Essays on Efficiency, Fairness and Strategy-Proofness in Allocation Problems with Exact Capacity Constraints

T.C.A. Madhav Raghavan

Thesis submitted to

The Indian Statistical Institute, New Delhi

in partial fulfilment of the requirements for the award of the degree of

Doctor of Philosophy

in Economics

Economics and Planning Unit

Indian Statistical Institute

7 Shaheed Jeet Singh Sansanwal Marg,

New Delhi 110016,

India

Submitted: July 2014

Accepted: July 2015

Essays on Efficiency, Fairness and Strategy-Proofness in Allocation Problems with Exact Capacity Constraints

T.C.A. Madhav Raghavan

Thesis supervisor: Prof. Arunava Sen

Abstract

In this thesis we consider situations where heterogenous objects are to be distributed among a set of claimants. The objects in question are indivisible, so they cannot be split or shared. There is no money in this economy, so objects cannot be simply bought and sold. Allocations to claimants must be based on their preferences over objects alone. We impose the additional restriction that each object has an exact capacity constraint, such that each object is assigned either to a pre-specified (and fixed) number of agents, or it is not assigned at all. Further, we assume that the capacity constraint is the same for all objects. We call such a situation an allocation problem.

In Chapter 2 we consider the case where the exact capacity constraint is of size 2. Thus agents must be divided into pairs. Our quest is to find efficient rules in the framework. We propose a rule which we call the Partner Trading (PT) Rule, and show that it characterises the set of all rules in this model that satisfy the standard properties of strategy-proofness, limited influence, unanimity and neutrality. It is also group-strategy-proof and Pareto efficient. The PT rule can be thought of as a generalisation of the famous top trading cycles procedure to this particular environment.

In Chapter 3 we consider fair rules in this framework. We extend the exact capacity constraints to be of any size. We demonstrate that the well-known incompatibility between fairness and Pareto efficiency persists in this model too. We propose a rule which we call the Deferred Acceptance with Improvements (DAI) rule, which is fair and constrained efficient. We also identify a Pareto improvement procedure that always leads us to a fair and constrained efficient allocation in one iteration. We show, however, that the DAI rule is not strategy-proof.

In Chapter 4 we study group-strategy-proofness, which is the extension of strategy-proofness to groups of agents. This property comes in a standard form and a weak form. The distinction between the two forms is non-trivial as important rules in the literature fail the standard form but satisfy the weak form. It is well-known that in allocation models such as ours, a strategy-proof rule that is also non-bossy is (standard) group-strategy-proof. But the link between strategy-proofness and weak group-strategy-proofness is not as well established. We make steps towards this in this paper. We identify conditions (which we call ultra-weak Maskin monotonicity and weak non-bossiness) that are sufficient to ensure that a strategy-proof rule is weakly group-strategy-proof. These conditions are natural weaker forms of commonly used axioms in the literature. We also demonstrate that the conditions are 'weak enough', in that a rule satisfying them may not be (standard) group-strategy-proof.

Acknowledgements

I would like to express my sincere gratitude to everyone who has helped me over the past few years. The PhD is a long and difficult journey, and one I could not have completed alone.

First and foremost, I would like to thank Prof Arunava Sen for being the supervisor every PhD candidate wishes for. He has been patient above all else. He indulged me in my fickleness and inability to stick to a subject, especially in the early years. He has been firm when I have needed him to be, which is often. He has provided incisive guidance in crucial moments, finding clarity where I couldn't. If there is anything of merit in these pages, it is because of him.

I would also like to thank Prof Debasis Mishra for being a pillar of support through the entire process. The reading groups organised by Prof Sen and him were invaluable in keeping us up-to-date with recent theoretical developments, and keeping me focused on research. His insightful comments and feedback on previous iterations of the papers in the thesis have been influential in their improvement. My debt to him is immense.

The faculty and administration at the ISI, Delhi, also deserve special mention. Profs Satya Das, Chetan Ghate, Abhiroop Mukhopadhyay, Bharat Ramaswami, Tridip Ray, E Somanathan and others have made us all feel like we belong, first as teachers and later as sources of encouragement. Behind the scenes, Mrs Niranjana Gupta and the kind folk in the accounts department ensured that the monthly stipends kept coming. Deepmala in the department office has cheerfully borne the brunt of all our paperwork. The library staff have often gone out of their way to help. The scholarships and travel support received from the ISI are also gratefully acknowledged.

My colleagues in the PhD programme have been compatriots in crime, in the sharing of joys and misery equally. Abdul Quadir, Anup Pramanik, Mihir Bhattacharya, Mridu Goswami, Sonal Yadav, Soumendu Sarkar, in particular, this would not have been possible without them.

Aditya Wig should be able to see reflections of himself in every page of this thesis. Many arguments in here are the end products of long and late-night discussions over cold-coffee and worse. He has brought a level head to both my excitement at breakthroughs and surliness at roadblocks. I am especially grateful for the recent commonality in our cause.

No acknowledgement of help received can be complete without a profuse thanks to my family. Ma and Baba, especially, firstly for gently guiding me into doing a PhD in the first place and then for keeping me firmly bound to it when I would have rather been doing almost anything else. My brother Sharad's equanimity in the face of any obstacle has been an inspiration for someone like me who continually imagines obstacles in his way. My aunts and uncles have provided much encouragement. Chandra Attai, in particular, thank you for your unshakeable conviction that I could do this, even when I didn't believe it myself. My grandmothers have found it hard to contain their love, and I have been the lucky recipient of many a spillover. And Poto, our ageing Labrador, whose patient company has been a daily comfort.

There have been many others - friends, teachers, family - who have left an indelible mark on this journey in their own particular ways. I thank them all. I am also grateful to two anonymous examiners for their comments on this thesis.

And yet, despite the overwhelming and unconditional support that I have been fortunate to receive, writing the thesis has often been a lonely process. I am grateful for the strength to deal with isolation and solitude long enough to have done it.

Madhav Raghavan
July 2014 (Updated: July 2015)

For Anna and Thatha, who should have been here to see it

Contents

1 Introduction **1**
- 1.1 Allocation Problems . . . 1
- 1.2 No Money, More Problem . . . 1
- 1.3 What We Look For in a Good Rule . . . 3
- 1.4 Summary of Chapters . . . 4

2 Efficient Pairwise Allocation via Priority Trading **10**
- 2.1 Introduction . . . 10
- 2.2 Literature Review . . . 11
- 2.3 An Informal Discussion . . . 12
- 2.4 Notation and Definitions . . . 16
- 2.5 Partner Trading Rules . . . 17
- 2.6 A Detailed Example . . . 26
- 2.7 Features of the PT Rule and Special Cases . . . 28
- 2.8 Axioms for Pairwise Project Allocation Rules . . . 30
- 2.9 Characterisation Result . . . 35
- 2.10 Conclusion . . . 37
- 2.11 Appendix: Proof of Theorem 1 . . . 38

3 Fair Allocation with Exact Capacity Constraints **44**
- 3.1 Introduction . . . 44
- 3.2 The Literature . . . 47
- 3.3 An Informal Discussion . . . 48
- 3.4 Notation and Definitions . . . 53
- 3.5 Properties of Allocation Rules . . . 54
- 3.6 Pareto Improvements . . . 56
- 3.7 Deferred Acceptance with Improvements . . . 58
- 3.8 Strategic Aspects of the DAI Rule . . . 62
- 3.9 Conclusion . . . 63
- 3.10 Appendix A: Proof of Theorem 2 . . . 64

4 Sufficient Conditions for Weak Group-Strategy-Proofness **67**
- 4.1 Introduction . . . 67
- 4.2 Notation and Definitions . . . 68
- 4.3 Standard Strategy-Proofness Properties . . . 69
- 4.4 Strategy-Proof Rules and their Group-Strategy-Proofness . . . 71
- 4.5 Weaker Conditions on Allocation Rules . . . 73
- 4.6 Results . . . 75
- 4.7 Conclusion . . . 79

5 Bibliography **80**

1

Introduction

1.1 Allocation Problems

Consider a situation where heterogenous objects are to be distributed among a set of claimants. The objects in question are indivisible, so they cannot be split or shared. Different claimants may want different objects and there is a limited supply of them. What's more, there is no money in this economy, so objects cannot be simply bought and sold.

We shall call such a situation an allocation problem. Examples abound in real life. For instance, a teacher in a school may wish to assign projects to his or her students. Each student may get one project or many students may share projects, but there is no buying or selling involved. Similarly, a manager may wish to assign tasks to his or her workers. A task is not assigned by money changing hands, however much the worker eventually might be paid for completing it. The same is true of a school that wishes to offer a selection of courses to its students. It must determine which students get to attend which course, particularly when there is large demand for some of them. This determination however does not depend on the 'price' of a course.

When we look around, a surprisingly wide class of other problems can be considered in this light. In a wholly different context, the problem of determining kidney donations for transplants is also an allocation problem. It is ethically questionable to buy or sell human organs. Yet there is a need for kidneys for transplants. How is this need met? Hospitals usually have a organ bank, which is built up over time via donations by 'good Samaritans', or the will of the deceased. Sometimes it may even be possible for couples to trade kidneys, provided they are compatible. Once kidneys become available, though, the question remains as to how to assign them among claimants.

Or take college admissions. Around the world, the assignment of college seats to students takes place on considerations of merit or other criteria. Usually students become eligible for seats via a competitive examination, and their relative grades determine the success of their application. Related to this is the matching problem faced by hospitals and medical residents. Hospitals wish to secure the best residents, while graduating medical students wish to apply to the best hospitals for their residency requirements. Any centralised matching process to deal with these situations becomes a solution to an allocation problem.

Another interpretation of allocation problems is that of matching people with other people. The classical example is called the marriage market. This is the problem of matching men and women with each other based on how much they desire to be together. Similarly, the so-called roommate problem is one of pairing tenants with each other. In these contexts, speed dating programs, online matrimony web sites and housing forums make their money essentially by solving allocation problems.

1.2 No Money, More Problem

There are two key features common to the situations above. The first is the idea of a centralised planner or designer. Many of these transactions could take place at an individual level. But there is also a coordination benefit to centralisation, especially in terms of the proper use of scarce resources.

For instance, there is no point for a school to have empty seats if there are students in another neighbourhood that need them. A centralised process could allocate spare seats more efficiently than schools doing it by themselves.

The second common feature is the absence of monetary compensation. The normal practices of setting a price or conducting negotiations or selling to the highest bidder do not generally work here. Instead, allocations have to be made using criteria that have to do with entitlements or notions of 'deservingness'. These criteria are usually distilled into the form of relative 'rights', 'priorities' or 'entitlements' that individuals have over different objects or vice versa.

For example, in the college admissions setting, students compete for colleges on the basis of grades obtained in school-leaving examinations, and in some cases may receive bonuses for other criteria such as living in the same neighbourhood as the college, or having a sibling attend the same college, and so on. Students with higher grades or those that better meet the requirements stand a better chance of admission. In the problem of allocating office space to faculty, rooms are usually assigned with reference to seniority. Elsewhere, office tasks are assigned to workers based on measures of their competence. Courses are assigned to students based on completion of prerequisite courses or on CGPA. In each case, the set of criteria used to differentiate between claimants can be collapsed into a ranking over the claimants involved, with higher-ranked people said to be more eligible for the respective object.

The kidney exchange market, while more complicated in theory and practice, also essentially operates on the notion of priority. Because kidneys are generally more scarce than patients needing them, there is often a lengthy waiting list. A kidney that becomes available goes preferentially to patients who are higher on the list. Thus priorities can also take the form of a queue, where allocations are made on a first-come-first-served basis.

Each problem is different and has its own particular features. In its abstract form, however, an allocation problem can be represented by the following information:

1. There are some indivisible[1] objects (houses, offices, courses, kidneys, college seats, etc.) that are to be assigned.

2. There are some individuals (we call them agents) who claim these objects.

3. Each agent may receive at most one object. This is not a necessary requirement of all models, but it does apply to a number of settings, including the ones we wish to consider in this thesis.

4. Agents have a clear idea of what they want, represented by their preferences. These preferences could be simply over what they might receive. Or it might also matter to them what other agents receive. In cases where many agents may be assigned together, such as students in a school or workers in a team, they may also care about who shares the task or space with them. In this thesis, we will restrict attention to cases where agents are 'selfish', in that they only care about their own assignments. Preferences are represented as rankings over objects.

5. The objects in turn are imbued with priorities. Priorities are also rankings, this time over agents. They are meant to capture the relative eligibility of each agent for that object. Each

[1] Indivisibility is important, to distinguish these models from cases where objects can be divided among agents. A usual example of the latter problem is that of dividing a cake, or splitting a sum of money. Such problems are beyond the scope of this thesis. In our case, while a college may have many seats to offer, we consider each seat to be indivisible.

object may have its own ranking over agents. For example, a manager wishing to assign tasks to employees may wish to give priority to engineers for engineering positions, lawyers for legal positions and so on. Or objects may have the same ranking over agents. For example, a school with multiple seats may evaluate candidates based on grades in some common examination. In this case students with higher grades are more eligible for any of the seats in the school.

1.3 What We Look For in a Good Rule

A rule takes all this information and prescribes an allocation, i.e., who gets what. It is always possible to make assignments in some arbitrary and ad hoc fashion. Except in rare cases, however, these allocations will be unsatisfactory - there may be cause for complaints, or there may be waste. Thus the designer's quest is to come up with 'good' solutions, formulated by way of those that satisfy normative criteria that are desirable for individuals as well as the society as a whole. We now discuss some of these criteria.

For instance, we may be concerned about the fairness of an allocation. An allocation is deemed unfair if an agent receives an object even though he or she is less eligible for it than some other agent who also desires it. For example, a student with a higher GPA may feel legitimately aggrieved if a seat in a course he or she desires is instead given to some other student with a lower GPA, especially when the explicit criteria for assigning seats is GPA. We would like to avoid such situations wherever possible.

We may also be concerned about the efficiency of the overall allocation, i.e., minimising waste. An allocation is efficient if there is no way for us to improve it, in the sense that there is no other allocation that gives at least one agent something that he desires more, while at the same time not making any other agent worse off. It would be inefficient to have empty seats in a school while there are students who seek admission there. Granting even one of these students admission makes him or her better off, without making anyone else worse off. Efficiency is desirable to the extent that it seeks to optimises the use of scarce resources.

Another important consideration is that of strategy. In all the allocation problems above, an agent's 'true' preferences are unknown, and must be elicited by the rule designer or implementer. A shrewd agent may be tempted to game the system if it advantageous to do so. In particular, if lying about one's preferences results in a better assignment, there is no reason to believe that an agent will not falsify revealed information. If agents were encouraged to game the system in this way, it could make a mess of the designer's aims.

Thus it is desirable to provide agents with incentives to truthfully reveal their preferences. This notion has also been proposed on the basis of informational simplicity. If it is always the case that truthfully revealing preferences makes an agent as well off as lying, then he or she gains nothing by spending time pondering what others may or may not do, and can be concerned with simply evaluating and reporting his or her own preferences. A rule satisfying this property is said to be strategy-proof.

Other considerations may have an egalitarian agenda. One may seek to limit the undue influence that one individual may have on another's assignment. In particular, we may wish to avoid situations where one agent 'bosses' another, in that the former can determine what the latter gets, without any effect on her own assignment. It would be undesirable, say, for a student picking a course to dictate to other students which courses they should or shouldn't pick for themselves. A rule that prevents this occurrence is called non-bossy.

Or, from the point of view of the objects, one may wish to treat all objects the same way, without discriminating between them. For example, a centralised admissions process may wish to treat all schools equally, without being biased for or against any of them. Such a rule would avoid referencing any of the schools by name in its procedure. A rule satisfying this property is 'neutral'.

There are several other properties of varying desirability. The challenge for a social planner faced with an allocation problem is to pick a set of criteria that he or she considers important to that context, and to find a solution to the problem that satisfies them.

1.4 Summary of Chapters

In the first two chapters in this thesis we make some additional restrictions on the nature of the objects we wish to allocate. These have to do with their capacity constraints.

In many cases it is natural to assume that objects have a maximum capacity constraint. For instance, a school may have limited seats to offer, such that it cannot admit any more students than that. It is also possible that in some situations objects have a minimum capacity constraint. A college wishing to offer a course to its students may require that at least ten students sign up for it, say. It might be infeasible for the course to be offered to any fewer students than that - because of the costs involved or limited facilities.

Capacity constraints, especially minimum capacities, have an effect on the existence of good solutions to allocation problems. The range of feasible solutions may decrease substantially. Moreover, if capacity constraints are incompatible, no feasible solutions may exist at all.

Our central theme in this thesis is the study of allocation problems with exact capacity constraints. What we assume is that objects not only have maximum and minimum capacities, but also that they coincide. Thus each object has an *exact capacity constraint.* An object may be assigned to only exactly that number of agents. We will also assume that the exact capacity constraint is the same for all objects so that they are all perfectly substitutable with each other.

We will save further discussion of the restrictions to the description of the chapters themselves, to which we now turn.

1.4.1 Efficient Pairwise Allocation via Priority Trading

In this chapter we restrict our attention to cases in which each heterogenous object must be assigned to exactly two agents if it is to be assigned at all. To differentiate this from the classical single-unit object allocation problem, we will refer to these objects as 'projects', and the model as one of pairwise project allocation.

Such partnerships problems are common occurrences. Police precincts usually send officers out in pairs. Airlines allocate flight routes to pilots and co-pilots. Teachers often assign projects to pairs of students. Many hostels and dormitories allocate rooms to pairs of roommates. Managers may have tasks that require exactly two workers to perform them.

Individuals are concerned only about the assignment they receive. So the police officer will only care about his or her beat, the student will only care about which project she is assigned, and the hosteller is only interested in the room he is assigned. In particular, individuals will not care about the identity of the partner who is also assigned that project, even though there must be one. We will also assume that preferences are strict, in that there are no two projects to which any individual is indifferent.

An allocation in this context will be the assignment of projects to individuals, such that each project is assigned either to nobody or to exactly two people. Conceptually, it is as if each project will have two copies, such that if one copy is assigned to some agent, then the other copy *must* be assigned to some other agent as well.

It should be pointed out at this stage that this problem differs from the classical house allocation problem in another significant way. There are more projects available than can be feasibly assigned to agents. Thus there is also an inbuilt project selection problem, where m projects need to be selected from a set of size at least $m+1$. If, instead, the number of projects were less than or equal to what could be feasibly assigned to agents, then many of the classical procedures in the literature would be appropriate to use in this context.

What we seek in this context is a rule or a class of rules. A rule is a prescription of an allocation (or a procedure to determine an allocation) for any configuration of individual preferences. We will formulate such a rule (we call it the partner trading (PT) rule). The PT rule can be considered to be an adaptation of the famous top trading cycles (TTC) procedure to our environment.

The partner trading (PT) rule proceeds in stages. The information at each stage is captured by a 'state' vector, which essentially is a partial allocation that keeps track of agents' assignments up to that stage. This state information serves as input for functions that specify the relative order of agents who get to choose subsequently. These functions are called a 'primary list', a 'preferred partner list', a 'proposal' and a 'partner inheritance table', respectively. Collectively we call them the 'entitlement' for that state.

Agents may receive their assignments in one of two ways - via a proposal or by trading.

Note that for feasibility the total number of different projects that can be assigned in any feasible allocation is some integer m. At each stage in which the number of different projects assigned is less than m (we call this an interim state), the primary list will identify agents who can guarantee their assignments by picking distinct projects from the ones that remain. The preferred partner list will identify the agents who can join these agents in their choices, should they choose the same project. The primary list is ordered, in that there is a ranking that governs the sequence in which they may exercise their choice.

We will develop the notion of a proposal to capture how assignments are made in interim states. Based on the primary list and the preferred partner list, a pairing is an ordered pair of agents, such that the first agent in the pair is from the primary list, while the second agent in the pair is either the corresponding preferred partner, or is a distinct agent from the primary list. The set of all possible pairings is ordered by a relation which we call a proposal. The ordering determines the sequence in which pairings are considered. Given a proposal, we start with the first pairing and check if the top-ranked project for each agent in the pairing (from among the projects available at this stage) is the same. If so, we deem this an acceptable pairing and assign both agents that project. If not, we go on to the next pairing determined by the ordering. In this way, we evaluate all the pairings until we find the first acceptable pairing. If there is no acceptable pairing, we assign all agents in the primary list their top-ranked projects (which must be distinct, since otherwise we would have found an acceptable pairing). We then update the state, which might result in an interim state, a trading state, or a terminal state. If it is an interim state, we repeat the above step, with the primary list, preferred partner list, and proposal corresponding to this state.

For trading states, which occur when m distinct projects have been assigned to at least one agent by the steps above, we use the TTC procedure. For each project that has already been assigned, but has only been assigned to one agent thus far, there is a designated 'preferred partner' who can

be thought of as owning the rights to partner this agent in her assignment. The preferred partner's identity is determined by a partner inheritance table, which lists the sequence in which unassigned agents may become the partner for an unpartnered agent. The first agent in the corresponding partner inheritance path is the preferred partner, and is yet to receive her assignment. She could always guarantee being a partner of this agent for herself. But she could also put the partner status up for trade. If there is another similarly placed agent who is the preferred partner of some other agent, and they each desire each other's partner's projects, they can swap. This also applies to cycles of more than two agents. Any cycles are honoured, and the agents removed along with their assignments. If there are any unpartnered agents remaining, their preferred partners are determined by the highest remaining agent in the corresponding inheritance path. In each such case, at least one cycle must emerge, and thus in a finite number of steps, all remaining agents receive their assignments. We update the state (which is now a terminal state) and stop. Note that once we get to a trading state, only the TTC procedure applies until all agents receive their assignments.

The PT rule is then a collection of the above sub-procedures. For any combination of agent preferences, the rule proceeds by starting with the null state and performing either a proposal evaluation or a TTC evaluation, depending on the nature of the current state, and making some assignments. There will always be at least one assignment made in every stage. In subsequent stages, the assignments made thus far will become inputs for the selection of agents for the sub-procedures above. The entire procedure terminates when all agents have received their assignment.

We will impose some desirable properties on the rule. More precisely, the properties are strategy-proofness, limited influence, unanimity and neutrality. Our main result will be a characterisation, in that we will show that the class of PPT rules we identify are exactly those rules that satisfy the combination of these properties.

1.4.2 Fair Allocation with Exact Capacity Constraints

In this paper we model a situation where a college wishes to offer a selection of courses to its students and requires that each student sign up for exactly one of these courses. In turn, each course has a minimum and maximum capacity and can admit students only within those capacities. For the sake of this paper we make the further assumption that the minimum and maximum capacities for all courses are equal, and the same. That is, for example, each course may admit only exactly twenty-five students, say.

In contrast to other papers with minimum capacity constraints, we allow for the fact that a course need not be assigned at all. There are more courses available than may be feasibly assigned together, so in effect the college must determine the solution to a two-part problem: not only must the college decide which selection of courses from the total will be offered, but also which students will be assigned which course.

Course allocations are made on the basis of preference and priorities. Each student has as his or her private information a strict ranking over the available courses, which we call a preference ordering, or simply a preference. This information must be elicited by the college and in general we may wish to award students their preferred courses, as far as possible. On the other hand, each course has a strict ranking over the students, which we call a priority. In contrast to preferences, this priority information is commonly known and fixed, and may be determined by transparent criteria such as GPA, prerequisites, and so on. Priority information captures which students are

more eligible for which courses.

So an allocation problem for a college is a collection of students, courses, capacities, course priorities and student preferences. It must use this information to produce a feasible allocation - one in which each student is assigned a course, and every course is assigned to its exact capacity (or to no one). In effect, since the first four are commonly known, the problem becomes one of producing a feasible allocation for any combination of elicited student preferences.

The college wishes that the allocation should satisfy some desirable properties. The first class of properties has to do with fairness. An allocation is deemed unfair for some student if there is a course that she prefers and there is another less eligible student who is assigned that course instead of her. The former student then can be said to have a case of justifiable envy towards the latter. A fair process will avoid this possibility.

The second class of properties has to do with efficiency. An efficient process eliminates waste. In particular, an allocation is inefficient if there is another allocation in which each student receives a course she likes as much, and some student receives a course she strictly prefers. In this case we say that the latter allocation 'Pareto dominates' the former.

However, it has been well documented in the literature that fairness and efficiency are incompatible in the most general environments. We demonstrate by example that this incompatibility persists even in our model with exact capacity constraints.

In the model without minimum constraints, a weaker version of efficiency, called constrained efficiency, is compatible with fairness. A constrained efficient allocation is efficient *within the set* of feasible and fair assignments.[2] The classical Gale-Shapley Deferred Acceptance Rule for instance is simultaneously fair and constrained efficient.

However, the DA rule is not directly applicable to models with minimum constraints without requiring extra information. We shall discuss the nature of this extra information in more detail and why it is necessary. However, we can find a modified DA rule that is simultaneously fair and constrained efficient. This rule requires no extra information. We call this rule the DA rule with improvements (DAI).

The DAI rule works as follows. We first exogenously select a set of courses that can exactly accommodate all students, and run the conventional DA rule on the restricted environment with only these courses. By the properties of the DA rule, such an allocation will be fair as well as internally constrained efficient [3]. However, our initial selection of courses is arbitrary. There is the possibility that there exists some other allocation, with a *different* selection of courses, that Pareto dominates this one. Our main contribution is to provide a procedure to identify these Pareto improvements whenever they exist. Moreover, we show how we can carry out these efficiency improvements in a manner that preserves the fairness of the original allocation. The composite process is therefore fair and constrained efficient. We also show that the procedure can reach a constrained efficient allocation in just one complete round.

Next we come to questions of strategy. A third property that we would like our rule to satisfy is non-manipulability on the part of students. Since student preferences are private information, the college would like to ensure that students are incentivised to report their true preferences. We consider the strategic aspects of the rule. The DAI rule is not strategy-proof. Thus it is not a

[2] If some allocation Pareto dominates a constrained efficient allocation, then it must either be infeasible or unfair to some student.

[3] An allocation is internally constrained efficient if it is constrained efficient *for that particular selection* of courses. It need not be constrained efficient in general.

complete generalisation of the DA procedure to the environment with exact capacity constraints.

1.4.3 Sufficient Conditions for Weak Group-Strategy-Proofness

In this paper, we concentrate on the fact that the key feature of preferences in all allocation models is that they are private to each individual. The designer or implementer of the solution to the allocation problems must elicit this information from individuals before making assignments. Individuals may reveal any preferences at all.

It is assumed that individuals may seek to game the system if it is to their advantage. If falsely revealing preferences gives an agent an object she prefers to what she might get if she instead truthfully revealed her preferences, then there is no reason to believe that a rational agent would not do so. A desirable property that a designer would like the allocation rule to satisfy is immunity from such undue gain for deviating agents. In particular, a strategy-proof rule ensures that it is a dominant strategy for every individual to truthfully reveal her preferences.

There are many allocation problems where even individually strategy-proof rules do not exist. Demanding more from such rules is futile. However, in problems where strategy-proof rules exist, it is natural to ask whether the immunity from manipulation can be extended to coordinated actions by groups of agents as well. This property is called group-strategy-proofness.

Group-strategy-proofness comes in two forms. Both look at the consequences for a group of agents who deviate by reporting different preferences. The standard form requires that it should not be the case that all deviating agents are as well off as before in terms of their original preferences, and some agent is strictly better off. A weaker form requires that it should not be the case that all deviating agents are strictly better off as a result. It is clear to see that the standard form implies the weak form while the converse is not true in general.

There is a clear connection between individual and group-strategy-proofness. In a wide class of allocation problems, group-strategy-proofness can be shown to be equivalent to a combination of strategy-proofness and a property called non-bossiness. Non-bossiness preempts situations where one agent can be bossy with another by affecting her assignment without changing her own.

Several important strategy-proof rules in the literature are also group-strategy-proof. The list includes inheritance rules (Pápai (2000)), other generalisations of top trading cycles rules (Abdulkadiroğlu and Sönmez (1999), Pycia and Ünver (2013)) and sequential and serial dictatorships (Svensson (1999), Pápai (2001), Hatfield (2009)). These rules are non-bossy and are also weakly group-strategy-proof by default.

However, there are also important strategy-proof rules that are weakly group-strategy-proof but not group-strategy-proof. The famous Gale-Shapley Deferred Acceptance (DA) rule (Gale and Shapley (1962)) is a case in point. Kojima (2010) shows that it is impossible for a stable rule to be non-bossy. Since the DA rule always produces a stable outcome, it is bossy. Thus it cannot be group-strategy-proof. Yet Hatfield and Kojima (2009) show that under general conditions (including the ones that apply in our model) the DA rule is weakly group-strategy-proof.

