Equivalence results in the allocation of indivisible objects: A unified view

Thiam Lee[*]and Jay Sethuraman[†]

August 2011

Abstract

This paper studies probabilistic mechanisms that allocate indivisible objects to agents by hierarchical exchange using the top-trading cycles algorithm. The main result of this paper is a general technique for proving that seemingly different probabilistic mechanisms are in fact equivalent. This approach simplifies and unifies several equivalence results in the literature. The same technique is used to generalize these results to mechanisms in which the priority structure for each object is given by a tree (instead of a linear ordering of the agents).

1 Introduction

The problem of allocating a finite number of indivisible objects to a set of agents has been studied extensively, since the pioneering work of Shapley and Scarf [15], and has served as a useful model in many real-world settings such as the assignment of schools to students [1] and the design of kidney exchanges [12]. The early literature was mostly on two distinct models: the *housing market* model of Shapley and Scarf in which each agent was endowed with an object; and the *house allocation* model, studied by Hylland and Zeckhauser [7] in which there were no endowments. There is by now a well-developed literature on each of these models as well as on a hybrid version, first proposed by Abdulkadiroğlu and Sönmez [2], in which some—but not all—agents are endowed with objects. We recommend the recent surveys of Sönmez and Ünver [17] for an overview of the literature on allocating indivisible objects, and of Pathak [10] for applications of these ideas to student placement.

[*]IEOR Department, Columbia University, New York, NY; `thl2102@columbia.edu`

[†]IEOR Department, Columbia University, New York, NY; `jay@ieor.columbia.edu`. Research supported by NSF grant CMMI-0916453.

In this paper our focus is exclusively on ordinal settings in which agents submit strict preference orderings over the objects. Shapley and Scarf formulated their housing market model in this setting, and described the top-trading cycle (TTC) algorithm (attributed to Gale) that computes a unique reallocation of the objects to the agents. They further showed that this allocation is in the core of the natural cooperative game associated with this problem, and also that it can be sustained as a competitive equilibrium. Roth and Postlewaite [14] later established the uniqueness of the core, and Roth [13] showed that the core mechanism is strategyproof (submitting true preference orderings is a dominant strategy for each agent). Bird [3] generalized this result and showed that the core is in fact group strategyproof (submitting true preference orderings is a dominant strategy for any group of agents).

For the house allocation problem, a fundamental mechanism in this setting is the *Random Priority* (RP) mechanism (see Zhou [19], and Abdulkadiroğlu and Sönmez): a random ordering of the agents is drawn, and the agents are invited to choose objects in this order. The resulting allocation (for any given ordering) is Pareto efficient, so the outcome of the mechanism can be thought of as a lottery over Pareto efficient assignments. It is easy to see that RP is strategyproof and treats equals equally: agents with identical preference orderings receive identical (probabilistic) allocations. The literature on these problems evolved fairly independently until Abdulkadiroğlu and Sönmez thought of the following mechanism for the house allocation problem: endow each agent with a random object, each possible endowment of the objects to the agents equally likely; and find the unique reallocation given by the TTC algorithm. They called this the *random endowment* (RE) mechanism. A natural question is to understand the relationship between RP and RE. Somewhat surprisingly, Abdulkadiroğlu and Sönmez showed that these mechanisms are equivalent: given any preference structure for the agents, RP and RE lead to the same probability distribution over Pareto efficient assignments. Since their result, there have been a number of papers that establish the equivalence of seemingly different mechanisms for a variety of models assignment models: For example, Pathak and Sethuraman [11] show that in assigning students to schools (using the TTC mechanism), the mechanism in which all the schools use the same (random) priority ordering of the agents is equivalent to the mechanism in which every school generates its priority ordering randomly and independently.

Our paper is inspired by a recent paper of Carroll [5] that puts forth a general framework that subsumes many (but not all) of the known equivalence results in the literature. Specifically, Carroll introduces the notion of a priority framework in which the priority orderings of the objects are specified in terms of "roles," which are placeholders for the agents. The priority framework is instantiated by picking a bijection from the set of roles to the set of agents, resulting in a priority ordering of the agents for each object. Carroll showed that if the bijection from roles to agents is chosen uniformly at random, and if the TTC algorithm is used to compute the allocation, then the outcome—a probability distribution over matchings—is independent of the priority framework! The equivalence of RP and RE follows because each of them can be modeled by a fixed priority framework. Carroll also introduced a more general priority structure for the objects by using an exogeneous partition of the agents into groups. There is a fixed priority structure for the objects in terms of the given groups, but each group is free to order its roles any way it likes across the various objects. As before, the roles of each group are instantiated by a random bijection to the agents in that group, and this is done independently for each group. Carroll showed that the final outcome is the same regardless of how each group orders its roles across the various objects. In proving his result, Carroll observed that the straightforward counting approach of Pathak and Sethuraman does not easily extend to this model. Nevertheless Carroll used a combination of techniques and gave essentially a bijective technique to prove his main result. In another recent paper, Ekici [6] proved a new equivalence result in the hybrid model where some (but not all) agents are endowed with objects. For this model, Ekici showed the equivalence of a natural random priority mechanism, first proposed by Abdulkadiroğlu and Sönmez, to a variant of the random endowment mechanism in which some agents may be endowed with multiple objects and others none; agents of the latter sort are nonetheless granted an "inheritance" right. We defer the details of this mechanism to Section 4.1, but note that this result is neither subsumed by Carroll's general model, nor by a similar looking equivalence result of Pathak and Sethuraman. Ekici's proof is again bijective.

Motivated by these recent equivalence results we describe a general technique to prove the equivalence of two mechanisms (Section 3). Our approach is related to that of Pathak and Sethuraman: like that approach we rely on induction (on the number of agents), and express the outcome of a mechanism in terms of its outcome on smaller instances. Given two mechanisms, we focus on those terms that appear in one but not the other, and argue that the overall contribution of such terms is zero. The key difference is in the way this last fact is established. Pathak and Sethuraman used a counting argument to do this, whereas we replace this counting argument with a simple bijection. The advantage is that one can now apply this argument more broadly. In particular, the results of Carroll, Ekici, and generalizations of these results can all be established easily using our general technique, see Section 4. In Section 5 we show how the same technique can be used to prove equivalence results when the priority structures form an inheritance tree in the sense of Papai [9]. Our main contribution therefore is a unified approach that sheds light on all the equivalence results in the literature, in addition to suggesting new ones.

2 TTC Mechanisms

Let A denote the set of agents and S the set of objects, with $|A| = |S|$. Each agent has a strict preference ordering over the objects. Each agent wishes to be assigned exactly one object, and each object can be assigned to at most one agent. Our focus shall be on the family of algorithms called *top-trading cycles* (TTC) invented by Gale and first described by Shapley and Scarf [15]. This algorithm operates in phases; At the beginning of each phase, there is a set of (remaining) agents, a set of (remaining) objects, and each object has a top-priority agent (among the ones that remain). Consider the graph with a node for each (remaining) agent, and an arc from node i to node k if agent i's most preferred object is one for which agent k has top priority. Note that self-loops are possible: an arc (i, i) exists if agent i has the highest priority for his most-preferred (remaining) object. This graph, referred to as the TTC graph, must have a cycle c (which may be simply a self-loop), because every node has out-degree 1 and there are finitely many nodes. Every agent in c is matched with the object he most prefers among the ones that remain; the agents in c along with their matched objects are removed from the problem, and the top-priority agent for each remaining object is updated if necessary. We will usually refer to this process as *clearing the cycle c*. If any agent (equivalently object) remains, the next phase starts in which the same algorithm is applied to the remaining objects and agents; Otherwise, all the agents have been matched and the algorithm terminates.

