Resource Task Games

Jessica L. Newman
University of Southampton
Southampton, UK
jln1g19@soton.ac.uk

Enrico Gerding
University of Southampton
Southampton, UK
eg@soton.ac.uk

Enrico Marchioni
University of Southampton
Southampton, UK
e.marchioni@soton.ac.uk

Baharak Rastegari
University of Southampton
Southampton, UK
b.rastegari@soton.ac.uk

ABSTRACT

In this work, we introduce Resource Task Games (RTGs), a model of cooperative strategic interactions generalising Wooldridge and Dunne's Coalitional Resource Games. In RTGs, agents are endowed with different types of resources, which can be put towards graded completion of certain tasks. Agents have preferences over the states of completion of these tasks and can allocate resources in cooperation with other agents. We introduce a notion of core for RTGs and investigate the existence and computation of stable outcomes and core-related closure properties. We show that RTGs are sufficiently expressive to encode Transferable Utility (TU) games efficiently, providing a construction from an arbitrary TU game to an RTG that preserves the core. We provide the computational complexity classes of problems relating to the core of these games, including bounds on the polynomial hierarchy for each problem.

KEYWORDS

Coalitional games; Resources; Computational complexity; Multiagent systems; Core; Coalitions; Transferable Utility Games

ACM Reference Format:
Jessica L. Newman, Enrico Gerding, Enrico Marchioni, and Baharak Rastegari. 2025. Resource Task Games. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

1 INTRODUCTION

Understanding the emergence of cooperation between self-interested agents is one of the key aspects in the study multiagent systems. Modelling how coalitions are formed and to what degree and in what sense they are beneficial for the agents involved is crucial for the development of formal representations trying to capture cooperative interactions. In recent years, borrowing notions and techniques from classical cooperative game theory, theoretical research in multiagent systems has put forward several different approaches and computational models with the goal of offering a formal treatment of agents' cooperative behaviour (see [13]). Within this framework, a substantial amount of work has been devoted to the study of how cooperation might emerge between resource-bound agents who allocate their assets to achieve their private goals (for instance, the logical approaches in [1, 2, 15, 26, 33]).

This work follows the above-mentioned trend and introduces Resource Task Games (RTGs), a model of cooperative strategic interactions where agents allocate resources at their disposal in order to complete certain tasks. RTGs generalise Coalitional Resource Games (CRGs), introduced by Wooldridge and Dunne [18, 35]. In CRGs, agents each have a set of goals and are endowed with finite amounts of discrete resources that can be allocated towards achieving these goals. A goal is completed once it has received a certain amount of each required resource. Every agent ultimately aims at achieving at least one of their goals, and, at the same time, minimising the amount of resources they allocate. Agents can form coalitions by pooling their resources together, and a coalition is successful whenever it has enough collective resources to achieve at least one goal in the goal set of each agent in the coalition.

Similar to CRGs, in RTGs, agents are endowed with different types of resources. However, they extend CRGs in two fundamental aspects. First, agents in RTGs have tasks rather than goals: these tasks can have different levels of completion depending on the combination of resources allocated to them. Second, each agent has individual preferences over the states of completion of each task: these preferences are specified by a real-valued function. RTGs offer then a quantitative generalisation of the CRGs model where agents can pool together resources to achieve tasks to different degrees, and can capture complex preferences of the results of their allocations, going beyond the simple positive or negative achievement of qualitative goals.

This work is structured as follows. In Section 2, we formally introduce RTGs and define appropriate notions of coalition and coalition structure. In Section 3, we define the concepts of stable outcome and (pessimistic) core and examine some of their properties in relation to dependencies between agents. In Section 4, we show that RTGs are sufficiently expressive to encode Transferable Utility games [25], by giving an efficient construction from an arbitrary TU game to a RTG so that the core of the former is preserved in the latter. Section 5 investigates the computational complexity of three problems in RTGs: verifying if a resource assignment provides the lowest utility for a particular agent, checking if a given coalition structure is in the core, and checking if the core is non-empty. For each problem, we give bounds on the polynomial hierarchy [4].

Section 6 covers related work on cooperative games, while Section 7 concludes with some remarks on future work.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

2 PRELIMINARIES

Let us start with a toy example.

Example 2.1. *There are 3 power plants with different capacities for producing energy, and an energy company they can sell to. The company will pay proportional to the amount of energy sold to them, but will only pay for at most 100GWh at once. Power plants can work together to increase the amount of energy sold at once. The company wants to encourage selling more in one go, so power plants working together will each receive full payment for the amount of collective energy sold, as if they had each sold it individually. Thus agents in a coalition prefer the other members to sell more energy, since they will also see the benefit without any additional cost.*

We next define RTGs and show how the above example can be formally modelled in Example 2.3.

2.1 Resource Task Games

A Resource Task Game Γ is a tuple:

$$\Gamma = (N, R, \textbf{type}, t_1, \ldots, t_k, \textbf{en}, (v_i)_{i \in N}, (cost_i)_{i \in N})$$

where:

- $N = \{1, \ldots, n\}$ is a set of n agents.
- $R = \{r_1, \ldots, r_m\}$ is a set of m resource types.
- **type** : $R \to \{\text{discrete}, \text{continuous}\}$ is a function that determines whether each resource type is continuous (it can be expressed in real-valued quantities) or discrete (it can only be expressed in integer quantities).
- $t_1, \ldots, t_k$ are k different *task functions*. Each $t_a : Y_1 \times \ldots \times Y_m \to \mathbb{R}$ can have an arbitrary real-valued outcome depending on the amount of each resource type assigned to it. Each Y_j represents the input of the corresponding resource r_j. $Y_j = \mathbb{R}_0^+$ if r_j is continuous, and $Y_j = \mathbb{N}$ if r_j is discrete. We use the words 'task' and 'task function' interchangeably.
- **en** : $N \times R \to \mathbb{R}_0^+$ is a function that specifies how much of each resource each agent is endowed with. For discrete resources, the endowment is constrained to $\mathbb{N}$.
- Each $v_i : \mathbb{R}^k \to \mathbb{R}$ is the *value function* for agent i, which maps outputs of task functions into real numbers, representing agent i's preferences over the outcomes of tasks.
- Each $cost_i : \mathbb{X}_i \to \mathbb{R}$ is the *cost function* for agent i, representing the cost to agent i for each possible resource assignment they could make. Each $\mathbb{X}_i$ is the set of feasible resource assignments for i, see Definition 2.2.

Agents in RTGs are able to make *resource assignments* to tasks, choosing how much of each of their resources go towards each task. For an agent i, a resource assignment X_i is a $k \times m$ matrix (where k is the number of task functions and m is the number of resource types) with entries in $\mathbb{R}$. A value $x_{a,b}$ at row a and column b indicates an assignment of $x_{a,b}$ amount of resource r_b to task t_a. When discussing elements of more than one resource assignment, we differentiate using the agent name as a superscript; for example the assignment of resource r_b to task t_a by agent i will be $x^i_{a,b}$ and by agent j will be $x^j_{a,b}$.

Definition 2.2. *A resource assignment X_i is feasible if:*

- *For any discrete resource r_b, for all tasks t_a, $x^i_{a,b} \in \mathbb{N}$*
- *For any continuous resource r_b, for all tasks t_a, $x^i_{a,b} \in \mathbb{R}$*
- *For each resource r_b, the total amount of that resource assigned across all tasks by an agent i should not be greater than their endowment: $\sum_{j=1}^{k} x^i_{j,b} \le \textbf{en}(i, r_b)$*

Given a subset C of agents, we denote the concatenation of individual resource assignments X_i from each $i \in C$ as X_C. We use $-C$ as shorthand for $N \setminus C$, so X_{-C} would represent a collection of resource assignments for all agents outside C.