There is a non-trivial distinction between the two properties. It is useful therefore to ask the question: what makes a strategy-proof rule weakly group-strategy-proof but not group-strategy-proof? Barberà et al. (2010) consider a related question in the context of domains of preferences. In particular, they provide conditions on domains guaranteeing that for all rules defined on them, individual and weak group-strategy-proofness become equivalent.

Non-bossiness is too strong a condition and is not necessary for weak group-strategy-proofness.

We identify two fairly weak properties, which we call partial weak Maskin Monotonicity and weak non-bossiness respectively, that are sufficient to guarantee that a strategy-proof rule is also weakly group-strategy-proof.

We show the robustness of these conditions. That is, we can find examples of rules that violate one or other of these properties in turn. We also show that these properties are 'weak enough', in that there are rules that satisfy these properties (and are thus weakly group-strategy-proof) but are not group-strategy-proof.

2

Efficient Pairwise Allocation via Priority Trading

2.1 Introduction

In this chapter we restrict our attention to cases in which each heterogenous object must be assigned to exactly two agents if it is to be assigned at all. To differentiate this from the classical single-unit object allocation problem, we will refer to these objects as 'projects', and the model as one of pairwise project allocation.

Such partnerships problems are common occurrences. Police precincts usually send officers out in pairs. Airlines allocate flight routes to pilots and co-pilots. Teachers often assign projects to pairs of students. Many hostels and dormitories allocate rooms to pairs of roommates. Managers may have tasks that require exactly two workers to perform them.

Individuals are concerned only about the assignment they receive. So the police officer will only care about his or her beat, the student will only care about which project she is assigned, and the hosteller is only interested in the room he is assigned. In particular, individuals will not care about the identity of the partner who is also assigned that project, even though there must be one. We will also assume that preferences are strict, in that there are no two projects to which any individual is indifferent.

An allocation in this context will be the assignment of projects to individuals, such that each project is assigned either to nobody or to exactly two people. Conceptually, it is as if each project will have two copies, such that if one copy is assigned to some agent, then the other copy *must* be assigned to some other agent as well.

It should be pointed out at this stage that this problem differs from the classical house allocation problem in another significant way. There are more projects available than can be feasibly assigned to agents. Thus there is also an inbuilt project selection problem, where m projects need to be selected from a set of size at least $m+1$. If, instead, the number of projects were less than or equal to what could be feasibly assigned to agents, then many of the classical procedures in the literature would be appropriate to use in this context.

What we seek in this context is a rule or a class of rules. A rule is a prescription of an allocation (or a procedure to determine an allocation) for any configuration of individual preferences. We will formulate such a rule (we call it the partner trading (PT)[1] rule). We will impose some desirable properties on the rule. Our main result will be a characterisation, in that we will show that the class of PT rules we identify are exactly those rules that satisfy the combination of properties[2] we consider. The properties as well as the rule will be discussed at length in subsequent sections. The PT rule can be considered to be an adaptation of the famous top trading cycles (TTC) procedure to our environment.

The paper is organised as follows. Section 2.2 presents a review of the literature on object allocation and recent extensions to project allocation. Section 2.3 is an informal discussion of the

[1]The PT rule makes assignments via an iterative process. In each stage, there are two ways in which an agent may receive his or her assignment - either by finding an acceptable partner or by trading preferred partner status with each other. As we shall see, some of these ideas are familiar in the literature and some are novel to this rule.

[2]More precisely, the properties are strategy-proofness, limited influence, unanimity and neutrality.

model. We will show how some of the common approaches used to assign objects efficiently fail in our context of pairwise project allocation. We will also discuss the workings of the PPT rule.

Next we get to the formal part of the paper. Section 2.4 presents the notation and main definitions that we use throughout the paper. Section 2.5 contains the formal specification of the PT rule. Section 2.6 presents a detailed example of its working. Section 2.7 discusses the features of the rule and presents some special cases of PPT rules. Section 2.8 discusses the properties ('axioms') that we wish for our rule to satisfy. Section 2.9 presents our main result. Section 2.10 concludes and discusses possible extensions of this model. The proofs are relegated to the appendix.

2.2 Literature Review

The typical allocation problem deals with assigning indivisible objects. It is generally assumed that objects are such that each may be allocated to only one agent. The classical studies in the literature designate these objects as houses (Shapley and Scarf (1974)), and the general framework is called house allocation. Shapley and Scarf (1974) also introduce Gale's top trading cycles (TTC) procedure, which is an iterated procedure in which agents initially own objects and trade with each other. Trading is done via a 'pointing' mechanism that represents favourable trades via cycles. The TTC procedure will be discussed at length later, as it lies at the centre of a number of papers on efficient object allocation. The TTC procedure is also a feature of our rule.

The TTC is a robust rule. Roth and Postlewaite (1977) show how the TTC allocation coincides with the unique core allocation when preferences are strict. The TTC is also strategy-proof (Roth (1982b)). Moreover, the TTC solution is the only Pareto-efficient, individually rational and strategy-proof rule (Ma (1994), Svensson (1999)). It is also group-strategy-proof (Bird (1984)).

Allocation mechanisms that are strategy-proof and Pareto efficient have been well-covered in the literature. Pápai (2000) characterises the set of Pareto efficient, group-strategy-proof and reallocation-proof mechanisms, and derives a wide class of functions called hierarchical exchange rules. Hierarchical exchange rules are an extension of the TTC procedure, via a generalisation of the initial endowment structure and by defining inheritance rules for unassigned objects. Also, Pycia and Ünver (2013) independently characterise the class of group-strategy-proof and Pareto efficient rules in this context.

Another generalisation of the TTC procedure can be found in Abdulkadiroğlu and Sönmez (1999) who consider a mixed model of house allocation and a housing market. They provide a strategy-proof, Pareto efficient and individually rational mechanism. The key feature of their model is the presence of exogenous property rights.

Under the assumption of neutrality, Svensson (1999) shows that the only strategy-proof and non-bossy mechanism in this model is the serial dictatorship. The serial dictatorship is one in which there is an exogenous fixed order of agents who get to successively pick their best options from the set available to them after previous choices have been made. Rhee (2011) extends this result to the case where each object must be assigned to a couple. Couples are ranked according to a hierarchy, and one agent in each couple serially selects an object. The other agent in the couple shares this object. The serial dictatorship is Pareto efficient in both cases. Pápai (2000) shows how the serial dictatorship can be embedded in a hierarchical exchange rule based on the TTC.

When we consider the case of multiple agents sharing an object, the extreme case is when one object must be assigned to all agents. Here, the seminal result in the literature is of course the Gibbard-Satterthwaite Theorem. Independently proposed by Gibbard (1973) and Satterthwaite

(1975), it was shown that the only strategy-proof and onto rule is the dictatorial one. This result has been replicated in numerous instances (see, for example, Sen (2001), Barberà (1983)), and is also derived as a consequence of the serial dictatorship result in Svensson (1999). As in that paper, this applies to the public goods case as well. It also applies to voting rules.

In some models, agents may receive more than one object. Pápai (2001) characterises the sequential dictatorship as the only rule that is strategy-proof, non-bossy and satisfies citizen sovereignty. Strengthening non-bossiness to total non-bossiness yields the serial dictatorship in this model. Hatfield (2009) considers a model where each agent has a capacity that must be filled exactly. He shows under a certain restriction on preferences that the only strategy-proof, Pareto optimal and nonbossy rule is a sequential dictatorship. Furthermore, he shows that the only strategy-proof, Pareto optimal, nonbossy, and neutral mechanisms are serial dictatorships.

The model in Hatfield (2009) is essentially a model of exact capacity constraints on the part of agents. In the context of capacity constraints for objects, Hylland and Zeckhauser (1979) present a model where objects have some maximum capacity constraint, and propose a strategy-proof and efficient rule. Fragiadakis et al. (2012) and Ehlers et al. (2011) also deal with capacity constraints. In particular, the objects in their models have minimum as well as maximum constraints. The former paper is relevant to our model, in that the TTC extensions they produce are related to our rule. However, they do not provide a characterisation of efficient and group-strategy-proof rules in the context of minimum and maximum capacities. The latter paper is concerned with fairness as opposed to efficiency.

2.3 An Informal Discussion

In this section we discuss some alternative approaches to pairwise project allocation. In particular, we show how the rules commonly used in object allocation need to be modified or extended to fit this context. After that we discuss the key features of our rule.

2.3.1 Other Approaches

Why is pairwise project allocation different from object allocation? Why can we not use the rules that have been shown to work in the single-unit case here as well? In what follows we attempt to demonstrate the reasons why.

Consider, as a natural starting point, the serial dictatorship. This is also known as the serial priority rule. The rule works as follows: there is an exogenous and fixed ordering of agents (agent I comes before agent J who comes before agent K, and so on). Each agent gets to pick his or her top-ranked object in turn. So the first agent is guaranteed to receive his or her top-ranked object in all cases. The second agent is faced with the objects that are left over after the first agent has made his or her selection. The second agent is then guaranteed to receive her top-ranked object from all the ones that remain. The only object she cannot receive is the one that has been selected by the first agent. Every subsequent agent in the ordering picks from the set of objects left behind after earlier agents have made their choice.

The serial priority rule has been characterised by Svensson (1999) as the only rule that is strategy-proof, non-bossy and neutral. Rhee (2011) extends this result to the case (similar to ours) where agents are organised in pairs. The equivalent rule to serial priority in this context works as follows: there is an ex ante separation of agents into pairs, an exogenous ranking of pairs, and a

fixed selection of one agent from each pair. These selected agents are ordered according to how the pairs are ordered. The rule applies the serial priority method to these agents, as in the single object case. The other agent in the pair is automatically assigned the object his or her partner has selected. Rhee shows that that this extended rule is also characterised by the axioms of strategy-proofness, non-bossiness and neutrality.

However, while the original rule is also Pareto efficient, the extended rule is no longer so. To see this, note that the non-selecting partners in each pair in the extended rule have no say in their assignments. In particular, it may be that two non-selecting partners would actually prefer to be assigned each other's object, i.e., belong to a different pair. The rule does not allow any such profitable swaps, and is thus inefficient. Since we look for Pareto efficient rules in the pairwise model, this rule will not serve our purpose.

So consider the following thought experiment. Each project in our model can be considered to comprise two 'copies'. If one copy is assigned to some agent, the other copy must be assigned to some other agent as well. We could then label each copy as a distinct object in itself. For example, project a could be separated into a_1 and a_2. As long as we ensure that if one of them is assigned, that the other must be as well, there is no threat to feasibility from this approach.

So we could apply the original serial priority rule to this case with $2x$ 'objects', where x is the number of original projects. Agents are ordered, and would select according to their order. We would suitably restrict the set of objects available to later agents to ensure feasibility of the overall assignment in the original context of projects. This rule, as the original one was, will also be strategy-proof, non-bossy and neutral.[3] However, it is no longer characterised by those properties. That is, when we translate this model back to the original pairwise project allocation setting, we can find other rules as well that satisfy those properties, that are not serial or even sequential priority rules. So if we seek a full characterisation of this class of rules, we have to look beyond purely serial or sequential priority.

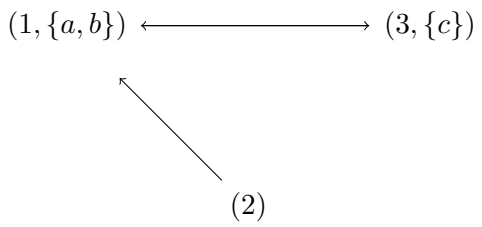
So let us consider the famous top trading cycles (TTC) rule attributed to David Gale (see Shapley and Scarf (1974)). In brief, the extended TTC works as follows:

Each object is initially owned by one agent, who brings it to the market for trade[4]. Some agents may initially own more than one object, while others may own none at all. The procedure works in stages. In any stage, each agent who is yet to receive an assignment points to the owner of the object she most prefers from the ones that are available. A top trading cycle is made up of agents who successively point to the next agent, with the last agent pointing to the first. A cycle can be a singleton, such that an agent points to herself (she owns the object she most prefers.) Since there is a finite number of agents, at every stage there must always be a cycle. Agents in a cycle trade their objects along the cycle until they receive the object they desire. This becomes their assignment and such agents leave the market along with those objects. If there are still agents and objects left unassigned, the procedure repeats in the reduced market. If preferences are strict, then given an initial ownership, the resulting allocation is unique.

The TTC rule is illustrated by an example. Suppose there are three agents $(1, 2, 3)$ and three objects (a, b, c). Suppose agent 1 initially owns a, b and agent 3 initially owns c. Agent 1 desires c, while agents 2 and 3 desire b. The TTC procedure would look as follows:

[3] As an added bonus, it will also be Pareto efficient.

[4] Objects that an agent initially owns form a part of his or her 'endowment'



Agent 1 'points to' agent 3 who owns c, and agents 2 and 3 in turn point to agent 1 who owns b. The cycle in this stage is between agents 1 and 3, who consequently trade those objects. The TTC would assign b to agent 3 and c to agent 1.

As discussed earlier, TTC rules and their generalisations to inheritance rules (Pápai (2000)) are indeed group-strategy-proof and Pareto efficient. An inheritance rule in the above example would also specify how agent 2 'inherits' the remaining object a. The TTC procedure in the second stage would just be agent 2 pointing to herself, and a would become her assignment.

Sequential and serial priority rules form a sub-class of inheritance rules. To see this, note that a serial priority effectively grants initial ownership of all objects to some agent. This agent can always pick his or her top-ranked object from among them. Additionally, the rule specifies that all objects left over after his or her selection are inherited by some other fixed agent. This agent can always pick her top-ranked object from among them. The remaining objects are inherited by some fixed third agent, and so on.

However, when faced with projects, the idea of initial ownership needs to be generalised as well. Since there are two copies, each project could be initially collectively owned by a pair of agents. Different projects may be owned by different pairs of agents. But note that pairwise project allocation implicitly involves not just the assignment of projects, but also the question of how to select the assigned projects in the first place. How do we select the projects? What makes one pair's endowment superior to another's? When do we decide that this project can be traded while another one cannot? Unless the number of projects is exactly equal to capacity, there will always be this selection problem. Thus a specification of a TTC rule will have to account for these possibilities as well.

There is another conceptual problem with naively using the TTC rule in this setting. Note that the success of the TTC procedure rests on the existence of a distinct cycle in every stage of pointing. We can then unambiguously trade objects along the cycle until the agents are satisfied. The cycle itself arises because each agent desiring an object can unambiguously point to a single owner of that object. But how would that work here, where objects have two copies? If an agent desires an object owned by two other agents, who are both available, then to whom should she point? There exists the possibility now of multiple overlapping cycles. This problem has been addressed in Hakimov and Kesten (2014) and Morrill (2014), where agents owning different copies of the same object are ranked by their respective priorities under that object.

We do indeed base our rule around the TTC, but we take a different approach. As a result, we see that the TTC rule cannot be directly used in this framework. Our rule is therefore more complicated than the simple TTC rule itself.

2.3.2 The Partner Trading Rule

The partner trading (PT) rule proceeds in stages. The information at each stage is captured by a 'state' vector, which essentially is a partial allocation that keeps track of agents' assignments up to that stage. This state information serves as input for functions that specify the relative order of agents who get to choose subsequently. These functions are called a 'primary list', a 'preferred partner list', a 'proposal' and a 'partner inheritance table', respectively. Collectively we call them the 'entitlement' for that state.

Agents may receive their assignments in one of two ways - via a proposal or by trading.

Note that for feasibility the total number of different projects that can be assigned in any feasible allocation is some integer m. At each stage in which the number of different projects assigned is less than m (we call this an interim state), the primary list will identify agents who can guarantee their assignments by picking distinct projects from the ones that remain. The preferred partner list will identify the agents who can join these agents in their choices, should they choose the same project. The primary list is ordered, in that there is a ranking that governs the sequence in which they may exercise their choice.

We will develop the notion of a proposal to capture how assignments are made in interim states. Based on the primary list and the preferred partner list, a pairing is an ordered pair of agents, such that the first agent in the pair is from the primary list, while the second agent in the pair is either the corresponding preferred partner, or is a distinct agent from the primary list. The set of all possible pairings is ordered by a relation which we call a proposal. The ordering determines the sequence in which pairings are considered. The ordering is based on the rank of the first agent in each pairing in the primary list. There are three criteria that govern the ordering of pairs. The first condition is that, for every primary list agent, the pairing with her corresponding preferred partner should be ranked above any other pairing where she is the first agent. The second criterion is that for any two primary list agents, the pairing with the higher ranked agent as the first agent should be ranked above the reciprocal pairing. And the third condition requires that, for a given primary list agent, the pairing of higher ranked agents in the primary list as the second agent should be ranked higher than pairings with lower ranked agents in the primary list as the second agent. These criteria produce a complete ranking over all pairings, which we call a proposal.

Given a proposal, we start with the first pairing and check if the top-ranked project for each agent in the pairing (from among the projects available at this stage) is the same. If so, we deem this an acceptable pairing and assign both agents that project. If not, we go on to the next pairing determined by the ordering. In this way, we evaluate all the pairings until we find the first acceptable pairing. If there is no acceptable pairing, we assign all agents in the primary list their top-ranked projects (which must be distinct, since otherwise we would have found an acceptable pairing). We then update the state, which might result in an interim state, a trading state, or a terminal state. If it is an interim state, we repeat the above step, with the primary list, preferred partner list, and proposal corresponding to this state.

For trading states, which occur when m distinct projects have been assigned to at least one agent by the steps above, we use the TTC procedure. For each project that has already been assigned, but has only been assigned to one agent thus far, there is a designated 'preferred partner' who can be thought of as owning the rights to partner this agent in her assignment. The preferred partner's identity is determined by a partner inheritance table, which lists the sequence in which unassigned agents may become the partner for an unpartnered agent. The first agent in the corresponding partner inheritance path is the preferred partner, and is yet to receive her assignment. She could

always guarantee being a partner of this agent for herself. But she could also put the partner status up for trade. If there is another similarly placed agent who is the preferred partner of some other agent, and they each desire each other's partner's projects, they can swap. This also applies to cycles of more than two agents. Any cycles are honoured, and the agents removed along with their assignments. If there are any unpartnered agents remaining, their preferred partners are determined by the highest remaining agent in the corresponding inheritance path. In each such case, at least one cycle must emerge, and thus in a finite number of steps, all remaining agents receive their assignments. We update the state (which is now a terminal state) and stop. Note that once we get to a trading state, only the TTC procedure applies until all agents receive their assignments.

The PT rule is then a collection of the above sub-procedures. For any combination of agent preferences, the rule proceeds by starting with the null state and performing either a proposal evaluation or a TTC evaluation, depending on the nature of the current state, and making some assignments. There will always be at least one assignment made in every stage. In subsequent stages, the assignments made thus far will become inputs for the selection of agents for the sub-procedures above. The entire procedure terminates when all agents have received their assignment.

2.3.3 Criteria for Allocation Rules

In this paper we are concerned with the following four criteria for rules. The first is strategy-proofness, which ensures that it is always a dominant strategy for every agent to truthfully report his or her preferences. The second is a composite criterion which we call 'limited influence'. The first part of this criterion is familiar in the literature as the non-bossiness condition. Non-bossiness stipulates that an agent may not affect another agent's assignment via a change in reported preferences, if she does not change her assignment as well. The second part is new to this paper but similar versions have appeared in other papers as well. It seeks to limit the influence that an agent has on the assignment of certain projects via a change in reported preferences, even if her own assignment changes. We shall discuss this axiom in more detail in a later section.

The third criterion for a rule is unanimity, which states that a rule should respect the self-selection of agents into feasible pairs. If agents report preferences such that it is feasible to give every agent her top-ranked project, than a unanimous rule must do so. The final condition is neutrality, which requires the rule to treat all projects symmetrically. The main result in this paper is that a PT rule is characterised by the above four axioms.

Next, we go to the formal part of the paper, where we define all the above notions rigorously.

2.4 Notation and Definitions

The model is described below.

- There is a finite set of *agents* $\mathcal{N} = \{1, ..., i, j, , k, ..., N\}$ and a finite set of *projects* $\mathcal{Z} = \{a, b, c, d, ...\}$. We denote the 'null object' as $\emptyset$, and assume that $\emptyset \in \mathcal{Z}$. We assume that $|\mathcal{N}| = 2m$ for some integer $m \geq 2$ and that $|\mathcal{Z}| \geq m + 1$.[5]

[5]This assumption of at least two 'pairs' and three projects is equivalent to the assumption in object allocation models of at least two 'agents' and three objects.

- An *allocation* $x \in \mathcal{Z}^N$ with $x = (x_1, ..., x_N)$ is a vector that associates a project with each agent. For any agent $i \in \mathcal{N}$, $x_i \in \mathcal{Z}$ is the *assignment* of agent i in x. An allocation x is *feasible* if, for all $a \in \mathcal{Z}$, $|\{j \in \mathcal{N} : x_j = a\}| \in \{0, 2\}$. That is, an allocation is feasible if it assigns each project to exactly two agents, or to nobody. The set of all feasible allocations is given by $\mathcal{A}$.

- Preferences over assignments are strict. Formally, agent $i \in \mathcal{N}$ has *preferences*, denoted R_i, that are given by a binary relation over $\mathcal{Z}$. For any a, b, aR_ib is interpreted as 'project a is at least as good as project b for agent i under preferences R_i'. The binary relation is reflexive (for all a, aR_ia), complete (for all a, b, aR_ib or bR_ia), transitive (for all a, b, c, aR_ib and bR_ic imply aR_ic) and antisymmetric (for any a, b, aR_ib and bR_ia imply $a = b$). The associated strict relation is given by P_i, such that aP_ib if aR_ib and $a \neq b$. For any a, b, aP_ib means 'a is preferred by i to b under preferences R_i'.

- Agent preferences over allocations are selfish, in that they care only about the assignment they receive. Agents are indifferent between all allocations that give them the same assignment. An agent's preferences between two allocations that give her different assignments are governed by her preferences over the respective assignment she receives.

- A collection of preferences for all agents is called a *preference profile*, or simply a *profile*, and is denoted by $R = (R_1, ..., R_N)$. The set of all preference profiles is $\mathcal{R}$. In this model we shall usually suppress reference to $\mathcal{R}$, with the understanding that we operate on the full domain of preferences everywhere. As is the convention, we write R_{-i} for a sub-profile of preferences of all agents other than i. Similarly, for a subset of agents M, we write R_M and R_{-M} to denote the sub-profile of preferences of agents in subsets M and $\mathcal{N} \setminus M$, respectively.

- A *pairwise project allocation rule (P-PAR)* is a function $f : \mathcal{R} \to \mathcal{A}$ that maps every preference profile to a feasible allocation. For any agent i, $f_i(R)$ is the assignment she receives at preference profile R according to the rule f. Similarly, for any subset of agents M, $f_M(R)$ is the M-dimensional vector of assignments of M at R, according to f.

2.5 Partner Trading Rules

2.5.1 States

We first present some useful terminology. The notion of a state will be useful to keep track of agents' assignments as the algorithmic procedure in our rule unfolds.

Formally, a *state* is a vector $s \in \mathcal{Z}^N$ with $s = (s_1, ..., s_N)$ such that:

1. $|\{j \in \mathcal{N} : s_j = a\}| \leq 2$ for any $a \in \mathcal{Z}$
2. $|\{a \in \mathcal{Z} : s_i = a \text{ for some } i \in \mathcal{N}\}| \leq m$.

A state is essentially a partial allocation. The two conditions are necessary for feasibility. The first condition ensures that no project is associated with more than two agents for any state, and the second condition ensures that not more than m projects are associated with any state. Let

the set of all states be denoted $\mathcal{S}$[6]. It follows from the definition above that the set of feasible allocations is a subset of the set of states, i.e., $\mathcal{A} \subset \mathcal{S}$.

For a state s, we define the set of *assigned agents in* s as $N(s) = \{i \in \mathcal{N} : s_i \neq \emptyset\}$, the set of *unassigned agents in* s as $\bar{N}(s) = \{i \in \mathcal{N} : s_i = \emptyset\}$, the set of *assigned projects in* s as $Z(s) = \{a \in \mathcal{Z} : s_i = a \text{ for some } i \in \mathcal{N}\}$, the set of *partially assigned projects in* s as $\hat{Z}(s) = \{a \in \mathcal{Z} : s_i = a \text{ for exactly one } i \in \mathcal{N}\}$, the set of *unassigned projects in* s as $\bar{Z}(s) = \{a \in \mathcal{Z} : s_i \neq a \text{ for all } i \in \mathcal{N}\}$, and the *number of remaining projects in* s as $m'(s) = m - |Z(s)|$. It is clear that $\hat{Z}(s) \subseteq Z(s)$, $\mathcal{N} = N(s) \cup \bar{N}(s)$, $\mathcal{Z} = Z(s) \cup \bar{Z}(s)$ and $0 \leq m'(s) \leq m$ for any s.

If, in addition, $m'(s) > 0$, we call s an *interim state*. Note that $|\bar{N}(s)| \geq 2$ for any interim state s. It shall be convenient to call the null vector $(\emptyset, ..., \emptyset)$ the *null state*. It is clear that the null state is an interim state. For reasons that shall become clear, if $m'(s) = 0$ but $|\bar{N}(s)| > 0$, we call s a *trading state*. If $m'(s) = 0$ and $\bar{N}(s) = 0$, we call s a *terminal state*. For any preference profile, our rule will start with the null state and progressively assign projects to agents in stages until we reach a terminal state. It is easy to see that a terminal state is a feasible allocation.

For example, let $\mathcal{N} = \{1, ..., 6\}$ and let $\mathcal{Z} = \{a, b, c, d\}$. Then the null state is given by $(\emptyset, ..., \emptyset)$, an interim state could be the vector $(a, \emptyset, b, b, \emptyset, \emptyset)$, a trading state could be the vector $(a, c, b, b, \emptyset, \emptyset)$, and a terminal state could be (a, c, b, b, c, a).

Given two states s and s', we say that s *is a prior state of* s' (alternatively, s' *is a subsequent state of* s) if:

1. $s_i \neq \emptyset \implies s'_i = s_i$
2. $m'(s') < m'(s)$

The first condition requires that any agent with a non-empty assignment in s has the same assignment in s'. The second condition requires that more projects be assigned (partially assigned) in s' than in s. In essence, s' is formed by taking s and assigning additional projects to more agents.

2.5.2 Entitlements

Primary and Preferred Partner Lists

As discussed earlier, Rhee (2011) presents a pairwise allocation model and characterises a rule similar to the serial priority rule. That is, there is an ex ante separation of agents into pairs, an exogenous ranking of pairs, and a fixed selection of one agent from each pair. These selected agents are ordered according to how the pairs are ordered. The rule applies the serial priority method to these agents, as in the single object case. The other agent in the pair is automatically assigned the object his or her partner has selected. Rhee shows that that this extended rule is also characterised by the axioms of strategy-proofness, non-bossiness and neutrality.

Even in our more general case, we use this result as a starting point. Thus we retain the notion of an agent in each pair that selects a project, and the idea that the other agent in the pair can always claim the same project that her partner has chosen. The main difference is that the identity of the pairs, and the respective agents in the pair, are no longer ex ante fixed, and depend instead on the features of the rule.

[6]Strictly, $\mathcal{S}$ depends on $\mathcal{N}$, but since $\mathcal{N}$ is fixed in this model, we slightly abuse notation by suppressing reference to $\mathcal{N}$ in the specification of $\mathcal{S}$.

Given an interim state s, where there are still $m'(s)$ projects left to be assigned, we develop the notion of a primary list, which identifies $m'(s)$ unassigned agents who, if they all picked different projects as their top-ranked project, could guarantee those projects as their assignment. We also identify, for each of these agents, a preferred partner (called the preferred partner list), who can join them in their assignments by commonly declaring that project as their top-ranked project.

Formally, let s be an interim state. A *primary list* is an ordered collection of agents $\alpha(s) = (i, j, \ldots, K)$ such that:

(PL1). $i \in \bar{N}(s)$ for all $i \in \alpha(s)$

(PL2). $i \neq j$ for all $i, j \in \alpha(s)$

(PL3). $K = m'(s)$

Condition PL1 requires that each agent in the primary list be unassigned in this state. Condition PL2 requires each agent in the primary list to be distinct. And Condition PL3 requires that there be $m'(s)$ agents in the primary list.

Since the primary list is ordered, for any agent $i \in \alpha(s)$, we denote $i's$ position in $\alpha(s)$ by $\alpha_i(s)$. So we say i *precedes* j *in* $\alpha(s)$ if $\alpha_i(s) < \alpha_j(s)$. Also, we denote the first agent in $\alpha(s)$ by $\alpha^1(s)$, the second agent as $\alpha^2(s)$, and so on.