Note that we talk of the TTC *family* of algorithms rather than the TTC algorithm because the final allocation depends on how the top priority agent for each object is specified. Thus the mechanisms we discuss differ in how the agents are prioritized for each object. The following two examples, which are well-known mechanisms in the literature, give an idea of how mechanisms can differ within the framework just described. In each case, the agents are assigned objects using the TTC algorithm, but applied to different priority profiles for the objects.

Example 1. Random Priority (RP). Let $(\sigma_1, \sigma_2, \ldots, \sigma_n)$ be a permutation of the agents chosen uniformly at random, that is, every permutation of the agents is equally likely. Let i^* be the smallest i for which agent σ_i still remains in the problem. Then, the top-priority agent (in any phase) for *every* remaining object is σ_{i^*}. Equivalently, the top-priority agent for every remaining object in phase k is the agent σ_k.

Example 2. Random Endowment (RE). Let $(\sigma_1, \sigma_2, \ldots, \sigma_n)$ be a permutation of the agents chosen uniformly at random, that is, every permutation of the agents is equally likely. For each $j = 1, 2, \ldots, n$, the top-priority agent for the j-th object is σ_j. Note that if the j-th object still remains in the problem, so will agent σ_j, so every remaining object will have a top-priority agent at every point in time.

For both RP and RE, the priority orderings of the objects are determined randomly, so the outcome is a *random* matching, which can be described as a *probability distribution over perfect matchings*. This naturally leads us to the notion of *equivalence* of mechanisms, defined as follows.

Definition 1. Two mechanisms are *equivalent* if for any instance, every perfect matching occurs with the same probability under both mechanisms.

Although RP and RE look different and are described in different terms, they lead to the *same probability distribution over perfect matchings*, as was shown independently by Abdulkadiroğlu and Sönmez [2] and Knuth [8]. Thus, RP and RE are equivalent mechanisms.

3 Equivalence of Mechanisms: A general approach

Recent results have shown equivalences between mechanisms that allow for more complex priority structures [5, 6, 11]. We review in Section 4 two of these results and show how their proofs can be simplified. We unify and generalize these recent results in §5, where we formulate a model in which the priority structure for each object is given by an *inheritance* tree. The proof technique used in all of our proofs is essentially the same, and is discussed explicitly in this section.

Recall that different mechanisms in the TTC family differ only in how the priority structures for the various objects are determined. All the TTC mechanisms we consider satisfy the following *persistence* property:

(i) Once an agent has top priority for an object, he retains it until he is matched (to that object or a different one);

and the following *inheritance* property:

(ii) The top-priority agent for a remaining object at any time may depend only on the set of remaining agents and objects as well as the partial matching guaranteed by the all past cycles formed in the TTC graph.

Note that on a given priority profile for the objects it is possible for the corresponding TTC graph for $\mathbb{M}$ to have multiple cycles; by property (i), however, any cycle that is not cleared will *persist* in the TTC graph until it is cleared. This enables us to write the outcome of mechanism $\mathbb{M}$ as a simple recursion. To do this, we first set up some notation.

Let $\mathbb{M}(A, \pi)$ be the probability that mechanism $\mathbb{M}$ applied to the set of agents A results in the matching π.[1] For any $l \geq 1$, let $\mathcal{C}^l_{\mathbb{M}}$ be the collection of l disjoint top-trading cycles (also

[1]Strictly speaking, we should also specify the set of objects that are "available." However the set of available objects will be clear from the context, justifying the notation.

called a *cycle product* of size l) that could be simultaneously present initially when mechanism $\mathbb{M}$ is used. Each cycle can be written in a canonical way: the smallest-numbered agent in the cycle appears as the first agent in the cycle; for a cycle-product of size l, we arrange the cycles in ascending order of the first agents. Note that this representation is unique. For any cycle product c, let $\nu(c)$ be the induced matching, and $P_{\mathbb{M}}(c)$ be the probability that cycle product c is present in the TTC graph under mechanism $\mathbb{M}$. Finally, for matchings π and π', $\pi' \subseteq \pi$ if π' is a submatching of π (i.e., every matched pair in π' is also a matched pair in π); in that case, $\pi \setminus \pi'$ denotes the matching π restricted to the agents and objects that do not appear in π'. Then:

$$\mathbb{M}(A, \pi) = \sum_{l \geq 1} \sum_{\substack{c \in \mathcal{C}^l_{\mathbb{M}} \\ \nu(c) \subseteq \pi}} (-1)^{l-1} P_{\mathbb{M}}(c)\, \mathbb{M}(A \setminus c, \pi \setminus \nu(c)). \tag{1}$$

Eq. (1) can be justified in a straightforward way using the inclusion-exclusion principle and property (i).[2] Note that the information about the cycles that have been cleared is implicit in the definition of the residual problem.

As priority structures grow increasingly complex so does the behavior of the mechanism, so comparing different TTC mechanisms is not always easy. However, Eq. (1) suggests a simple way to think about the equivalence of two TTC mechanisms. Suppose there is another TTC mechanism $\mathbb{M}'$ satisfying properties (i) and (ii) such that

(iii) every collection of cycles that could be simultaneously present in the TTC graph of $\mathbb{M}$ can also be simultaneously present in the TTC graph of $\mathbb{M}'$. That is, $\mathcal{C}^l_{\mathbb{M}} \subseteq \mathcal{C}^l_{\mathbb{M}'}$ for any l; and

(iv) for any $c \in \bigcup_{l \geq 1} \mathcal{C}^l_{\mathbb{M}}$, $P_{\mathbb{M}}(c) = P_{\mathbb{M}'}(c)$.

Then,

$$\begin{aligned}
&\mathbb{M}'(A, \pi) \\
&\quad = \sum_{l \geq 1} \sum_{\substack{c \in \mathcal{C}^l_{\mathbb{M}'} \\ \nu(c) \subseteq \pi}} (-1)^{l-1} P_{\mathbb{M}'}(c)\, \mathbb{M}'(A \setminus c, \pi \setminus \nu(c)) \\
&\quad = \sum_{l \geq 1} \sum_{\substack{c \in \mathcal{C}^l_{\mathbb{M}'} \cap \mathcal{C}^l_{\mathbb{M}} \\ \nu(c) \subseteq \pi}} (-1)^{l-1} P_{\mathbb{M}'}(c)\, \mathbb{M}'(A \setminus c, \pi \setminus \nu(c)) + \sum_{l \geq 1} \sum_{\substack{c \in \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}} \\ \nu(c) \subseteq \pi}} (-1)^{l-1} P_{\mathbb{M}'}(c)\, \mathbb{M}'(A \setminus c, \pi \setminus \nu(c)) \\
&\quad = \sum_{l \geq 1} \sum_{\substack{c \in \mathcal{C}^l_{\mathbb{M}} \\ \nu(c) \subseteq \pi}} (-1)^{l-1} P_{\mathbb{M}}(c)\, \mathbb{M}'(A \setminus c, \pi \setminus \nu(c)) + \sum_{l \geq 1} \sum_{\substack{c \in \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}} \\ \nu(c) \subseteq \pi}} (-1)^{l-1} P_{\mathbb{M}'}(c)\, \mathbb{M}'(A \setminus c, \pi \setminus \nu(c)),
\end{aligned} \tag{2}$$

[2]We appeal to the inclusion-exclusion principle to avoid double-counting. Properties (i) and (ii) together imply that the outcome of the mechanism does not depend on which subset of cycles of the TTC graph is cleared during a phase of the TTC algorithm.

where the first term of the last equality follows from properties (iii) and (iv) of mechanism $\mathbb{M}'$.