Given some collection of resource assignments X_C, we use $t_j(X_C)$ as shorthand for $t_j(\sum_{i \in C} x^i_{1,j}, \ldots \sum_{i \in C} x^i_{m,j})$. In other words, for each resource we sum over each agent's assignment of that resource to task j, and use this as the input to the function.

Each of the k inputs to the value function represents the output of each of the k tasks, so that agent preferences over different states of task can be encoded. The cost function for each agent defines how they value each of their feasible resource assignments. The value and cost functions together induce a utility for each agent over joint resource allocations from all agents.

Given a collection of resource assignments for all agents, X_N, the utility of agent i is defined as:

$$u_i(X_1, \ldots X_n) := v_i(t_1(X_N)), \ldots, t_k(X_N)) - cost_i(X_i)$$

So, we calculate the state of each task based on the collective resource allocation, and work out how the agent values this particular combination of task states. We then subtract the agent's personal cost for the resources they spent to achieve this.

Example 2.3. *Consider the setting in Example 2.1. This can be modelled as a game:*

$$\Gamma = (N = \{1, 2, 3\}, \{r_{energy}\}, \{continuous\}, t_{sell}, \textbf{en}, (v_i)_{i \in N}, (cost_i)_{i \in N})$$

Where $t_{sell}(x_{energy}) = \min(100, x_{energy})$, and $v_i(t_{sell}(x_{energy})) = t_{sell}(x_{energy})$, $cost_i(X_i) = x^i_{sell,energy}$ for all $i \in N = \{1, 2, 3\}$, and $\textbf{en}(1, r_{energy}) = 40$, $\textbf{en}(2, r_{energy}) = 60$, $\textbf{en}(3, r_{energy}) = 100$.

So, there is a single task that increases linearly with the sum of resource r_{energy} allocated to it until reaching a cap of 100, representing the payment for that amount of energy. All agents would prefer to see this task at as high a value as possible, but wish to use as little of their resource r_{energy} as possible. Agent 1 is endowed with 40 GWh of r_{energy}, agent 2 with 60 GWh, and agent 3 with 100 GWh. Let us consider the resource assignments $X_1 = X_2 = (0), X_3 = (100)$. The utilities for this assignment are $u_1(X_N) = u_2(X_N) = 100$, but $u_3(X_N) = 100 - 100 = 0$. Alternatively, for resource assignments $X_1 = (40), X_2 = (50), X_3 = (50)$, the utilities will be $u_1(X_N) = 60, u_2(X_N) = u_3(X_N) = 50$

2.2 Coalitions and Coalition Structures

Agents in an RTG are able to form coalitions, in which all the agents within the coalition commit to a joint action. We define a *coalition* λ as a tuple containing the subset C of agents within the coalition, and for each agent i in C a resource allocation chosen from i's space of feasible resource allocations, i.e.

$$\lambda = (C, (X_i)_{i \in C}), \quad \text{where } C \subseteq N,\ X_i \in \mathbb{X}_i$$

Recall that this can be abbreviated to $\lambda = (C, X_C)$. We will denote the set of all coalitions in a game Γ as Λ_Γ. If a coalition contains N, the set of all agents, we will call it a *grand coalition*.

In order to define agent preferences over coalitions, we need a way of handling the externalities outside each coalition: the resource assignments of agents outside of the coalition set C. To get a utility for agent i in a coalition λ, a reasonable[1] choice would be to fix the resource assignments of those inside the coalition, and take the minimum value of agent i's utility over all possible resource assignments external to the coalition to get a worst-case or risk-averse utility for each agent. We call this the *pessimistic utility*, and denote the pessimistic utility of agent i in coalition λ as $u_i^{\downarrow}(\lambda)$:

$$u_i^{\downarrow}(C, X_C) := \inf_{X_{-C}} (u_i(X_C, X_{-C}))$$

Example 2.4. *Continuing Example 2.3, consider the coalition:*

$$\lambda = (\{1, 2\}, X_1 = (40), X_2 = (20))$$

For both agents, the worst case resource assignment for agent 3 *is* (0)*. Therefore,* $u_1^{\downarrow}(\lambda) = 20$ *and* $u_2^{\downarrow}(\lambda) = 40$.

We can then define a preference ordering $\succ_i^{\downarrow}$ for each agent i over coalitions for which they are a member:

$$\lambda' \succ_i^{\downarrow} \lambda \quad \text{iff} \quad u_i^{\downarrow}(\lambda') > u_i^{\downarrow}(\lambda)$$

It is easy to see that the pessimistic utility for each agent in a coalition $\lambda = (C, X_C)$ is monotonic with respect to expansions of C: new agents joining the coalition do not decrease the utility of the agents in λ, as their resource assignment is already accounted for. From this, we can construct an inclusion ordering over coalitions such that, for two coalitions $\lambda = (C, X_C)$ and $\lambda' = (S, X_S)$, $\lambda \subseteq \lambda'$ if and only if $C \subseteq S$ and X_C agrees with X_S on overlapping agents. Then:

Proposition 2.5. *For any two coalitions λ and λ' and any agent i in both, if $\lambda \subseteq \lambda'$ then $u_i^{\downarrow}(\lambda) \le u_i^{\downarrow}(\lambda')$.*

We define the operation of disjoint union on coalitions with no shared agents as follows:

Definition 2.6. *For coalitions $\lambda_1 = (C, X_C), \lambda_2 = (S, X_S)$ where $C \cap S = \emptyset$, we define the disjoint union as $\lambda_1 \sqcup \lambda_2 = (C \cup S, X_C, X_S)$.*

It follows from Proposition 2.5 that agents always prefer the disjoint union of two coalitions at least as much as either coalition separately:

Corollary 2.7. *For any two disjoint coalitions λ_1, λ_2 and any agent i in λ_j, $j \in \{1, 2\}$, $u_i^{\downarrow}(\lambda_j) \le u_i^{\downarrow}(\lambda_1 \sqcup \lambda_2)$.*

An outcome of an RTG is a *coalition structure*. This is a partition of the N agents in the game into coalitions. The space of all possible coalition structures in a game Γ is be denoted by Σ_Γ. We will say the utility of a particular coalition structure σ for agent i will be the pessimistic utility of i for the coalition i appears in. If we denote by $\lambda_{\sigma,i}$ the coalition in σ which contains i, then the utility of a coalition structure σ for i is: $u_i^{\downarrow}(\sigma) := u_i^{\downarrow}(\lambda_{\sigma,i})$.

Example 2.8. *Continuing from Example 2.3, a possible coalition structure could be:*

$$\sigma = \{(\{1, 2\}, X_1 = (40), X_2 = (20)), (\{3\}, X_3 = (60))\}$$

with utilities: $u_1^{\downarrow}(\sigma) = u_1^{\downarrow}((\{1, 2\}, X_1 = (40), X_2 = (20))) = 20$; $u_2^{\downarrow}(\sigma) = u_2^{\downarrow}((\{1, 2\}, X_1 = (40), X_2 = (20))) = 40$; $u_3^{\downarrow}(\sigma) = u_3^{\downarrow}((\{3\}, X_3 = (60))) = 0$. *Note that, even though a total of* 120 *units of r_{energy} is assigned to t_{sell}, each coalition's payment is based only on the resources provided within the coalition.*

We call a coalition structure that contains only a grand coalition a *grand coalition structure.*

3 THE CORE

One of the most important solution concepts in cooperative game theory is the core [25]. This is the set of 'stable outcomes', in the sense that in one such outcome no subset of agents are incentivised to deviate from the current outcome and form their own coalition. A coalition structure σ is *blocked* by a coalition $\lambda = (C, X_C)$ iff all members of the coalition strictly prefer λ to their coalition in σ:

$$\lambda \text{ blocks } \sigma \text{ iff } \lambda \succ_i \lambda_{\sigma,i} \text{ for all } i \in C$$

We say a coalition structure σ is in the *pessimistic core* or *p-core* of a game Γ if it is not blocked by any coalition $\lambda \in \Lambda_\Gamma$.

