utility generation algorithm.* Note that the generated assignment provides a reasonable lower bound on the utility of the optimal assignment for that instance. Also note that the generation algorithm does not take into account the lookback value m, so we can expect different approximations of optimal values, depending on m, even for the same instance.

5 Experiments and Results

All simulations are run on high utility instances with agent preferences sampled from a Gaussian distribution centered at zero with a standard deviation of 1/4th. All games were run for 12 time steps with 5 unique skills.

5.1 Utility of Heuristic Algorithms

For each configuration of instance size ($|N| = 10, 20, 30$), we generated 30 independent high-utility instances. Each instance was evaluated with both greedy algorithms and local search under different lookback factors ($m = 0, 3, 5$). For greedy assignments, each method was used to generate 1000 assignment policies where the highest yielded utility determined the algorithm's performance. For local search, we ran 100 iterations and retained the highest utility encountered.

Results:

As the size of the agent set grows (Figures 1 and 2) the total utilities yielded by the algorithms become more extreme when local search is involved; however, the results normalized to the generated rewards appear to converge. Local searching over each task assignment generated the worst performing assignments, whereas local search made the best assignments of the methods.

When introducing a myopic lookback factor of 3 (Figures 3 and 4) the algorithms perform comparatively worse. As the lookback increases to a large enough value of 5, where utilities are compounded over more steps, the yeilded utilities surpass the non-dynamic instances where utilities remain constant. This increase is not reflected in the normalized results (Figure 4) of the best performing algorithm, local search.

5.2 Effect of Preferences on Utility

To understand how agent preferences for loyalty and variety affect their utility, agents were categorized into three groups based on the distribution of their bias parameter ϵ, which encodes their preference for variety (low ϵ) or loyalty (high ϵ). The bottom 33% of agents were designated as low ϵ, the middle 33% as mid ϵ, and the top 33% as high ϵ. This grouping allowed us to investigate how preferences impact interaction patterns and individual outcomes. All experiments used a lookback window of 3.

Results:

As the size of the agent set grows (Figures 1 and 2) the total utilities yielded by the algorithms become more extreme when local search is involved; however, the results normalized to the generated rewards appear to converge. Local searching over each task assignment generated the worst performing assignments, whereas local search made the best assignments of the methods.

When introducing a myopic lookback factor of 3 (Figures 3 and 4) the algorithms perform comparatively worse. As the lookback increases to a large enough value of 5, where utilities are compounded over more steps, the yeilded utilities surpass the non-dynamic instances where utilities remain constant. This increase is not reflected in the normalized results (Figure 4) of the best performing algorithm, local search.

In 20- and 30-agent instances (Figure 6 and 7), agents with stronger preferences—whether for loyalty or variety—tend to contribute more to the overall assignment utility, suggesting that more