Suppose we are able to show that

$$\sum_{l\geq 1} \sum_{\substack{c\in \mathcal{C}^l_{\mathbb{M}'}\setminus \mathcal{C}^l_{\mathbb{M}} \\ \nu(c)\subseteq \pi}} (-1)^{l-1} P_{\mathbb{M}'}(c)\ \mathbb{M}'(A\setminus c, \pi \setminus \nu(c)) \;=\; 0. \tag{3}$$

Then expression (2) simplifies to

$$\mathbb{M}'(A,\pi) = \sum_{l\geq 1} \sum_{\substack{c\in \mathcal{C}^l_{\mathbb{M}} \\ \nu(c)\subseteq \pi}} (-1)^{l-1} P_{\mathbb{M}}(c)\ \mathbb{M}'(A\setminus c, \pi \setminus \nu(c)).$$

Notice that this is exactly the recursion defining $\mathbb{M}(A,\pi)$, assuming $\mathbb{M}(A',\pi') \;=\; \mathbb{M}'(A',\pi')$ for all $A' \subsetneq A$ and all matchings π' involving the agents in A'. This sets the stage for an inductive proof.[3]

We shall therefore focus on proving (3), which we do using the following general idea. For any $c \in \bigcup_{l\geq 1} \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}}$ pick two special agents x_1 and x_2 from c and write the cycles in c such that x_1 appears at the beginning of its cycle, x_2 appears at the beginning of its cycle if it is in a different cycle than x_1, and all other cycles not containing x_1 or x_2 are written in the canonical fashion[4]:

if x_1 and x_2 appear in the same cycle, say $(x_1a_0a_1\dots a_kx_2b_0b_1\dots b_l)$, then split the cycle into two cycles $(x_1a_0a_1\dots a_k)$ and $(x_2b_0b_1\dots b_l)$; if x_1 and x_2 appear in distinct cycles, say $(x_1a_0a_1\dots a_k)$ and $(x_2b_0b_1\dots b_l)$, then merge the two cycles as $(x_1a_0a_1\dots a_kx_2b_0b_1\dots b_l)$. We call this the *transformation* T applied to the cycle product c. Suppose that for any cycle product $c \in \bigcup_{l\geq 1} \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}}$, there is a method of picking the two agents x_1 and x_2 such that the following properties hold:

(a) The choice of x_1 and x_2 depends only on the *set* of agents in c, not on its cycle structure;

(b) $T(c) \in \bigcup_{l\geq 1} \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}}$, and $P_{\mathbb{M}'}(T(c)) = P_{\mathbb{M}'}(c)$.

We record a few properties of this transformation T: first, $T(T(c)) = c$, so that T is self-inverse; second, $T(c)$ has one less or one more cycle than does c; and finally, the matchings induced by c and $T(c)$ are identical, therefore the residual problems are identical as well. These observations collectively imply that T is a bijection between $\{c \in \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}'} : \nu(c) \subseteq$

[3]Any assumption on the priority structure for the problem involving the agents in A must be preserved for the reduced problem A'. This is generally easily verified.

[4]In all the applications of this idea, every $c \in \bigcup_{l\geq 1} \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}}$ has at least two candidates for the roles of x_1 and x_2, so this can always be done.

π, l odd$\}$ and $\{c \in \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}'} : \nu(c) \subseteq \pi, l$ even$\}$. Furthermore, $P_{\mathbb{M}'}(T(c)) = P_{\mathbb{M}}(c)$ for any $c \in \bigcup_{l\geq 1} \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}}$. Rewriting Eq. (3) as

$$\sum_{l \text{ odd}} \sum_{\substack{c \in \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}} \\ \nu(c) \subseteq \pi}} P_{\mathbb{M}'}(c)\, \mathbb{M}'(A \setminus c, \pi \setminus \nu(c)) \;=\; \sum_{l \text{ even}} \sum_{\substack{c \in \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}} \\ \nu(c) \subseteq \pi}} P_{\mathbb{M}'}(c)\, \mathbb{M}'(A \setminus c, \pi \setminus \nu(c)),$$

we note that its validity is immediate from the preceding discussion: for each term involving a cycle product c on the left, there is a corresponding term (involving cycle product $T(c)$) on the right such that the expressions are identical, and vice-versa.

Example 3. Consider an instance with 4 agents, with $A = \{1, 2, 3, 4\}$ and $S = \{a, b, c, d\}$, where the agents have strict preferences given by Table 1.

1	2	3	4
a	b	c	a
b	a	d	c
c	c	a	b
d	d	b	d

Table 1: Agent preferences for Example 3, from most- to least-preferred.

Let RP be $\mathbb{M}$ and RE be $\mathbb{M}'$. Consider the sets of cycle products under RP. These are simply

$$\begin{aligned} \mathcal{C}^1_{\mathbb{M}} &= \{(1), (2), (3), (4)\}, \\ \mathcal{C}^2_{\mathbb{M}} &= \mathcal{C}^3_{\mathbb{M}} = \mathcal{C}^4_{\mathbb{M}} = \emptyset. \end{aligned}$$

Similarly, the sets of cycle products under RE are

$$\begin{aligned} \mathcal{C}^1_{\mathbb{M}'} &= \mathcal{C}_{\mathbb{M}} \cup \{(12), (13), (23), (24), (34), (123), (132), (234), (243)\}; \\ \mathcal{C}^2_{\mathbb{M}'} &= \{(1)(2), (1)(3), (2)(3), (2)(4), (3)(4), (1)(23), (13)(2), (2)(34), \\ &\qquad (12)(3), (24)(3), (23)(4)\}; \\ \mathcal{C}^3_{\mathbb{M}'} &= \{(1)(2)(3), (2)(3)(4)\}; \\ \mathcal{C}^4_{\mathbb{M}'} &= \emptyset. \end{aligned}$$

Note that it is impossible for agents 1 and 4 to be part of the same cycle product since they both have a as their most preferred object. Any cycle product in RE that is not in RP must have at least two agents; let x_1 and x_2 be, respectively, the lowest-numbered agent and the second lowest-numbered agent in that cycle product. It is easy to verify that the associated transformation T (with this choice of x_1 and x_2) satisfies the claimed properties. For the cycle $c = (23) \in \mathcal{C}_{\mathbb{M}'} \setminus \mathcal{C}_{\mathbb{M}}$, note that $T(c) = (2)(3) \in \mathcal{C}_{\mathbb{M}'} \setminus \mathcal{C}_{\mathbb{M}}$, and that $P_{\mathbb{M}'}(c) = 1/12 = P_{\mathbb{M}'}(T(c))$. If $c' = (243)$, $T(c') = (24)(3)$; and $P_{\mathbb{M}'}(c') = 1/24$ as is $P_{\mathbb{M}'}(T(c'))$. Finally, if $\tilde{c} = (2)(3)(4)$, $T(\tilde{c}) = (23)(4)$; again, $P_{\mathbb{M}'}(\tilde{c}) = 1/24$ as is $P_{\mathbb{M}'}(T(\tilde{c}))$. We leave it to the reader to verify the bijection for each of the other cycle products.