A *preferred partner list* is a correspondence $\beta(s) : \{\alpha(s)\} \to \bar{N}(s)$ such that $\beta(\alpha_i(s)) \neq i$ for all $i \in \alpha(s)$ (call this Condition SL1). Condition SL1 requires that each preferred partner be unassigned in this state, and be distinct from the corresponding primary list agent. The same agent could be the preferred partner for multiple primary list agents, and the preferred partner could be some other agent from the primary list as well.

Primary lists and preferred partner lists are also related across states. In particular, given an interim state with a primary list and a preferred partner list, when some assignments are made and we reach a subsequent state, the new primary list and preferred partner list are related to the previous lists in the following ways. Let s be an interim state and let s' be a subsequent state to s. Then:

(PL4.) For $i, j \in \alpha(s)$, if $\alpha_i(s) < \alpha_j(s)$, $i \in \bar{N}(s')$ and $j \notin \bar{N}(s')$, then $\alpha_i(s') = \alpha_i(s)$.

(PL5.) For $i, j \in \alpha(s)$ such that $\alpha_i(s) < \alpha_j(s)$, if $i, j \notin \bar{N}(s')$ and $\beta(i) \in \bar{N}(s)$, then $\alpha_{\beta(i)}(s') = \alpha_i(s)$.

Condition PL4 says that if an agent from the primary list is no longer available in a subsequent state, then all agents that were higher ranked to her in the primary list in the previous state retain their ranks in the primary list of the subsequent state. Condition PL5 says that if two agents in the primary list are no longer available in a subsequent state, then if the preferred partner of the higher ranked agent is available in the subsequent state, she enters the primary list of the subsequent state in the position that her erstwhile partner occupied in the previous state.

The intuition behind Condition PL5 is as follows. At each state, each primary list agent has a preferred partner, who can guarantee the assignment of the same project by declaring it as her top-ranked preference (explained in detail below.) If this partner should remain unassigned when the primary list agent receives an assignment, then strategy-proofness will require that she receive an assignment that is weakly better than that project. The only way to guarantee this is by including her in the primary list so that she can get her top-ranked project in a later state.

Pairings and Proposals

At an interim state, if all agents in the primary list have different top-ranked projects, they will be assigned those projects. However, how is this determined? Also, what happens when all the top-ranked projects are not distinct? How are assignments evaluated? In this subsection we develop the notion of a pairing and a proposal, which is key to making assignments in interim states.

Essentially, we develop a suitable set of pairs of agents whose top-ranked preferences we jointly evaluate. The pairs are ordered according to a well-defined ranking function. We check each pair to see if the agents in that pair have the same top-ranked preference. The first pair that agrees is assigned the corresponding project.

The pairs are constructed carefully. Formally, a *pairing* is an ordered tuple of agents (i, j) such that:

(PA1). $i \in \alpha(s)$

(PA2). $j \in \{\beta(i), \alpha(s)\}$

(PA3). $j \neq i$

Condition PA1 requires that the first agent in the pairing be from the primary list. Condition PA2 requires that the second agent in the pairing be either the corresponding preferred partner, or an agent from the primary list. Condition PA3 requires both agents in the pairing to be distinct.

Given a primary list and a preferred partner list, we can generate the set of all possible pairings. Let the set of all pairings for a given state s be denoted $\mathcal{T}(s)$. Now we develop an ordering over the set of pairings. A *proposal* $\succ_s$ is a reflexive, complete, transitive and anti-symmetric relation over pairings in $\mathcal{T}(s)$ such that:

(PR1). $(i, \beta(i)) \succ_s (i, j)$ for every $i, j \in \alpha(s)$.

(PR2). $\alpha_i(s) < \alpha_j(s) \implies (i, j) \succ_s (j, i)$ for every $i, j \in \alpha(s)$.

(PR3). $\alpha_j(s) < \alpha_k(s) \implies (i, j) \succ_s (i, k)$ for every $i, j, k \in \alpha(s)$.

A proposal is an ordering over pairings. Condition PR1 requires that, for every primary list agent, the pairing with her corresponding preferred partner should be ranked above any other pairing where she is the first agent. Condition PR2 requires that for any two primary list agents, the pairing with the higher ranked agent as the first agent should be ranked above the reciprocal pairing. And Condition PR3 requires that, for a given primary list agent, the pairing of higher ranked agents in the primary list as the second agent should be ranked higher than pairings with lower ranked agents in the primary list as the second agent.

Note that a proposal $\succ_s$ is not unique for a given $\alpha(s), \beta(\alpha(s)$, as we demonstrate by example below. Note also that $\succ$ is not defined for trading states.

For example, let s be an interim state, and let let $\bar{N}(s) = \{1, 2, 3, 4, 5, 6\}$. The table below gives a proposal for the following three cases: (1) Primary list $\{1, 2, 3\}$, preferred partner list $\{2, 1, 4\}$; (2) Primary list $\{1, 2, 3\}$, preferred partner list $\{2, 1, 4\}$; (3) Primary list $\{1, 3, 5\}$, preferred partner list $\{2, 4, 6\}$.

$\succ_s^1$	$\succ_s^2$	$\succ_s^3$
(1,2)	(1,2)	(1,2)
(1,3)	(2,1)	(1,3)
(2,1)	(3,4)	(3,4)
(2,3)	(1,3)	(3,1)
(3,4)	(3,1)	(1,5)
(3,1)	(2,3)	(3,5)
(3,2)	(3,2)	(5,6)
		(5,1)
		(5,3)

It is easy to see that the pairings above satisfy Conditions PA1-PA3, and that the proposals satisfy Conditions PR1-PR3. Also, while the primary list and preferred partner list in the first two cases are the same, the proposals are different. This demonstrates that proposals need not be unique for a given combination of primary list and preferred partner list.

Partner Inheritance

Given a trading state s, m different projects have been assigned to different agents, but some may have been assigned to only one agent so far. Feasibility requires that the copy of these projects be assigned to another agent. The partner is designed to capture the rights that unassigned agents have to be the partner of the as-yet unpartnered agents in s.

For a trading state s and an agent i such that $|\{j \in \mathcal{N} : s_j = s_i\}| = 1$, the *partner inheritance path of i in state s* is a sequence of agents $(a^1, a^2, \ldots, a^K)$ such that:

(E1). $a_j \in \bar{N}(s)$ for all j

(E2). $a_j \neq a_k$ for all $j \neq k$

(E3). $a_1 = i$

(E4). $K = |\bar{N}(s)|$

For a given trading state s, a *partner inheritance* $E(s)$ is a specification of an inheritance path for every unpartnered agent.

Condition E1 requires all agents in the inheritance path to be unassigned agents. Condition E2 requires each agent in the inheritance path to be distinct. Condition E3 requires that the first agent in the path be the agent itself. And Condition E4 requires that all unassigned agents feature in each inheritance path.

The first available agent in each inheritance path is the preferred partner for the corresponding agent. If an agent in an inheritance path becomes unavailable, then the next agent in the path becomes the preferred partner. The nature of these transfers bears some resemblance to the inheritance rules in Pápai (2000), as we shall discuss subsequently.

For example, suppose there are three unassigned agents $(1, 2, 3)$, and three partially assigned projects (a, b, c) assigned to agents $(4, 5, 6)$, respectively. Then a possible partner inheritance table looks as follows:

$E_4(s)$	$E_5(s)$	$E_6(s)$
1	1	3
2	2	1
3	3	2

It is easy to check that the table above satisfies Conditions E1-E4. Note also that for an interim state s, there are no inheritance paths defined.

Entitlements

We are now ready to define an entitlement. For every state s, define the *entitlement at state* s as $\Gamma(s) = (\alpha(s), \beta(\alpha(s)), \succ_s, E(s))$. A collection of entitlements for every state is denoted Γ.

A partner trading rule f^Γ is a specification of Γ, i.e., an entitlement for every state $s \in \mathcal{S}$, along with an iterative procedure that prescribes an allocation based on these entitlements for every preference profile. In what follows, we describe this procedure.

2.5.3 Assignments

First we shall describe how assignments are made for a particular state. If the state is an interim state, we use the Proposal Evaluation described below to make assignments. If, instead, the state is a trading state, we use the TTC Evaluation as described subsequently to make assignments.

Proposal Evaluation

Let s be an interim state. Let $\mathcal{T}(s)$ be the set of pairings for this state, ordered by a proposal $\succ_s$. Let R be a preference profile. For every agent i , let $a_i = top(R_i, \bar{Z}(s))$ be the top-ranked project for agent i among unassigned projects, according to preferences R_i.

For a pairing $t(s) \in T(s)$, we say $t(s)$ is an *acceptable pairing* if $a_{t_1(s)} = a_{t_2(s)}$, i.e., if both agents in the pairing have the same top-ranked project from among unassigned projects.

To evaluate a proposal at interim state s, we check pairings one by one, according to the ordering, to see if they are acceptable. For the first acceptable pairing we encounter, we assign each agent i in that pairing the corresponding project a_i, and make no more assignments in this round[7]. If no pairing is acceptable, we assign every agent $i \in \alpha(s)$ her corresponding top-ranked project a_i.[8]

Note that, by construction of the proposal, if no acceptable pairing is found, this means that the top-ranked projects for agents in $\alpha(s)$ must all be distinct. (This is because each combination of pairings of agents in $\alpha(s)$ is included in the proposal.)

For example, suppose a proposal is given as in the following table, where the primary list is $\{1, 2, 3\}$ and the corresponding preferred partner list is $\{2, 1, 4\}$.

[7] Two agents receive their assignment.

[8] In this case, $m'(s)$ assignments are made.

$\succ_s$
(1,2)
(1,3)
(2,1)
(2,3)
(3,4)
(3,1)
(3,2)

Consider the following three preference profiles: (1) R^1 such that the top-ranked projects of agents (1,2,3,4) are (a, b, a, d); (2) R^2 such that the top-ranked projects of (1,2,3,4) are (a, b, b, a); (3) R^3 such that the top-ranked projects of (1,2,3,4) are (a, b, c, a), respectively. Then the acceptable proposals are given in boxes in the table below. Note that in the third case, there is no acceptable proposal, and so agents (1,2,3) are assigned (a, b, c), respectively.

R^1	R^2	R^3
(1,2)	(1,2)	(1,2)
$\boxed{(1,3)}$	(1,3)	(1,3)
(2,1)	(2,1)	(2,1)
(2,3)	$\boxed{(2,3)}$	(2,3)
(3,4)	(3,4)	(3,4)
(3,1)	(3,1)	(3,1)
(3,2)	(3,2)	(3,2)

TTC Evaluation

The top trading cycles method of assigning projects applies only in trading states. Note that in trading states, m distinct objects have been assigned to at least one agent each, and there is at least one partially assigned project, i.e., a project that has been assigned to exactly one agent thus far. The partner inheritance table specifies the inheritance path for all unpartnered agents.

In any TTC round, all agents who are preferred partners of at least one unpartnered agent 'point to' the agent who is the preferred partner of the assigned agent holding her top-ranked project among the set of partially assigned projects. A cycle is a distinct sequence of agents with each agent in the sequence pointing to the next agent, and the last agent pointing to the first. A cycle can be a singleton, with an agent pointing to herself. For any cycle that we might encounter, we trade partners along the cycle until each agent is the partner of the assigned agent holding her most preferred project. This becomes their assignments. We do this for all cycles that we find. At least one agent will receive her assignment as a result.

If there are any other unpartnered agents, their preferred partner status is 'inherited' by the next unassigned agent in the corresponding inheritance path. The TTC step is repeated with the remaining agents. Since at least one agent receives her assignment in every TTC step, and partner inheritance is well-defined, the procedure is guaranteed to terminate in a finite number of steps. Also, since there are exactly as many partially unassigned projects as there are unassigned agents in a trading step, each agent will receive exactly one project as her assignment.

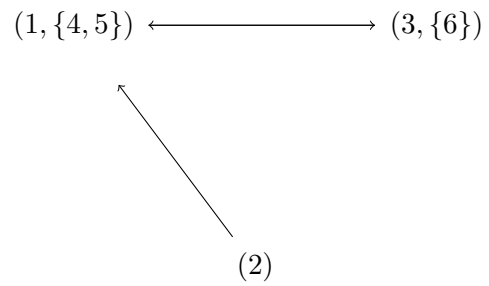
Formally, let s be a trading state and let $E(s)$ be the partner inheritance table. Let R be a preference profile. Let $N^1(s) = \bar{N}(s)$ be the set of unassigned agents in s, $Z^1(s) = \hat{Z}(s)$ be the set of partially assigned projects.

1. Pointing Step #k ($k \geq 1$): Each unassigned agent $i \in N^k(s)$ points to the agent in $N^k(s)$ who is the preferred partner of the agent who holds the project $top(R_i, Z^k(s))$.
2. A cycle is a set of agents $(i_1, i_2, ..., i_n = i_1)$ such that each agent i_j points to i_{j+1}.
3. A cycle must exist. For every cycle, agents in the cycle trade partners along the cycle until they are partners of the assigned agent who holds their most preferred project. The corresponding unassigned copy becomes their assignment. Remove these agents from $N^k(s)$ to get $N^{k+1}(s)$ and remove their assignments from $Z^k(s)$ to get $Z^{k+1}(s)$.
4. The preferred partner status for all unpartnered agents remaining is inherited by the first unassigned agent in $N^{k+1}(s)$ according to the corresponding inheritance path in $E(s)$.
5. If there are any unassigned agents, we go back to the Pointing Step #$(k+1)$. Otherwise we stop.

To continue the example given previously, suppose there are three unassigned agents $(1, 2, 3)$, and three partially assigned projects (a, b, c) assigned to agents $(4, 5, 6)$. Suppose the partner inheritance table looks as follows:

$E_4(s)$	$E_5(s)$	$E_6(s)$
1	1	3
2	2	1
3	3	2

Suppose agent 1 desires c according to R_1, while agent 2 desires a and 3 desires b according to R_2 and R_3, respectively. The TTC procedure would look as follows:



Agent 1 'points' to agent 3 who is the preferred partner of agent 6 (who is assigned c), and agents 2 and 3 in turn point at agent 1 who is the preferred partner of agents 4 and 5 (who are assigned a and b respectively. The cycle in this stage is between agents 1 and 3, who consequently trade partners. The TTC procedure at this stage would assign b to agent 3 and c to agent 1. Then the preferred partner status for agent 5 would be inherited by agent 2. In the next round of trading, there is only one cycle (agent 2 points to herself), and thus agent 2 is assigned a.

2.5.4 The Iterative Procedure

Fix a preference profile R. Then a PT rule $f^{\Gamma}(R)$ is determined by an iterative procedure with a finite number of stages. In the first stage $k = 1$ we start with the null state $s^0 = (\emptyset, ..., \emptyset)$. Each subsequent stage $k > 1$ of the procedure begins with a state s^{k-1} that captures the assignments of agents made up to and including stage $k - 1$.

At any stage k, we check if the corresponding state s^{k-1} is an interim state or a trading state. If it is an interim state, we perform the Proposal Evaluation for that state to make assignments as discussed. At least two agents are assigned a project in every interim state. We update the state s^{k-1} to reflect all assignments made up to and including stage k, giving us a new state s^k. We then proceed to stage $k+1$ of the procedure. Eventually we end up at either a terminal state or a trading state. For any terminal state, we stop. For a trading state, we perform the TTC Evaluation with partner inheritance as discussed. Note that all as-yet-unassigned agents receive their assignments in a trading state. So after this stage, we are at a terminal state.

The corresponding terminal state gives us the final allocation.

Formally:

Let R be a profile. Let Γ be given.

Stage 1

The state is s^0 (the null state). Let $\Gamma(s^0)$ be the entitlement for this state.

Proposal Step: Run the Proposal Evaluation for $T(s^0)$. For any agent i receiving an assignment at this step, update s_i^1 as this project. For all other agents j, $s_j^1 = s_j^0$. Go to the Verification Step.

Verification Step: If s^1 is a terminal state, we stop with the resulting allocation. If not, we proceed to Stage 2.

Stage k+1, $k \geq 1$

The state is s^k. Let $\Gamma(s^k)$ be the entitlement for this state. If s^k is an interim state, go to the Proposal Step. Otherwise go to the Trading Step.

Proposal Step: Run the Proposal Evaluation for $T(s^k)$. For any agent i receiving an assignment at this step, update s_i^{k+1} as this project. For all other agents j, $s_j^{k+1} = s_j^k$. Go to the Verification Step.

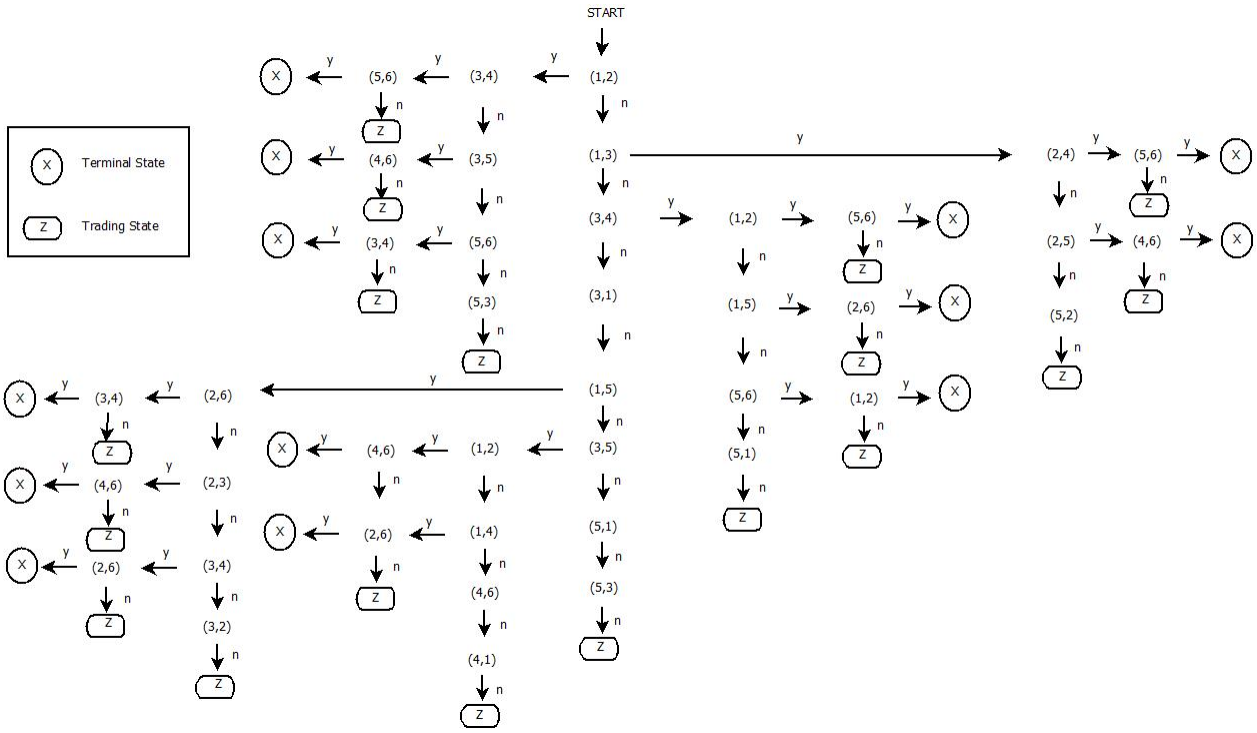
Trading Step: Run the TTC Evaluation for all agents in $\bar{N}(s^k)$. For any agent i receiving an assignment at this step, update s_i^{k+1} as this project. For all other agents j, $s_j^{k+1} = s_j^k$. Go to the Verification Step.

Verification Step: If s^{k+1} is a terminal state, we stop with the resulting allocation. If not, we proceed to Stage $k + 2$.

Note that $f^{\Gamma}(R)$ is unambiguously defined, as for every R and every agent i there is at most one stage in which i receives her assignment. The procedure is also finite as at least one agent receives her assignment at every stage.

2.6 A Detailed Example

Let $\mathcal{N} = \{1, 2, 3, 4, 5, 6\}$ and $\mathcal{Z} = \{a, b, c, d, e\}$ be a set of agents and projects, respectively. The entire proposal map is outlined in the figure below.



Let the preference profile R be given as follows:

R_1	R_2	R_3	R_4	R_5	R_6
a	c	d	b	a	d
c	d	b	c	c	c
b	b	c	a	d	a
d	a	a	d	b	b

Stage 1

The state s^0 is the null state. The primary list is $\{1, 3, 5\}$ and the preferred partner list is $\{2, 4, 1\}$.

Proposal Step: The proposal corresponding to s^0 is evaluated. The first acceptable pairing is $(1, 5)$ (the top-ranked projects in each previous pairing are different). Thus agents 1 and 5 are assigned the project a. The state $s^1 = \{a, \emptyset, \emptyset, \emptyset, a, \emptyset\}$.

Verification Step: s^1 is not a terminal stage, so we go to Stage 2.

Stage 2

The state s^1 is an interim state. The primary list is $\{2, 3\}$ and the preferred partner list is $\{3, 4\}$.

Proposal Step: The proposal corresponding to s^1 is evaluated. There are no acceptable pairings. Thus agent 2 is assigned c and agent 3 is assigned d. The state $s^2 = \{a, c, d, \emptyset, a, \emptyset\}$.

Verification Step: s^2 is not a terminal stage, so we go to Stage 3.

Stage 3

The state s^2 is a trading state. The preferred partner for agent 2 is agent 6 while the preferred partner for agent 3 is agent 4.

Trading Step: Agent 6 desires d and agent 4 desires c. Thus they trade partners. So agent 6 is assigned d and agent 4 is assigned c. The state $s^3 = \{a, c, d, c, a, d\}$.

Verification Step: The state s^3 is a terminal state, so we stop with the resulting assignment.

Instead, with the same entitlements, suppose the preference profile R is as follows:

R_1	R_2	R_3	R_4	R_5	R_6
a	c	a	b	c	d
c	d	b	c	a	c
b	b	c	a	d	a
d	a	d	d	b	b

Stage 1

The state s^0 is the null state. The primary list is $\{1, 3, 5\}$ and the preferred partner list is $\{2, 4, 1\}$.

Proposal Step: The proposal corresponding to s^0 is evaluated. The first acceptable pairing is $(1, 3)$. Thus agents 1 and 3 are assigned the project a. The state $s^1 = \{a, \emptyset, a, \emptyset, \emptyset, \emptyset\}$.

Verification Step: s^1 is not a terminal stage, so we go to Stage 2.

Stage 2

The state s^1 is an interim state. The primary list is $\{2, 5\}$ and the preferred partner list is $\{4, 2\}$.

Proposal Step: The proposal corresponding to s^1 is evaluated. The first acceptable pairing is $(2, 5)$. Thus agents 2 and 5 are assigned c. The state $s^2 = \{a, c, a, \emptyset, c, \emptyset\}$.

Verification Step: s^2 is not a terminal stage, so we go to Stage 3.

Stage 3

The state s^2 is an interim state. The primary list is agent 4 and the preferred partner is agent 6.

Proposal Step: The proposal corresponding to s^2 is evaluated. There is no acceptable pairing. Thus agent 4 is assigned b. The state $s^3 = \{a, c, a, b, c, \emptyset\}$.

Verification Step: s^3 is not a terminal stage, so we go to Stage 4.

Stage 4

The state s^3 is a trading state. The preferred partner for agent 4 is agent 6.

Trading Step: Agent 6 desires b. There are no other unassigned agents. So agent 6 is assigned b. The state $s^4 = \{a, c, a, b, c, b\}$.

Verification Step: The state s^4 is a terminal state, so we stop with the resulting assignment.

2.7 Features of the PT Rule and Special Cases

2.7.1 Similarities with Inheritance Rules

In the single-unit object allocation model, Pápai (2000) characterises the set of group-strategy-proof, Pareto optimal and reallocation proof rules via a class of rules called hierarchical exchange rules. As discussed previously, hierarchical exchange rules determine the initial endowment of each object to some agent. Assignments are made using the TTC procedure in rounds. Hierarchical exchange rules also specify the inheritance of unassigned objects to other agents as a function of the structure of previous assignments. Each object is owned by some agent, and agents may own more than one object. Once they have received their assignment, the remaining objects in their endowments get transferred to (or inherited by) some other agent.

One important sub-class of hierarchical exchange rules is what Pápai (2000) calls a fixed endowment exchange rule. In such a rule, the inheritance structure depends only on which agents have received their assignments and what assignment they have received. Such a rule also covers the serial priority rule. Our specification of endowment results effectively in a fixed endowment exchange rule.

Hierarchical exchange rules in general satisfy two properties called the Assurance Rule and the Twin Inheritance Rule. The Assurance Rule guarantees that an object that is in an agent's endowment remains in her endowment at any later stage in which she is yet to receive an assignment. The Twin Inheritance Rule says that if there are two preference profiles such that the hierarchical exchange rule at a certain stage results in the same agents with the same preferences being assigned the same objects, then the endowments determined at that stage for other agents must also be the same in the two cases.

In our model, the partner inheritance structure satisfies these two properties. An agent retains her preferred partner status as long as she is still unassigned. This is the Assurance Rule in other words. Furthermore, the partner inheritance table depends only on the state s. So for any two situations where the assignments and preferences of assigned agents up to that point are the same, i.e., we are in the same state, then the partner inheritance specification for those states will also be the same. This is basically the Twin Inheritance Rule.

Thus there is a connection between PT rules and fixed endowment inheritance rules, though they operate in different environments.

2.7.2 The Serial Dictatorship Rule

In the classical model, where each object may be assigned to at most one agent, the serial dictatorship (or serial priority) rule works as follows: There is an exogenous and fixed ordering of agents σ such that agents sequentially select projects in that order ($\sigma(1)$ selects first, $\sigma(2)$ goes next, and so on). Each agent selects her top-ranked projects from the ones that are available, given the choices of earlier agents in the sequence. It is easy to see that for any preference profile, the first agent always gets her top-ranked object, while the second agent always gets her top-ranked object whenever it is distinct from the selection of the first agent, and so on.

Extending this rule to our context is straightforward. However, the presence of copies in our model means that the feasible set of projects available to later agents in the sequence may be larger than in the classical case. So the serial dictatorship rule f^{SD} in our model works as follows: There

is an exogenous and fixed ordering of agents σ. The first agent in σ will always get to select her top-ranked project. But the second agent in σ may select not only a project that differs from the first agent, but also may select the copy of the project that is assigned to the first agent, because it is still available. In general, agents' choices will affect the feasible set for subsequent agents not only via restricting the available set of *different* projects, but also the availability of copies of projects already selected.

A PT rule f^{Γ} incorporates this special case in the following manner. Suppose with no loss of generality that there is an exogenous ordering of agents σ such that $\sigma(i)$ 'precedes' $\sigma(j)$ whenever $i < j$, where i and j are agents in $\mathcal{N}$. Then for any interim state s, let the primary list be the first $m'(s)$ agents according to σ, and let the corresponding preferred partners be the first available agent according to σ. For a trading state, let the partner inheritance table be such that agent i appears above agent j in the inheritance path of that project whenever i precedes j.

Thus, for example, in the null state, the primary list is $\{\sigma(1), \sigma(2), \sigma(3)\}$, while the preferred partner list is $\{\sigma(2), \sigma(1), \sigma(4)\}$.

The proposal is given as follows:

$\succ_s$
$(\sigma(1), \sigma(2))$
$(\sigma(1), \sigma(3))$
$(\sigma(2), \sigma(1))$
$(\sigma(2), \sigma(3))$
$(\sigma(3), \sigma(4))$
$(\sigma(3), \sigma(1))$
$(\sigma(3), \sigma(2))$

The partner inheritance table is given as follows:

$E_{\sigma(1)}(s)$	$E_{\sigma(2)}(s)$	$E_{\sigma(3)}(s)$
$\sigma(4)$	$\sigma(4)$	$\sigma(4)$
$\sigma(5)$	$\sigma(5)$	$\sigma(5)$
$\sigma(6)$	$\sigma(6)$	$\sigma(6)$

It is easy to see that $f^{SD} = f^{\Gamma}$ for this specification of entitlements, and thus the serial dictatorship rule is a special case of the PT rule.