Definition 3.1. $PCORE(\Gamma) = \{\sigma \in \Sigma_\Gamma | \neg\exists\lambda \in \Lambda_\Gamma \text{ s.t. } \lambda \text{ blocks } \sigma\}$.

By Corollary 2.7, we can infer that the p-core is closed under the operation of taking the disjoint union of coalitions within coalition structures. So, by taking the disjoint union of *all* coalitions within a coalition structure in the p-core, we can construct a grand coalition structure in the p-core with the same resource assignments for each agent. Therefore, to check the non-emptiness of the p-core, it is sufficient to check all possible grand coalition structures.

Proposition 3.2. *For a given coalition structure $\sigma = \{\lambda_1, \ldots, \lambda_\ell\}$, if $\sigma \in PCORE(\Gamma)$ then for the grand coalition $\sigma' = \{\lambda_1 \sqcup \ldots \sqcup \lambda_\ell\}$ it is also the case that $\sigma' \in PCORE(\Gamma)$*

Example 3.3. *Continuing from Example 2.8, the grand coalition structures in the p-core of our game are of the form:*

$$\sigma = \{(\{1, 2, 3\}, X_1 = (x), X_2, = (y), X_3 = (z))\}$$

Where $x + y + z = 100$ (and X_1, X_2, X_3 are all feasible). To see why any grand coalition is blocked when $x + y + z < 100$, define $d := 100 - (x + y + z)$; agents can each spend $d/3$ extra units of r_{energy} to obtain a net gain of $d/2$ utility.

3.1 Core Decomposition

We have seen that taking disjoint union of coalitions in a coalition structure will preserve membership of the p-core. There are also circumstances where we can split coalitions apart in a p-core coalition structure, and still have a coalition structure in the p-core. This is based on the *dependencies* between agents. We say an agent i is *dependent* on another agent j whenever j has some resource assignment that affects i's utility.

Definition 3.4. *Agent i is dependent on agent j if there is some resource assignment X_{-j} and two resource assignments X_j, X_j' such that $u_i(X_{-j}, X_j) \neq u_i(X_{-j}, X_j')$*

We say an agent i is *independent* of an agent j if i is not dependent on j. If we know an agent i is independent of an agent j, then neither adding nor removing agent j from a coalition containing

[1]and common, e.g. in the α-core in classical game theory [5] or in the semantics of coalition logic and ATL [27][3]

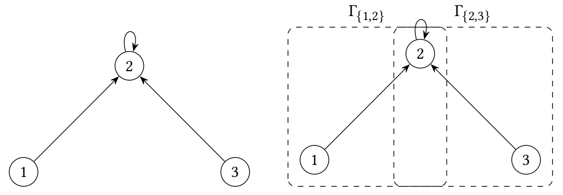

Figure 1: *Left:* a dependency graph for a Resource Task Game. *Right:* two dependency-closed subgames of the game.

i will affect i's utility. So, if we have two subsets of agents C and S such that all agents in each subset are independent of all agents in the other, then whenever we have a coalition $\lambda = (C \cup S, X_{C \cup S})$ we can split this into two coalitions $\lambda_1 = (C, X_C), \lambda_2 = (S, X_S)$ such that $\lambda = \lambda_1 \sqcup \lambda_2$ and all agents will earn the same utility in λ as in λ_1 or λ_2. If we have a coalition structure in the p-core, therefore, that contains such a λ, we know we can substitute λ for λ_1, λ_2 and the resulting coalition structure will also be in the p-core.

Since independence is defined over all possible resource assignments, we know whenever *any* coalition consisting of multiple independent subsets of agent forms, we are able to split it and preserve utilities. We can use this to decompose a game into smaller subgames in a way that is consistent with the p-core. If we take subgames with only dependency-closed sets of agents, we know the elements of the p-core of each subgame cannot be blocked by agents outside of this subgame, since they have no effect on utility.

Definition 3.5. *Given an RTG Γ with set of agents N, we call a subset C of agents dependency-closed iff for all $i \in C$, if i is dependent on $j \in N$ then $j \in C$.*

Definition 3.6. *Given an RTG Γ, the subgame of Γ restricted to C is the game Γ_C identical to Γ but with agent set C, domain of* ***en*** *restricted to $C \times R$, and $v_i, cost_i$ from only $i \in C$.*

Definition 3.7. *A collection of subgames of Γ form a cover of Γ when the union of all of their agent sets is the agent set of Γ.*

For any RTG we can create a dependency graph with agents as nodes and an edge (i, j) whenever i depends on j. Suppose we have a game with 3 agents where all 3 agents are endowed with some amount of resource r, and only one of the agents gains utility when this resource is allocated to a task. In Figure 1 we can see the dependency graph for this game, and a collection of dependency-closed subgames that form a cover of the game. We can reconstruct elements in the p-core of the whole game by finding elements of the p-core of each of the subgames. If none of the subgames share agents, we can simply take the product of all the p-cores of the subgames and close under disjoint union of coalitions to find the p-core of the whole game. Otherwise, we need to ensure agents shared in multiple subgames are making the same assignment in each. We will index our collection of subgames over a set I and call the collection of subgames $(\Gamma_i)_{i \in I}$ and their respective agent sets $(C_i)_{i \in I}$. To be able to generate a p-core element of Γ using a p-core element σ_i from each Γ_i, the following condition is required:

(Consistency) If some agent i appears in two different agent sets C_j, C_k then i's resource assignment in σ_j must be equal to i's resource assignment in σ_k. That is, if any games have overlapping agents, these agents are making the same resource assignment in each game.

A collection of p-core elements from dependency-closed subgames that meet this condition is called a *valid decomposition.*

Definition 3.8. *Suppose we have an RTG Γ with agent set N and collection of dependency-closed sets $(C_i)_{i \in I}$ such that each $C_i \subseteq N$ and $\cup_{i \in I} C_i = N$. Suppose we have a collection of games $(\Gamma_i)_{i \in I}$, such that each Γ_i is a subgame of Γ restricted to C_i. We will call a collection of coalition structures $(\sigma_i)_{i \in I}$* a valid decomposition *if each σ_i is in the p-core of Γ_i, and the consistency condition holds.*

Given a collection of p-core elements $(\sigma_i)_{i \in I}$ that forms a valid decomposition, any coalition structure σ will be in the p-core of the whole game if for each σ_i and each $\lambda \in \sigma_i$, there is some $\lambda' \in \sigma$ such that $\lambda \subseteq \lambda'$.[2] This ensures agents in multiple subgames are in a coalition with all of the coalition members across each of their subgames. For example, suppose for the game in Fig. 1 we had a valid decomposition $\sigma_{\{1,2\}} = \{(\{1, 2\}, X_1, X_2)\}$ and $\sigma_{\{2,3\}} = \{(\{2, 3\}, X_2, X_3)\}$. Then, $\sigma = \{(\{1, 2, 3\}, X_1, X_2, X_3)\}$ is guaranteed to be in the p-core of the whole game Γ. We can also go in the other direction: given an element σ of the p-core in an RTG, we can find p-core elements for a dependency-closed subgame by simply removing any agents from σ that do not appear in the subgame.