We can summarize the discussion so far as follows: To prove that mechanisms $\mathbb{M}$ and $\mathbb{M}'$ satisfying properties (i)-(iv) are equivalent, it is *sufficient* to verify the equivalence of $\mathbb{M}$ and $\mathbb{M}'$ when there is a *single* agent and to prove Eq. (3). The latter can be done by showing how to choose x_1 and x_2 such that the associated transformation T satisfies properties (a) and (b). In the sections that follow, we shall carry out these (two) steps for various pairs of mechanisms.

4 Linear Priority Structures

4.1 RP and RE in a model with endowments

We consider an assignment model in which some of the agents are endowed with objects. The most prominent application is in the setting of house allocation where some of the houses already have existing tenants who are willing to move but only to other houses that are better for them. The RP and RE mechanisms can be generalized in many ways to this setting, and several recent papers show the equivalence between the generalized RP and RE mechanisms, see Sömnez and Ünver [16], Pathak and Sethuraman [11] and Ekici [6]. To illustrate our general approach, we provide a simple proof of Ekici's result, which is also the most recent.

Suppose the agents A are partitioned into N and E, and the objects are partitioned into V and O, such that $|E| = |O|$ and $|N| = |V|$. The interpretation is that every agent in E is an "existing tenant," and every object in O is an "occupied house." The agents in N are "new tenants", and the "houses" in V are "vacant". Each agent in E is endowed with an object in O, for which he has the highest priority. Ekici [6] proposes natural generalizations of the RP and RE mechanisms to this setting (we abuse notation and continue to call these RP and RE), which we describe next. (Recall that the matching is found by applying the TTC mechanism to a priority structure for the objects; the RP and RE mechanisms differ only in how the priority structure—to which the TTC algorithm is applied—is generated.)

RP. Let $(\sigma_1, \sigma_2, \ldots, \sigma_n)$ be a permutation of the agents chosen uniformly at random, that is, every permutation of the agents is equally likely. Let i^* be the smallest $i \in \{1, 2, \ldots, n\}$ for which agent σ_i is still in the problem. The priority structure for the remaining objects is determined as follows: each object in V has σ_{i^*} as the top-priority agent; if the agent endowed with $o \in O$ still remains in the problem, he retains top-priority for o, otherwise σ_{i^*} has top priority for o.

RE. Let $(\sigma_1, \sigma_2, \ldots, \sigma_n)$ be a permutation of the agents chosen uniformly at random, that is, every permutation of the agents is equally likely. Suppose the objects are ordered in an arbitrary (but fixed) manner so that there is a first object, a second object, and so on. For each $j = 1, 2, \ldots, n$, if the j-th object is in V, then its initial top-priority agent is σ_j; otherwise, the jth object is in O and its initial top-priority agent is the agent $i \in E$ to whom it is endowed. In this latter case, agent σ_j becomes the "inheritor" of that agent i: When agent i departs, every object for which i has top-priority at that stage but is still unassigned will now have σ_j as its top-priority agent. Note that the inheritor of an agent $i \in E$ may be another agent $i' \in E$ who has already been assigned an object, triggering a chain of inheritances until an inheritor yet to be assigned an object is found. It is a simple matter to verify that each remaining object will have a top priority agent at any stage.

We now give an alternative proof that RP and RE are equivalent in this setting. First observe that both mechanisms satisfy properties (i)-(iv), with RP playing the role of $\mathbb{M}$ and RE the role of $\mathbb{M}'$ in the definitions of those properties. Additionally, when there is a single agent it is clear that these mechanisms are equivalent. Finally, the residual problem after clearing a cycle product c is simply a smaller instance of the original problem, under both RP and RE. For RP this is immediate; for RE, it is possible that some objects in O remain, but without the agents who owned them, as these agents departed with other objects; in that case, we simply place these objects in V (for the smaller problem). Consequently, we may use the induction argument from Section 3.

To complete the proof of equivalence, we need to be precise about the choice of x_1 and x_2, given any cycle that is present in the TTC graph of RE, but not in that of RP. Let A^* be the agents whose most-preferred object is in V. Notice that each cycle product in the TTC graph for RP involves *at most* one agent from A^*; the TTC graph for RE contains all of these cycle products but could contain other cycle products involving two or more agents from A^*. Consequently, any cycle product $c \in \bigcup_{l \geq 1} \mathcal{C}^l_{\mathbb{M}'} \setminus \mathcal{C}^l_{\mathbb{M}}$ contains at least two agents from A^*. Given any cycle product c that could be present in the TTC graph of RE, but not in that of RP, let x_1 and x_2 be, respectively, the smallest and second smallest-numbered agents in $c \cap A^*$. This choice of x_1 and x_2 satisfies property (a). It is simple to verify that the cycle products c and $T(c)$ occur with the same probability in RE, satisfying property (b).

4.2 Assignment models with agent groups

We turn now to a model, introduced by Carroll [5], in which we are given an exogenous partition of the agents into groups. Furthermore, for each object, we are given a (fixed) priority ordering of the groups. This is useful in modeling situations in which agents within a group should be treated equally, but agents across groups can be prioritized. Carroll showed that this model, described in more detail next, is rich enough to subsume most of the equivalence results in the literature, the only notable exception being the result of Ekici discussed in the previous section. In this model, Carroll proposed two mechanisms—the Random Serial Dictatorship in Groups mechanism (RSDIG) and the Within-Groups Top Trading Cycles mechanism (WGTTC)—and showed that they are equivalent.

As before, we are given a set A of agents, a set S of objects, with $|A| = |S|$. Agents have strict preferences over the objects. Suppose the agents are partitioned into groups $G_1, G_2, \dots G_m$, for some $m \geq 1$. Each group G_i has an associated role group with the same number of members, $R_i = \{r^i_1, r^i_2, \dots, r^i_{|G_i|}\}$. Each object's priority structure is given by an ordering of the roles, such that all the roles belonging to a given group occur consecutively (a property Carroll [5] calls *group-respecting*); however the roles of a given group can be ordered differently in the priority orderings of different objects. One interpretation of these two features is as follows: each object is first endowed to some group and passes to a different group only when all the members of the orginal group have all been matched; however, a given group may choose to distribute the object it owns among its roles any way it pleases. (This latter feature allows constraints such as objects a and b cannot have the same top-priority agent.) The priority structure for the objects is specified in terms of roles, so following Carroll [5] we call this a *priority framework*. The two mechanisms studied by Carroll differ only in the priority frameworks to which the TTC algorithm is applied:

RSDIG. The roles R_i of each group G_i always appear in the order $r^i_1, r^i_2, \dots, r^i_{|G_i|}$.