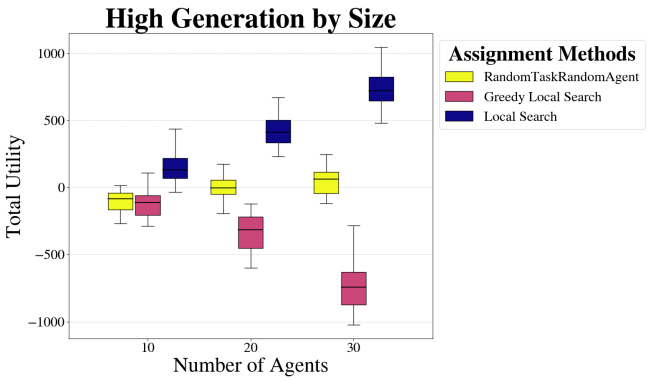


Figure 1. Number of agents vs. total utility; 5 skills, lookback 3

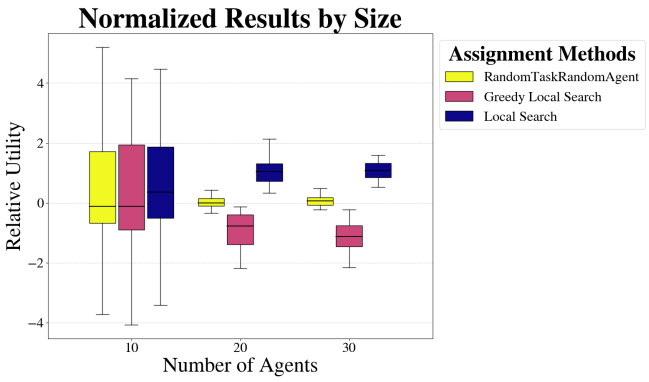


Figure 2. Number of agents vs. normalized utility; 5 skills, lookback 3

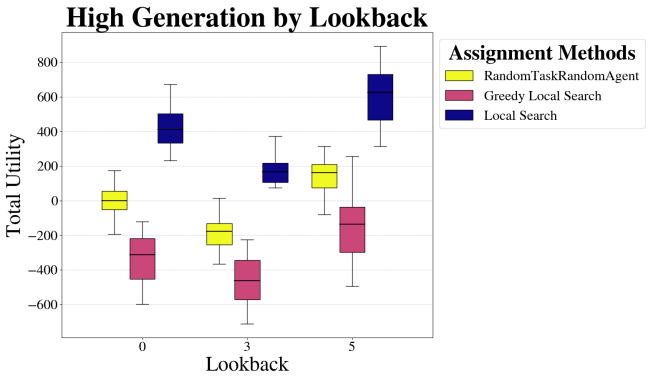


Figure 3. Myopic lookback factor m vs. total utility; 20 agents, 5 skills

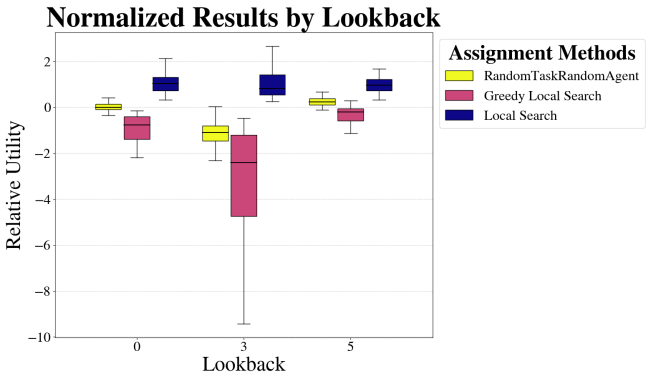


Figure 4. Myopic lookback factor m vs. normalized utility; 20 agents, 5 skills

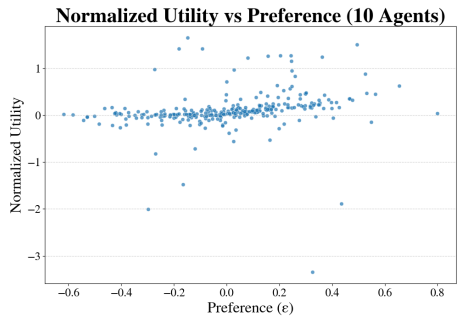


Figure 5. Normalized utility of each agent versus preference from local search (10 agents) for local search

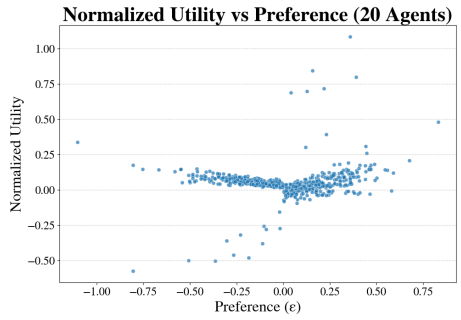


Figure 6. Normalized utility of each agent versus preference from local search (20 agents)

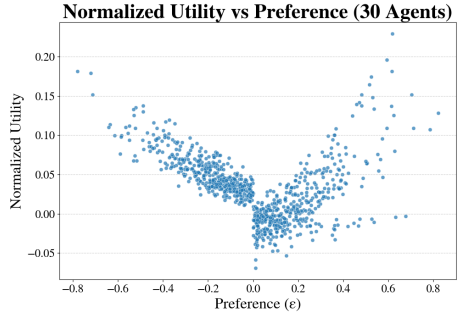


Figure 7. Normalized utility of each agent versus preference from local search (30 agents)

extreme values of ϵ correlate with higher relative utility. This pattern is less pronounced in the 10-agent case (Figure 5). Additionally, low-ϵ agents show occasional negative utility outliers, while high-ϵ agents more often appear among the top utility contributors. Around $\epsilon = 0$ in the 30-agent instances there is a clear drop in utility from negative ϵ agents to positive ϵ agents.