Proposition 3.9. *Given an RTG Γ, dependency-closed subsets of agents $(C_i)_{i \in I}$ such that $\bigcup_{i \in I} C_i = N$, and a collection of games $(\Gamma_i)_{i \in I}$, such that each Γ_i is a subgame of Γ restricted to C_i: A coalition structure σ is in the p-core of Γ iff there exists a valid decomposition $(\sigma_i)_{i \in I}$ such that $\forall \sigma_i, \forall \lambda \in \sigma_i$ there is a $\lambda' \in \sigma$ such that $\lambda \subseteq \lambda'$.*

Proof. ($\Leftarrow$) We are given a valid decomposition $(\sigma_i)_{i \in I}$ and want to show that any σ is in the p-core if for each σ_i, for each $\lambda \in \sigma_i$, there is a $\lambda' \in \sigma$ such that $\lambda \subseteq \lambda'$. Each σ_i is in the p-core of some subgame Γ_i of Γ restricted to some C_i. Let us suppose σ in Γ was blocked via some coalition λ that contains members in C_i, where these members are receiving strictly higher utility than in σ. Since the subsets are dependency closed, the only agents that can affect the utility of agents in C_i are also in C_i. Therefore, when we restrict λ to only members of C_i, call this λ', the agents in this coalition would still have the same utility - so λ' also blocks σ. Since resource assignments are identical and the coalitions are no larger, agents cannot earn more utility in σ_i than σ, and λ' is a possible coalition in Γ_i, so must also block σ_i. However, σ_i is in the p-core of Γ_i by assumption, so, σ must be in the p-core of Γ.

($\Rightarrow$) We have some σ in the p-core of Γ. Take some dependency-closed subset of agents in Γ, and call this C_i. We will form a coalition structure σ_i in the subgame Γ_i of Γ restricted to C_i, by removing all agents and resource assignments from coalitions in σ that are not in Γ_i. Due to dependency closure, agents in σ_i earn the same utility as in σ. Therefore, if σ_i were blocked through some coalition, this same coalition would block σ - which is, by assumption, in the core. It can also be verified that this way of constructing each σ_i gives us a valid decomposition that meets the Consistency condition. □

[2]Taking the largest such σ (and therefore the σ with the smallest coalitions) will give all other coalition structures that meet these conditions, since they can be obtained through a sequence of disjoint union operations on the largest σ

If the core is empty for one of these subgames, we are unable to construct a valid decomposition, so there is no way to retrieve coalition structures in the core of the whole game.

Corollary 3.10. *An RTG Γ has an empty p-core iff there exists a dependency-closed subset of agents C such that the subgame restricted to C has an empty p-core.*

In this notion of dependence we check against every possible resource assignment all agents can make. This means we can use the subgames it generates to split *any* coalition structure in the p-core. However, there may be additional ways of splitting specific coalition structures. To find these, we need a more local notion of dependence, where we have some resource assignments fixed.

Definition 3.11. *For some subset of agents $C \subseteq N$ such that $i \notin C$, agent i is dependent on agent j given a resource assignment X_C if there exists some resource assignment $X_{-(C\cup\{j\})}$ and resource assignments X_j, X'_j such that $u_i(X_C, X_{-(C\cup\{j\})}, X_j) \neq u_i(X_C, X_{-(C\cup\{j\})}, X'_j)$.*

Intuitively, if agents i and j share a coalition but i is independent of j under the current resource assignment of the coalition, then it makes no difference to i whether j is in the coalition or not. This is also true of sets of agents; if an agent i is (locally) independent of all agents in a set S, then i is also (locally) independent of their collective choice X_S. Similarly, we can show that if i is independent of sets of agents T, S, they are also independent of $T \cup S$.

Suppose we have a coalition structure in the p-core containing $\lambda = (C \cup S, X_C, X_S)$ where C are locally independent of S given X_C and vice versa. Then, if we split agents C and S into two separate coalitions with the same resource assignments, we know this coalition structure will also be in the p-core.

Proposition 3.12. *Given σ in the p-core containing λ, the coalition structure σ' where λ has been substituted for coalitions $\lambda_1 = (C, X_C), \lambda_2 = (S, X_S)$ such that $\lambda = \lambda_1 \sqcup \lambda_2$, but with all other coalitions identical, will also be in the p-core if all agents in C are independent of all agents in S given resource assignment X_C and all agents in S are independent of all agents in C given resource assignment X_S.*

Proof. Suppose all agents in C are independent of all agents in S given resource assignment X_C and vice versa. This means for any agent in C, there are no assignments X_S, X'_S, $X_{-(C\cup S)}$ such that $u_i(X_C, X_{-(C\cup S)}, X_S) \neq u_i(X_C, X_{-(C\cup S)}, X'_S)$; no matter the assignment of agents outside C and S, there is no way for S to affect the utility of agents in C when X_C is being played. Therefore $u_i^{\downarrow}(\lambda_1) = u_i^{\downarrow}(\lambda)$ for all $i \in C$. The same is true symmetrically of agents in S. Agents outside of of C or S have the same coalition in both σ and σ' so have unchanged utility. □

Whilst we have given dependency in terms of actions from agents, we could have equivalently viewed this as dependencies on *resources themselves*, since resources act identically even when assigned by different agents. We could then look at which agents hold the resources that some agent depends on.

4 ENCODING OTHER GAMES

In this section, we show that RTGs are sufficiently expressive to encode TU games in a way that preserves the core. See Chapter 17 of [25] for an overview of TU games. We show that, for each TU game G, we can construct an RTG Γ such that there is a correspondence between the core of G and the p-core of Γ.

TU games are specified with a tuple (N, v), where $v : \mathcal{P}(N) \to \mathbb{R}$ assigns each coalition a value. An *outcome* of a TU game is a tuple $(CS, \vec{u})$ where CS is a partition of N representing which coalitions form and $\vec{u}$ is an n-dimensional vector where each index u^i is the payoff to agent i. An outcome has the restriction that for each $C \in CS$, the sum of payoffs to members of C cannot be any more than $v(C)$; so, $\sum_{i\in C} u^i \leq v(C)$.

The *core* of a TU game is the set of outcomes such that no subset of agents could form a coalition in which they all earn strictly higher utility. If we denote the space of outcomes for a game G as Σ_G, then:

$$CORE(G) = \left\{ (CS, \vec{u}) \in \Sigma_G \;\middle|\; \forall C \subseteq N. \sum_{i\in C} u^i \geq v(C) \right\}$$

Given an arbitrary TU game $G = (N, v)$, we will construct an RTG Γ that has corresponding core elements. The general idea is to have tasks and resource assignments that correspond to moves in the original game. TU games essentially have two decisions for each agent: which coalition they wish to join, and what share of the value of the coalition they wish to obtain. So, we will have a task t_C for each coalition C where assigning a resource to this task corresponds to making a commitment to join the coalition C in the original game, and a task t_i for each agent that will allow that agent i to request a share of the coalition's value based on the quantity of resource assigned to it. For simplicity, we will only work with games where $v(C) > 0$ for any C, although it is not hard to extend to more general TU games. More explicitly, given $G = (N, v)$ our constructed game is:

$$\Gamma_G = (N, R, \mathbf{type}, (t_C)_{C\subseteq N}, (t_i)_{i\in N}, \mathbf{en}, (v_i)_{i\in N})$$

where:

- N is the set of agents in G
- $R = \{(r_{join_i})_{i\in C}, r_{value}\}$
- For each r_{join_i}, $\mathbf{type}(r_{join_i})$ = discrete, and $\mathbf{type}(r_{value})$ = continuous.
- A task t_C for each $C \subseteq N$, such that
$$t_C(X_N) = \begin{cases} 1 \text{ if } \sum_{i\in C} x^i_{C,join_i} = |C| \\ 0 \text{ otherwise} \end{cases}$$
and a task t_i for each $i \in N$ such that $t_i(X_N) = \sum_{j\in N} x^j_{i,value}$ - so t_i just returns the amount of r_{value} allocated to it.
- For each $i, j \in N$ where $i \neq j$, $\mathbf{en}(i, r_{join_i}) = 1$, $\mathbf{en}(i, r_{join_j}) = 0$, and $\mathbf{en}(i, r_{value}) = max_{C\subseteq N}(v(C))$
- a value function for each $i \in N$ such that
$$v_i((t_C)_{C\subseteq N}, (t_i)_{i\in N}) = \sum_{\{C\subseteq N \mid i\in C\}} \left(t_C \cdot v(C) \cdot \frac{t_i}{\sum_{j\in C} t_j} + 1 \right)$$
where $v(C)$ is the value of coalition C in G. Here we use each t_C and t_i as a placeholder for the output of the corresponding task. To make this well defined, we will say v_i returns 0 whenever some $t_C = 1$ but $t_j = 0$ for all $j \in C$.