WGTTC. For each group G_i, there is no restriction on the ordering of roles, and the ordering may differ from object to object.

In each of these cases the roles have to be instantiated before applying the TTC algorithm. This is done by choosing, for each group G_i, a bijection uniformly at random from its set of roles R_i to the set of agents G_i. In such a case, Carroll showed that the RSDIG and WGTTC mechanisms are equivalent. We give an alternative proof of this result.

Observe that both mechanisms satisfy properties (i)-(iv), with RSDIG as $\mathbb{M}$ and WGTTC as $\mathbb{M}'$. Furthermore, it is clear the two mechanisms are equivalent where there is a single agent. Also, the residual problem after clearing a cycle product c is simply a smaller version of the original problem, for both RSDIG and WGTTC. So we may use the induction argument of §3. It remains to show there is an appropriate way to pick x_1 and x_2 for transformation T so that $P_{\mathbb{M}'}(c) = P_{\mathbb{M}'}(T(c))$.

Observe that each cycle product in the TTC graph for RSDIG involves *at most* one agent from each group. The TTC graph for WGTTC contains all of these cycle products, but also includes those containing *two or more* agents from *some* group. Let c be a cycle product containing two or more agents from some group. Consider the smallest labeled group with two or more agents in c. Let x_1 be the smallest and and x_2 the second-smallest labeled agents from that group in c, and consider the canonical transformation T described earlier. The resulting cycle product $T(c)$ corresponds to the cycle product induced by the same role-to-agent map as c except that the roles of x_1 and x_2 are swapped. Since x_1 and x_2 are in the same group, it follows that the role-to-agent maps inducing c and $T(c)$ both occur with the same probability. This verifies properties (a)-(b) so the equivalence result follows.

5 Tree-based inheritance with groups and endowments

We are ready to formulate a general object assignment model. As before we have a set A of agents and a set S of objects. Each agent wishes to be assigned exactly one object. Suppose the agents are partitioned into A and E and the objects are partitioned into V and O. Each object $o \in O$ is endowed to an agent $\rho(o) \in E$, and each agent $i \in E$ is endowed with *at least* one object in O. (Different objects in O may be endowed to the same agent.) Thus $|O| \geq |E|$ and $\rho(O) = E$. As in our earlier discussion, one can think of the agents in E as the existing tenants and the objects in O as the houses they occupy in a house allocation model; in that case, the agents in N and the houses in V are, respectively, the new tenants and the new houses. Agents have strict preferences over the objects. The priority structure of the objects—a distinguishing feature of our model—is specified by an *inheritance tree*, which generalizes the models of Carroll [5], as well as the earlier models of Papai [9] and Svensson and Larsson [18]. In particular, the priority structure allows for objects to be endowed to certain agents, admits group hierarchies, and additionally allows the inheritance of an object to depend on the partial matching at that stage.

The agents are also partitioned into groups $G_1, G_2, \ldots, G_m$ so that $G_i \cap G_j = \emptyset$ for all $i \neq j$ and $\bigcup G_i = A$. Note that composition of each of these groups is unrestricted: G_i may have a non-empty intersection with A or E or both. However the group membership of an agent may affect his priority for certain objects. Each agent group G_i is associated with a set of *roles* R_i with $|R_i| = |G_i|$, with $R_i \cap R_j = \emptyset$ for all $j \neq i$. The priority structure for each object is specified in terms of roles. When the mechanism is run, the roles, R_i, are instantiated with the agents in G_i, uniformly at random: each mapping of the roles R_i to the agents in G_i is equally likely. We say that an agent a or role r *owns* an object o if a or r has the topmost priority for o.

5.1 Inheritance Trees

The priority structure of each object is specified in terms of an inheritance *tree* as introduced by Papai [9], with one key difference: the nodes of the tree are populated by *roles* rather than the agents.

Every object $o \in S$ has an associated *inheritance tree* Γ_o, which is a rooted tree graph with directed arcs. If there are n agents in all, Γ_o has one root node (level 0), $n-1$ nodes at level 1, $(n-1)(n-2)$ nodes at level 2, etc., each of which is labeled with a role. Each node has exactly one incoming arc except the root (which has none), and each node at level k has $(n-k-1)$ outgoing arcs, each of which terminates at a node at level $k+1$ and is labeled with an object other than o and the k objects appearing as labels in the unique path from the root to that node. Moreover, in any path from the root to a leaf, each role label should appear exactly once. The agent who owns the object is the one whose role is the label of the root.

The tree Γ_o defines an inheritance plan for the object o in the following sense: consider a path from the root node to a node at level k, and suppose the labels of the nodes and arcs in the path are $r_0 - o_0 - r_1 - o_1 - r_2 - \ldots - r_{k-1} - o_{k-1} - r_k$. Then the agent with role r_k owns o if for all $i = 0, \ldots, k-1$ the agent with role r_i is assigned object o_i, and the agent with role r_k is still unassigned. (The assignment of agents with roles other than r_i, $i = 0, \ldots, k$ to objects other than $o, o_0, o_1, \ldots, o_{k-1}$ is immaterial.) Note that the definition of an inheritance tree ensures that in any partial assignment of objects to agents, either object o is assigned, or there is a unique maximal path originating from the root of Γ_o using only agents and objects in that partial assignment whose terminal node is the role of an unmatched agent, who next owns o.

We are now ready to specify the priority structure for each object $o \in S$. For $o \in V$, the priority structure is given by the inheritance tree Γ_o, with the roles instantiated uniformly at random from the corresponding groups; each object $o \in O$ is owned by the agent $\rho(o)$ until that agent is assigned an object; if $\rho(o)$ is assigned an object, but o is still unassigned, its ownership will be governed by its inheritance tree Γ_o.

Suppose the TTC mechanism is applied to a problem where the priority structure for each object is given by an inheritance tree. We describe how the trees are updated when a cycle c is cleared. Typically this process involves trimming branches and contracting arcs so that information that is no longer useful is discarded; the resulting updated tree will involve only those objects and roles that are still "unassigned". Let X be the set of objects that are assigned and let Y be the roles that these objects are assigned to when the TTC mechanism is applied. For each $x \in X$, let $\lambda(x)$ be the role in Y that x is assigned to. The inheritance trees are updated as follows:

- Discard the inheritance trees Γ_o for each $o \in X$;

- For each $x \in X$, and for each node labeled $\lambda(x)$ in every remaining inheritance tree, delete every outgoing arc from that node with a label other than x (and the subtree rooted at the other end of that arc). So for each $x \in X$, the only arc that emanates from any node labeled $\lambda(x)$ will be labeled x.
- Contract every arc with a label of x emanating from a node labeled $\lambda(x)$ (this is the same as deleting the arc x, and moving the subtree rooted at its head node to its tail node).

Observe that the updates to the inheritance trees removes all the paths that can no longer be realized and retains the paths that *may* still be realized. We end this section with an illustrative example.