2.7.3 The 'No Agent Always Gets Her Top-Ranked Project' Rule

An interesting special case of the PT rule is one in which there is no agent that always gets her top-ranked project. The reason this is interesting is that in the classical model, the rules characterised by the same collection of axioms as we have here specify an agent who always gets her top-ranked project. In the case where all agents share an object, strategy-proofness and unanimity together guarantee dictatorship. On the other hand, in the house allocation model, strategy-proofness, unanimity and neutrality guarantee the serial dictatorship. But as can be seen from the example below, it is possible in the pairwise project allocation framework to have a rule that satisfies all the axioms, yet does not have an agent who always gets her top-ranked project.

Let $\mathcal{N} = \{1, 2, 3, 4, 5, 6\}$, let $\mathcal{Z} = \{a, b, c, d\}$, and consider the null state. Let the primary list be $\{1, 2, 3\}$, the preferred partner list be $\{2, 1, 4\}$, and let the proposal be given as in the table below.

$\succ$
(1,2)
(2,1)
(3,4)
(1,3)
(3,1)
(2,3)
(3,2)

Then for the following preference profile, the assignments for agents $\{1, 2, 3, 4\}$ are given in boxes:

R_1	R_2	R_3	R_4	R_5	R_6
a	**b**	**a**	**a**	c	c
b	a	b	b	d	d
c	c	c	c	a	a
d	d	d	d	b	b

With the same entitlement, if the preference profile is the following, the assignments are given in boxes:

R_1	R_2	R_3	R_4	R_5	R_6
a	b	**b**	**b**	c	c
b	**a**	b	a	d	d
c	c	c	c	a	a
d	d	d	d	b	b

Instead, if the preferences are as given below, the assignments also change accordingly:

R_1	R_2	R_3	R_4	R_5	R_6
a	**a**	a	a	c	c
b	b	**b**	**b**	d	d
c	c	c	c	a	a
d	d	d	d	b	b

It is clear from inspection that none of the agents $\{1, 2, 3, 4\}$ always get their top-ranked projects.

2.8 Axioms for Pairwise Project Allocation Rules

We describe below the axioms that we impose on P-PARs, and present some key initial results. Most of these axioms are standard in the literature.

Strategy-proofness is a condition which requires truth-telling to be a dominant strategy for all agents. In other words, given the reports of all other agents, an agent must be as well off reporting her true preferences as any other preferences. When this is true for all agents and all preferences, the mechanism is said to be strategy-proof. Formally:

Axiom **1.** A P-PAR f is *strategy-proof (SP)* if, for all preference profiles R, all agents $i \in \mathcal{N}$, and all preference orderings R_i':

$$f_i(R) R_i f_i(R'_i, R_{-i})$$

The next axiom, which we call the limited influence axiom, identifies conditions under which an agent may not affect another agent's assignment. The axiom has two parts. The first part is identical to the non-bossiness axiom that is pervasive in the literature on assignment rules. The condition was introduced by Satterthwaite and Sonnenschein (1981) and requires that an agent not be able to affect other agents' outcomes without affecting her own.

The second part of the limited influence axiom is new. It states that if an agent cannot obtain a particular project that she desires over her assignment (given other agents' preferences) then she cannot influence the assignment of that project. We first state the axiom formally and discuss it below.

Axiom **2.** A P-PAR f satisfies *limited influence (LIN)* if:

1. (LIN1) (Non-bossiness (NB)) For all preference profiles R, all agents $i \in \mathcal{N}$, and all preference orderings R'_i:

$$[f_i(R'_i, R_{-i}) = f_i(R)] \implies [f(R'_i, R_{-i}) = f(R)]$$

2. (LIN2) For all preference profiles R, all projects $a \in \mathcal{Z}$, all agents $i \in \mathcal{N}$ such that $aP_i f_i(R)$, and for all R'_i such that $f_i(R'_i, R_{-i}) \neq a$:

$$[f_j(R) = a] \implies [f_j(R'_i, R_{-i}) = a] \text{ for all } j \in \mathcal{N}$$

Limited influence merits more discussion. LIN1, which is essentially non-bossiness, negates any effect that an agent can have on other agents' assignments in cases where she does not change her own assignment. Its main justification is that it keeps the distribution of influence in the allocation process from unduly depending on any one agent. Another justification has to do with its strategic effects. Also, its original use by Satterthwaite and Sonnenschein (1981) is on the basis of considerations of informational simplicity. Non-bossiness disqualifies rules in exchange economies that "assign all the resources to one or the other of two agents depending upon some arbitrary feature of some third agents preferences."[9]. However, Thomson (2014) also notes that its main value is in providing technical support for characterisation results.

LIN2 negates any effect that an agent can have on the assignment of a project that she also desires but cannot obtain. Suppose that there is an agent and a project she desires. For whatever reason, she is not assigned this project. Then for any reported preference in which she does not get assigned that project, it should not be the case that she can influence who else gets or does not get that project. In a sense, it says that if the agent does not have the 'rights' to that project, then she should not be able to influence who else has those rights.

Both LIN1 and LIN2 have the most effect when used in conjunction with strategy-proofness (Thomson (2013)). To see this, suppose that an agent is unable to affect his assignment to his advantage by misrepresenting his or her preferences. Then strategy-proofness is met for this agent. Such a misrepresentation may yet affect some other agents assignment. If this other agent benefits from it, there is an incentive for the second agent to approach the first agent and suggest the manipulation. LIN1 applies to cases where an agent is unable to change her own assignment at all,

[9] See Thomson (2014)

while LIN2 applies to cases where the misrepresentation does not yield the desired project (which is ensured by strategy-proofness).

LIN2 is very similar to a condition used by Pápai (2000). In fact, LIN1 and LIN2 play an important role in the characterisation of inheritance rules. In that paper, LIN2 is not assumed directly, and instead emerges as a consequence of the combination of strategy-proofness, non-bossiness and an additional condition called reallocation-proofness. Reallocation-proofness "rules out the possibility that two individuals can gain by jointly manipulating the outcome and swapping objects ex post, when the collusion is self-enforcing in the sense that neither party can lose by reporting false preferences in case the other party does not adhere to the agreement and reports honestly".[10] The corresponding version of reallocation-proofness in this model is complicated, and so we do not use reallocation-proofness. We instead take LIN2 to be a primitive requirement of the rule.

There are important single-unit object allocation rules in the literature that satisfy LIN1 and LIN2 and others that do not. As mentioned above, inheritance rules satisfy both axioms. Therefore so do sequential and serial priority rules. However, the Deferred Acceptance (DA) rules and their generalisations typically do not satisfy either condition. It is possible for an agent in the DA rule to affect the assignment of an object even when she cannot obtain it herself, whether she changes her assignment as a result or not. Thus it is a non-trivial condition to impose on project allocation rules. Moreover, the two conditions are independent, as we demonstrate by examples in Section 2.8.1.

Group-strategy-proofness is a stronger condition than strategy-proofness. It ensures that groups of agents do not have profitable deviations, i.e., if a group of agents deviates by reporting different preferences, then a group-strategy-proof rule ensures that it is not the case that all agents in the deviating group are at least as well off as before, and some agent strictly better off. Formally, a P-PAR f is *group-strategy-proof* if, for all profiles R, there does not exist a set of agents $M \subseteq \mathcal{N}$, and a preference sub-profile R'_M, such that $f_i(R'_M, R_{-M})R_i f_i(R)$ for all $i \in M$, and $f_j(R'_M, R_{-M})P_j f_j(R)$ for some $j \in M$.

In a wide class of assignment models, including ours, group-strategy-proofness is equivalent to the combination of strategy-proofness and non-bossiness. We reproduce the proof from Pápai (2000) here.

LEMMA **1.** *A P-PAR is group-strategy-proof if and only if it is strategy-proof and non-bossy.*

Proof: It is clear that group-strategy-proofness implies strategy-proofness (let the group size be unity). To see that it implies non-bossiness as well, consider a preference profile R, agents $i, j \in \mathcal{N}$, and preferences R'_i such that $f_i(R'_i, R_{-i}) = f_i(R)$ but $f_j(R'_i, R_{-i}) \neq f_j(R)$. Since preferences are strict, either $f_j(R'_i, R_{-i})P_j f_j(R)$ or $f_j(R)P_j f_j(R'_i, R_{-i})$. In the first case, agents i, j can manipulate at R via (R'_i, R_j), and in the second case, agents i, j can manipulate at (R'_i, R_{-i}) via (R_i, R_j). In either case, group-strategy-proofness is violated.

To show the converse, let f satisfy SP and NB. Consider a subset of agents M, a preference profile R and a sub-profile R'_M, such that for all $i \in M$, we have that $f_i(R'_M, R_{-M})R_i f_i(R)$. For each $i \in M$, consider a preference ordering $\hat{R}_i$ such that we move her assignment $f_i(R'_M, R_{-M})$ to the top of her preference $\hat{R}_i$, and leave the other projects ranked the same as they are in R_i. By SP, $f_i(\hat{R}_i, R_{-i}) = f_i(R)$. Hence by NB, $f(\hat{R}_i, R_{-i}) = f(R)$. Repeating for all agents in M, we have that $f(\hat{R_M}, R_{-M}) = f(R)$. Also, by SP and NB, $f(\hat{R_M}, R_{-M}) = f(R'_M, R_{-M})$. So $f(R'_M, R_{-M}) = f(R)$, and f is group-strategy-proof. ■

[10] See Pápai (2000).

Thus by requiring our rule to satisfy strategy-proofness and limited influence, we ensure that the rule is group-strategy-proof.

For any preference P_i and any subset $X \subseteq \mathcal{Z}$, let $top(P_i, X)$ denote the top project in X according to P_i. When we mean the top-ranked project from the full set $\mathcal{Z}$, we will often suppress the set notation and refer to it simply as $top(P_i)$. Correspondingly, for a preference profile P, let $top(P, X)$ denote the N-dimensional vector of top-ranked projects in X according to preferences in P. Similarly, $top(P)$ is the vector of top-ranked projects in $\mathcal{Z}$ according to preferences in P.

Unanimity is a full-range condition which says that if agents' preferences are such that it is feasible to give each agent her top-ranked project, then the mechanism must do so. In our model, this means that if the vector of top-ranked preferences in a particular profile is such that agents are naturally divided into m pairs, then a mechanism satisfying unanimity must prescribe those exact pairs. Formally:

Axiom **3.** A P-PAR f is *unanimous (U)* if, for all preference profiles R:

$$[top(R) \in \mathcal{A}] \implies [f(R) = top(R)]$$

A stronger condition than unanimity is Pareto efficiency. We say that an assignment is Pareto efficient if it is not possible to make an agent strictly better off while keeping all agents at least as well off as earlier. Formally, a P-PAR f is *Pareto efficient* if, for any preference profile R, there is no feasible assignment $x \in \mathcal{A}$ such that $x_i R_i f_i(R)$ for all $i \in \mathcal{N}$, with $x_j P_j f_j(R)$ for some $j \in \mathcal{N}$.

The combination of strategy-proofness, limited influence and unanimity gives us Pareto efficiency. Pápai (2001) proves a very similar result using a condition called citizen sovereignty, which is weaker than unanimity.

Lemma **2.** *A P-PAR satisfying strategy-proofness, non-bossiness and unanimity is Pareto efficient.*

Proof: Let f be strategy-proof, non-bossy and unanimous, and let R be a preference profile. Suppose $f(R)$ is not a Pareto efficient assignment. Then there exists some feasible assignment $x \in \mathcal{A}$ such that $x_i R_i f_i(R)$ for all $i \in \mathcal{N}$ and $x_j P_j f_j(R)$ for some $j \in \mathcal{N}$. Construct a preference profile $\hat{R}$ from R such that for all agents i, x_i is ranked top in $\hat{R}_i$ while other projects are ranked the same as in R_i.

Consider some agent i and the profile $(\hat{R}_i, R_{-i})$. By strategy-proofness, $f_i(\hat{R}_i, R_{-i}) = f_i(R)$, and so $f(\hat{R}_i, R_{-i}) = f(R)$ by non-bossiness. Repeating for all agents, we have that $f(\hat{R}) = f(R)$. By construction, $top(\hat{R}) = x$. Since $x \in \mathcal{A}$, by unanimity we have that $f(\hat{R}) = x$. But $x \neq f(R)$. This is a contradiction. Hence $f(R)$ is Pareto efficient. Since R was arbitrary, $f(R)$ is Pareto efficient for all R and so f is a Pareto efficient rule. ■

Neutrality ensures that a rule treats all projects symmetrically, and does not distinguish between them on the basis of their names. That is, if for a particular preference profile we were to perform a swap operation on a pair of projects, exchanging their positions in each agent's preferences, then such a swap must reflect exactly in the final assignments as well. When this is true for all projects and all profiles, we say a mechanism is neutral. Formally:

Axiom **4.** A P-PAR is *neutral (NEU)* if, for all preference profiles R and all permutations[11] π of $\mathcal{Z}$:

$$f(\pi R) = \pi f(R)$$

[11] A permutation applied to a collection of objects X is a bijection $\pi : X \to X$ that associates each object in X with a unique object in X (possibly itself). In our case, we use it to mean a relabelling of projects such that a collection of

2.8.1 Independence of Axioms

To show that these axioms are independent, we now provide examples of rules satisfying all but one of the axioms in turn.

Strategy-proofness: Consider a rule that operates like a PT rule with the following modification. There are three agents $\{i, j, k\}$ such that for any preference profile R, if $top(R_i) = top(R_j)$ then the preferred partner for agent i is j, and is k otherwise. Also, j is not in the primary list at any interim state s. Let R be a profile where $top(R_i) = top(R_k) = a$, $top(R_j) = b$ and a is ranked second in R_j, and a and b are the last two projects in the preferences of all other agents. Then $aP_jf_j(R)$. But j can manipulate via a preference R'_j in which a is ranked top, since $f_j(R'_j, R_{-j}) = a$. This rule violates strategy-proofness. It is easy to check that it satisfies limited influence, unanimity and neutrality.

Limited influence 1: Consider a rule like the PT rule, but with the following modification. Let the first pairing in the null state be $(1, 2)$. If agents 1 and 2 are assigned the same project, and if their second-ranked projects are the same, then the first pairing for the next state is (i, j), whereas if the second-ranked projects in their preferences differ, then the first pairing for the next state is (k, l), where $i \neq j \neq k \neq l$. Like the PT rule, this rule satisfies strategy-proofness, unanimity, neutrality and limited influence 2, but violates LIN1, i.e., is bossy.

Limited Influence 2: Let $N = 6$ and $q = 2$. Consider a rule that works like the PT rule, with the following modifications. Let R be a profile. The first pairing for the null state is $(1, 2)$. If $top(R_1) \neq top(R_2)$, we look at agent 3's preferences. If $top(R_3) = top(R_1)$ and the second-ranked project in R_3 is distinct from $top(R_2)$, then the second pairing is $(1, 4)$. If $top(R_3) = top(R_1)$ and the second-ranked project in R_3 is the same as $top(R_2)$, then the second pairing is $(1, 5)$. We specify the rest of the entitlements suitably.

Consider the preference profiles R and R' given below. The assignments are given in boxes.

R

1	2	3	4	5	6
[a]	[b]	a	a	[a]	a
c	d	[b]	[d]	d	[d]
d	c	c	b	c	c
b	a	d	c	b	b

R'

1	2	3	4	5	6
[a]	[b]	a	[a]	a	a
c	d	[c]	d	d	d
d	c	b	b	[c]	c
b	a	d	c	b	[b]

projects exchange their names. For example, under π, project a may now be called project b ($\pi(a) = b$), which is now called project c ($\pi(b) = c$), which in turn is called project a ($\pi(c) = a$). The permutation π applied to a preference profile R (written as πR) or an assignment vector x (written as πx) permutes the projects in the preferences or the assignment according to the permutation applied to the underlying set of projects $\mathcal{Z}$.

Note that from R to R' only agent 3's preferences change. She does not get a which she prefers to her assignment, but still affects the assignment of a (to agent 5 in R and agent 4 in R'). Thus this rule violates LIN2. It is easy to see that it satisfies strategy-proofness, LIN1, unanimity and neutrality.

Unanimity: Consider a rule[12] that divides agents into fixed pairs $M_1, ..., M_m$ and indexes one agent $i(M_i)$ in each pair. For any preference profile R, run the sequential priority rule with agents $i(M_i)$. Whatever project they select becomes the assignment of the corresponding pair M_i. This rule violates unanimity. It is easy to check that it satisfies strategy-proofness, limited influence and neutrality.

Neutrality: Consider a rule that works like the PT rule, with the following modification. There is a set of agents $\{i, j, k\}$ and projects a, b such that for any profile R with $top(R_i) = a$, $i \in \alpha(s^0)$, where s^0 is the null state, the first pairing for S^0 is (i, j), and for any profile R' such that $top(R'_i) = b$, the first pairing for s^0 is (i, k). This rule violates neutrality. It is easy to check however that it satisfies strategy-proofness, limited influence and unanimity.

2.9 Characterisation Result

We are now ready to state our main characterisation theorem.

Theorem **1.** *A pairwise-project assignment rule is strategy-proof, unanimous, neutral and satisfies limited influence if and only if it is a partner trading rule.*

The proof is in the appendix. Here we provide the intuition behind the arguments of the proof.

2.9.1 Sufficiency

We begin by proving two lemmas. The first says that if there is an agent and a project she desires over her assignment at a profile, then this project must either have been fully assigned by the PT rule at an earlier stage, or if this project is not assigned at all, then m different projects must have been at least partially assigned by the time she gets her assignment. The second lemma shows that an agent cannot affect the assignment of any agent who receives her project in a stage before when this agent is assigned her project. We use these lemmas to show that the PT rule satisfies the axioms.

- Strategy-proofness: Suppose for a profile there is an agent who prefers some project to her own assignment. Either this project is assigned at that profile or it is not. If it is, then by the lemma it must have been fully assigned at a stage before when this agent gets her assignment. If it is not assigned to anyone, then m different projects must have already been assigned by that stage, by the same lemma. Since no agent can affect the assignment of any agent who is assigned a project at an earlier stage, a unilateral deviation on the part of this agent will not get her the project, and strategy-proofness is satisfied.

[12] This rule is identical to the one proposed by Rhee (2011).

- Limited influence 1: It can easily be seen that no agent can be bossy with another agent who receives her assignment at an earlier stage or the same stage (by the second lemma). We show that no agent can be bossy with an agent who receives her assignment at a later stage. This involves showing that the entitlement at this stage is a function only of the agents who receive their assignments and the projects they receive. As long as this agent receives the same project, the entitlement remains the same. Thus the assignments at the next stage are independent of this agent. And the same is true for all later stages as well. Thus the earlier agent cannot affect the assignment of the later agent.

- Limited influence 2: If there is a project that this agent prefers, then it is being assigned at an earlier stage than when she gets her assignment. Thus she cannot affect the assignment of this project to earlier agents even if she receives a different project later. The same is true if the project is not assigned to anyone at all.

- Unanimity: The PT rule makes assignments based on the top-ranked projects of agents among projects not fully assigned. Thus for a unanimous profile, agents are only ever assigned their top-ranked projects, and so the allocation must be the vector of top-ranked projects.

- Neutrality: It is easy to see that neither the entitlement nor the iterative procedure of the PT rule depends on the identity of the project. Thus for any profile R and any reshuffling of the names of projects, the allocation must incorporate the reshuffling as well, as the various partners remain the same.

2.9.2 Necessity

To prove the converse, we have to show that any rule satisfying the axioms is a PT rule. This requires two steps: One, we construct the entitlements for any state. Two, we show that assignments are made for any preference profile via the iterative procedure using those entitlements.

First we construct the entitlements for any state.

We show that for any interim state there is a pair of agents that can guarantee the assignment of any unassigned project among themselves by declaring it their top option. The proof begins by showing that for an identical preference profile, Pareto efficiency implies that some pair gets the top-ranked project. We then show that strategy-proofness and limited influence mean that no other agent can affect the assignment of this pair. Neutrality then implies that this is true for all unassigned projects. Since our selection of interim state was arbitrary, this allows us to construct the pairing for all such states.

We prove a 'trading' lemma that allows us to trace the sequence in which acceptable pairings may be found. We also prove a lemma that states that any pair of agents that can guarantee a project between them must have at least one of them receiving a weakly preferred project to that one.

We use these lemmas to construct the acceptable pairings for any state. We also order them, generating the proposal, and identify the primary and preferred partner lists for each state.

After this, we build the partner inheritance table for each trading state. For any interim state, we show that any project assigned to only one agent must have another agent who is the preferred partner. We show that trade must happen wherever possible, and so repeating for all unpartnered agents and all subsets of remaining agents, we are able to generate the inheritance paths for each unpartnered agent for that trading state, thereby generating the partner inheritance table.

Collectively, this gives us the entitlement for each state.

Finally, we show that the rule satisfying the axioms must behave like a PT rule. That is, using the entitlements generated above, we show that the rule prescribes assignments in stages in a manner consistent with the iterative process specified by the PT rule. This completes the proof.

2.10 Conclusion

In this paper we have developed a rule to make assignments in the pairwise project allocation framework. We have shown that this PT rule specifies a class that is characterised by the properties of strategy-proofness, limited influence, unanimity and neutrality. In what follows, we discuss some possible extensions of this model.

The first target of any extension of this rule will be from pairwise allocation to arbitrary group sizes, i.e., where each project must be assigned to exactly q agents or to nobody, where $1 \leq q \leq |\mathcal{N}|$.[13]

Note that assuming $q = 1$ returns us to the classical object allocation setup. Following from the work of Svensson (1999), it is straightforward to show that our rule would translate into serial dictatorships. At the other extreme, when $q = |\mathcal{N}|$, such that each project must be assigned either to everyone or no one, we return to the public goods setting, and our rule will become a dictatorship in the sense of Gibbard (1973) and Satterthwaite (1975).

Thus the problem effectively becomes one of determining what happens when $2 < q < |\mathcal{N}|$. In principle, this would involve expanding the components of the entitlement and then refining the iterative procedure to handle larger groups.

As far as expanding entitlements is concerned, some tasks are easier than others. The definition of a pairing can easily be generalised to account for larger group sizes that can commonly guarantee a top-ranked project. The existence of such a larger group for any interim state is also easy to show.

One difficulty lies in specifying proposals. In our model, proposals are orderings over pairs of agents. When considering larger groups, however, there is more than one agent who can join any agent who receives a project. In fact, more than one agent *must* join any agent who receives a project. Thus the notion of a proposal cannot be used directly. We have some insight but not a clear picture about how this would work.

Difficulties also arise when considering a possible TTC round. More than one agent may be the preferred partner of an unpartnered agent. These agents have no a priori distinction. Thus if there is some other agent who desires this partner in a TTC round, she has more than one option of agent to point to. Determining which trade, if any, is honoured in this case is a non-trivial exercise. The presence of possibly multiple overlapping cycles will require the use of a tie-breaker or the introduction of some other component of an entitlement. We have not found a way around this problem yet.

Other extensions, even in the pairwise framework, would be to obtain a full characterisation of group-strategy-proof and Pareto optimal rules. This will require dropping the axioms of neutrality and LIN2. Dropping neutrality will spread the pairwise 'ownership' of each project to possibly a different pair of agents. Dropping LIN2 leaves the rule in a possibly endogenous situation, where the preferences of other agents could influence the initial pairings. These remain open problems.

[13] Another related extension would be to drop the requirement that each project have the same exact capacity constraint.

2.11 Appendix: Proof of Theorem 1

In this section we present the proof of our main characterisation theorem.

2.11.1 Sufficiency Proof

We must show that the PT rule satisfies the axioms.

Lemma **3.** *Let f^Γ be the PT rule, let R be a profile, let i be some agent and $a \in \mathcal{Z}$ a project such that $aP_i f_i^\Gamma(R)$. Suppose i receives her assignment in stage k. Then, by some stage $k' < k$, either (1) a is fully assigned, or (2) m different projects are assigned to other agents.*

Proof: Let f^Γ be the PT rule, let R be a profile, let i be some agent and $a \in \mathcal{Z}$ a project such that $aP_i f_i^\Gamma(R)$. Note that at any stage of the PT rule, an agent gets her assignment by virtue of her top-ranked project from those that are available, whether as a part of a proposal or by trading. Suppose i receives her assignment $f_i(R)$ in stage k. She gets it by pointing to $f_i(R)$. By the properties of the PT rule, this means that project a is no longer available in stage k otherwise she would be pointing to it instead. Thus if project a is assigned to other agents, it must have been fully assigned by at most stage $k-1$. Alternatively, project a is no longer available because m other projects have been assigned. Again, this must have happened at most by stage $k-1$. ■

Given a preference profile R and agents i and j, we say that agent i *affects the assignment of agent j* if, for some R_i', we have that $f_i(R_i', R_{-i}) = f_j(R)$ but $f_j(R_i', R_{-i}) \neq f_j(R)$.

Lemma **4.** *Let f^Γ be a PT rule. For any profile R, consider agent i and let k be the stage in which she receives her assignment. Then i cannot affect the assignment of any agent j who receives her assignment in a stage earlier than k.*

Proof: The null state entitlement is given exogenously and does not depend on agent i. Assignments in the first stage are based on preferences of agents that receive their assignments in that stage. Agent i cannot influence their preferences, and cannot influence the order in which pairings are evaluated, and so she cannot influence their assignments. Thus she cannot affect the state after the first stage, and cannot affect the entitlements in the second stage either. Let the state at any stage $k' < k$ be given. Assignments are based on preferences of agents other than i. Agent i cannot affect their preferences, the order of proposals, or assignments. Thus the state in $k'+1$ is given independently of i. So are entitlements. By induction this is true for all $k' < k$. So agent i cannot affect any assignments made in stages before when she gets her assignment. ■

Strategy-proofness: Let R be a profile, a some project and i an agent such that $aP_i f_i(R)$. Suppose i gets her assignment in stage k. There are two possibilities: (1) $f_M(R) = a$ for some M. Then by Lemma 3 agents in M are assigned project a before stage k. (2) $f_j(R) \neq a$ for all $j \in \mathcal{N}$. By Lemma 3, this means that m distinct projects have been assigned by stage k. Since by Lemma 4, agent i cannot affect the entitlement at any earlier stage, she cannot get a for any preference R_i'. Thus in each case a unilateral deviation will not get her the project, and SP is satisfied.

Limited influence 1: It can easily be seen that no agent can be bossy with another agent who receives her assignment at an earlier stage or the same stage, since by Lemma 4 she cannot affect the assignments at earlier stages. To see that an agent cannot affect the assignment of later agents without changing her own assignment, see that the entitlement formed at the end of the stage where she receives her assignment depends on the identities of agents receiving their assignments in that stage and the projects they receive. She cannot affect any other agent who receives her assignment in the same stage without changing her own assignment. So if she continues to get the same project, the state at the end of the stage remains the same, and so does the entitlement. Later entitlements are independent of her preferences. So she cannot affect the assignment of any subsequent agent as long as she gets the same project. Thus f^{Γ} satisfies LIN1.

Limited influence 2: Let R be a profile, i some agent and some project a such that $aP_i f_i(R)$. If a is assigned by the PT rule for R, then by Lemma 3, there is some pair M and some stage k where both agents in M have received a as their assignment. Since by Lemma 4 agent i cannot affect entitlements at earlier stages, she cannot affect the assignment of this project to earlier agents as long as they desire it. This is true even if she were to receive some other project. If a is not assigned to any agent for R, then by Lemma 3, m different projects have already been assigned, and agent i cannot influence the assignment of project a. Thus f^{Γ} satisfies LIN2.

Unanimity: Let R be a unanimous preference profile. At any stage, an agent participating in a proposal or a partner inheritance table does so via her top-ranked project among those that are not fully assigned. Thus every assignment that is made is the respective agent's top-ranked project. So $f^{\Gamma}(R) = top(R)$, and f^{Γ} is unanimous.

Neutrality: It is easy to see that neither the entitlement nor the iterative procedure of the PT rule depends on the identity of the project (it does depend on the identity of the agents). Thus for any profile R and any permutation π of $\mathcal{Z}$, we have that $f^{\Gamma}(\pi R) = \pi f^{\Gamma}(R)$ and so f^{Γ} is neutral.

2.11.2 Necessity Proof

We start by showing that at any interim state there is a pair of agents that can guarantee their own assignment of any unassigned project by commonly declaring it as their top-ranked project.