The coalition tasks t_C for each $C \subseteq N$ encodes which coalitions are formed in the outcome of a game. Each agent i has precisely 1 unit of r_{join_i} which they can use to join a coalition: each t_C returns 1 only if all agents i in C allocate their r_{join_i} resource to it. Under a resource assignment X_N, we interpret an output of 1 from $t_C(X_N)$ to mean the coalition C has formed. Each v_i is a sum over each coalition i can be a part of, and the t_C term guarantees we only take the summand for a coalition that has formed. For such a coalition, each agent gets a share of $v(C)$ determined on the amounts of r_{value} they have allocated to each of their t_i tasks. For example, if they have all allocated the same amount, they get an equal split of $v(C)$. Forming any coalition also grants each agent +1 utility, so that agents are always incentivised to form some coalition even when they do not receive any share of $v(C)$.

We only wish to look at outcomes of this constructed game Γ_G that correspond to the original transferable utility game G. Therefore, we will define the notion of a *well-formed* solution of Γ_G. This is a coalition structure σ with collective resource allocation X_N such that:

- All agents are properly allocated to a coalition; for each $i \in N$, there is exactly one $C \subseteq N$ such that $i \in C$ and $t_C(X_N) = 1$.
- For any C where $t_C(X_N) = 1$, for each $i \in C$ at least one of the tasks t_i where agents request their share of the coalition's value is non-zero ($\sum_{i \in C} t_i(X_N) \neq 0$).
- The coalitions in σ match those encoded in the tasks: for each C where $t_C(X_N) = 1$, for any two agents $i, j \in C$, if for a coalition $\lambda \in \sigma$ it is the case that $i \in \lambda$ then it is also the case that $j \in \lambda$.

We can recover outcomes of the TU game G from well-formed outcomes of the game Γ_G, and vice versa.

Given an outcome of G in the form (CS, u), we get a corresponding grand coalition structure in Γ_G with $\{(N, X_{(CS,u)})\}$ where in $X_{(CS,u)}$, for each $C \in CS$ and $i \in C$, $x^i_{C,join_i} = 1$ (i's endowment of r_{join_i} is assigned to t_C) and each agent allocates u^i (their payoff in G) of their r_{value} resource to task t_i. Given a well-formed outcome of Γ_G in the form $\{\lambda_1, \ldots, \lambda_k\}$, we first collect together the resource assignments X_i from each agent across the coalitions to get a collective resource assignment X_N. We can then construct a TU game outcome (CS, u) by:

- adding C to CS for any $C \subseteq N$ such that $t_C(X_N) = 1$
- iterating through each agent i and setting their payoff u^i to $\frac{t_i(X_N) \cdot v(C)}{\sum_{j \in C} t_j(X_N)}$, where C is the coalition corresponding to the task t_C to which i has allocated their unit of r_{join_i}. So the agent's TU payoff is a proportion of the coalition value $v(C)$ based on the share of resource r_{value} allocated between all members of their chosen t_C task.

Remark 4.1. *Mapping between TU outcomes and well-formed outcomes in Γ_G is strictly order-preserving w.r.t. agent utility.*

Since each agent in a coalition receives a normalised share of utility such that the sum is $v(C)$, we can only recover Pareto-optimal outcomes of G - i.e. an outcome $(CS, \vec{u})$ where for all $C \in CS$, $\sum_{i \in C} u^i = v(C)$. However, this is not really a problem, as the core of G can only contain Pareto-optimal outcomes, and any blocked coalition structure will always have at least one Pareto-optimal coalition blocking it.

A well-formed coalition is one that appears in some well-formed coalition structure. A non-well-formed coalition is one that doesn't appear in any well-formed coalition structure. By the definition of a non-well-formed coalition, at least one agent receives 0 (the minimum possible) pessimistic utility. Therefore:

Lemma 4.2. *A non-well-formed coalition cannot block any coalition structure.*

In a well-formed coalition with a single agent, that agent by definition receives strictly greater than 0 utility, so:

Lemma 4.3. *Every non-well-formed coalition structure σ is always blocked by some coalition*

We will now show that, for every coalition structure in the core of G, there is a corresponding well-formed grand coalition structure in the p-core of Γ_G and vice versa.

Proposition 4.4. $(CS, u) \in \text{CORE}(G) \iff \{(N, X_{(CS,u)})\} \in \text{PCORE}(\Gamma_G)$

Proof. ($\implies$) Take a (CS, u) in the core of G. Suppose the corresponding coalition structure $\sigma = \{(N, X_{(CS,u)})\}$ (as detailed above) was blocked. It must be blocked by a well-formed coalition by lemma 4.2. Take any well-formed coalition structure that contains this coalition. In this coalition, some subset of agents C all receive strictly higher utility than in σ. Since it is well-formed, we can retrieve a corresponding outcome in the TU game (CS', u') where, due to remark 4.1, the agents in C are receiving strictly higher utility than in our original outcome (CS, u). By the definition of well-formed outcomes, this subset of agents C form a coalition $S \in CS'$: therefore, (CS', u') would block (CS, u). Since we know (CS, u) is in the core, our assumption was incorrect and σ is in the p-core.
($\impliedby$) Take a coalition structure $\sigma = \{(N, X)\}$ in the p-core of Γ_G. By Lemma 4.3 then $\{(N, X)\}$ must be well-formed and so corresponds to some outcome (CS, u) in G. Suppose this is blocked by some outcome (CS', u') where for some coalition $C \in CS'$, all agents in C are receiving a strictly higher payoff. This corresponds to a well-formed coalition structure $\{(N, X'_N)\}$. If we restrict this to just the agents in C we get a coalition $\lambda = (C, X'_C)$ that, due to 4.1, blocks σ. This contradicts the fact that σ is in the core, so (CS, u) must be in the core of G. □

As a consequence of the closure properties described in section 3.1, the p-core of Γ_G also contains the coalition structures where we split coalitions of Γ_G based on the coalitions of G they are assigned to via the t_C tasks - these are still well-formed and correspond to exactly the same coalition structures in G as their grand coalition counterparts, however.