5.3 Satisfaction of Loyal Agents

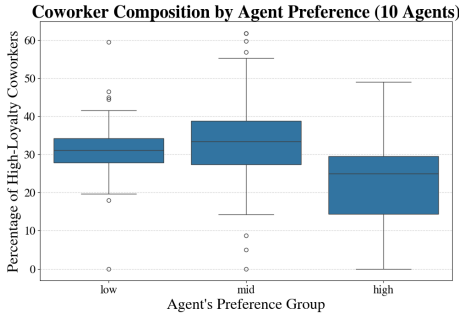


Figure 8. Percentage of loyal agents in assignment versus preference group from local search (10 agents)

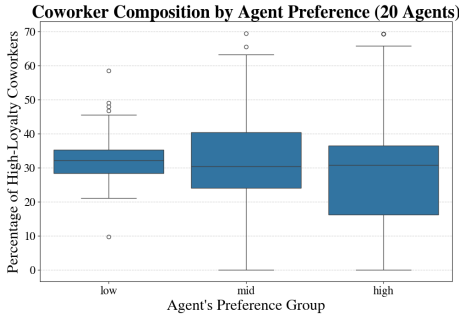


Figure 9. Percentage of loyal agents in assignment versus preference group from local search (20 agents)

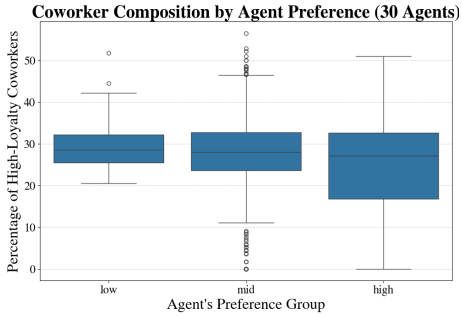


Figure 10. Percentage of loyal agents in assignment versus preference group from local search (20 agents)

For each agent, we recorded the types (terciles) of coworkers they were grouped with at each timestep. This metric captures the distribution of coworker preferences encountered by each agent over time.

Results:

In 10-agent instances (Figure 8), high-ϵ agents generally work with slightly fewer other high-ϵ agents, compared to their low and

mid-ϵ counterparts, though a few outliers do exhibit higher co-loyalty. At 20 and 30 agents (Figure 9 and 10), this distinction becomes less pronounced, with the average co-loyalty appearing more evenly distributed across preference groups with more outliers. In all three instance types there is higher variation in the co-loyalty values of high-ϵ agents relative to low-ϵ agents.

5.4 Persistence of Groups vs. Preference

We measured the average rate at which each agent's coworker set changed between consecutive timesteps. Turnover was defined as the symmetric difference over the union of the agent's coworker sets from adjacent timesteps.

Results:

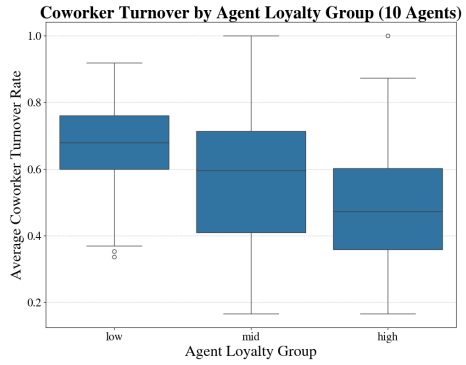


Figure 11. Average coworker turnover rate versus loyalty group from local search (20 agents)

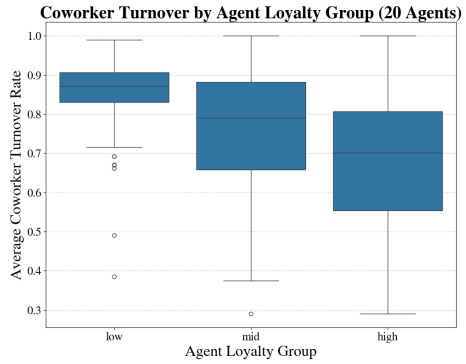


Figure 12. Average coworker turnover rate versus loyalty group from local search (20 agents)

Across all instance sizes, assignments found by local search were more likely to change the set of coworkers around low-ϵ agents relative to high-ϵ agents (Figure 11, 12, and 13). High-ϵ and mid-ϵ agents have higher variance in turnover rate compared to low-ϵ agents. As the size of the instances increases, low-ϵ agents tend to have higher average turnover rates among their coworkers.