Lemma **5.** *Consider an interim state s with $m'(s) \geq 1$. Fix preferences of agents in $N(s)$ as $R_{N(s)}$. Let f satisfy SP, LIN, U and NEU. Then there exists a pair of agents $M(s) \subseteq \bar{N}(s)$ such that, for any $R_{\bar{N}(s)}$ and any $a \in \bar{Z}(s)$, $[top(R_{M(s)}) = (a, a)] \implies [f_{M(s)}(R) = (a, a)]$.*

Proof: Let R be a profile such that $R_{N(s)}$ is as given and $R_i = R_j$ for all $i, j \in \bar{N}(s)$. Without loss of generality, let $top(R_{\bar{N}}(s)) = (a, a, ..., a)$, with $a \in \bar{Z}(s)$. Since f is Pareto efficient and $m'(s) \geq 1$, there must exist a pair of agents $M \subseteq \bar{N}(s)$ such that $f_M(R) = (a, a)$. Let $R'_{\bar{N}(s)\setminus M}$ be an arbitrary sub-profile for agents in $\bar{N}(s) \setminus M$. We will first show that $f_M(R_{N(s)}, R_M, R'_{\bar{N}(s)\setminus M}) = (a, a)$.

Pick a $j \in \bar{N}(s) \setminus M$ and consider the sub-profile (R'_j, R_{-j}). Since $aP_j f_j(R)$, SP implies that $f_j(R'_j, R_{-j}) \neq a$. If $f_j(R'_j, R_{-j}) = f_j(R)$ then $f(R'_j, R_{-j}) = f(R)$ by LIN1. In particular, $f_M(R'_j, R_{-j}) = (a, a)$. Instead, suppose $f_j(R'_j, R_{-j}) \neq f_j(R)$. Then $[f_M(R) = (a, a)] \implies [f_M(R'_j, R_{-j}) = (a, a)]$ by LIN2.

Repeating for all other agents in $\bar{N}(s) \setminus M$, we have that $f_M(R_{N(s)}, R_M, R'_{\bar{N}(s)\setminus M}) = (a, a)$.

It follows from SP and LIN1 that this is true for any R_M with $top(R_M) = (a, a)$. Since f satisfies NEU, this is true for all $a \in \bar{Z}(s)$. Thus, for any preference sub-profile $R_{\bar{N}(s)}$ and any $a \in \bar{Z}(s)$ with $top(R_M) = (a, a)$, we have that $f_M(R) = (a, a)$. Set $M(s) = M$ as determined above. ■

For convenience, we shall denote the pair $M(s^0)$ where s^0 is the null state as M^*.

Next, we define the notion of the pair-option-set. Fixing a sub-profile of preferences of other agents, the *pair-option-set* of a pair M at that sub-profile is the set of projects that it can receive if both agents in the pair list it as their top preference. Formally:

DEFINITION **1.** Let M be a pair of agents, and let R_{-M} be an arbitrary sub-profile for the other agents. The *pair-option-set of* M *at* R_{-M} is denoted $o_M(R_{-M})$ such that:

$$o_M(R_{-M}) = \{a \in \mathcal{Z} | \exists R'_M : [top(R'_M) = (a, a)] \implies [f_M(R'_M, R_{-M}) = (a, a)]\}$$

We now state and prove a lemma that will be useful in constructing proposals. Lemma 6 identifies conditions under which agents in pairs with a non-empty pair-option-set can 'trade' projects with each other. In particular, if there are two pairs with non-empty pair-option sets and at least one agent claims the project associated with the opposite pair, then a rule satisfying our axioms must honour the swap. Formally:

LEMMA **6.** *Let f satisfy SP, LIN and U. Let R be a profile. For a pair of agents $M = \{i, j\}$, suppose that $top(R_i) = a$ and $top(R_j) = b$, and let $a \in o_M(R_{-M})$. If there is a pair $M' = \{k, l\}$ such that $top(R_k) = top(R_l) = a$ and $b \in o_{M'}(R_{-M'})$, then $f_j(R) = b$ and $f_k(R) = a$ or $f_l(R) = a$.*

Proof: Suppose for contradiction that $bP_j f_j(R)$. Construct R'_j such that b and a are ranked first and second, and all other projects are ranked the same as in R_j. By SP, $f_j(R'_j, R_{-j}) \neq b$. Since $a \in o_M(R_{-M})$, we have that $f_M(R'_j, R_{-j}) = a$. Note that $aP_k f_k(R'_j, R_{-j})$, $aP_l f_l(R'_j, R_{-j})$ and $b \in o_{M'}((R'_j, R_{-\{\mathcal{N}\setminus\{M'\cup\{j\}\}}}))$. Construct R'_k, R'_l such that a and b are ranked first and second respectively, and all other projects are ranked the same as in R_k, R_l respectively. By SP, $f_{k,l}(R'_{\{j,k,l\}}, R_{-\{j,k,l\}}) \neq a$.

As $b \in o_{M'}((R'_j, R_{-\{\mathcal{N}\setminus\{M'\cup\{j\}\}}}))$, it follows that $f_{M'}(R'_{\{j,k,l\}}, R_{-\{j,k,l\}}) = (b, b)$. By LIN, we have that $f_j(R'_{\{j,k,l\}}, R_{-\{j,k,l\}}) = a$. This violates PE as agent j and either k or l can swap assignments making them both strictly better off while keeping other agents as well off as before. This is a contradiction. So $f_j(R'_{\{j,k,l\}}, R_{-\{j,k,l\}}) = b$. But $b \in o_{M'}((R'_j, R_{-\{M'\setminus\{j\}\}}))$, so at least one of k, l must get something they strictly prefer to b. Without loss of generality, let $f_k(R'_{\{j,k,l\}}, R_{-\{j,k,l\}}) = a$. By SP and LIN, this means that $f_j(R) = b$ and $f_k(R) = a$. ■

We prove another useful lemma. Lemma 7 says that if f satisfies SP, LIN and U, then for any preference profile and any pair of agents with a non-empty pair-option-set, at least one agent in the pair must be assigned a project that she weakly prefers to her top-ranked project in the pair-option-set. Formally:

LEMMA **7.** *Let f satisfy SP, LIN and U. Let R be a preference profile and let M be a pair of agents such that $o_M(R_{-M}) \neq \emptyset$. For each $i \in M$, let $a_i = top(R_i, o_M(R_{-M}))$. Then $f_j(R) R_j a_j$ for some $j \in M$.*

Proof: Let f satisfy SP, LIN and U, let R be a preference profile and let M be a pair of agents with $o_M(R_{-M}) \neq \emptyset$. Let $a_i = top(R_i, o_M(R_{-M}))$ for each $i \in M$. Define $Y = \{a_i | i \in M\}$. It follows that $1 \leq |Y| \leq 2$.

Case 1: Suppose $|Y| = 1$. Let $Y = \{b\}$, i.e., $a_i = b$ for all $i \in M$. For contradiction, suppose that $bP_i f_i(R)$ for all $i \in M$. For each $i \in M$, construct R'_i such that b is the top-ranked project in R'_i and all other projects are ranked the same as in R_i. Consider some $j \in M$. Since $bP_j f_j(R)$, by SP and LIN1 it follows that $f(R'_j, R_{-j}) = f(R)$. Repeating for the other agent in M, we get that $f(R'_M, R_{-M}) = f(R)$. In particular, $f_M(R'_M, R_{-M}) \neq (b, b)$. But $b \in o_M(R_{-M})$, so by definition $top(R'_M) = (b, b)$ implies $f_M(R'_M, R_{-M}) = (b, b)$. This is a contradiction. So $f_i(R)R_i b$ for at least one agent $i \in M$.

Case 2: Suppose $|Y| = 2$. Let $M = \{i, j\}$ and without loss of generality let $top(R_i) = a$ and $top(R_j) = b$. For contradiction, suppose that $aP_i f_i(R)$ and $bP_j f_j(R)$. Consider agent i. Construct R'_i such that a and b are ranked first and second in R'_i, and all other projects are ranked the same as in R_i. It follows from SP that $f_i(R'_i, R_{-i}) \neq a$. Since $b \in o_M(R_{-M})$, we have by SP and LIN1 that $f_M(R'_i, R_{-i}) = (b, b)$. For agent j, construct R'_j such that b and a are ranked first and second respectively, and other projects are ranked the same as in R_j. It follows from SP and LIN1 that $f_M(R'_M, R_{-M}) = (b, b)$.

Instead, consider agent j and the profile (R'_j, R_{-j}). By a symmetric argument, $f_M(R'_j, R_{-j}) = (a, a)$. And repeating for agent i, we have that $f_M(R'_M, R_{-M}) = (a, a)$. We thus have two outcomes for M at the same preference profile. Since $a \neq b$, this is a contradiction. Thus $f_i(R)R_i a$ or $f_j(R)R_j b$. ■

COROLLARY **1.** *For every R there is an $i \in M^*$ such that $f_i(R) = top(R_i)$.*

Proof: By Lemma 5, for every $a \in \mathcal{Z}$ and every $R_{-M_1^*}$, $a \in o_{M^*}(R_{-M^*})$. ■

PROPOSAL VECTOR

In what follows we show how to generate the proposals for an arbitrary interim state.

1. Let s be some interim state, and let $m'(s) \geq 2$.

2. By Lemma 5, there exists a pair of agents $M(s) \subseteq \bar{N}(s)$ such that $a \in o_{M(s)}(R_{-\bar{N}(s)\setminus M(s)})$ for all $R_{-\bar{N}(s)\setminus M(s)})$ and all $a \in \bar{Z}(s)$. Without loss of generality, let $M(s) = \{1, 2\}$.

3. Note that, by Lemma 5, for all $R_{M(s)}$ with $top(R_1, \bar{Z}(s)) = top(R_2, \bar{Z}(s)) = a$, we must have that $f_M(s) = (a, a)$, for all $a \in \bar{Z}(s)$. Thus $(1, 2)$ and $(2, 1)$ are both acceptable pairings by the definition above.

4. Consider agent 1. Consider a sub-profile $R_{\bar{N}(s)}$ where $top(R_1) = b$ and a is ranked second, and $top(R_k) = a$ for all $k \in \bar{N}(s), k \neq 1$. By Lemma 6, $f_1(R) = b$. By PE, there is a pair $M_1 \subset \bar{N}(s)$ such that $f_{M_1}(R) = a$.

   CLAIM **1.** For any $R_{\bar{N}(s)}$ and $c, d \in \bar{Z}(s)$, if $top(R_1) = c$ and $top(R_{M_1}) = (d, d)$, then $f_1(R) = c$ and $f_{M_1}(R) = (d, d)$.

Proof: Let $R_{\bar{N}(s)}$ be as in the construction above. Then we have $f_1(R) = b$ and $f_{M_1}(R) = a$. Consider any agent $k \neq 1$ and $k \notin M_1$, and let R'_k be some arbitrary preference. We have that $aP_k f_k(R)$. So by SP, $f_k(R'_k, R_{-k}) \neq a$. By LIN, $f_{M_1}(R'_k, R_{-k}) = (a, a)$. Suppose $f_1(R'_k, R_{-k}) \neq b$. Then by Lemma 5, $f_1(R'_k, R_{-k}) = a$. This contradicts LIN. Thus $f_1(R'_k, R_{-k}) = b$. Repeating for all other agents, we get that assignments of agent 1 and agents in M_1 are independent of other unassigned agents' preferences. By neutrality, this must be true for all such configurations of preferences. In particular, for any $c, d \in \bar{Z}(s)$, if $top(R_1) = c$ and $top(R_{M_1}) = (d, d)$, then $f_1(R) = c$ and $f_{M_1}(R) = (d, d)$. ■

5. So M_1 is an acceptable pairing, by definition. Now, consider agent 2. Consider a sub-profile $R_{\bar{N}(s)}$ where $top(R_2) = b$ and a is ranked second, and $top(R_k) = a$ for all $k \in \bar{N}(s), k \neq 2$. By Lemma 6, $f_2(R) = b$. By PE, there is a pair $M'_1 \subset \bar{N}(s)$ such that $f_{M'_1}(R) = a$.

6. By the reasoning in Claim 1 above, M'_1 is also an acceptable pairing.

7. Now, fixing the preferences of earlier agents, we can repeat the above step for each acceptable pairing (once for each agent in the pairing) to get the next acceptable pairing. We must take care to vary the top-ranked projects for each agent in the pairing so that they are distinct from the top-ranked projects of the agents whose preferences we have already fixed.

8. Repeating in this way, we can generate all acceptable pairings. We end up in a situation where there are $m'(s)$ agents with distinct top-ranked projects, which they are all assigned, and this does not depend on the preferences of other agents. This produces the primary list for this state.

9. Now it remains to order them. The agents in the primary list are ordered by the sequence in which they were fixed. Their proposals are ordered by the sequence in which they were considered. The reciprocal pairings are suitably inserted. The first partner for each primary agent is the preferred partner, thereby generating the preferred partner list.

10. It is easy to see that Conditions PA1-PA3 are satisfied by the pairings and Conditions PR1-PR3 are satisfied by the proposal.

Partner Inheritance

Let s be a state and R be a profile. Let $E(s) = (\emptyset, ..., \emptyset)$. For each partially assigned project $a \in \hat{Z}(s)$, let i be the agent such that $s_i = a$. With no loss of generality, let $a = top(R_i)$. Construct a sub-profile $R'_{\bar{N}(s)}$ such that $a = top(R'_j)$ for all $j \in \bar{N}(s)$. It must be that $f_k(R_{N(s)}, R'_{\bar{N}(s)}) = a$ for some $k \in \bar{N}(s)$. Repeating for all other unpartnered agents, we have the first preferred partners for each. Fixing the assignments of each of them in turn, we can construct the rest of the partner inheritance table. Thus we have $E(s)$ for this state.

It is easy to check that the partner inheritance table $E(s)$ satisfies (E1-E3). To see that E4 is also satisfied, consider any agent and any unpartnered agent that she is the preferred partner for. Note that from the construction above, other agents prefer the project but cannot get it. Thus by LIN they cannot affect the assignment of this project. So at all other states where this project is partially assigned to the same agent, and this agent remains unassigned, she can claim that project when she desires it.

Entitlements

For any interim state, we use the above steps to generate the primary list, the preferred partner list and the proposals. Similarly, for any trading state, we can generate the partner inheritance table. Thus we can get the complete entitlement.

Iterative Procedure

It remains to show that the assignments must work as described, in that the iterative procedure must be followed. Let R be a preference profile and let Γ be the entitlements as determined above.

Assignments at stage 1:

The null state s^0 is an interim state. We use the proposal generated above to find the first acceptable pairing. If there is no acceptable pairing, then all agents in the primary list are assigned their (distinct) top-ranked projects.

Assignments at stage $k+1$:

Let the state s^k be the partial allocation up to stage k. The entitlement $(\alpha(s^k), \beta(\alpha(s^k)), \succ_{s^k}, E(s^k))$ is specified by construction. Fix the preferences and assignments of agents receiving their assignments up to and including stage k.

If s^k is a trading state, then for all $i \in \hat{E}(s^k)$ and $a \in \hat{E}_i(s^k)$, if $top(R_i, \bar{Z}(s^k) \cup \hat{Z}(s^k))$, then we have that $f_i(R) = a$. If there is a set of agents $(i_1, ..., i_k \equiv i_1)$, with $i_j \in \hat{E}(s^k)$ for all j, and $a_{i_j} \in \hat{E}_{i_j}(s^k)$ such that $a_{i_{j-1}} P_{i_j} a_{i_j}$ for all j, then if $f_{i_j}(R) \neq a_{i_{j-1}}$ for any j then PE is violated. Thus all trades must occur.

Instead, if s^k is an interim state, we use the proposal generated above to find the first acceptable pairing. If there is no acceptable pairing, then all agents in the primary list are assigned their (distinct) top-ranked projects. Stop if the resulting state is a terminal state. Otherwise update the state and go to the next stage.

In each stage at least one agent receives an assignment. Thus the procedure is guaranteed to terminate in a finite number of steps. Moreover, assignments are made in according with the PPT rule procedure. This completes the proof.

3

Fair Allocation with Exact Capacity Constraints

3.1 Introduction

A premier engineering college in New Delhi, India, offers as a part of its course structure a number of optional courses or 'electives' from the social science department. Every student, in addition to his or her core science and engineering courses, must complete over the five years of the program a certain number of such electives. The social science department offers a variety of courses, from history to political science to philosophy, even economics.

However, as facilities and faculty are limited, so are the total number of courses offered in any particular term. Thus the college stipulates that a student may opt for only one elective in a term, so that there are enough seats in a term for everybody who desires one.

It is not feasible to offer a course if there are few takers for it, because of the cost of time and facilities. The college wishes to offer only those courses that are enough in demand. Thus there is in effect a minimum capacity constraint operating on a course, such that the college finds it infeasible to offer the course to fewer students than that. Also, since classroom or laboratory sizes are limited, there is a maximum number of students that each course can accommodate.

The college wishes to give priority for these electives to students in their second and third years of the program. It feels this is an appropriate time to finish the elective requirements, as the last two years are typically very intensive in the chosen specialisations. Within this broad preference for students by year, the college gives priority to students with a higher GPA, and also sometimes to a subject-specific progression of prerequisites. Thus, for example, a second year student with a 3.6 GPA who has done elementary microeconomics could be given priority for intermediate microeconomics over a fourth year student with a 3.8 GPA.

Before the start of each term, the college asks students to submit their preferences over some potential set of courses to be offered in that term. Based on this information, plus the capacities and the priorities of the various courses, it must prescribe the course allocation for that term. How should it do so?

In this paper we model such a situation. In particular, we look at the case where a college wishes to offer a selection of courses to its students and requires that each student sign up for exactly one of these courses. In turn, each course has a minimum and maximum capacity and can admit students only within those capacities. For the sake of this paper we make the further assumption that the minimum and maximum capacities for all courses are equal, and the same. That is, for example, each course may admit only exactly twenty-five students, say. While this may seem like an overly restrictive assumption, we believe this is a natural starting point. Relaxing this assumption will not materially change our results, though it will complicate the process.

In keeping with our motivating example, we allow for the fact that a course need not be assigned at all.[1] There are more courses available than may be feasibly assigned together, so in effect the college must determine the solution to a two-part problem: not only must the college decide which selection of courses from the total will be offered, but also which students will be assigned which

[1] This is a departure from other allocation models with minimum constraints. In these papers, each object *must* be assigned up to its minimum constraint.

course.

Course allocations are made on the basis of preference and priorities. Each student has as his or her private information a strict ranking over the available courses, which we call a preference ordering, or simply a preference. This information must be elicited by the college and in general we may wish to award students their preferred courses, as far as possible. On the other hand, each course has a strict ranking over the students, which we call a priority. In contrast to preferences, this priority information is commonly known and fixed, and may be determined by transparent criteria such as GPA, prerequisites, and so on. Priority information captures which students are more eligible for which courses.

So an allocation problem for a college is a collection of students, courses, capacities, course priorities and student preferences. It must use this information to produce a feasible allocation - one in which each student is assigned a course, and every course is assigned to its exact capacity (or to no one). In effect, since the first four are commonly known, the problem becomes one of producing a feasible allocation for any combination of elicited student preferences.

The college wishes that the allocation should satisfy some desirable properties. The first class of properties has to do with fairness. An allocation is deemed unfair for some student if there is a course that she prefers and there is another less eligible student who is assigned that course instead of her. The former student then can be said to have a case of justifiable envy towards the latter. A fair process will avoid this possibility.

The second class of properties has to do with efficiency. An efficient process eliminates waste. In particular, an allocation is inefficient if there is another allocation in which each student receives a course she likes as much, and some student receives a course she strictly prefers. In this case we say that the latter allocation 'Pareto dominates' the former.

However, it has been well documented in the literature that fairness and efficiency are incompatible in the most general environments. We demonstrate by example that this incompatibility persists even in our model with exact capacity constraints.

In the model without minimum constraints, a weaker version of efficiency, called constrained efficiency, is compatible with fairness. A constrained efficient allocation is efficient *within the set* of feasible and fair assignments.[2] The classical Gale-Shapley Deferred Acceptance Rule for instance is simultaneously fair and constrained efficient. We shall describe this rule next. It will provide the starting point for our model.

3.1.1 The Deferred Acceptance Rule

In the general model with no minimum capacity constraints, the Deferred Acceptance (DA) rule is fair and constrained efficient. It works in a series of rounds, as follows.

In the first round, all students apply to their most preferred course. Any course that receives more applications than its maximum capacity is forced to reject the excess students, provisionally accepting the rest. The students that are rejected are those that are the lowest in that course's priority among its pool of applicants. In the next round, all rejected students apply to their next preferred course that has not rejected them already. A course considers its existing applications plus any fresh ones it might receive, and provisionally accepts the top students according to its priority, rejecting the lowest ones that are excess to capacity. If any student is rejected, we go to

[2]If some allocation Pareto dominates a constrained efficient allocation, then it must either be infeasible or unfair to some student.

the next round. The rule terminates in any round in which no student is rejected. All provisional acceptances become final.

Consider the following example. There are three agents $\{1, 2, 3\}$ and three objects $\{a, b, c\}$. Preference and priority information are given in the table below:

Priorities			Preferences		
a	b	c	1	2	3
1	2	3	b	a	a
2	3	1	a	c	b
3	1	2	c	b	c

In the first round, agent 1 applies to her top-ranked object b while agents 2 and 3 apply to a. Since a has two applicants, it rejects the lower-ranked one according to its priority, which is agent 3. Agents 1 and 2 are tentatively assigned b and a respectively. In the next round, agent 3 applies to her next preferred object, which is b. Now b has two applicants, so it rejects the lower-ranked one according to its priority, which is agent 1. Agents 2 and 3 are tentatively assigned a and b respectively. In the next round, agent 1 applies to his next preferred object, which is a. Again, object a must reject one application, and so agent 2 is rejected. In the final round, agent 2 applies to c. There are no more rejections and the rule terminates here, giving us the final allocation $((1, a), (2, c), (3, b))$.

The DA rule is not directly applicable to models with minimum constraints without requiring extra information. We shall discuss the nature of this extra information in more detail and why it is necessary. However, we can find a modified DA rule that is simultaneously fair and constrained efficient. This rule requires no extra information. We call this rule the DA rule with improvements (DAI).

The DAI rule works as follows. We first exogenously select a set of courses that can exactly accommodate all students, and run the conventional DA rule on the restricted environment with only these courses. By the properties of the DA rule, such an allocation will be fair as well as internally constrained efficient [3]. However, our initial selection of courses is arbitrary. There is the possibility that there exists some other allocation, with a *different* selection of courses, that Pareto dominates this one. Our main contribution is to provide a procedure to identify these Pareto improvements whenever they exist. Moreover, we show how we can carry out these efficiency improvements in a manner that preserves the fairness of the original allocation. The composite process is therefore fair and constrained efficient. We also show that the DAI rule needs only one iteration to produce a constrained efficient assignment from any initial selection of courses.

Next we come to questions of strategy. A third property that we would like our rule to satisfy is non-manipulability on the part of students. Since student preferences are private information, the college would like to ensure that students are incentivised to report their true preferences. This is done by eliminating the possibility of undue gain that a student may make by unilaterally reporting false preferences. This is called a profitable manipulation. A rule that can ensure that there is no profitable manipulation for any student is called strategy-proof. It has the additional pleasant hallmark of informational simplicity. Students need only consider what they truly prefer, and need not worry about other students' strategies. The DAI rule is not strategy-proof. However, we show that the opportunities for manipulation on the part of students are limited.

[3] An allocation is internally constrained efficient if it is constrained efficient *for that particular selection* of courses. It need not be constrained efficient in general.

The paper is organised as follows. Section 3.2 provides a literature review on the DA rule and its properties. We also talk about fair assignments and Pareto improvement procedures in other contexts. Section 3.3 is an informal discussion on minimum capacity constraints, our rule, and the justification of our procedure. Section 3.4 begins the formal section of the paper by providing the notation that we use throughout. Section 3.5 discusses the axioms of fairness and efficiency. Here we show the key incompatibility between the two properties. Section 3.6 outlines our Pareto improvement procedure and presents our existence result. Section 3.7 lays out the DAI rule. Section 3.8 discusses its strategic aspects. Section 3.9 concludes. All proofs are relegated to the appendix.

3.2 The Literature

In the context of object allocation, it is well known that Pareto efficiency and fairness are incompatible. There are usually two approaches followed in the literature.

On efficiency, most papers are typically generalisations or special cases of the famous top trading cycles mechanism (TTCM) attributed to David Gale (Shapley and Scarf (1974)), and include inheritance rules (Pápai (2000), Pycia and Ünver (2013)), and sequential priority rules (Svensson (1999)[4], Hatfield (2009), Pápai (2001) etc.) Such rules are Pareto efficient and strategy-proof but not fair. Since we are interested in fairness, we do not pursue this line of research here.

Fairness is the one-sided analogue property to stability in the two-sided matching environment[5] (see for example Abdulkadiroğlu and Sönmez (2003)). In this literature, there are several key results. The famous Gale-Shapley agent-optimal stable mechanism (AOSM)(Gale and Shapley (1962)) always produces a stable matching in the two-sided case. Its direct adaptation to the one-sided model (called the deferred acceptance (DA) rule, described above) always produces a justified-envy-free or fair matching. Moreover, this mechanism Pareto-dominates any other mechanism that is stable/fair (Gale and Shapley (1962)). Also, the DA rule is strategy-proof (Roth (1982b), Dubins and Freedman (1984)). In fact, the DA rule is the only mechanism that is individually rational, fair, non-wasteful, and strategy-proof (Alcalde and Barberà (1994)). There are other characterisations of the DA rule as well (see for example Kojima and Manea (2010)).

In the one-sided model, such as ours, object 'preferences' are interpreted as priorities. Under certain conditions on these priorities, the DA rule and the TTCM can be shown to be equivalent. In particular, Kesten (2006) shows the equivalence of the DA rule and the TTCM when priorities are acyclical, i.e., satisfy a very particular restriction. For more on acyclicity, see also Ergin (2002).

Some recent papers have introduced the problem of minimum constraints to the allocation model. In a closely related paper, Fragiadakis et al. (2012) examine a model where each object has a minimum capacity that must be satisfied. They introduce adaptations of the DA procedure that satisfy combinations of properties. In particular, their ESDA procedure is strategy-proof and satisfies a weak version of fairness and non-wastefulness, while the MSDA procedure satisfies an even weaker notion of fairness, but is non-wasteful and strategy-proof. However, in their model they assume that all objects *must* be assigned, thus all minimum capacities must be satisfied. This is in contrast to our model, where some objects may not be assigned at all. Thus our results are qualitatively different from theirs.

[4] Svensson (1999) deals with serial priority rules, which is a further restriction of sequential priority rules.

[5] Two-sided matching, such as marriage markets, involves private preferences on both sides of the market.

In another related model, Ehlers et al. (2011) and Abdulkadiroğlu and Ehlers (2006) look at the case where agents may be classified according to type, and each object has a minimum capacity for each student type, which must be satisfied. They too show that there might not exist assignments that satisfy standard fairness and non-wastefulness properties, whereas constrained non-wasteful assignments which are fair for same type students always exist. They introduce a 'controlled' version of the (DA) rule with an improvement stage that finds a Pareto optimal assignment among such assignments.

In the context of Pareto improvements, Erdil and Ergin (2008) show how to go from any stable assignment to the student-optimal stable assignment when indifferences may exist in priorities. Indifferences are broken according to an arbitrary tie-breaker, and then the DA rule is used to get a stable assignment. However, because of the arbitrariness of the tie-breaker, and the presence of indifferences, this assignment may not be a Pareto-efficient stable assignment. But the repeated use of what they call the stable-improvement-cycles technique leads inevitably to a Pareto-efficient stable assignment, if one exists. Their key result, similar in spirit to ours, is that if an assignment is Pareto dominated by another stable assignment, then there must exist a stable-improvement-cycle. However, their model considers indifferences and not minimum capacities, and is thus substantially different from our model here.

3.3 An Informal Discussion

Suppose there are M courses and N students. We assume that the capacity of each course is the same (and equal to q) in order to not discriminate between different combinations of courses, i.e., so that every combination of courses is equally likely to appear in a feasible allocation. The total available seats in all the courses is then qM.

If $qM < N$, then no feasible allocation exists, as there are not enough seats for each student. If $qM = N$, then minimum capacities become irrelevant, as there are only as many seats as students. Thus if we run the conventional DA rule as described in the introduction with maximum capacities set as q for each course, we will always get a feasible allocation. Moreover, this allocation will be fair, constrained efficient, and the rule will be strategy-proof.

However, consider the case where $qM > N$. There are more seats available than students to fill them. In particular, let $qm = N$ for some $m < M$. A feasible allocation will thus contain some *selection of m courses* from the whole set, since not all can be simultaneously offered. So now we have a two part problem in which we have to determine not only the subset of m courses that will be offered, but also the allocation of q students to each course.