Example 4. Consider an instance of the problem with 4 agents $\{1, 2, 3, 4\}$ and 4 houses $\{a, b, c, d\}$, and suppose role r_i is mapped to agent i for each i. Focus on object a, whose inheritance tree is given by Γ_a in Figure 1(a). We illustrate in Figures 1(b)–(d) how ownership and the inheritance tree of object a evolves under some sequence of submatchings, say $(1 \leftarrow b)$, $(3 \leftarrow d)$, $(2 \leftarrow c)$, and $(4 \leftarrow a)$ in the given order. Initially, a is owned by agent 1, who is assigned b in phase 1. Following this, a's inheritance tree is updated to Γ'_a, and agent 2 becomes the new owner of a. When agent 3 is assigned d, 2 continues to own a, but a's inheritance tree is updated to Γ''_a. Then, agent 2 is assigned c at which point agent 4 becomes the owner of a; this is shown in the tree Γ'''_a. Finally 4 is matched with a so a is removed from the problem.

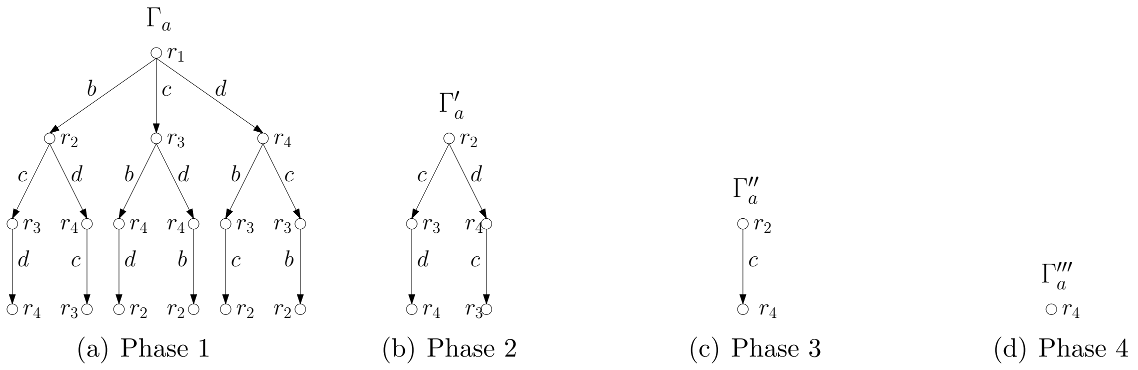


Figure 1: The evolution of Γ_a in Example 4.

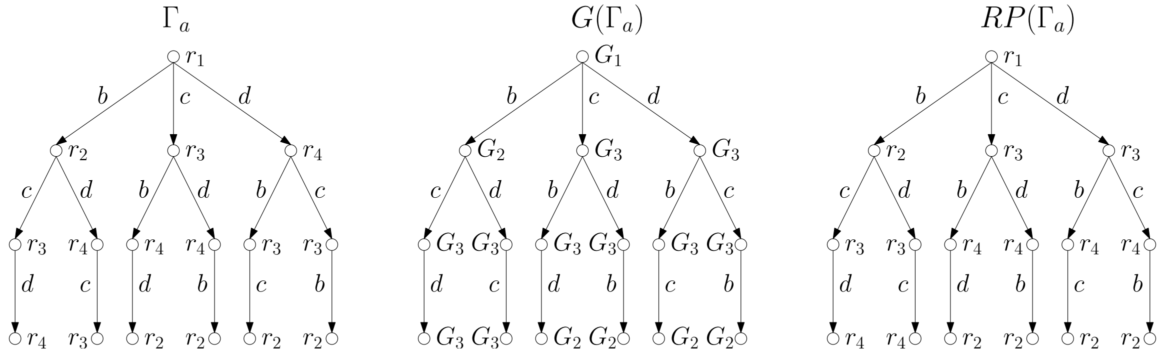
5.2 Equivalence results

We generalize the equivalence results discussed earlier to settings in which the priority structure for each object is an inheritance tree. To state the equivalence result formally, we need the following definitions.

Given an inheritance tree Γ, define $G(\Gamma)$ to be the tree obtained by replacing each role at the nodes of Γ with the group to which that role belongs. Inheritance trees Γ and Γ' are *G-similar* if $G(\Gamma)$ and $G(\Gamma')$ are identical.

An inheritance tree is a *random priority* inheritance tree if in every root-to-leaf path, the roles belonging to each group G_i appear in ascending order (from the lowest index to the highest index). Notice that when all the inheritance trees in the problem are random priority inheritance trees, each $o \in V$ owned by a member of a group G_i will be owned by the *same* role. Given an arbitrary inheritance tree Γ, there is a natural relabeling of the nodes that gives rise to the associated random priority inheritance tree $RP(\Gamma)$: find $G(\Gamma)$ and apply the random-priority labeling to it. It follows that if Γ and Γ' are G-similar, $RP(\Gamma) = RP(\Gamma')$.

For a partial assignment (or matching) π, and an object $s \in S$, let $g_\pi(s)$ be the group to which s is assigned; if s is not assigned to any agent in π, define $g_\pi(s)$ to be zero. Two partial assignments π and π' are *G-equivalent* if $g_\pi(s) = g_{\pi'}(s)$ for all $s \in S$. In other words, π and π' must assign the same objects, and each object assigned by them must be assigned to the same group under both π and π' (although the actual agents or roles to which s is assigned may differ). Given an inheritance tree Γ and a partial assignment π, let Γ^π be the updated inheritance tree after the objects and roles in π are removed from the problem. An inheritance tree Γ is *G-invariant* if $G(\Gamma^\pi) = G(\Gamma^{\pi'})$ for any G-equivalent partial assignments π and π'. Note that if Γ is G-invariant, then so is any inheritance tree Γ' that is G-similar to Γ.



(a) Inheritance tree (b) Replacing roles with groups (c) Induced random priority inheritance tree

Figure 2: A role-inheritance tree Γ_a; $G(\Gamma_a)$, the tree after replacing its roles with groups, and $RP(\Gamma_a)$ the random-priority inheritance tree induced by Γ_a when the groups are $G_1 = \{1\}$, $G_2 = \{2\}$, $G_3 = \{3, 4\}$.

Example 5. Consider the inheritance tree Γ_a in Figure 2(a). Let the groups be $G_1 = \{1\}$, $G_2 = \{2\}$, $G_3 = \{3, 4\}$. After relabeling the roles with their associated groups, we get $G(\Gamma_a)$

as shown in Figure 2(b). The random-priority tree induced by Γ_a, $RP(\Gamma_a)$, is shown in Figure 2(c): in every path from the root to any leaf the roles of each group appear consecutively, and in ascending order. Moreover, both Γ_a and $RP(\Gamma_a)$ are G-invariant.