6 Discussion

We have seen that our local search algorithm can outperform the instance generation algorithm as the loyalty and variety preferences become more common. We conjecture that the effects of those preferences dominate the baseline inter-agent utilities — in other words,

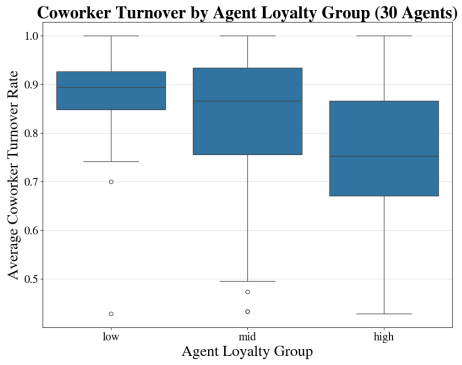


Figure 13. Average coworker turnover rate versus loyalty group from local search (30 agents)

it is easy to make agents happy by respecting those preferences than by focusing on baseline utilities, as the lookback, m, increases. It seems that local search is better able to leverage those preferences than the assignment generation approaches.

Overall, our results indicate that agents with more extreme ϵ values accrue more utility over the course of assignments. When instance sizes get larger, such as in the 20- and 30- agent case the average turnover rate of low-ϵ agents is very high, $\sim$85-90% (Figures 12 and 13). It is not clear whether this is inherent in the problem or in the local search approach.

From Figure 6 we see that when ϵ is a relatively low positive or negative value, local search tends to prioritize variety-preferenced agents. This is most likely because the set of agents an agent *hasn't* worked with is larger than the number of agents that they *have* worked with. Thus, it is easier, statistically, to satisfy a desire for variety than one for loyalty, as the skill demands of tasks vary.

From Figure 9 we see that while on average the percentage of high-loyalty agents an agent works with stays the same between loyalty groups, the higher variance associated with the high-loyalty preference group. Furthermore, while the average turnover rate does decrease from low- to high-ϵ (Figure 12), the actual compositions don't show a tendency for high-ϵ agents to cluster together. This suggests that a few select loyalties are prioritized by the algorithm.

7 Conclusions and Future Work

We have introduced a model that combines skill-based coalition formation, interagent preferences, and variety/loyalty based dynamics on those preferences. We showed that the problem of finding an optimal assignment is NP-hard, and investigated several heuristic approaches to agent assignment. Future work includes using a bi-criteria approach to include the agency's utilities for tasks to explore which agent satisfiability metrics are preserved when a company decides to prioritize its own goals, and completing the fixed-parameter analysis of this problem. We are also interested in considering tasks that stretch over multiple time steps, particularly when combined with likelihoods of task completion.

Another one of the open questions for this work is about the prevalence of loyalty and variety preferences for different (human) populations. For instance, we conjecture that computer scientists have a higher preference for loyalty than the general population.

References

[1] E. Andrejczuk, R. Berger, J. A. Rodriguez-Aguilar, C. Sierra, and V. Marín-Puchades. The composition and formation of effective teams:

computer science meets organizational psychology. *The Knowledge Engineering Review*, 33:e17, 2018.

[2] E. Andrejczuk, F. Bistaffa, C. Blum, J. A. Rodríguez-Aguilar, and C. Sierra. Synergistic team composition: A computational approach to foster diversity in teams. *Knowledge-Based Systems*, 182:104799, 2019.

[3] H. Aziz, F. Brandt, and H. G. Seedig. Optimal partitions in additively separable hedonic games. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence-Volume Volume One*, pages 43–48, 2011.

[4] H. Aziz, F. Brandl, F. Brandt, P. Harrenstein, M. Olsen, and D. Peters. Fractional hedonic games. *ACM Transactions on Economics and Computation (TEAC)*, 7(2):1–29, 2019.

[5] Y. Bachrach and J. S. Rosenschein. Coalitional skill games. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems Volume 2*, pages 1023–1030, 2008.

[6] Y. Bachrach, R. Meir, K. Jung, and P. Kohli. Coalitional structure generation in skill games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, pages 703–708, 2010.

[7] N. Boehmer, M. Bullinger, and A. M. Kerkmann. Causes of stability in dynamic coalition formation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5499–5506, 2023.

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

[9] M. Bullinger and S. Kober. Loyalty in cardinal hedonic games. In *Proc. IJCAI*, 2021.

[10] M. Bullinger and R. Romen. Stability in online coalition formation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 9537–9545, 2024.

[11] G. Chalkiadakis and C. Boutilier. Sequentially optimal repeated coalition formation under uncertainty. *Autonomous Agents and Multi-Agent Systems*, 24:441–484, 2012.