Once we have the courses selected, we can run the regular DA rule to get a fair and constrained efficient allocation. So let us look at the course selection problem first. There are at least three approaches we could take. We could:

1. Systematically add courses to our collection until we reach m
2. Systematically eliminate courses from the available set until we are left with m
3. Start with an arbitrary selection of m courses and progressively exchange courses until we have the 'right' combination.

We consider each of these approaches in turn.

3.3.1 Course Addition

Assume we have to select the required number of courses somehow. A natural starting point would be the sequential priority rule, in which students are ordered according to some ranking, and successively pick their desired courses until the requisite number is reached. In such a scenario, the remaining students would then be assigned one or other of these selected courses. This approach has a few features that are worth discussing.

Firstly, consider the ordering of students that determines the sequence in which they pick the courses. This assumption has some intuitive justification in the context of course selection (since students can be ranked according to their CGPA), and has parallels in other matching problems. In the US military, for example, cadets are ranked according to a single merit list based on overall performance (Sönmez and Switzer (2011)). In school choice problems, in most countries students are required to take a common exam, which serves as the basis for admissions to many schools (Abizada and Chen (2011)). Fragiadakis et al. (2012) use an overall ranking, which they call a 'master list', as a key ingredient in their models.

Secondly, we may not always be able to respect the selection of earlier students in the sequential priority. This is because the sequential priority ranking may be different from how a course ranks students in its own course priority. So there may be subsequent students in the sequential priority who are higher-ranked in a course's priority than the student who originally selected it. In this case we cannot continue to assign it to the student who selected it without violating fairness.

To see this, consider the following example: Suppose there are four students $\{1, 2, 3, 4\}$ and three courses $\{a, b, c\}$, and each course has a capacity of 2. Suppose the sequential priority is $(1, 2, 3, 4)$, students $1, 3, 4$ most prefer course a, student 2 prefers course b, and the priority for course a is $(3, 4, 1, 2)$. If we follow the sequential priority rule, student 1 goes first and will pick course a. Then student 2 will pick course b. Now if student 1's choice is to be respected, then only one of students $3, 4$ can be assigned course a. But whichever one of them does not get assigned a will feel justified envy towards student 1. Thus this allocation will not be fair.

Even with the sequential priority rule, the selection of a course and its allocation to the selecting student can only be provisional. But this gives rise to the possibility of manipulation. A student higher in the sequential priority may deliberately pick a course in order to affect the choices of later students. In turn, this might grant her a preferred course. Such a rule may not be strategy-proof.

3.3.2 Course Elimination

Suppose instead that we start with the entire set of courses, looking for a way to eliminate undesired courses. Picture the DA rule, and suppose that in the first round students apply to their preferred course among the entire set on offer. The provisional allocation in this round typically will be infeasible. Infeasibility in this context comes in two flavours: (1) There may be one or more courses that receive more applications than their capacities; and (2) There may be one or more courses that receive less than their capacities.

The DA rule tells us what to do in the first case, i.e., of *oversubscription*. We simply appeal to the relevant course's priority, and reject the application of the lowest-ranked students that are excess to capacity. But the second possibility (which we call *undersubscription*) is more problematic. What if there is more than one course that receives less applications than it requires? Do we eliminate only one of them? If so, which one? Alternatively, do we reject the applications of some of those students? If so, which ones?

We could assume a ranking over courses, such that we eliminate the lowest-ranked undersubscribed course and reassign its applicants. However, this introduces a discrimination between courses. It may result in the same set of courses being offered in most allocations. Plus, we would like to avoid a situation where a History course is always favoured over a Political Science course, say.

Instead, we could assume a ranking over students, such that an undersubscribed course that receives applications from higher-ranked students is retained in favour of another that has only lower-ranked applicants. However, consider the following scenario: students who have applied to course a are the highest-ranked in its priority, but lowest-ranked in the overall ranking. Course a is undersubscribed and so these students are rejected. Course a is eliminated as a result. These students apply to course b. They are higher ranked in course b's priority than some of the existing applicants, who are now in turn rejected. These students prefer course a to any other course. But course a is now eliminated, so either they cannot apply to it, or doing so violates fairness in terms of the original rejected students. There are also significant efficiency losses in this scenario.

We also cannot let the choice of course to be eliminated depend on student preferences without running the risk of manipulation.

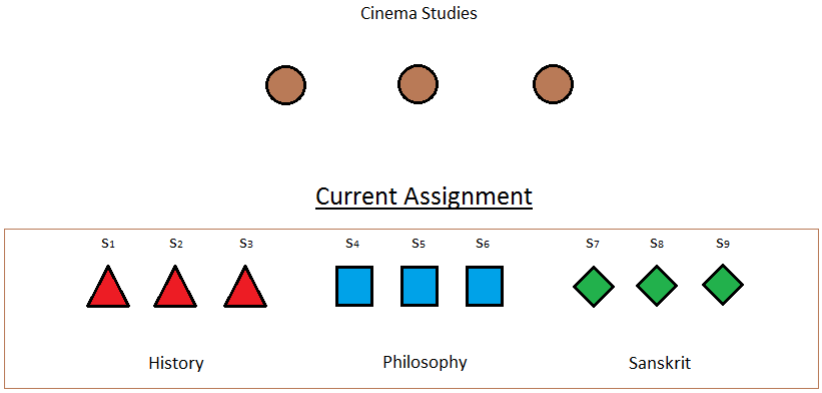
3.3.3 Our Model

In this paper we take the third approach. We design a rule that first makes an allocation with some arbitrary selection of courses, and then consider efficiency improvements.

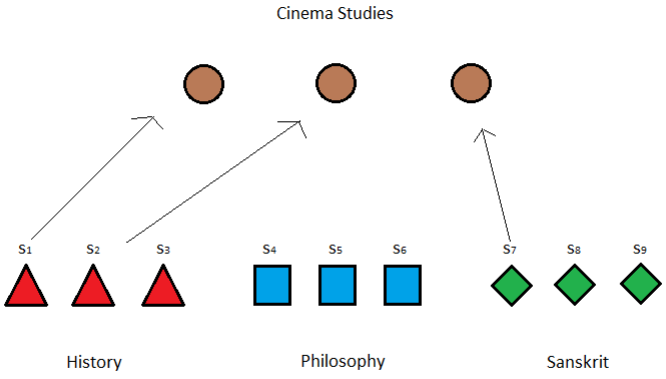
So first we arbitrarily select m courses that can exactly accommodate all students, i.e, such that $qm = N$. We have left out some courses at this stage. We then run the classical DA rule on the restricted environment with just this selection of courses.

By the properties of the DA rule, we are guaranteed an allocation that is fair and internally constrained efficient. As mentioned earlier, internal constrained efficiency means that there is no other fair allocation *with the same selection of courses* that Pareto dominates this one. But this allocation may not be constrained efficient in general. In particular, since the initial selection of courses was exogenous, and did not depend on the preference profile, it might be that there is an unassigned course that some students prefer to their current allocation.

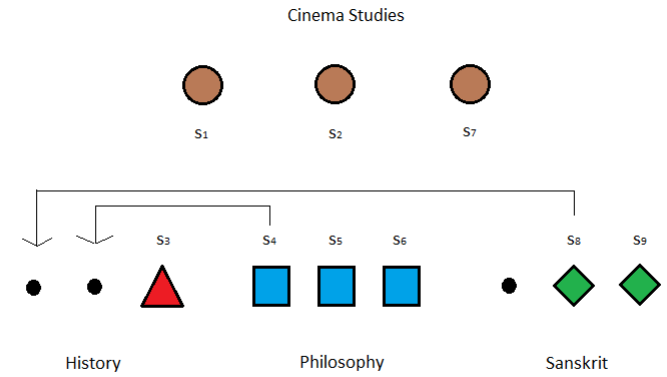
So consider the following scenario where students s_1 to s_9 are assigned courses in History, Philosophy and Sanskrit in a fair and internally constrained efficient manner. There are three seats in each course. But suppose that there is additionally a course in cinema studies which could be offered to them.



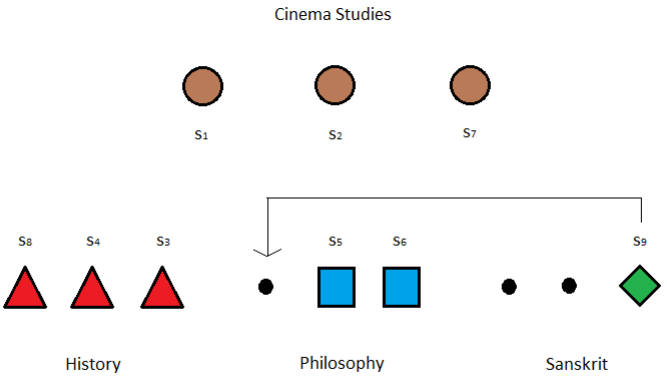
If there are enough claimants for the Cinema Studies course, i.e., students who desire the course at the current allocation, it may be possible to transfer the top claimants to the course (see figure below).



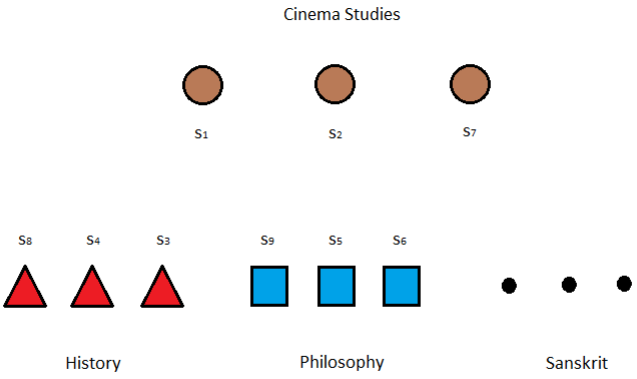
But doing so will leave vacancies in the courses they leave behind. Thus this provisional allocation is not feasible, though it is fair and also a Pareto improvement on the original allocation. So next we see if we can fill the slots that are left behind. Note that at each stage if we are able to fill a slot we must do so by moving the *top claimant of that slot* according to the priority of the relevant course. This is essential to ensure fairness of the overall allocation. Suppose we can do so as below.



This will leave new vacancies, which we fill again, if possible.



Suppose the result is an allocation as in the figure below.



By following this sequential improvement procedure, we have achieved not only a feasible allocation, but one that is also fair and a Pareto improvement on the original allocation.

In essence we have built a Pareto improvement chain for every slot in the new course, such that each chain terminates in a slot in some other course. The course where the chains terminate can

be removed and replaced with the new course. Of course, the scenario can be more complicated, in that the chains may be interlinked with those for another course. But the key insight is that if at any point if we can find a collection of chains, one for each slot in a course, such that all chains terminate in slots in the same courses, then we can add the new courses and delete the old ones. This will give us a Pareto improving allocations that is also fair and feasible. In particular, the original allocation is not constrained efficient.

In fact, our first theorem characterises precisely this situation. It shows that the original allocation is internally constrained efficient but not constrained efficient if and only if such a collection of improvements exists. That is, there is a sequence of fairness-preserving moves that leads us from every slot in a new course to every slot in some course that will be now dropped. We also outline a procedure to find such improvements.

We thus build a composite rule, the Deferred Acceptance with Improvements (DAI), which works as follows. We exogenously select some courses, run the DA rule to get an allocation, and check it for improvements. If improvements exist, and if there is more than one possible improvement, then we select[6] one of them and run the DA rule again for the new selection of courses. This resulting allocation will be constrained efficient and fair. We also show that a constrained efficient allocation can be found by just running this entire procedure once.

We then test the DAI rule for strategic implications. The DAI rule is not strategy-proof. But the possibilities of manipulation are limited. We show that no student can induce the favourable addition of a course or favourably change her assignment within the set of assigned courses via a manipulation. The reason the rule is not strategy-proof is that a student can cause a potentially Pareto improving course to be removed from contention in favour of another course that is also in contention and that she prefers. We demonstrate by example and show that this is the only way a student can manipulate.

It should be pointed out that for the same preference profile, DAI rules can result in different outcomes, depending not only on the initial choice of courses, but also possibly on the later selection from among improvements, in case there are more than one of them. Thus there is not a one-to-one correspondence for DAI rules between preference profiles and allocations.

In the sections that follow we formally demonstrate the results and arguments above.

3.4 Notation and Definitions

- There is a finite set of *students* $\mathcal{S} = \{s, t, u, ...\}$, and a finite set of *courses* $\mathcal{C} = \{c, d, e, ...\}$.
- Each course $c \in \mathcal{C}$ has an *exact capacity* of q. We assume that $|\mathcal{S}| = qm$ for some integer m and that $|\mathcal{C}| > m$.
- An *allocation* is a mapping $\mu : S \cup C \to 2^{\mathcal{S} \cup \mathcal{C}}$ such that:
  1. $\mu(s) \in C$ for all $S \in \mathcal{S}$.
  2. $\mu(c) \subseteq S$ for all $c \in \mathcal{C}$.
  3. $\mu(s) = c$ if and only if $s \in \mu(c)$.

  In an allocation, every student is mapped to a course, every course is mapped to a set of students, and a student is mapped to a course if and only if she belongs to the set of students

[6] The selection of improvements is a non-trivial exercise, and we shall discuss it in greater detail in a later section.

that the course in turn maps to. For an allocation μ, a student s and a course c, $\mu(s)$ and $\mu(c)$ refer to student s and course c's *assignment* in μ, respectively.

- An allocation μ is *feasible* if $|\mu(c)| \in \{0, q\}$ for all $c \in \mathcal{C}$, and $|\mu(s)| = 1$ for all $s \in \mathcal{S}$. An allocation is feasible if every course is assigned either to no student, or to exactly q students, and every student is assigned to exactly one course. Let $\mathcal{A}$ denote the set of all feasible allocations.

- Preferences over assignments are strict. Formally, student $s \in \mathcal{S}$ has *preferences*, denoted R_s, that are given by a binary relation over $\mathcal{C}$. The binary relation is reflexive (for all c, cR_sc), complete (for all c, d, cR_sd or dR_sc), transitive (for all c, d, e, cR_sd and dR_se imply cR_se) and antisymmetric (for any c, d, cR_sd and dR_sc imply $c = d$). Here cR_sd is interpreted as 'course c is at least as good as course d for student s under preferences R_s'. The associated strict relation is given by P_s, such that cP_sd if cR_sd and $c \neq d$. For any c, d, cP_sd means 'c is preferred by s to d under preferences R_s'. We assume that all courses are acceptable to all students.

- Agent preferences over allocations are selfish, in that they care only about the assignment they receive. Agents are indifferent between all allocations that give them the same assignment. An agent's preferences between two allocations that give her different assignments are governed by her preferences over the respective assignment she receives.

- A collection of preferences for all agents is called a *preference profile*, or simply a *profile*, and is denoted by $R = (R_1, ..., R_N)$. The set of all preference profiles is $\mathcal{R}$. In this model we shall usually suppress reference to $\mathcal{R}$, with the understanding that we operate on the full domain of preferences everywhere. As is the convention, we write R_{-i} for a sub-profile of preferences of all agents other than i. Similarly, for a subset of agents M, we write R_M and R_{-M} to denote the sub-profile of preferences of agents in subsets M and $\mathcal{N} \setminus M$, respectively.

- Each course $c \in \mathcal{C}$ has a strict ranking $\succ_c$ over students, which we call a *priority*. A priority is given by a reflexive, complete, transitive and antisymmetric binary relation over $\mathcal{S}$. For any priority $\succ_c$ and students s and t, $s \succ_c t$ means that 'student s has a higher priority for course c than student t'. We assume that all students are acceptable to all courses. We refer to a collection of priorities for all courses as a priority structure, and denote it as $\succ$. The priority structure is exogenous, fixed, and common knowledge.

- An *allocation problem* is the tuple $(\mathcal{S}, \mathcal{C}, q, \succ, R)$. When $\mathcal{S}, \mathcal{C}, q, \succ$ are fixed, as in what follows, we will refer to an allocation problem simply as R.

- An *allocation rule* (or simply a *rule*) $f : \mathcal{R} \to \mathcal{A}$ solves every allocation problem, i.e., associates every preference profile R with a feasible allocation. For any student s, $f_s(R)$ is the assignment she receives at preference profile R according to the rule f. Similarly, for any subset of students T, $f_T(R)$ is the $|T|$-dimensional vector of assignments of T at R, according to f.

3.5 Properties of Allocation Rules

The first property we wish to implement is fairness. An allocation is unfair if there is a student who does not get a course she desires, and loses out to some other student who is below her in that

course's priority. If that is the case, then this student justifiably envies the other student. We wish to avoid such a situation. Formally:

DEFINITION **2.** For a preference profile R, An allocation μ is *unfair to student* s if there is a course c and a student t such that $cP_s\mu(s)$, $t \in \mu(c)$ and $s \succ_c t$, i.e., if student s prefers student t's assignment (course c) to her own, and also has a higher priority for c than t. An allocation μ is *fair* if it is not unfair to any student. A rule f is *fair* if the allocation $f(R)$ is fair for every preference profile R.

The second class of properties we are interested in has to do with efficiency. In its strong form, we say an allocation is efficient if there is *no other* allocation that Pareto dominates it, i.e., in which all students are at least as well off according to their preferences, and at least one student is strictly better off, in the sense that she is assigned a course that she prefers. Formally:

DEFINITION **3.** For a preference profile R, An allocation μ is *Pareto efficient* if there is no other allocation ψ such that $\psi(s)R_s\mu(s)$ for all $s \in \mathcal{S}$ and $\psi(t)P_t\mu(t)$ for some $t \in \mathcal{S}$. A rule f is *Pareto efficient* if, for all preference profiles R, the allocation $f(R)$ is Pareto efficient.

Pareto efficiency is incompatible with fairness in general. The following example, which we adapt from Roth (1982a), demonstrates this.

EXAMPLE **1.** Let $s_1, s_2, s_3, s_4, s_5, s_6$ be six students and let a, b, c be three schools with exactly two seats each. The constraint structure, priority structure and preference profile is given in the following table.

$\succ_a$	$\succ_b$	$\succ_c$	P_{s_1}	P_{s_2}	P_{s_3}	P_{s_4}	P_{s_5}	P_{s_6}
s_1	s_3	s_3	ⓑ	ⓑ	ⓐ	ⓐ	a	a
s_2	s_4	s_4	a	a	b	b	b	b
s_5	s_1	s_1	c	c	c	c	ⓒ	ⓒ
s_6	s_2	s_2						
s_3	s_5	s_5						
s_4	s_6	s_6						

It is easy to check that the allocation in boxes is the only fair allocation, which is not Pareto efficient as it is dominated by the allocation in circles. However, the allocation in circles, while Pareto efficient, is not fair as s_5 envies s_4 for course a.

So we weaken our definition of efficiency to constrained efficiency. An allocation is constrained efficient if there is *no other feasible and fair allocation* that Pareto dominates it. Formally:

DEFINITION **4.** An allocation μ is *constrained efficient* if there is no other feasible and fair allocation ψ such that $\psi(s)R_s\mu(s)$ for all s, and $\psi(t)P_t\mu(t)$ for some t.

For an allocation μ, let C_μ denote the *set of courses assigned in* μ, i.e., $C_\mu = \{c \in \mathcal{C} | \mu(s) = c$ for some $s \in \mathcal{S}\}$. Next we define the notion of internal constrained efficiency, which has to do with efficiency with respect to the courses selected in a particular allocation.

DEFINITION **5.** An allocation μ is *internally constrained efficient* if there is no other fair and feasible allocation ψ with $C_\psi = C_\mu$, such that $\psi(s)R_s\mu(s)$ for all s, and $\psi(t)P_t\mu(t)$ for some t.

An allocation that is Pareto efficient is also constrained efficient, and an allocation that is constrained efficient is also internally constrained efficient, but the reverse implications may not be true.

3.6 Pareto Improvements

In this section we elaborate on a technique that allows us to recover constrained efficiency from an internally constrained efficient allocation. Note that an internally constrained efficient allocation need not be constrained efficient, because we do not consider courses outside of the allocation when evaluating the former. There may be some other course that is not assigned, that enough students prefer, such that we can suitably transfer and reassign students making some of them better off without making anybody else worse off, while preserving fairness. We build the notion of an improvement to capture such situations.

3.6.1 Pareto Chains

Since each course has an exact capacity q, we say that each course $c \in \mathcal{C}$ has q *slots* $\{o_1^c, o_2^c, ..., o_q^c\}$, denoting the available positions in the course. We denote a generic slot for a course c as o^c.

For an allocation μ and a course c, let $D_\mu(c) = \{s \in \mathcal{S} | cP_s\mu(s)\}$ be the set of students who *desire* c *at* μ. Let D_μ refer to the collection $\{D_\mu(c)\}_{c\in\mathcal{C}}$. We call D_μ the *claimant profile at* μ.

Also, for course c and any subset of students T, let $top_k(\succ_c, T)$ denote the k *top students in* c*'s priority among students in* T. It should be noted that, depending on the size of the set T, that $top_k(\succ_c, T)$ may contain fewer than k students.

First, we define a Pareto chain. Suppose we are given an allocation at a preference profile. Consider some course that is unassigned at this allocation. A Pareto chain is a sequence of distinct students and their assignments such that the first student in the chain prefers this unassigned course, and each subsequent student prefers the assignment of the student immediately before her in the chain. A Pareto chain will be associated with a particular slot in the unassigned course.

Definition **6.** Let μ be an allocation, let R be a preference profile, let $c \notin C_\mu$ be a course, and let o^c be a slot in c. We say that a finite sequence of students and courses $\gamma = \{(s_k, c_k)\}_{k=1}^K$, with $1 \leq k \leq K$, is a *Pareto chain originating at* o^c if:

1. $s_i \neq s_j$ for all $i, j \in \{1, ..., K\}$
2. $s_i \in \mu(c_i)$ for all $i \in \{1, ..., K\}$
3. Set $c_0 \equiv c$. Then $c_{i-1}P_{s_i}c_i$ for all $i \in \{1, ..., K\}$.
4. For each c_{i-1}, $s_i \in top_q(\succ_{c_{i-1}}, D_\mu(c))$.

We call a particular pair (s_k, c_k) a *link* in the chain. The first condition requires that all students in the chain be distinct, while the second condition pairs each student in a link with her corresponding assignment. The third condition highlights the chain quality, by requiring that each student prefer the assignment of the previous student in the chain, with the first student preferring the unassigned course. The fourth condition states that each student in the chain is among the top q students for the course she desires among those who desire it at that allocation.

We now introduce some notation to simplify exposition. For a chain γ and a particular k, we denote the corresponding s_k as $\gamma^s(k)$ and the corresponding c_k as $\gamma^c(k)$. We denote the last student in the chain, $\gamma^s(K)$, by $\bar{\gamma}^s$, and the last course in the chain, $\gamma^c(K)$, by $\bar{\gamma}^c$. Similarly, we denote the origination course for a chain γ by $\gamma^c(0)$. Denote a collection of chains satisfying the above properties by Γ.

So, for a course c and some slot o^c, if a Pareto chain exists, it looks like this, where the arrows mark the direction of preference, and where $\gamma^c(0) = c$:

$$\begin{array}{c} \gamma^c(0) \\ \hline \uparrow \\ (\gamma^s(1), \gamma^c(1)) \\ \uparrow \\ (\gamma^s(2), \gamma^c(2)) \\ \uparrow \\ \dots \\ \uparrow \\ (\bar{\gamma}^s, \bar{\gamma}^c) \\ \hline \end{array}$$

A Pareto chain is designed to capture a potential sequence of transfers that would make some students better off, while preserving the assignments of other students, in a fair manner. That is:

DEFINITION **7.** For an allocation μ, a course $c \notin C_\mu$ and a Pareto chain γ associated with some slot $o^c \in c$, we say that an allocation μ' is an *upgrade of* μ *via* γ if $\mu'(\gamma^s(k)) = \mu(\gamma^s(k-1))$ for all k and $\mu'(s) = \mu(s)$ for all other students.

It is clear that an upgrade, if it exists, is a Pareto improving allocation, since at least one student in the chain is strictly better off and all other students are equally well off. It is also fair. However, an upgrade may not be feasible. The new assignment, formed by giving students in the Pareto chain their desired courses, may not satisfy the requirement of each course being assigned to exactly q students. To additionally ensure feasibility, we now provide the notion of an *improvement*, which is a collection of distinct Pareto chains that originate and terminate 'coherently'.

DEFINITION **8.** Let μ be an allocation with courses C_μ. Let C be a selection of m courses with $C \neq C_\mu$. A collection of chains Γ^C is an *improvement of* μ if:

1. For any $c \in C \setminus C_\mu$, we have that $|\{\gamma_i \in \Gamma^C : \gamma_i^c(0) = c\}| = q$.
2. For any $c' \in C_\mu \setminus C$, we have that $|\{\gamma_i \in \Gamma^C : \bar{\gamma}_j^c = c'\}| = q$.
3. $\bigcap_{\Gamma^C} \gamma_j^s(k) = \phi$.

The first condition requires there to be a Pareto chain in the collection for every slot in every course that we seek to add. The second condition requires that there be a Pareto chain terminating in every slot in a course that we seek to drop. The third condition requires each student in the collection of chains to be distinct.

We are now ready to state our first theorem. Theorem 2 says that an internally constrained efficient allocation is not constrained efficient when we can find an improvement for some other set of courses such that the chains terminate 'coherently'. In other words, when we perform the upgrade related to the improvement, then the resulting allocation is feasible.

THEOREM **2.** *Let R be a profile and let μ be an internally constrained-efficient allocation for some selection of courses C_μ at R. Then, μ is not constrained efficient if and only if there exists a selection of courses $C \neq C_\mu$ such that Γ^C is an improvement of μ.*

The proof is in Appendix A. We provide an informal discussion here.

Theorem 2 characterises the situation where an allocation is internally constrained efficient but not constrained efficient. If this is true, then there must be some other courses such that, when we build the Pareto chains for slots in those courses, the resulting collection of chains is an improvement of the original allocation, and also that the resulting upgrade is feasible.

In one direction, it is clear that if such an improvement exists, the resulting upgrade is feasible, fair, and Pareto dominates the original allocation, and thus the latter cannot be constrained efficient.

To prove the converse, we assume that the allocation is not constrained efficient. Then there must be a constrained efficient allocation which Pareto dominates it. Furthermore, this constrained efficient allocation must differ from the original allocation by at least one course. (Otherwise the original course could not have been internally constrained efficient.) We build the collection of Pareto chains for the courses in this allocation, and show that the collection is an improvement according to the definition, and that each chain terminates in a course that is not present in the constrained efficient allocation. We build these chains iteratively, identifying the top claimant for a course in each step, and upgrading students' assignments as we go along. The key insight is that at each stage some student is strictly better off, and no student is ever worse off, so eventually we get to the constrained efficient allocation, which is feasible and fair.

Thus we have a procedure that allows us to evaluate the constrained efficiency of an allocation. We use this feature to define our composite rule in the next section.

3.7 Deferred Acceptance with Improvements

The Deferred Acceptance with Improvements (DAI) rule operates in three stages. Firstly, we select some courses and make assignments following the classical DA procedure, restricting preferences to just those courses. By the properties of the DA rule, such an allocation is fair and internally constrained efficient. We then look for improvements to this allocation. If we find one, we build a new set of courses accordingly. The third stage is a repetition of the first stage for the new selection of courses. The resulting allocation will be fair, feasible and constrained efficient.

Initialisation

Exogenously select a subset of courses $C \subset \mathcal{C}$ such that $|C| = m$. Then $q|C| = |\mathcal{S}|$, i.e., there are just enough courses to exactly accommodate all students.

Assignment Step #1

Restrict preferences to courses in C, and run the classical Deferred Acceptance procedure restricted to these courses. That is:

1. In the first round, all students apply to their most preferred course in C. Any course c receiving more than q applicants provisionally accepts the applications of the top q students among the applicants according to $\succ_c$, rejecting the rest.

2. In any subsequent round, rejected students apply to their most preferred course in C that has not rejected them already. Each course c evaluates its existing applicants (if any) plus new applicants (if any), provisionally accepting the top q applications according to $\succ_c$, and rejecting the rest. If any course rejects a student, we go to the next round. The procedure stops in a round where there are no more rejections.

It is clear that since $q|C| = |\mathcal{S}|$, that this procedure will terminate in a finite number of steps with each course in C receiving exactly q applicants, since any course receiving $k > q$ applicants in any round must reject $k - q$ applications. Courses outside C receive no applicants. Moreover, each student is assigned to a course in C. Thus the resulting allocation is feasible. By the properties of the classical DA rule, this allocation is fair and internally constrained efficient.

By Theorem 2, we know that to check if this allocation is constrained efficient, we need only to look for improvements. So we run the Pareto improvement procedure as follows:

Improvement Step

Let μ be the allocation generated by the Assignment Step.