5 COMPLEXITY RESULTS

In this section we use notions from complexity theory relating to the polynomial hierarchy, an overview of which can be found in [4]. For the following problems we assume the RTGs have a finite number of possible coalition structures[3] and that all task, value, and cost functions are computable in polynomial time. We will look at three problems relating to the p-core:

WORST-CASE RESOURCE ASSIGNMENT
Given RTG Γ, coalition $\lambda = (C, X_C)$, resource assignment X_{-C}, agent $i \in C$

[3]so a finite number of feasible resource assignments

Question Check whether X_{-C} gives agent i's pessimistic utility for λ (whether $u_i(X_C, X_{-C}) = u_i^{\downarrow}(\lambda)$)

PCORE MEMBERSHIP
Given RTG Γ, coalition structure $\sigma \in \Sigma_\Gamma$
Question Check whether $\sigma \in PCORE(\Gamma)$

PCORE NON-EMPTINESS
Given RTG Γ
Question Check whether $PCORE(\Gamma) \neq \emptyset$

Lemma 5.1. *WORST-CASE RESOURCE ASSIGNMENT is in* co-np

Proof. Take the complement problem, of verifying whether the utility for a particular resource assignment X_{-C} is not equal to $u_i^{\downarrow}(\lambda)$. We could non-deterministically 'guess' another resource assignment X'_{-C}, and in polynomial time calculate whether $u_i(X_C, X_{-C}) > u_i(X_C, X'_{-C})$. If so, $u_i(X_C, X_{-C}) \neq u_i^{\downarrow}(\lambda)$ and X'_{-C} is our certificate of this, so the complement problem is in np. □

Proposition 5.2. *PCORE MEMBERSHIP is* Π_2^p*-complete*

Proof. The class Π_2^p can also be seen as co-np with an np (or co-np) oracle. Supposing we had such an oracle, we will show that the complement of PCORE MEMBERSHIP (checking whether a given coalition structure σ is blocked by some coalition λ), would be in np. For each agent i we will denote the coalition they appear in σ as $\lambda_{i,\sigma}$. We will non-deterministically guess a potential blocking coalition λ. We must use our oracle to be able to calculate pessimistic utilities in polynomial time, so we will also guess a resource assignment X^i_{-C} for each agent $i \in C$ to be the worst-case resource assignment from the perspective of i in λ, and for each $i \in C$ where $\lambda_{i,\sigma} = (S, X_S)$ for some S we take some X^i_{-S} as our guess for the worst-case resource assignment for i in $\lambda_{i,\sigma}$. We can use our oracle to check WORST-CASE RESOURCE ASSIGNMENT for each $i \in C$ with X^i_{-C} and λ, and with X^i_{-S} and $\lambda_{i,\sigma}$. Assuming our guess of a blocking λ and the worst-case guesses are all correct, we can verify that for all $i \in C$, $u_i^{\downarrow}(\lambda) > u_i^{\downarrow}(\lambda_{i,\sigma})$ in polynomial time. Therefore, our original problem PCORE MEMBERSHIP is in Π_2^p.

For hardness, we will also use the complement problem. We will reduce from the Σ_2^p-complete problem $2QBF_\exists$ [30]; determining the truth of a quantified boolean formula of the form $\exists\vec{x}\forall\vec{y}.\phi(\vec{x},\vec{y})$. We will denote the full set of variables V. Without loss of generality, we can assume the assignment $\phi(\vec{0})$ is false.[4] From this, we can construct an RTG with an 'existential' agent which will control the variables in $\vec{x}$ and a 'universal' agent which will control the variables in $\vec{y}$. Our existential agent will be trying to satisfy the formula ϕ, whereas the universal agent will be trying to ensure it is not satisfied. We will call the existential agent x, the universal agent y, and the set of both agents N. We will have a discrete resource type for each variable, and each agent will be endowed with exactly 1 unit of each resource corresponding to a variable they control. There will be a task t_{v_i} for each variable, that is affected only by the resource type of the corresponding variable; this will correspond to assigning a valuation to the variable in ϕ. If the resource for variable v_i is assigned to t_{v_i}, it will return 1. Else, t_{v_i} returns 0. We then encode ϕ in the value function for each agent. For the existential agent x, their value function will be:

$$v_x((t_{v_i})_{i\in V}) = \begin{cases} 1 \text{ if } \phi \text{ is satisfied under assignment } (t_{v_i})_{i\in V} \\ 0 \text{ otherwise} \end{cases}$$

For the universal agent y, their value function will be:

$$v_y((t_{v_i})_{i\in V}) = \begin{cases} 1 \text{ if } \neg\phi \text{ is satisfied under assignment } (t_{v_i})_{i\in V} \\ 0 \text{ otherwise} \end{cases}$$

Cost functions will always return 0. So, the agents controlling existential variables only gain utility when ϕ is satisfied, and the agents controlling universal variables only gain utility when ϕ is not satisfied.

Take $\sigma = (N, \vec{0})$ — the grand coalition structure where no agents assign any resources — as our given coalition structure. Since ϕ is unsatisfied here by definition, $u_x^{\downarrow}(\sigma) = 0$ and $u_y^{\downarrow}(\sigma) = 1$. If it is true that $\exists\vec{x}\forall\vec{y}.\phi(\vec{x},\vec{y})$, there is some assignment we will call SAT_x to x's variables that satisfies ϕ against all assignments of y's variables. The coalition $\lambda = (\{x\}, SAT_x)$ blocks σ as $u_x^{\downarrow}(\lambda) = 1$. If it is false that $\exists\vec{x}\forall\vec{y}.\phi(\vec{x},\vec{y})$, then $\forall\vec{x}\exists\vec{y}.\neg\phi(\vec{x},\vec{y})$. Since $u_y^{\downarrow}(\sigma) = 1$, there is no coalition containing y that could block σ, as they cannot earn strictly higher utility. If x alone defects, for any assignment to x's variables, there is always an assignment to y's that leaves ϕ unsatisfied, so x can never have strictly higher pessimistic utility. Therefore, σ is blocked iff $\exists\vec{x}\forall\vec{y}.\phi(\vec{x},\vec{y})$. □

Proposition 5.3. *PCORE NON-EMPTINESS is* Π_2^p*-hard and in* Σ_3^p

Proof. Σ_3^p can be viewed as the class of problems in np with a Π_2^p oracle. With non-determinism, we can guess an arbitrary coalition structure σ and use a Π_2^p oracle to decide the problem PCORE MEMBERSHIP on this σ in constant time. If any σ is a p-core member, then we know the p-core is non-empty. Therefore, the problem is in Σ_3^p.

For hardness with respect to Π_2^p, we will provide a reduction from $2QBF_\forall$ [30]. This is the problem of determining the truth of a quantified boolean formula of the form $\forall\vec{y}\exists\vec{x}.\phi(\vec{x},\vec{y})$. We will have a similar construction to the previous problem, with a universal agent y controlling the variables in $\vec{y}$ and seeking to leave ϕ unsatisfied, and an existential agent x controlling $\vec{x}$ and seeking to satisfy ϕ. We will use the same tasks, value, and cost functions from the construction in proposition 5.2.

We will now append 3 extra agents to this game; agents a, b, and c. We will construct some extra task functions and assign resources such that, whenever $\neg\forall\vec{y}\exists\vec{x}.\phi(\vec{x},\vec{y})$, these agents are incentivised to play a hedonic game [9] with no core. In a hedonic game, agent preferences are defined over the coalitions they can be a part of. Our game will be based on the following preference profiles:

$$\{a,b\} \succ_a \{a,c\} \succ_a \{a\} \succ_a \{a,b,c\}$$

$$\{b,c\} \succ_b \{a,b\} \succ_b \{b\} \succ_b \{a,b,c\}$$

$$\{a,c\} \succ_c \{b,c\} \succ_c \{c\} \succ_c \{a,b,c\}$$

[4]to see this, note that given an instance of $2QBF_\exists$, we can replace ϕ with $\phi \wedge x_{n+1}$, where x_{n+1} is a new existential variable. This preserves the truth/falsity of the formula $\exists\vec{x}\forall\vec{y}.\phi(\vec{x},\vec{y})$

It can be seen that every coalition structure in this game is blocked. We can use a similar construction as in section 4 to represent this, having a r_{join_i} resource which agents a, b, and c are endowed with precisely 1 unit of, and a task t_C for each possible coalition. A task t_C will return 1 if all agents in C have allocated their r_{join_i} resource to it, and 0 otherwise. We will associate a function f_i defined over the outputs of tasks $(t_C)_{C\subseteq\{a,b,c\}}$ with each of these 3 agents to encode their preference profiles - $f_i((t_C)_{C\subseteq\{a,b,c\}})$ will return 0.4 if the output of t_C is 1 for the top C in $\succ_i$, 0.3 for the second C, 0.2 for the third, and 0.1 for the bottom C. If no $t_C = 1$, then it will return -1.