[12] B. A. Craker, R. M. Erdmann, and K. Metzger. The mutual preference method: Capitalizing on student relationships to guide group formation. *The American Biology Teacher*, 84(9):570–572, 2022.

[13] G. Diehl and J. A. Adams. The viability of domain constrained coalition formation for robotic collectives. *Swarm Intelligence*, pages 1–42, 2024.

[14] J. A. Donndelinger, A. Weaver, J. C. Bates, and T. Russell. Automating project team formation with heterogeneous project preferences and skill mix constraints. In *ASEE 2021 Gulf-Southwest Annual Conference*, 2021.

[15] M. Farhangian, M. K. Purvis, M. Purvis, and B. T. R. Savarimuthu. Modeling the effects of personality on team formation in self-assembly teams. In *PRIMA 2015: Principles and Practice of Multi-Agent Systems: 18th International Conference, Bertinoro, Italy, October 26-30, 2015, Proceedings 13*, pages 538–546. Springer, 2015.

[16] J. Galak, J. P. Redden, and J. Kruger. Variety amnesia: Recalling past variety can accelerate recovery from satiation. *Journal of Consumer Research*, 36(4):575–584, 2009.

[17] J. Galak, J. Kruger, and G. Loewenstein. Is variety the spice of life? it all depends on the rate of consumption. *Judgment and Decision making*, 6(3):230–238, 2011.

[18] B. P. Gerkey and M. J. Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, 23(9):939–954, 2004.

[19] A. Hussain. Workplace conflicts and its effect on employee productivity: A mediating role of workplace politics. *International Journal of Psychosocial Rehabilitation*, 24(3):2774–2783, 2020.

[20] J. Juárez, C. Santos, and C. A. Brizuela. A comprehensive review and a taxonomy proposal of team formation problems. *ACM Computing Surveys (CSUR)*, 54(7):1–33, 2021.

[21] R. Karp. Reducibility among combinatorial problems, 1972.

[22] J. Lang, A. Rey, J. Rothe, H. Schadrack, and L. Schend. Representing and solving hedonic games with ordinal preferences and thresholds. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1229–1237, 2015.

[23] C. Mérida-Campos and S. Willmott. The effect of heterogeneity on coalition formation in iterated request for proposal scenarios. *EUMAS*, 223, 2006.

[24] C. Merida-Campos and S. N. Willmott. Modelling coalition formation over time for iterative coalition games. 2004.

[25] G. Monaco, L. Moscardelli, Y. Velaj, et al. Hedonic games with social context. In *ICTCS*, pages 24–35, 2018.

[26] N.-T. Nguyen, A. Rey, L. Rey, J. Rothe, and L. Schend. Altruistic hedonic games. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 251–259, 2016.

[27] E. Nunes, M. Manner, H. Mitiche, and M. Gini. A taxonomy for task allocation problems with temporal and ordering constraints. *Robotics and Autonomous Systems*, 90:55–70, 2017.

[28] S. of Human Resource Management. The real costs of recruitment, 2024. URL http://web.archive.org/web/20080207010024/http://www.808multimedia.com/winnt/kernel.htm.

[29] PayScale. Strategies to improve employee retention report, 2023. URL https://www.payscale.com/research-and-insights/retention-report/.

[30] S. S. Rangapuram, T. Bühler, and M. Hein. Towards realistic team formation in social networks based on densest subgraphs. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1077–1088, 2013.

[31] T. Schlueter and J. Goldsmith. Super altruistic hedonic games. In *The Thirty-Third International Flairs Conference*, 2020.

[32] T. C. Service and J. A. Adams. Coalition formation for task allocation: Theory and algorithms. *Autonomous Agents and Multi-Agent Systems*, 22(2):225–248, 2011.

[33] J. Sevilla, J. Lu, and B. E. Kahn. Variety seeking, satiation, and maximizing enjoyment over time. *Journal of Consumer Psychology*, 29(1): 89–103, 2019.

[34] O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial intelligence*, 101(1-2):165–200, 1998.

[35] G. Stavrou, P. Adamidis, J. Papathanasiou, and K. Tarabanis. Team formation: A systematic literature review. 2023.

[36] A. Tchaikovsky. *Service Model*. Tor Books, 2024.

[37] L. Tran-Thanh, T.-D. Nguyen, T. Rahwan, A. Rogers, and N. R. Jennings. An efficient vector-based representation for coalitional games. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, pages 383–389, 2013.