Recall that for an allocation μ and a course c, $D_\mu(c) = \{s \in \mathcal{S} | cR_s\mu(s)\}$ is the set of students who desire c at μ. Similarly, $D_\mu = \{D_\mu(c)\}_{c\in\mathcal{C}}$ is the claimant profile at μ.

Also, for course c and any subset of students T, $top_k(\succ_c, T)$ denotes the k top students in c's priority among students in T.

1. For each $c \notin C_\mu$, we build the Pareto chains for the slots of c as follows:
   (a) Let $S_c = top_q(\succ_c, D_\mu(c))$. If $|S_c| < q$, discard this course. If not:
   (b) Create a chain γ_i for each slot $o_i^c \in c$, with $\gamma_i^s(1) = s_i$ for some distinct $s_i \in S_c$, and let $\gamma_i^c(1) = \mu(s_i)$.
   (c) For every $k > 1$, and every i, let $\gamma_i^s(k) = top(\succ_{\gamma_i^c(k-1)}, D_\mu(\gamma_i^c(k-1)))$ if this student exists, and let $\gamma_i^c(k) = \mu(\gamma_i^s(k))$.
   (d) At any stage, if the same student repeats in any two chains, remove her from the chain she less prefers and remove all subsequent students in that chain.
   (e) At any stage, if all chains contain the same course, then remove all additional links and go on to the next course.
   (f) Otherwise repeat from step (c) until no other links can be formed.
   (g) Repeat the above steps for all other courses $c \notin C_\mu$.
2. When all chains are built, search for a collection of subchains Γ such that:
   (a) For any c with a $\gamma \in \Gamma$ such that $\gamma^c(0) = c$, we have that $|\{\gamma_i \in \Gamma : \gamma_i^c(0) = c\}| = q$.
   (b) For any c' with a $\gamma \in \Gamma$ such that $\bar{\gamma}^c = c'$, we have that $|\{\gamma_i \in \Gamma : \bar{\gamma}_i^c = c'\}| = q$.
   (c) $\bigcap_{\Gamma^C} \gamma_{ij}^s(k) = \phi$.

If we find such a collection of subchains (and there may be more than one collection), we *select*[7] one of these collections Γ, generate C' from C by adding the courses corresponding to the first nodes of chains in Γ and deleting the courses corresponding to the final nodes of chains in Γ.

Assignment Step #2

Finally, we repeat the Assignment Step with C'. The resulting allocation will be feasible, fair and constrained efficient.

[7] We discuss in the next sub-section how to select from the possible collections.

3.7.1 Example

Let $\{a, b, c, d, e\}$ be five courses and let $\{1, 2, 3, 4, 5, 6\}$ be six students, and suppose that $q = 2$. The preference and priority information is given in the following table.

$\succ_a$	$\succ_b$	$\succ_c$	$\succ_d$	$\succ_e$	P_1	P_2	P_3	P_4	P_5	P_6
1	6	2	1	4	d	a	d	e	e	b
4	3	5	3	2	a	e	b	d	b	a
5	5	1	6	3	b	d	c	a	a	d
6	2	3	4	5	c	c	a	c	c	e
2	1	4	5	6	e	b	e	b	d	c
3	4	6	2	1						

Suppose we start with the initial course selection $\{a, b, c\}$. Then the DA rule produces the allocation given below in boxes:

$\succ_a$	$\succ_b$	$\succ_c$	$\succ_d$	$\succ_e$	P_1	P_2	P_3	P_4	P_5	P_6
1	6	2	1	4	d	a	d	e	e	b
4	3	5	3	2	a	e	b	d	b	a
5	5	1	6	3	b	d	c	a	a	d
6	2	3	4	5	c	c	a	c	c	e
2	1	4	5	6	e	b	e	b	d	c
3	4	6	2	1						

Now, to check if the allocation is constrained efficient, we build the potential Pareto chains for unassigned courses $\{d, e\}$.

d		e	
o_1^d	o_2^d	o_1^e	o_2^e
↑	↑	↑	↑
(1,a)	(3,b)	(4,a)	(2,c)
↑	↑	↑	
(2,c)	(5,c)	(5,c)	

Thus we have the choice of selecting either e or d, and dropping c.

3.7.2 Selecting the Improvement

As can be seen from the example above, there may be several improvements that may arise as a result of the procedure. We have to select one of them. We could of course select one of these improvements arbitrarily, and we would be left with a fair and feasible allocation that is a Pareto improvement of the original. But how we select the improvement will matter in terms of the overall efficiency of the procedure. We claim that it is possible to reach from any arbitrary initial selection of courses to a selection that guarantees a constrained efficient allocation via a single round of the Improvement Step. This requires us to always be able to select a constrained efficient allocation at this stage.[8]

[8]Or, alternatively, to select a set of courses that, when we run the Assignment Step #2 with those courses, produces a constrained efficient allocation for that profile.

In what follows, we describe how to efficiently select a improvement from all the ones that are available after the Improvement Step.

First we need some notation. Let R be a preference profile and let μ be the allocation produced by the DA rule for some selection of courses C. We call μ the *base allocation at R for C*. Let $\mu^1, ...\mu^k$ be k allocations resulting from k different improvements generated by the Improvement Step. For each μ^i, let $S(\mu, \mu^i) = \{s \in \mathcal{S} : \mu^i(s) \neq \mu(s)\}$ be the set of all students whose assignments change from μ to μ^i. For any two allocations μ^i, μ^j, we say that μ^i *dominates* μ^j *with respect to* μ if $S(\mu, \mu^j) \subseteq S(\mu, \mu^i)$ and for all $s \in \mathcal{S}$, $\mu^i(s) R_s \mu^j(s)$. In other words, one allocation dominates another with respect to the base allocation if all the students whose assignments change in one also have their assignments change in the other, and all students are at least as well off in the former.

Selecting an Improvement

Let R be a preference profile and let μ be the base allocation for some selection of courses C. Let $\mu^1, ...\mu^k$ be k the allocations resulting from k different improvements generated by the Improvement Step. For each μ^i, let $S(\mu, \mu^i)$ be as defined above, i.e., the set of all students whose assignments change from μ to μ^i.

For any two μ^i, μ^j:

1. If μ^i dominates μ^j with respect to μ, we discard μ^j.
2. If not, we keep both courses.

We repeat the above check for every remaining pair of improvements. We call the set that remains after elimination of dominated improvements the set of *valid improvements.* From this set, we pick an improvement arbitrarily.

Moreover, any selection from the set of valid improvements would get us to a constrained efficient allocation, at most by running the Assignment Step #2. To see this, let μ_v be a valid improvement. Suppose μ_v itself has an improvement μ'_v. Then it follows that μ'_v is an improvement of μ. But then, if μ'_v dominates μ_v (which an improvement does), then μ_v could not have been a valid improvement of μ. Thus a selection from the set of valid improvements does not itself have an improvement, and is thus externally constrained efficient. To get a constrained efficient allocation, at best we need to run the Assignment Step #2 with these courses.

Example

To continue with the example in the previous section, we have two improvements. The associated course selections are $C_1 = \{a, b, d\}$ and $C_2\{a, b, e\}$.

If we select the improvement C_1 and the courses $\{a, b, d\}$, we get the following allocation:

$\succ_a$	$\succ_b$	$\succ_c$	$\succ_d$	$\succ_e$	P_1	P_2	P_3	P_4	P_5	P_6
1	6	2	1	4	d	a	d	e	e	b
4	3	5	3	2	a	e	b	d	b	a
5	5	1	6	3	b	d	c	a	a	d
6	2	3	4	5	c	c	a	c	c	e
2	1	4	5	6	e	b	e	b	d	c
3	4	6	2	1						

Instead, if we select $C_2 = \{a, b, e\}$, the allocation with courses $\{a, b, e\}$ is:

$\succ_a$	$\succ_b$	$\succ_c$	$\succ_d$	$\succ_e$	P_1	P_2	P_3	P_4	P_5	P_6
1	6	2	1	4	d	a	d	[e]	e	[b]
4	3	5	3	2	[a]	[e]	[b]	d	b	a
5	5	1	6	3	b	d	c	a	[a]	d
6	2	3	4	5	c	c	a	c	c	e
2	1	4	5	6	e	b	e	b	d	c
3	4	6	2	1						

We see that $S(C_1) = \{1, 2, 3, 5\}$ and $S(C_2) = \{2, 4, 5\}$. Thus no improvement dominates the other, and so we can pick either selection of courses. However, we shall see in the next section that this would have strategic implications.

3.7.3 Properties of the DAI Rule

We see from the example that the resulting allocation is constrained efficient and fair. Theorem 3 tells us that this is always true.

Theorem **3.** *The DAI procedure is fair and constrained efficient.*

Proof: The properties of the DA rule guarantee fairness, and the absence of an improvement guarantees constrained efficiency by Theorem 2. ■

3.8 Strategic Aspects of the DAI Rule

In this section we explore the strategic properties of the DAI rule. Strategy-proofness is a property that ensures that it is a dominant strategy for all students to truthfully report their preferences. In a strategy-proof rule, no student can do strictly better by falsely reporting some other preferences, given everybody else's reported preferences. Formally:

Definition **9.** A mechanism f is *strategy-proof* if, for every profile of preferences R, every student s, and every report R'_s, we have that $f_s(R) R_s f_s(R'_s, R_{-s})$.

The DAI rule is not strategy-proof, even though the underlying DA rule is. There can be situations where a student can cause a potentially Pareto improving course to be removed from contention in favour of another. Note that she would not have any incentive to do this if she is not part of both improvements. But if she is part of both improvements, then she may be differently well off if one course is picked over the other. It is possible that an agent could render one of those improvements infeasible. If there were no other options, this would cause the rule to select the other course. And if this makes her better off, there is a potential for manipulation.

We show the manipulability of the rule by example.

3.8.1 Example

To continue the example in the previous section, let us reproduce in the tables below the allocations that would result if we selected courses $\{a, b, d\}$ or $\{a, b, e\}$ at the Improvement Step, and re-ran the Assignment Step:

If we select courses $\{a, b, d\}$, after the Assignment Step we would get the following allocation:

$\succ_a$	$\succ_b$	$\succ_c$	$\succ_d$	$\succ_e$	P_1	P_2	P_3	P_4	P_5	P_6
1	6	2	1	4	[d]	[a]	[d]	e	e	[b]
4	3	5	3	2	a	e	b	d	[b]	a
5	5	1	6	3	b	d	c	[a]	a	d
6	2	3	4	5	c	c	a	c	c	e
2	1	4	5	6	e	b	e	b	d	c
3	4	6	2	1						

Instead, if we select $\{a, b, e\}$, the Assignment Step would produce:

$\succ_a$	$\succ_b$	$\succ_c$	$\succ_d$	$\succ_e$	P_1	P_2	P_3	P_4	P_5	P_6
1	6	2	1	4	d	[a]	d	[e]	[e]	[b]
4	3	5	3	2	[a]	e	[b]	d	b	a
5	5	1	6	3	b	d	c	a	a	d
6	2	3	4	5	c	c	a	c	c	e
2	1	4	5	6	e	b	e	b	d	c
3	4	6	2	1						

Note that agent 5's assignments are changing in both allocations. Thus he is part of an improvement in both cases. But agent 5 can manipulate. To see this, consider a preference R'_5 with cP'_5b. It is easy to see that the improvement with courses $\{a, b, d\}$ would no longer exist. Thus the only improvement would be with courses $\{a, b, e\}$, and the rule would select this one. But now agent 5 receives e which he prefers to b, which is what he was assigned in the other improvement. Thus agent 5 has an incentive to drop the improvement that gives him b in favour of the one that gives him e.

3.9 Conclusion

In this chapter we have formulated a fair and constrained efficient solution to the problem of allocating courses with exact capacity constraints to students based on preferences and priorities. In the process we have outlined a Pareto improvement procedure that allows us to find a constrained efficient allocation from any internally constrained efficient allocation, in a manner that preserves the fairness of the original allocation.

This is particularly useful in the context of the college in our motivating example. For any number of reasons, the college may have a pre-specified selection of courses that it would rather offer to the students. Using this as the default offering, our rule allows the college to replace some courses with others when necessary, especially when enough students would rather have some other course than one from their selection. The default selection of courses will be altered in a fair and efficiency-improving way.

The DAI rule is not strategy-proof. Yet the possibilities for manipulation are limited.

A future direction of research in this paper would be to consider the time complexity of the algorithm underlying the DAI rule. This would be useful especially when comparing the efficiency of the rule as compared to the brute force approach. We have already shown that running the procedure once is enough to guarantee a fair and constrained efficient allocation. The procedure involves two repetitions of the DA procedure, which is polynomial time. It is the building of the

Pareto chains that will determine the time efficiency of the procedure. Erdil and Ergin (2008) show that their improvement algorithm is polynomial time, so a similar result in this case would be of some use as well.

We can extend the analysis in this chapter in multiple directions. It can be seen that the procedure we have outlined can also be used in cases where courses have different capacities from each other. Care must be taken however while constructing improvements in this case, as not every course can be replaced by any other course. Thus the set of feasible allocations will be altered in this scenario, and some courses may end up being complementary to others.

Another related extension would be to relax the requirements of exact constraints, and allow courses to have different minimum and maximum constraints. This would increase the set of feasible allocations and possibly allow the rule to gain even more in terms of efficiency.

3.10 Appendix A: Proof of Theorem 2

Let R be a preference profile, let μ be a feasible, fair and internally constrained-efficient allocation at a selection of courses C_μ.

Sufficiency

In one direction, suppose there exists a selection of courses C such that, for $C' = C \setminus C_\mu$, there is an improvement $\Gamma^{C'}$ such that $\bar{\gamma}^s \in C_\mu \setminus C$ for all $\gamma \in \Gamma^{C'}$. Let μ' be the upgrade of μ via $\Gamma^{C'}$. Since $\bar{\gamma}^s \in C_\mu \setminus C$ for all $\gamma \in \Gamma^{C'}$, and $\bar{\gamma}_i \neq \bar{\gamma}_j$ for all $i \neq j$, we have that if $c \in C_\mu \setminus C$, then $c \notin C_{\mu'}$. Thus $C_{\mu'} = C$, and μ' is feasible. By construction, μ' is fair, and Pareto dominates μ. Thus μ is not constrained efficient.

Necessity

We first prove a useful lemma.

Lemma **8.** *Let μ and ψ be fair assignments, let ψ be constrained efficient, and let ψ Pareto dominate μ, i.e., $\psi(s)R_s\mu(s)$ for all s, and $\psi(t)P_t\mu(t)$ for some t. Then:*

(a) For every course $c \in C_\psi$, we have that $|D_\mu(c)| \geq q$.

(b) Let $c \in C_\psi$, let $s = top(\succ_c, D_\mu(c))$, and suppose that $cP_s\mu(s)$. Then $\psi(s)R_sc$, and in particular, $\psi(s)P_s\mu(s)$.

Proof: Let μ and ψ be fair assignments, let ψ be constrained efficient, and let ψ Pareto dominate μ, i.e., $\psi(s)R_s\mu(s)$ for all s, and $\psi(t)P_t\mu(t)$ for some t.

(a) Pick some $c \in C_\psi$, and let $S_c = \{s \in \mathcal{S} | \psi(s) = c\}$. We have that $\psi(s)R_s\mu(s)$ for all $s \in S_c$ since ψ Pareto dominates μ. Thus $s \in D_\mu(c)$ for all $s \in S_c$, and since $|S_c| = q$, we have that $|D_\mu(c)| \geq q$.

(b) Suppose $cP_s\psi(s)$. Since $s = top(\succ_c, D_\mu(c))$, we have in particular that $s \succ_c s'$ for all $s' \in \psi(c)$. This violates fairness of ψ. Thus $\psi(s)R_sc$. Since $cP_s\mu(s)$, it follows that $\psi(s)P_s\mu(s)$.

■

Suppose μ is not constrained efficient. Then there exists a constrained efficient allocation ψ that Pareto dominates μ. Let C be the courses at ψ, and let $C' = C \setminus C_\mu$. Note that we have $C_\psi \neq C_\mu$, because otherwise ψ is a rearrangement of μ, violating the internal constrained efficiency of μ. We will construct $\Gamma^{C'}$.

Initialisation:

1. Let ν be a tracking allocation and set $\nu = \mu$. Let D_ν be the associated claimant profile.
2. Let $C' = \{c_1, ..., c_r\}$, and for every $c_i \in C'$, let $\{o_1^{c_i}, ..., o_q^{c_i}\}$ be the slots in c_i.
3. For each course $c \in C'$ and a slot o_j^c, create a chain γ by setting the opening node $\gamma^c(0) = c$.
4. Set $i = 1, j = 1, k = 1$.

Construction Step:
Round k:

1. Let $s_{ij}(k) = top(\succ_{\gamma_{ij}^c(k-1)}, D_\nu(\gamma_{ij}^c(k-1)))$.

   Claim **2.** For any $k \geq 1$, $D_\nu(\gamma_{ij}^c(k-1)) \neq \phi$.

   Proof: In any round, ψ Pareto dominates ν (see Claim 3 below). Thus for every round k, $|D_\nu(\gamma_{ij}^c(k-1)| \geq q$, by Lemma 8(a). ■

2. Upgrade $\nu(s_{ij}(k)) = \gamma_{ij}^c(k-1)$

   Claim **3.** The temporary allocation ν is fair. Also, either $\psi = \nu$ or ψ Pareto dominates ν.

   Proof: In the first round, μ is fair so ν is fair. In every subsequent round, ν is fair because if a student's assignment changes from the previous round, she must be one of the top q claimants for the course she is newly assigned, and so no other student justifiably envies her.

   Suppose $\psi \neq \nu$. We show Pareto dominance. In the first round, ψ Pareto dominates μ. In any subsequent round k, since by Lemma 8(b) we have that $\psi(s) R_s \nu(s)$ for all students in round $k-1$, the same is true at round k also after the upgrade.

   We show that the relation is strict for some student. Since $\psi \neq \nu$, there must be a student s such that $\nu(s) \in C_\mu \setminus C$. By construction, $\nu(s) = \mu(s)$. Thus by Lemma 8(b), we have that $\psi(s) P_s \nu(s)$ in the previous round, and thus in this round too since her assignment does not change. So ψ Pareto dominates ν.

   ■

3. Set $\gamma_{ij}^s(k) = s_{ij}(k)$ and $\gamma_{ij}^c(k) = \mu(s_{ij}(k))$.

   Claim **4.** $s_{ij}(k) \neq s_{ij}(l)$ for all $l < k$.

Proof: Suppose not. Consider $s_{ij}(l), s_{ij}(l+1), ..., s_{ij}(k) \equiv s_{ij}(l)$. By construction, each student prefers the earlier student's assignment, which is in μ. Thus we have a Pareto cycle in μ, and μ is not internally constrained efficient, which is a contradiction. ■

4. If $\gamma^s_{ij}(k) = \gamma^s_{i'j'}(k')$ and $\gamma^c_{ij}(k) = \gamma^c_{i'j'}(k')$ for some i', j', k', then remove $\gamma^s_{i'j'}(k'')$ and $\gamma^c_{i'j'}(k'')$ for all $k'' \geq k'$.

   CLAIM **5.** If $\gamma^s_{ij}(k) = \gamma^s_{i'j'}(k')$ and $\gamma^c_{ij}(k) = \gamma^c_{i'j'}(k')$, then $\gamma^s_{ij}(k+m) = \gamma^s_{i'j'}(k'+m)$ and $\gamma^c_{ij}(k+m) = \gamma^c_{i'j'}(k'+m)$ for all $m \geq 0$.

   Proof: Suppose $\gamma^s_{ij}(k) = \gamma^s_{i'j'}(k')$ and $\gamma^c_{ij}(k) = \gamma^c_{i'j'}(k')$. Then we have that $top(\succ_{\gamma^c_{ij}(k+1)}, D_\nu(\gamma^c_{ij}(k+1))) = top(\succ_{\gamma^c_{i'j'}(k'+1)}, D_\nu(\gamma^c_{ij}(k'+1)))$. Thus $\gamma^s_{ij}(k+1) = \gamma^s_{i'j'}(k'+1)$ and $\gamma^c_{ij}(k+1) = \gamma^c_{i'j'}(k'+1)$. Repeating for all $m \geq 1$, we have the desired result. We remove all duplicated students from $\gamma_{i'j'}$ because by construction, $\gamma^c_{ij}(k) P_{s_{ij}(k)} \gamma^c_{i'j'}(k')$, and so she must get at least $\gamma^c_{ij}(k)$ by Lemma 8(b). ■

If $\gamma^c_{ij}(k) \in C \cap C_\mu$, we go to Round $k+1$. Otherwise $\gamma^c_{ij}(k) \in C_\mu \setminus C$, and we go to the Verification Step.

Verification Step:

1. We repeat for all slots in a course, i.e., if $j < q$, then we increment j, set $k = 1$, and go back to the Construction Step.
2. We repeat for all courses, i.e., if $j = q$ but $i < r$, we increment i, set $j = 1, k = 1$, and go back to the Construction Step.
3. If there is a course c_i and a slot $o^{c_i}_j$ such that, in the relevant chain γ_{ij}, we have that $\bar{\gamma}_{ij} \notin C_\mu \setminus C$, we repeat the Construction Step for that course.

CLAIM **6.** The process is guaranteed to terminate in a finite number of steps.

Proof: The number of courses and students is finite, and at every step some student is strictly improving her assignment. ■

When the process completes, we have the improvement $\Gamma^{C'}$ as required, with each chain originating in a new slot, and terminating in a discarded course.

4

Sufficient Conditions for Weak Group-Strategy-Proofness

4.1 Introduction

In this chapter we continue to consider a situation where heterogenous objects are to be distributed among a set of claimants. Same as before, the objects in question are indivisible, so they cannot be split or shared. There is no money in this economy so no compensation is possible.

Objects are to be assigned to individuals based on their preferences. Preferences in this context are rankings over the set of objects, one for each individual, such that an object preferred to another is also ranked higher than it. We assume that all objects can be ranked, that each object is acceptable to each individual, and that any ranking is possible in principle.

Moreover, the key feature of preferences is that this information is private to each individual. The designer or implementer of the solution to the allocation problems must elicit this information from individuals before making assignments. Individuals may reveal any preferences at all.

It is assumed that individuals may seek to game the system if it is to their advantage. If falsely revealing preferences gives an agent an object she prefers to what she might get if she instead truthfully revealed her preferences, then there is no reason to believe that a rational agent would not do so. A desirable property that a designer would like the allocation rule to satisfy is immunity from such undue gain for deviating agents. In particular, a strategy-proof rule ensures that it is a dominant strategy for every individual to truthfully reveal her preferences.[1]

There are many allocation problems where even individually strategy-proof rules do not exist. Demanding more from such rules is futile. However, in problems where strategy-proof rules exist, it is natural to ask whether the immunity from manipulation can be extended to coordinated actions by groups of agents as well. This property is called group-strategy-proofness.

Group-strategy-proofness comes in two forms. Both look at the consequences for a group of agents who deviate by reporting different preferences. The standard form requires that it should not be the case that all deviating agents are as well off as before in terms of their original preferences, and some agent is strictly better off. A weaker form requires that it should not be the case that all deviating agents are strictly better off as a result. It is clear to see that the standard form implies the weak form while the converse is not true in general.

There is a clear connection between individual and group-strategy-proofness. In a wide class of allocation problems, group-strategy-proofness can be shown to be equivalent to a combination of strategy-proofness and a property called non-bossiness. Non-bossiness preempts situations where one agent can be bossy with another by affecting her assignment without changing her own.

Several important strategy-proof rules in the literature are also group-strategy-proof. The list includes inheritance rules (Pápai (2000)), other generalisations of top trading cycles rules (Abdulkadiroğlu and Sönmez (1999), Pycia and Ünver (2013)) and sequential and serial dictatorships (Svensson (1999), Pápai (2001), Hatfield (2009)). These rules are non-bossy and are also weakly

[1] This property has also been motivated by its informational simplicity. In a strategy-proof environment, an agent has to only consider her own preferences and not worry about what other agents might or might not do.

group-strategy-proof by default.

However, there are also important strategy-proof rules that are weakly group-strategy-proof but not group-strategy-proof. The famous Gale-Shapley Deferred Acceptance (DA) rule (Gale and Shapley (1962)) is a case in point. Kojima (2010) shows that it is impossible for a stable rule to be non-bossy. Since the DA rule always produces a stable outcome, it is bossy. Thus it cannot be group-strategy-proof. Yet Hatfield and Kojima (2009) show that under general conditions (including the ones that apply in our model) the DA rule is weakly group-strategy-proof.

There is a non-trivial distinction between the two properties. It is useful therefore to ask the question: what makes a strategy-proof rule weakly group-strategy-proof but not group-strategy-proof? [2]

Non-bossiness is too strong a condition and is not necessary for weak group-strategy-proofness. What we seek in this paper is a condition (or set of conditions) that is weaker than non-bossiness that is nevertheless enough to guarantee weak group-strategy-proofness. We identify two fairly weak properties, which we call partial weak Maskin Monotonicity and weak non-bossiness respectively, that are sufficient to guarantee that a strategy-proof rule is also weakly group-strategy-proof.

We show the robustness of these conditions. That is, we can find examples of rules that violate one or other of these properties in turn. We also show that these properties are 'weak enough', in that there are rules that satisfy these properties (and are thus weakly group-strategy-proof) but are not group-strategy-proof.

A similar exercise is conducted in a recent working paper by Barberà et al. (2014) where they draw connections between strategy-proofness and weak group-strategy-proofness in different allocation environments. In particular, they show that it is the features of the models of house allocation, matching and division that result in the relationship between the two properties. Our work is independent of theirs. Their motivation is similar to ours but the formal results are different. We discuss this paper and the conditions therein in a later section.

The paper is organised as follows. In Section 4.2 we provide the formal notation that we use throughout the paper. In Section 4.3 we formally describe the various forms of strategy-proofness and highlight some well-known connections between them. In Section 4.4 we discuss some of the important strategy-proof rules in the literature and why they satisfy one or other variant of group-strategy-proofness. In Section 4.5 we present the partial weak Maskin Monotonicity and weak non-bossiness properties and relate them to other conditions in the literature. Section 4.6 presents our main result and proof. In this section we also show that our conditions are independent. Section 4.7 concludes.

4.2 Notation and Definitions

The details of the model are given below:

- There is a finite set of *agents* $\mathcal{N} = \{1, ..., i, j, , k, ..., N\}$ and a finite set of *objects* $\mathcal{Z} = \{a, b, c, d, ...\}$.
- An *allocation* $x \in \mathcal{Z}^N$ with $x = (x_1, ..., x_N)$ is a vector that associates an object with each agent. For any agent $i \in \mathcal{N}$, $x_i \in \mathcal{Z}$ is the *assignment* of agent i in x.

[2] Barberà et al. (2010) consider a related question in the context of domains of preferences. In particular, they provide conditions on domains guaranteeing that for all rules defined on them, individual and weak group-strategy-proofness become equivalent. A similar exercise is conducted in le Breton and Zaporozhets (2009).

- Depending on the context of the model, an allocation may be *feasible* or otherwise. The set of all feasible allocations in an allocation problem is given by $\mathcal{A}$.

- Preferences over assignments are strict. Formally, agent $i \in \mathcal{N}$ has *preferences*, denoted R_i, that are given by a binary relation over $\mathcal{Z}$. For any a, b, aR_ib is interpreted as 'object a is at least as good as object b for agent i under preferences R_i'. The binary relation is reflexive (for all a, aR_ia), complete (for all a, b, aR_ib or bR_ia), transitive (for all a, b, c, aR_ib and bR_ic imply aR_ic) and antisymmetric (for any a, b, aR_ib and bR_ia imply $a = b$). The associated strict relation is given by P_i, such that aP_ib if aR_ib and $a \neq b$. For any a, b, aP_ib means 'a is preferred by i to b under preferences R_i'.

- Agent preferences over allocations are selfish, in that they care only about the assignment they receive. Agents are indifferent between all allocations that give them the same assignment. An agent's preferences between two allocations that give her different assignments are governed by her preferences over the respective assignment she receives.

- A collection of preferences for all agents is called a *preference profile*, or simply a *profile*, and is denoted by $R = (R_1, ..., R_N)$. The set of all preference profiles is $\mathcal{R}$. In this model we shall usually suppress reference to $\mathcal{R}$, with the understanding that we operate on the full domain of preferences everywhere. As is the convention, we write R_{-i} for a sub-profile of preferences of all agents other than i. Similarly, for a subset of agents M, we write R_M and R_{-M} to denote the sub-profile of preferences of agents in subsets M and $\mathcal{N} \setminus M$, respectively.