So, for example, if $t_{\{a,c\}} = 1$ then $f_a((t_C)_{C\subseteq\{a,b,c\}}) = 0.3$. We can now give a value function for these agents $i \in \{a,b,c\}$ in our game:

$$v_i((t_{v_i})_{i\in V}, (t_C)_{C\subseteq\{a,b,c\}}) = \begin{cases} 1 & \text{if } \phi \text{ is satisfied under} \\ & \text{assignment } (t_{v_i})_{i\in V} \\ f_i((t_C)_{C\subseteq\{a,b,c\}}) & \text{otherwise} \end{cases}$$

So, if ϕ is satisfied they get the same utility as the existential agent x. If not, they get the utilities from the hedonic game.

If $\forall\vec{y}\exists\vec{x}.\phi(\vec{x},\vec{y})$ then for any valuation of $\vec{y}$, x has some valuation of $\vec{x}$ that satisfies ϕ. In a grand coalition structure where ϕ is satisfied, x and the hedonic agents are receiving the maximum possible utility, so have no incentive to deviate. If y deviates with move $\vec{y}$, they assume that the corresponding valuation of $\vec{x}$ that satisfies ϕ is played so they would receive 0 utility. Therefore, these grand coalition structures are not blocked and the p-core is nonempty.

For the other direction, consider when $\forall\vec{y}\exists\vec{x}.\phi(\vec{x},\vec{y})$ is false: i.e. $\exists\vec{y}\forall\vec{x}.\neg\phi(\vec{x},\vec{y})$. Take a grand coalition structure where ϕ is satisfied; y has a valuation that always leaves ϕ unsatisfied, so they gain strictly higher utility by defecting and playing this, blocking this coalition structure. Now consider a grand coalition structure where ϕ is unsatisfied; agents a, b, and c are now receiving the utilities from their hedonic game. Since this game has no core, then no matter the current resource assignment there will be at least one of a, b, or c that can gain utility by playing the assignment that corresponds to the blocking structure in the hedonic game. Therefore, all grand coalition structures are blocked, which is sufficient for checking core emptiness, and $\forall\vec{x}\exists\vec{y}.\phi(\vec{x},\vec{y})$ if and only if the p-core is nonempty. □

Currently, the higher and lower bounds for complexity of PCORE NON-EMPTINESS have not been matched. The same is true of the analogous problem for CRGs, where in [18] it was shown to be NP-HARD. Since CRGs are a special case of RTGs, we can import our upper bound of Σ_3^p.

6 RELATED WORK

Many approaches to encoding cooperative games are similar to RTGs and CRGs, where we specify the strategic power of individual agents and induce the coalitional power from this. In Cooperative Boolean Games [17] — a cooperative extension to [22] — each agent is in control of some boolean variables, and preferences are encoded by a boolean formula. Łukasiewicz Resource Games are introduced in [24] as a generalisation of CRGs that allow for games to be compactly represented with formulae in a many-valued Łukasiewicz logic.

A similar but less compact model is that of Qualitative Coalitional Games [34], where a set of outcomes is specified for each coalition. Each outcome is a set of qualitative goals, and an agent is satisfied whenever any of their desired goals is achieved - but since the sets of goals are arbitrarily chosen, we can have more general spaces of outcomes. There are several extensions to these, adding concepts such as temporality [28] or agent preferences over goals [19].

In a similar spirit to CRGs are Coalitional Skill Games [6–8], where each agent has a set of skills and can join together to complete tasks with requisite skills; completing a set of tasks rewards the coalition with some amount of utility which can be shared between them arbitrarily. Of particular interest are Hedonic Skill Games [21], which prescribe a share of utility based on how many tasks each agent contributed their skills to in a coalition - prohibiting the 'freeloading' behaviour seen in Example 2.3, where agents gain utility while contributing nothing.

Much work on representations of games is in the Transferable Utility space. One approach is to have a combinatorial structure, from which we can calculate the worth of a given coalition as the solution to a combinatorial optimisation problem. See [16] for a survey focusing on calculating solutions to various combinatorial optimization games. A more general (but not always succinct) approach would be the MC-nets of [23].

A similar property related to the ability to split the core of RTGs into dependency-closed subsets was investigated in [10] in regards to the Nash Equilibria of Boolean Games, and extended to the core of Cooperative Boolean Games in [29]. We use different notions of dependence and core, however.

It is also worth mentioning quantitative extensions to games with qualitative goals such as [11, 12], and some models of 2-player games with resource bounds and quantitative goals, such as [14, 20].

7 CONCLUSION

We have presented Resource Task Games, a new class of games that extend Coalitional Resource Games with tasks with arbitrary states of completion, and arbitrary preferences over these states of completion. We have shown the core of these games to be closed under the union of coalitions, as well as the splitting apart of coalitions with mutually independent agents. We demonstrated that RTGs are able to encode Transferable Utility games, with a construction that preserves the core. We have also given computational complexity bounds for checking if a resource assignment is the worst-case according to a given agent, checking if a specific coalition structure is in the core, and checking existence of the core; each problem is harder than the analogous problem for CRGs, since to calculate pessimistic utility in CRGs it is sufficient to assume all agents outside the coalition are allocating no resources.

It would be interesting to investigate further the classes of games that can be represented in RTGs; the approach used for TU games may apply more generally to NTU games. Additionally, there are many further decision problems that would be natural to investigate, especially focusing on the resource limitation aspect of RTGs. A parameterised complexity approach, as used in [31] w.r.t. to CRGs, may also show which aspects make these decision problems so hard. Similarly, problems may become more tractable when we restrict the number of different types of agent, as in [32].

REFERENCES

[1] Natasha Alechina, Brian Logan, Nguyen Hoang Nga, and Abdur Rakib. 2009. Expressing Properties of Coalitional Ability under Resource Bounds. In *Logic, Rationality, and Interaction*, Xiangdong He, John Horty, and Eric Pacuit (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 1–14.

[2] Natasha Alechina, Brian Logan, Hoang Nga Nguyen, and Abdur Rakib. 2011. Logic for coalitions with bounded resources. *Journal of Logic and Computation* 21, 6 (2011), 907–937. https://doi.org/10.1093/logcom/exq032

[3] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. 2002. Alternating-time temporal logic. *J. ACM* 49, 5 (sep 2002), 672–713. https://doi.org/10.1145/585265.585270

[4] Sanjeev Arora and Boaz Barak. 2009. *The polynomial hierarchy and alternations.* Cambridge University Press, 95–105.