Theorem 1. *Fix $A = (E, N)$, $S = (O, V)$ and ρ. Let $\{\Gamma_s\}$ be a set of inheritance trees on (A, S) that are G-invariant. Then the TTC mechanism using $\{\Gamma_s\}$ and the TTC mechanism using $\{RP(\Gamma_s)\}$ are equivalent.*

Proof. Let $\mathbb{M}$ be the TTC mechanism applied to $\{RP(\Gamma_s)\}$ and let $\mathbb{M}'$ be the TTC mechanism applied to $\{\Gamma_s\}$. With this interpretation, property (i) is immediate; property (ii) follows from the G-invariance of $\{\Gamma_s\}$ for $\mathbb{M}'$, and from the G-invariance of $\{RP(\Gamma_s)\}$ for $\mathbb{M}$. (Note that G-invariance of $\{RP(\Gamma_s)\}$ is implied by G-invariance of $\{\Gamma_s\}$.) Properties (iii) and (iv) are also easily verified for the two mechanisms. Futhermore, it is clear that the two mechanisms are equivalent when there is a single agent.

Consider the TTC graph for $\mathbb{M}$ and $\mathbb{M}'$ in the residual problem under any sequence of cleared cycles. Since the sequence of cycles cleared are the same, the inheritance trees are updated in both mechanisms similarly (the same arcs are deleted and contracted, though the role labels removed may differ). Moreover, the remaining agents and objects are the same in both mechanisms. It follows that the updated inheritance trees under $\mathbb{M}$ are the same as the the random-priority trees induced by the updated inheritance trees under $\mathbb{M}'$ (a relabeling of roles may be necessary), so we can use the induction argument from §3.

It remains to show that there is an appropriate way to pick x_1 and x_2 for transformation T that satisfy properties (a)-(b). Let A^* be the agents whose most-preferred object is in V. Observe that each cycle product in the TTC graph for $\mathbb{M}$ has *at most* one agent in $A^* \cap G_k$ for each k. The TTC graph for $\mathbb{M}'$ contains these cycle products as well (in addition to zero or more cycle products containing at least two agents from A^* from the same group). Consider the lowest labeled group G^* with at least two agents from $A^* \cap c$. Let x_1 and x_2 be, respectively, the lowest and second-lowest labeled agents in $A^* \cap c \cap G^*$. This satisfies property (a). Additionally, the resulting cycle product $T(c)$ is exactly the cycle product induced by the same role-to-agent map as c except that the roles mapping to x_1 and x_2 are swapped. Since x_1 and x_2 are in the same group, the role-to-agent maps inducing c and $T(c)$ both occur with the same probability, so property (b) is satisfied as well, and the equivalence result follows. $\square$

Corollary. *Fix $A = (E, N)$, $S = (O, V)$ and ρ. Let $\{\Gamma_s\}$ and $\{\Gamma'_s\}$ be two sets of inheritance trees on (A, S) that are G-invariant, and let Γ_s be G-similar to Γ'_s for each $s \in S$. Then the TTC mechanism using $\{\Gamma_s\}$ and the TTC mechanism using $\{\Gamma'_s\}$ are equivalent.*

Proof. As $\{\Gamma_s\}$ and $\{\Gamma'_s\}$ are G-similar, $\{RP(\Gamma_s)\}$ and $\{RP(\Gamma'_s)\}$ are identical, and the result is immediate from Theorem 1 $\square$

Remark. One can think of G-invariance as a natural generalization of the notion of a *group respecting* priority structure as defined by Carroll [5] (see §4.2). While this restriction is a technical necessity in the equivalence proof, it is not an unreasonable one: For example, in allocating public housing with agents grouped by income level, it may be undesirable for a house to pass from a low income group to a high income group and back to the low income group. This cannot occur if the inheritance structure is group respecting. Following Carroll, we could define an inheritance tree to be *group respecting* if in any path from the root to a leaf, all roles from the same group appear consecutively. It is easy to verify that every G-invariant tree is group respecting, but the converse is not true. Example 6 demonstrates the necessity of G-invariance for the equivalence result to hold. It also shows that equivalence may not hold when trees are group respecting but not G-invariant. If the inheritance tree is equivalent to a linear priority order as in Carroll [5], it can be shown that an inheritance tree is G-invariant if and only if it is group-respecting.

Example 6. To demonstrate the necessity of G-invariance of all trees, consider the following 4-agent example. Let $A = \{1, 2, 3, 4\}$, $S = \{a, b, c, d\}$, with three groups $G_1 = \{1, 2\}$ (with corresponding roles $\{r_1, r_2\}$), $G_2 = \{3\}$ (with roles $\{r_3\}$) and G_3 (with roles $\{r_4\}$). Suppose the agents' preferences are given by Table 2, and that the inheritances for the objects are defined by the following trees: for a, Γ_a as given in 3(a); for b, Γ_b induced by the linear ordering of roles $r_1 \succ r_2 \succ r_3 \succ r_4$; for c, Γ_c induced by the linear ordering $r_2 \succ r_1 \succ r_3 \succ r_4$; and for d, Γ_d induced by the linear ordering $r_1 \succ r_2 \succ r_3 \succ r_4$. Notice that Γ_b, Γ_c and Γ_d are G-invariant. On the other hand, although Γ_a is group-respecting it is not G-invariant since the two matchings $(1 \leftarrow b, 2 \leftarrow c)$ (induced by the role mappings $r_1 \to 1, r_2 \to 2$) and $(1 \leftarrow c, 2 \leftarrow b)$ (induced by $r_1 \to 2, r_2 \to 1$) are G-equivalent but result in top-priority roles from different groups for a (G_2 and G_3 respectively).

1	2	3	4
c	c	a	a
b	b	d	d
a	a	c	c
d	d	b	b

Table 2: Agent preferences for Example 6.

It is simple to verify the following: under $\{\Gamma_s\}$ two matchings $(1 \leftarrow b, 2 \leftarrow c, 3 \leftarrow a, 4 \leftarrow d)$ and $(1 \leftarrow c, 2 \leftarrow b, 3 \leftarrow a, 4 \leftarrow d)$ are output with equal probability; under $\{RP(\Gamma_s)\}$ two matchings $(1 \leftarrow b, 2 \leftarrow c, 3 \leftarrow d, 4 \leftarrow a)$ and $(1 \leftarrow c, 2 \leftarrow b, 3 \leftarrow d, 4 \leftarrow a)$ are output with equal probability. Notice that under $\{\Gamma_s\}$ 3 is always assigned a and b is always assigned b, whereas under $\{RP(\Gamma_s)\}$ the situation is reversed in that 3 is always assigned b and 4 is always assigned a. So the two mechanisms are not equivalent.

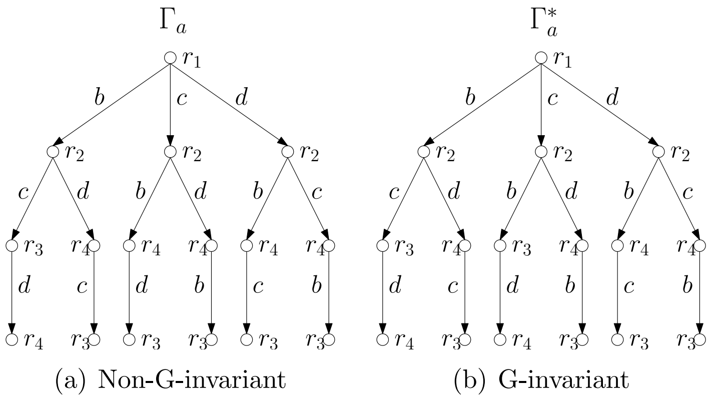


Figure 3: Two inheritance trees for object a in Example 6.