[5] Robert J. Aumann. 1961. The core of a cooperative game without side payments. *Trans. Amer. Math. Soc.* 98, 3 (March 1961), 539–539. https://doi.org/10.1090/s0002-9947-1961-0127437-2

[6] Yoram Bachrach, Reshef Meir, Kyomin Jung, and Pushmeet Kohli. 2010. Coalitional Structure Generation in Skill Games. *Proceedings of the AAAI Conference on Artificial Intelligence* 24, 1 (Jul. 2010), 703–708. https://doi.org/10.1609/aaai.v24i1.7620

[7] Yoram Bachrach, David C. Parkes, and Jeffrey S. Rosenschein. 2013. Computing cooperative solution concepts in coalitional skill games. *Artificial Intelligence* 204 (2013), 1–21. https://doi.org/10.1016/j.artint.2013.07.005

[8] Yoram Bachrach and Jeffrey S. Rosenschein. 2008. Coalitional skill games. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2* (Estoril, Portugal) *(AAMAS '08)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1023–1030.

[9] Anna Bogomolnaia and Matthew O. Jackson. 2002. The Stability of Hedonic Coalition Structures. *Games and Economic Behavior* 38, 2 (2002), 201–230. https://doi.org/10.1006/game.2001.0877

[10] Elise Bonzon, Marie-Christine Lagasquie-Schiex, and Jérôme Lang. 2009. Dependencies between players in Boolean games. *International Journal of Approximate Reasoning* 50, 6 (2009), 899–914. https://doi.org/10.1016/j.ijar.2009.02.008 Ninth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2007.

[11] Patricia Bouyer, Romain Brenguier, Nicolas Markey, and Michael Ummels. 2012. Concurrent Games with Ordered Objectives. In *Foundations of Software Science and Computational Structures*, Lars Birkedal (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 301–315.

[12] Nils Bulling and Valentin Goranko. 2021. Combining quantitative and qualitative reasoning in concurrent multi-player games. *Autonomous Agents and Multi-Agent Systems* 36, 1 (Nov. 2021). https://doi.org/10.1007/s10458-021-09531-9

[13] Georgios Chalkiadakis, Edith Elkind, and Michael Wooldridge. 2012. *Computational Aspects of Cooperative Game Theory.* Springer International Publishing. https://doi.org/10.1007/978-3-031-01558-8

[14] Krishnendu Chatterjee and Laurent Doyen. 2012. Energy parity games. *Theoretical Computer Science* 458 (2012), 49–60. https://doi.org/10.1016/j.tcs.2012.07.038

[15] Dario Della Monica, Margherita Napoli, and Mimmo Parente. 2011. On a Logic for Coalitional Games with Priced-Resource Agents. *Electronic Notes in Theoretical Computer Science* 278 (2011), 215–228. https://doi.org/10.1016/j.entcs.2011.10.017 Proceedings of the 7th Workshop on Methods for Modalities (M4M'2011) and the 4th Workshop on Logical Aspects of Multi-Agent Systems (LAMAS'2011).

[16] Xiaotie Deng and Qizhi Fang. 2008. *Algorithmic Cooperative Game Theory.* Springer New York, New York, NY, 159–185. https://doi.org/10.1007/978-0-387-77247-9_7

[17] Paul Dunne, Wiebe Hoek, Sarit Kraus, and Michael Wooldridge. 2008. Cooperative Boolean Games. 1015–1022.

[18] Paul E. Dunne, Sarit Kraus, Efrat Manisterski, and Michael Wooldridge. 2010. Solving coalitional resource games. *Artificial Intelligence* 174, 1 (2010), 20–50. https://doi.org/10.1016/j.artint.2009.09.005

[19] Paul E. Dunne and Michael Wooldridge. 2004. Preferences in Qualitative Coalitional Games. In *3rd Conference on Autonomous Agents and Multi-Agent Systems.*

[20] A. Ehrenfeucht and J. Mycielski. 1979. Positional strategies for mean payoff games. *International Journal of Game Theory* 8, 2 (June 1979), 109–113. https://doi.org/10.1007/bf01768705

[21] Laurent Gourvès and Gianpiero Monaco. 2024. Nash Stability in Hedonic Skill Games. In *the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024).* Auckland, New Zealand. https://hal.science/hal-04626037

[22] Paul Harrenstein, Wiebe Hoek, John-jules Meyer, and Cees Witteveen. 2001. Boolean Games. *Proceeding of the Eighth Conference on Theoretical Aspects of Rationality and Knowledge (TARK VIII)* (07 2001).

[23] Samuel Ieong and Yoav Shoham. 2005. Marginal contribution nets: a compact representation scheme for coalitional games. In *Proceedings of the 6th ACM Conference on Electronic Commerce* (Vancouver, BC, Canada) *(EC '05)*. Association for Computing Machinery, New York, NY, USA, 193–202. https://doi.org/10.1145/1064009.1064030

[24] Enrico Marchioni and Michael Wooldridge. 2019. Łukasiewicz logics for cooperative games. *Artificial Intelligence* 275 (2019), 252–278. https://doi.org/10.1016/j.artint.2019.03.003

[25] Michael Maschler, Eilon Solan, and Shmuel Zamir. 2020. *Game Theory* (2 ed.). Cambridge University Press.

[26] Hoang Nga Nguyen, Natasha Alechina, Brian Logan, and Abdur Rakib. 2015. Alternating-time temporal logic with resource bounds. *Journal of Logic and Computation* 28, 4 (06 2015), 631–663. https://doi.org/10.1093/logcom/exv034 arXiv:https://academic.oup.com/logcom/article-pdf/28/4/631/25020788/exv034.pdf

[27] Marc Pauly. 2002. A Modal Logic for Coalitional Power in Games. *Journal of Logic and Computation* 12, 1 (02 2002), 149–166. https://doi.org/10.1093/logcom/12.1.149 arXiv:https://academic.oup.com/logcom/article-pdf/12/1/149/3657514/120149.pdf

[28] Thomas Ågotnes, Wiebe van der Hoek, and Michael Wooldridge. 2006. Temporal qualitative coalitional games. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems* (Hakodate, Japan) *(AAMAS '06)*. Association for Computing Machinery, New York, NY, USA, 177–184. https://doi.org/10.1145/1160633.1160662

[29] Luigi Sauro and Serena Villata. 2011. Dependency in Cooperative Boolean Games. *Journal of Logic and Computation* 23, 2 (09 2011), 425–444. https://doi.org/10.1093/logcom/exr030 arXiv:https://academic.oup.com/logcom/article-pdf/23/2/425/4030630/exr030.pdf

[30] Marcus Schaefer and Christopher Umans. 2002. SIGACT news complexity theory column 37. *SIGACT News* 33, 3 (Sept. 2002), 32–49. https://doi.org/10.1145/582475.582484 Updated version: https://ovid.cs.depaul.edu/documents/phcom.pdf.

[31] Tammar Shrot, Yonatan Aumann, and Sarit Kraus. 2009. Easy and hard coalition resource game formation problems - A parameterized complexity analysis. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS* 1, 433–440. https://doi.org/10.1145/1558013.1558072

[32] Tammar Shrot, Yonatan Aumann, and Sarit Kraus. 2010. On agent types in coalition formation problems. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1 - Volume 1* (Toronto, Canada) *(AAMAS '10)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 757–764.

[33] Nicolas Troquard. 2018. Rich Coalitional Resource Games. *Proceedings of the AAAI Conference on Artificial Intelligence* 32, 1 (Apr. 2018). https://doi.org/10.1609/aaai.v32i1.11437

[34] Michael Wooldridge and Paul E Dunne. 2004. On the computational complexity of qualitative coalitional games. *Artificial Intelligence* 158, 1 (2004), 27–73. https://doi.org/10.1016/j.artint.2004.04.002

[35] Michael Wooldridge and Paul E. Dunne. 2006. On the computational complexity of coalitional resource games. *Artificial Intelligence* 170, 10 (2006), 835–871. https://doi.org/10.1016/j.artint.2006.03.003