Now, consider if a has the G-invariant inheritance tree Γ_a^* given in Figure 3(b) instead, while the inheritance trees of the other objects Γ_s^* are the same as Γ_s. It is not difficult to check that the mechanisms induced by $\{\Gamma_s^*\}$ and $\{RP(\Gamma_s^*)\}$ are equivalent, in particular both output two possible matchings $(1 \leftarrow b, 2 \leftarrow c, 3 \leftarrow a, 4 \leftarrow d)$ and $(1 \leftarrow c, 2 \leftarrow b, 3 \leftarrow a, 4 \leftarrow d)$ with equal probability.

Remark. It is not difficult to see that inheritance trees allows for inheritance structure that cannot be captured by any linear ordering of roles. For example, consider a 4-agent instance where the agents $\{1, 2, 3, 4\}$ are in three groups $G_1 = \{1, 2\}$ (roles $\{r_1, r_2\}$), $G_2 = \{3\}$ (roles $\{r_3\}$) and $G_3 = \{4\}$ (roles $\{r_4\}$), and the objects are $\{a, b, c, d\}$. Suppose the inheritance tree a is given by Γ_a in Figure 4, which is G-invariant. If 1 is assigned b and 2 is assigned c, 3 becomes the owner of a. However, if instead 1 is assigned b and 2 is assigned d, 4 becomes the owner of a. It is not difficult to see that no group-respecting linear orderings may capture this sort of branching behavior.

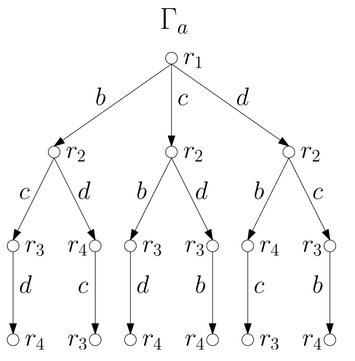


Figure 4: A G-invariant tree that cannot be represented by any linear inheritance.

6 Discussion

Our approach suggests the following general framework for thinking about allocating indivisible objects to a group of agents. Suppose we are given a set of agents with strict preferences over objects, an exogenous partition of the agents into groups; suppose also that agents are partitioned into "existing tenants" and "new agents," and objects into "occupied" and "vacant" objects. Each occupied object has an obvious top-priority agent, namely, the agent who occupies it. Assume each vacant object is endowed to a unique group. Finally, each group can assign the top-priority role any way it pleases for all of the vacant objects endowed to it. Each group instantiates its roles uniformly at random, either without repetition (a given role is equally likely to be mapped to any remaining agent in that group who is not yet mapped) or with repetition (each role is equally likely to be mapped to any remaining agent in that group). Each group make this decision independently, and can choose either type of mapping, regardless of what the other groups do. The TTC mechanism is used to clear (a subset of) cycles. If a vacant object remains, but its occupant does not, it is treated as a new object. If an object is endowed to a group, and its top-priority agent is no longer present, it is endowed to a different agent in the same group, assuming at least one such agent exists. If an object is endowed to a group, its top-priority agent is no longer present and no agent from that group is present, control of that object passes to another group (in a G-invariant) way. Our general approach shows that the final outcome is the same regardless of how the groups assign the objects they control amongst themselves. It is easy to see that all the mechanisms considered in our paper fit into this framework.

Consider the special case in which none of the agents are endowed with an object. Note that in this case all the mechanisms that fit this framework are strategyproof (SP), and always yield a Pareto efficient (PE) assignment. Furthermore, agents with identical preferences receive identical (probabilistic) allocations (ETE). The aforementioned result boils down to the statement that every mechanism in this framework is equivalent to the RP mechanism. An intriguing conjecture that suggests itself, in light of these equivalence results, is that the RP mechanism is characterized by these three properties. If we only require a subset of these properties, there are other mechanisms: giving an equal share of each object to each agent satisfies SP and ETE, but not PE; a serial dictatorship with an exogenous order of the agents satisfies SP and PE, but not ETE; and the probabilistic serial mechanism of Bogomolnaia and Moulin [4] satisfies PE and ETE, but not SP. The characterization result is, of course, of independent interest. We note, however, that this characterization of RP may be an alternative approach to the proof of the equivalence results established here.

References

[1] A. Abdulkadiroğlu, P. Pathak, and A. E. Roth. The New York City high school match. *American Economic Review, Papers and Proceedings*, 95(2):364–367, 2005.

[2] A. Abdulkadiroğlu and T. Sönmez. Random serial dictatorship and the core from random endowments in house allocation problems. *Econometrica*, 66(3):689–701, 1998.

[3] C. G. Bird. Group incentive compatibility in a market with indivisible goods. *Economic Letters*, 14(4):309–313, 1984.

[4] A. Bogomolnaia and H. Moulin. A new solution to the random assignment problem. *Journal of Economic Theory*, 100:295–328, 2001.

[5] G. Carroll. A general equivalence theorem for allocation of indivisible objects. Working Paper.

[6] Ö. Ekici. Fair and efficient discrete resource allocation: A market approach. Job Market Paper.

[7] A. Hylland and R. J. Zeckhauser. The efficient allocation of individuals to positions. *Journal of Political Economy*, 87(2):293–314, April 1979.

[8] D. E. Knuth. An exact analysis of stable allocation. *Journal of Algorithms*, 20(2):431–442, 1996.

[9] S. Pápai. Strategy-proof assignment by hierachical exchange. *Econometrica*, 68:1403–1433, 2000.

[10] P. Pathak. The mechanism design approach to student assignment. *Annual Reviews of Economics*, 3, Forthcoming.

[11] P. Pathak and J. Sethuraman. Lotteries in student assignment: An equivalence result. *Theoretical Economics*, 6:1–17, 2011.

[12] A. Roth, T. Sönmez, and M. U. Ünver. Kidney exchange. *Quarterly Journal of Economics*, 119(2):457–488, May 2004.

[13] A. E. Roth. Incentive compatibility in a market with indivisible goods. *Economic Letters*, 9:127–132, 1982.

[14] A. E. Roth and A. Postlewaite. Weak versus strong domination in a market with indivisible goods. *Journal of Mathematical Economics*, 4:131–137, 1977.

[15] L. Shapley and H. Scarf. On cores and indivisibility. *Journal of Mathematical Economics*, 1:23–28, 1974.

[16] T. Sömnez and M. U. Ünver. House allocation with existing tenants: An equivalence. *Games and Economic Behavior*, 52(1):153–185, 2005.

[17] T. Sönmez and M. U. Ünver. *Handbook of Social Economics*, volume 1A, chapter Matching, Allocation, and Exchange of Discrete Resources, pages 781–852. North-Holland, The Netherlands, 2011.

[18] L-G. Svensson and B. Larsson. Strategy-proofness, core, and sequential trade. *Review of Economic Design*, 9(2):167–190, 2005.

[19] L. Zhou. On a conjecture by gale about one-sided matching problems. *Journal of Economic Theory*, 52:123–135, 1990.