- An *allocation rule* $f : \mathcal{R} \to \mathcal{A}$ takes as input a preference profile R and prescribes an associated feasible allocation $f(R)$. For any agent i, $f_i(R)$ is the assignment she receives at preference profile R according to the rule f. Similarly, for any subset of agents M, $f_M(R)$ is the M-dimensional vector of assignments of M at R, according to f.

4.3 Standard Strategy-Proofness Properties

In what follows we set up the properties of strategy-proofness and the two versions of group-strategy-proofness and draw some important connections between them.

Strategy-proofness is a condition which requires truth-telling to be a dominant strategy for all agents. In other words, given the reports of all other agents, an agent must be as well off reporting her true preferences as any other preferences. When this is true for all agents and all preferences, the mechanism is said to be strategy-proof. Formally:

Definition **10.** A rule is *strategy-proof* if there does not exist a profile R, an agent i, and preferences R_i' such that:

$$f_i(R_i', R_{-i})P_if_i(R)$$

Group-strategy-proofness is a stronger condition than strategy-proofness. In its standard form, it ensures that groups of agents do not have profitable deviations, i.e., if a group of agents deviates by reporting different preferences, then a group-strategy-proof rule ensures that it is not the case that all agents in the deviating group are at least as well off as before, and some agent strictly better off. Formally:

DEFINITION **11.** A rule f is *group-strategy-proof* if there does not exist a profile R, a subset of agents M, and a sub-profile R'_M such that

$$f_i(R'_M, R_{-M})R_i f_i(R) \text{ for all } i \in M \text{ and } f_j(R'_M, R_{-M})P_j f_j(R) \text{ for some } j \in M$$

Weak group-strategy-proofness ensures that groups of agents do not have strictly profitable deviations, i.e., if a group of agents deviates by reporting different preferences, then a group-strategy-proof rule ensures that it is not the case that all agents in the deviating group are strictly better off than before. Formally:

DEFINITION **12.** A rule f is *weakly group-strategy-proof* if there does not exist a profile R, a subset of agents M, and a sub-profile R'_M such that:

$$f_i(R'_M, R_{-M})P_i f_i(R) \text{ for all } i \in M$$

It is clear that group-strategy-proofness implies weak group-strategy-proofness, which in turn implies strategy-proofness.

Non-bossiness is an axiom that is pervasive in the literature on assignment rules. The condition was introduced by Satterthwaite and Sonnenschein (1981) and requires that an agent not be able to affect other agents' outcomes without affecting her own.

DEFINITION **13.** A rule f is *non-bossy* if, for all preference profiles R, for all agents i, and all reports R'_i, we have:

$$[f_i(R'_i, R_{-i}) = f_i(R)] \implies [f(R'_i, R_{-i}) = f(R)]$$

Non-bossiness negates any effect that an agent can have on other agents' assignments in cases where she does not change her own assignment. Its main justification is that it keeps the distribution of influence in the allocation process from unduly depending on any one agent. Another justification has to do with its strategic effects. Also, its original use by Satterthwaite and Sonnenschein (1981) is on the basis of considerations of informational simplicity. Non-bossiness disqualifies rules in exchange economies that "assign all the resources to one or the other of two agents depending upon some arbitrary feature of some third agent's preferences."[3]. However, Thomson (2014) also notes that its main value is in providing technical support for characterisation results.

In the presence of non-bossiness, group-strategy-proofness is equivalent to strategy-proofness in a wide class of models. This has been demonstrated in several contexts[4], so we state it here without proof. However, it must be noted that this equivalence is not true for all indivisible object allocation models. In any situation, for example, where objects can be separated as multiple types (see Hideo Konishi and Wako (2001)), or where markets can be segmented so that not all allocations are reachable (see Pápai (2003)), the equivalence does not hold.

PROPOSITION **1.** *A rule f is strongly group-strategy-proof if and only if it is strategy-proof and non-bossy.*

[3] See Thomson (2014)

[4] See, for example, Pápai (2000) and Chapter 2 of this thesis.

4.4 Strategy-Proof Rules and their Group-Strategy-Proofness

In this section we present some frequently used and important strategy-proof rules in the literature on allocation problems. We show by example how these rules satisfy (or fail to satisfy) the variants of group-strategy-proofness presented above.

4.4.1 Constant rules

A constant rule is one that prescribes the same allocation for every preference profile. It is easy to see that such rules are trivially strategy-proof and group-strategy-proof. A partially constant rule is one that is constant for some agents, i.e., assigns the same object to them at every preference profile. Such agents can never be part of a strictly profitable deviating group. If every deviating group contains one such agent, this rule will be weakly group-strategy-proof. Of course, if any group is allowed to deviate, then this condition is only satisfied if all but one agent get a constant allocation, which makes for a limited case.

However, these rules can still be bossy. Consider the following example: There are two agents $\{1, 2\}$ and three objects $\{a, b, c\}$. For any profile R, the rule assigns object a to agent 1 ($f_1(R) = a$) and to agent 2 assigns the top-ranked object in R_1 that is distinct from a ($f_2(R) = b$ if bP_1c and $f_2(R) = c$ if cP_1b). This is a partially constant rule as agent 1 gets the same object at all profiles. Yet agent 1 can be bossy with agent 2. Thus this rule is not group-strategy-proof. Yet it is weakly group-strategy-proof.

4.4.2 Dictatorships and their variants

The serial dictatorship (or serial priority) rule works as follows: There is an exogenous and fixed ordering of agents σ such that agents sequentially select objects in that order ($\sigma(1)$ selects first, $\sigma(2)$ goes next, and so on). Each agent selects her top-ranked objects from the ones that are available, given the choices of earlier agents in the sequence. It is easy to see that for any preference profile, the first agent always gets her top-ranked object, while the second agent always gets her top-ranked object whenever it is distinct from the selection of the first agent, and so on.

The sequential dictatorship (or sequential priority) rule differs from this rule only by making the identity of subsequent agents dependent on the assignment of earlier agents in the sequence. Thus for instance depending on what object $\sigma(1)$ selects, the identity of $\sigma(2)$ could differ.

Both these classes of rules are strategy-proof and group-strategy-proof in a wide variety of contexts (see Svensson (1999), Pápai (2000), Pápai (2001), Rhee (2011), Hatfield (2009) and Chapter 2 of this thesis for more details).

Consider, however, a variant of the serial dictatorship that we call a 'pure' dictatorship. There is an exogenous and fixed ordering of agents σ such that for any preference profile, the first agent in the sequence $\sigma(1)$ gets her top-ranked object. The next agent $\sigma(2)$ gets the object that $\sigma(1)$ ranks second, and in general agent $\sigma(k)$ gets the k^{th}-ranked object according to preferences of $\sigma(1)$.

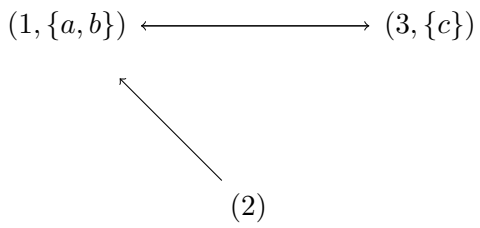
This rule is strategy-proof. Since it is bossy, it is not group-strategy-proof. Yet it is easy to check that this rule is weakly group-strategy-proof.

4.4.3 The Top Trading Cycles Rule

The famous top trading cycles (TTC) rule is attributed to David Gale (see Shapley and Scarf (1974)). In brief, the generalised TTC works as follows:

Each object is initially owned by one agent, who brings it to the market for trade[5]. Some agents may initially own more than one object, while others may own none at all. The procedure works in stages. In any stage, each agent who is yet to receive an assignment points to the owner of the object she most prefers from the ones that are available. A top trading cycle is made up of agents who successively point to the next agent, with the last agent pointing to the first. A cycle can be a singleton, such that an agent points to herself (she owns the object she most prefers.) Since there is a finite number of agents, at every stage there must always be a cycle. Agents in a cycle trade their objects along the cycle until they receive the object they desire. This becomes their assignment and such agents leave the market along with those objects. If there are still agents and objects left unassigned, the procedure repeats in the reduced market. If preferences are strict, then given an initial ownership, the resulting allocation is unique.

The TTC rule is illustrated by an example. Suppose there are three agents $(1, 2, 3)$ and three objects (a, b, c). Suppose agent 1 initially owns a, b and agent 3 initially owns c. Agent 1 desires c, while agents 2 and 3 desire b. The TTC procedure would look as follows:



Agent 1 'points to' agent 3 who owns c, and agents 2 and 3 in turn point to agent 1 who owns b. The cycle in this stage is between agents 1 and 3, who consequently trade those objects. The TTC would assign b to agent 3 and c to agent 1.

TTC rules and their generalisations to inheritance rules (Pápai (2000)) are group-strategy-proof. They are thus also weakly group-strategy-proof. (An inheritance rule in the above example would also specify how agent 2 'inherits' the remaining object a. The TTC procedure in the second stage would just be agent 2 pointing to herself, and a would become her assignment.)

4.4.4 The Deferred Acceptance Rule

In addition to agent preferences, some allocation models assume that each object in turn has a 'priority', which is a ranking over agents or subsets of agents. A priority essentially captures the relative eligibility of an agent for an object vis-a-vis another agent.

The Deferred Acceptance (DA) rule (Gale and Shapley (1962)) uses preference and priority information and works in a series of rounds, as follows.

In the first round, all agents apply to their most preferred object. Any object that receives more applications than its maximum capacity is forced to reject the excess agents, provisionally accepting

[5]Objects that an agent initially owns form a part of his or her 'endowment'

the rest. The agents that are rejected are those that are the lowest in that object's priority among its pool of applicants. In the next round, all rejected agents apply to their next preferred object that has not rejected them already. A object considers its existing applications plus any fresh ones it might receive, and provisionally accepts the top agents according to its priority, rejecting the lowest ones that are excess to capacity. If any agent is rejected, we go to the next round. The rule terminates in any round in which no agent is rejected. All provisional acceptances become final.

Consider the following example. There are three agents $\{1, 2, 3\}$ and three objects $\{a, b, c\}$. Preference and priority information are given in the table below:

Priorities			Preferences		
a	b	c	1	2	3
1	2	3	b	a	a
2	3	1	a	c	b
3	1	2	c	b	c

In the first round, agent 1 applies to her top-ranked object b while agents 2 and 3 apply to a. Since a has two applicants, it rejects the lower-ranked one according to its priority, which is agent 3. In the next round, agent 3 applies to her next preferred object, which is b. Now b has two applicants, so it rejects the lower-ranked one according to its priority, which is agent 1. In the next round, agent 1 applies to his next preferred object, which is a. Again, object a must reject one application, and so agent 2 is rejected. In the final round, agent 2 applies to c. There are no more rejections and the rule terminates here, giving us the final matching $((1, a), (2, c), (3, b))$.

The DA rule is strategy-proof. But it is bossy. To see this, consider a profile where only agent 2 changes her preferences such that she now ranks c above a and b. Evaluating the DA rule for this profile, we get that agent 2 continues to get c, as in the first profile, but now agents 1 and 3 swap a and b with each other. Thus despite not changing her own assignment, agent 2 affects the assignments of the other agents.

Thus the DA rule is not group-strategy-proof. It is, however, weakly group-strategy-proof. We will show that it satisfies our sufficient conditions for weak group-strategy-proofness.

4.5 Weaker Conditions on Allocation Rules

In this section we define our notions of partial weak Maskin monotonicity and weak non-bossiness which we will show are sufficient for a strategy-proof rule to be weakly group-strategy-proof. We first define some useful concepts.

The strict upper contour set of an object at a preference is the set of all objects that are preferred to it. Consider an agent i, a preference R_i and an object a. Formally, the *strict upper contour set of a at R_i* is given by $U(R_i, a) = \{b \in \mathcal{Z} : bP_ia\}$.

Let R be a preference profile and $f(R)$ the corresponding allocation produced by a rule f. For an agent i, we say that a preference R_i' is a *strict monotonic transformation of R_i at $f_i(R)$* if $U(R_i', f_i(R)) \subset U(R_i, f_i(R))$. A preference is a strict monotonic transformation of another if the strict upper contour set of the assignment at the old preference is a strict superset of the strict upper contour set of that object in the new preference. We say that a preference R_i' is a *monotonic transformation of R_i at $f_i(R)$* if the subset relation in the above condition is weak. A particular kind of monotonic transformation is an upper-contour-set preserving transformation where, as the name suggests, the upper contour set remains the same in the new preference as well.

In particular, a preference R_i' is an *upper-contour-set preserving transformation of* R_i *at* $f_i(R)$ if $U(R_i', f_i(R)) = U(R_i, f_i(R))$.

We can extend these concepts to profiles in a natural way. A profile R' is a *monotonic transformation of* R *at* $f(R)$ if R_i' is a monotonic transformation of R_i at $f_i(R)$ for all $i \in \mathcal{N}$. We say that R' is a *strict monotonic transformation of* R *at* $f(R)$ if in addition we have that R_j' is a strict monotonic transformation of R_j at $f_j(R)$ for some $j \in \mathcal{N}$. Similarly, a profile R' is an upper-contour-set preserving transformation of R at $f(R)$ if $U(R_i', f_i(R)) = U(R_i, f_i(R))$ for all $i \in \mathcal{N}$.

We will use these concepts to define our two conditions: partial weak Maskin monotonicity and weak non-bossiness.

4.5.1 Partial Weak Maskin Monotonicity

A rule is Maskin monotonic if the allocation at any profile that is a monotonic transformation of another remains the same. Weak Maskin monotonicity relaxes the requirement that the allocations remain the same. A rule is weakly Maskin monotonic if every agent weakly prefers her assignment at a monotonic transformation of a profile to her assignment at that profile.

Formally, a rule satisfies *Maskin monotonicity* (Maskin (1999)) if for all R, R' such that R' is a monotonic transformation of R at $f(R)$, we have that $f(R') = f(R)$. A rule satisfies *weak Maskin monotonicity* if for all R, R' such that R' is a monotonic transformation of R at $f(R)$, we have that $f_i(R')R_i'f(R)$ for all $i \in \mathcal{N}$.

Weak Maskin monotonicity plays an important role in the characterisation of DA rules (see Kojima and Manea (2010)). It is also shown in that paper that the DA rule is the only stable rule at an exogenously specified priority profile that satisfies weak Maskin monotonicity.

In this paper we weaken the notion of Maskin monotonicity further. Partial weak Maskin monotonicity weakens the condition by requiring that only at least one agent weakly prefer her assignment at a strict monotonic transformation. Formally:

Definition **14.** A rule satisfies *partial weak Maskin monotonicity* if for all R, R' such that R' is a strict monotonic transformation of R at $f(R)$, we have that $f_i(R')R_i'f(R)$ for some $i \in \mathcal{N}$.

It is clear that a Maskin monotonic rule is also weakly Maskin monotonic and that a weakly Maskin monotonic rule is also partially weak Maskin monotonic. But the reverse implications do not hold in general.

4.5.2 Weak Non-Bossiness

Our other main property is weak non-bossiness. As its name suggests, this is a relaxation of the requirements of non-bossiness. Non-bossiness requires that assignments for all agents remain fixed for any deviation by any agent at any profile that does not change her assignment. Weak non-bossiness restricts the type of preference for which this invariance is true. In particular, if an agent's assignment does not change when she changes her preference via a upper-contour-set preserving transformation of her original preferences, then no other agent's assignment should change. Weak non-bossiness places no restrictions on bossy behaviour at other kinds of preference changes. Formally:

DEFINITION **15.** A rule f is weakly non-bossy if, for any preference profile R, agent i and preferences R_i' such that R_i' is an upper-contour-set preserving transformation of R_i at $f_i(R)$, we have that:

$$[f_i(R_i', R_{-i}) = f_i(R)] \implies [f(R_i', R_{-i}) = f(R)]$$

Non-bossiness implies weak non-bossiness but the reverse implication does not hold in general.

4.6 RESULTS

We are now ready to state our main theorem. Theorem 4 shows that the combination of strategy-proofness, weak non-bossiness and partial weak Maskin monotonicity is sufficient to give us weak group-strategy-proofness.

THEOREM **4.** *If a strategy-proof rule f is weakly non-bossy and partially weak Maskin monotonic, then it is weakly group-strategy-proof.*

Proof: Let f be strategy-proof, weakly non-bossy and partially weak Maskin monotonic. Let R be a profile. Let M be a subset of agents, and R_M' be preferences for M such that $f_i(R_M', R_{-M})R_i f_i(R)$ for all $i \in M$. Define the profile R' as (R_M', R_{-M}).

For f to be weakly group-strategy-proof, we must show that $f_j(R') = f_j(R)$ for at least one agent $j \in M$. For contradiction, suppose that $f_i(R')P_i f_i(R)$ for all $i \in M$. Construct $\hat{R}$ such that for every $i \in M$, $top(\hat{R}_i) = f_i(R')$ and other objects are ranked the same as in R_i. Set $\hat{R}_j = R_j$ for all $j \notin M$.

By assumption, $f_i(R')P_j f_i(R)$ for all $i \in M$. Thus by construction $\hat{R}$ is an upper-contour-set preserving transformation of R at $f(R)$. Consider the sequence of profiles $R^0 = R, R^1 = (\hat{R}_1, R_{-1}), R^2 = (\hat{R}_{\{1,2\}}, R_{-\{1,2\}}), ..., R^N = \hat{R}$. By strategy-proofness, $f_1(R^1) = f_1(R^0)$ since $\hat{R}_1$ is an upper-contour-set preserving transformation of R_1 for $f_1(R^0)$ and by weak non-bossiness, $f(R^1) = f(R^0)$. Repeating the argument for other agents and noting that for every i, R_i^i is an upper-contour-preserving transformation of R_i at $f_i(R^{i-1})$, we have by strategy-proofness that $f_i(R^i) = f_i(R^{i-1})$ and by weak non-bossiness that $f(R^i) = f(R^{i-1})$. Thus $f(\hat{R}) = f(R)$.

It is easy to see that $\hat{R}$ is also a monotonic transformation of R' at $f(R')$. If it is not a strict monotonic transformation, then it must be an upper-contour-set preserving transformation. In that case, by the arguments above, we have that $f(\hat{R}) = f(R')$. So let $\hat{R}$ be a strict monotonic transformation of R' at $f(R')$. Then by partial weak Maskin monotonicity, we have that $f_J(\hat{R})\hat{R}_J f_J(R')$ for some $J \in M$. In particular, we can find a $J \in M$ such that $f_J(\hat{R}) = top(\hat{R}_J)$.

But then $f_J(\hat{R}) = f_J(R')$ (since by construction $top(\hat{R}_J) = f_J(R')$), $f_J(\hat{R}) = f_J(R)$ (as demonstrated above), but $f_J(R')P_j f_J(R)$ by assumption. Since preferences are strict, this is a contradiction. Thus our initial supposition was false, and there is at least one $i \in M$ such that $f_i(R') = f_i(R)$. Since R, M and R_M' was arbitrary, this is true for all R, for all M and for all subprofiles R_M'. Thus f is weakly group-strategy-proof. ■

4.6.1 DISCUSSION

As discussed earlier, partial weak Maskin monotonicity is a weaker version of the axiom of weak Maskin monotonicity used in the characterisation of DA rules by Kojima and Manea (2010). In an independent paper closely related to ours, Barberà et al. (2014) derive sufficient conditions for a

strategy-proof rule to also be (weakly) group-strategy-proof. We discuss the relation to this paper here. Their conditions of 'H-strategy-proofness' and 'H-top rich domains' are implicitly present in our assumptions as well, since we assume strategy-proofness, and consider the full domain of preferences. Their other two conditions are 'H-top monotonicity' and 'H-respectfulness', which are related to our conditions of partial weak Maskin monotonicity and weak non-bossiness, respectively. H-respectfulness is a weaker version of non-bossiness in two ways. Firstly, it imposes requirements only on some agents H such that $|H| \leq |N|$ (whereas ours is true for $|H| = |N|$). Secondly, the change in preferences is also restricted. H-respectfulness requires the change in preferences to be *both* upper- as well as lower-contour-set preserving. In our case, we insist on the change only to be upper-contour-set preserving, and is thus a weaker requirement in this context.

H-top monotonicity requires that when preferences of H agents change in such a way as to make their original assignments their top-ranked objects under the new preferences, then these agents should retain the same outcome at the new profile. We depart from this condition in two ways. Firstly, partial weak Maskin monotonicity does not require the change in preferences to make the original assignment the top-ranked object in the new preferences; instead we ask only for a strict monotonic transformation. Moreover, we do not insist that the outcome remain the same for all agents changing their preferences; instead, we require that at least one get an outcome she weakly prefers in the new profile (which may not necessarily be the same object.)

Thus we have two qualitatively different conditions from Barberà et al. (2014). Nevertheless, our result is similar in spirit - as both exercises draw a connection between strategy-proofness and (weak) group-strategy-proofness.

4.6.2 Independence of Conditions

We will now show that our two properties are independent.

Partial Weak Maskin Monotonicity

Consider a situation with three agents $\mathcal{N} = \{1, 2, 3\}$ and three objects $\mathcal{Z} = \{a, b, c, \}$. For any preference R_i and subset of objects X, let $bottom(R_i, X)$ be the last-ranked object in X according to R_i.

Let R be a preference profile. Then the rule is defined as follows: $f_2(R) = bottom(R_1, \mathcal{Z})$, $f_3(R) = bottom(R_2, \mathcal{Z} \setminus \{f_2(R)\})$, and $f_1(R) = bottom(R_3, \mathcal{Z} \setminus \{f_2(R), f_3(R)\})$.

Let R be a profile as given below. The allocation is given in boxes.

Preferences

R_1	R_2	R_3
c	a	b
a	**b**	**c**
b	c	a

This rule is strategy-proof. No agent can alter her own assignment via any change in preferences. To see this, suppose an agent changes her bottom-ranked object, affecting some other agent's assignment. This may in turn affect the third agent's assignment. However, the changing agent's assignment is predicated on the bottom-most available alternative, which will either be the same or

the object previously assigned by her to the next agent, which is less preferred to her assignment. Thus she never gets an object that she prefers to her original assignment.

Moreover, the assignment is invariant under any shuffling of the strict upper-contour set. Thus the rule satisfies weak non-bossiness. But the rule is not partially weak Maskin monotonic. To see this, consider the following profile R', where each preference R'_i is a strict monotonic transformation of R_i at $f_i(R)$.

Preferences

R'_1	R'_2	R'_3
a	b	c
$\boxed{\text{b}}$	$\boxed{\text{c}}$	$\boxed{\text{a}}$
c	a	b

Partial weak Maskin monotonicity requires that at least one of the agents gets an object weakly preferred to the original assignment at the new profile. But this is not the case here, as $f_i(R)P'_i f_i(R')$ for all $i \in \mathcal{N}$. Moreover, the rule is not weakly group-strategy-proof. In particular, agents $1, 2, 3$ can manipulate at R' via profile R making them all strictly better off according to R'.

Weak Non-bossiness

Let $\mathcal{N} = \{1, 2, 3\}$ be two agents and $\mathcal{Z} = \{a, b, c, d, e, h\}$ be a set of objects. For any preference R_i and set of objects X, let $bottom(R_i, X)$ denote the last-ranked object in X according to R_i. Let R be a profile and let the rule work as follows: $f_3(R) = h$. If bP_1a then $f_2(R) = d$. If aP_1b then $f_2(R) = e$. Also, $f_1(R) = bottom(R_2, \{a, b, c\})$.

Consider a preference profile R as in the table below. The allocation is marked in boxes.

Preferences

R_1	R_2	R_3
b	h	a
a	a	b
$\boxed{\text{c}}$	b	e
e	$\boxed{\text{d}}$	$\boxed{\text{h}}$
d	c	c
h	e	d

It is easy to see that this rule is strategy-proof. No agent can affect her own assignment via any other preference report. It is also partially weak Maskin monotonic as agent 3 gets object h for all preference profiles. However, it is not weakly non-bossy. In particular, let a profile R' be given as follows, where R'_1 is an upper-contour-set preserving transformation of R_1 at $f_1(R)$ and other preferences remain the same.

Preferences		
R'_1	R'_2	R'_3
a	h	a
b	a	b
[c]	b	e
e	d	[h]
d	c	c
h	[e]	d

Even though agent 1 gets the same object as before, the allocation is no longer the same as agent 2 now gets e. This rule is also not weakly group-strategy-proof. Consider the following profile R'':

Preferences		
R''_1	R''_2	R''_3
[b]	h	a
a	a	b
c	c	e
e	[d]	[h]
d	b	c
h	e	d

It is easy to see that agents $1, 2$ will manipulate at R' via (R''_1, R''_2).

4.6.3 The Rules

It should be reiterated here that the conditions outlined above are together sufficient only for weak group-strategy-proofness; they are in general not sufficient for group-strategy-proofness. In what follows we will elaborate by showing that the rules we described earlier that are weakly group-strategy-proof but not group-strategy-proof do indeed satisfy our properties.

'Pure' Dictatorship

Recall the 'pure' dictatorship rule in this context. There is an exogenous and fixed ordering of agents σ such that for any preference profile, the first agent in the sequence $\sigma(1)$ gets her top-ranked object. The next agent $\sigma(2)$ gets the object that $\sigma(1)$ ranks second, and in general agent $\sigma(k)$ gets the k^{th}-ranked object according to preferences of $\sigma(1)$.

It is clear that this rule is strategy-proof. Note that the upper-contour-set for the first agent is empty. Also, that any reshuffling of the upper-contour-set for other agents does not affect any agent's assignment. Thus the rule is weakly non-bossy. Also, the first agent always gets her top-ranked object for any profile, and so the rule is partially weak Maskin monotonic.

Deferred Acceptance

The Deferred Acceptance rule is strategy-proof (Roth (1982b), Dubins and Freedman (1984)). For any agent, and two objects she prefers to her assignment, reversing their order in the upper-contour-set will not affect the allocation. To see this, first note that the strategy-proofness of the DA rule

ensures that this agent gets the same assignment. For any other agent, if she gets a different assignment, it is either more or less preferred than the original assignment. If it is less preferred, this will result only if she is rejected by the original object, which is now assigned to some other agent. For this agent, the object cannot be strictly preferred, or the original DA allocation would be invalid. Thus this agent must also get an assignment less preferred to the original. We can then find a set of agents who all get assignments less preferred. This will cause a contradiction to the original DA assignment. Instead, if the assignment is preferred, then it must be that the original claimant of that object has received something else, which must be preferred as well. Thus we can find a set of agents collectively better off. This violates the constrained efficiency of the DA rule (citation). Thus no other agent's assignment will change Thus the DA rule is weakly non-bossy. It is also partially weak Maskin monotonic. In particular, if an agent strictly raises her assignment in her preference, then by strategy-proofness of the DA rule, she continues to receive the same object. Thus at least one agent is at least as well off in the new profile as the old one. This argument also follows from axioms of IR monotonicity and weak Maskin monotonicity used in the characterisation of the Deferred Acceptance rule by Kojima and Manea (2010).

4.7 Conclusion

The distinction between group-strategy-proofness and weak group-strategy-proofness is non-trivial as there are significant classes of rules that satisfy the weak property only. The connection between strategy-proofness and group-strategy-proofness have been well studied. In this chapter we contribute to this literature by identifying conditions that along with strategy-proofness are sufficient for weak group-strategy-proofness. These conditions, partial weak Maskin monotonicity and weak non-bossiness, are both weaker forms of well-known conditions in the literature. We also demonstrate that a rule satisfying all these properties need not be group-strategy-proof. Thus our conditions are 'weak enough' to fill the gap between the two versions of group-strategy-proofness.

A comment on extensions is in order. The conditions that we have identified are merely sufficient for weak group-strategy-proofness. Showing the necessity of these conditions is a difficult exercise. Firstly, note that weak group-strategy-proofness implies strategy-proofness. It is also possible to show that weak group-strategy-proofness implies partial weak Maskin monotonicity, but only when the number of agents that we consider is 2. Furthermore, weak non-bossiness is not a direct implication of weak group-strategy-proofness.

Finding suitable variants of these properties that are both necessary and sufficient for a full characterisation, i.e., to ensure that a strategy-proof rule is also weakly group-strategy-proof, is therefore an open question.

5

Bibliography

Bibliography

Abdulkadiroğlu, A. and L. Ehlers (2006): "Controlled school choice," Mimeo.

Abdulkadiroğlu, A. and T. Sönmez (1999): "House allocation with existing tenants," *Journal of Economic Theory*, 88, 233–260.

——— (2003): "School choice: A mechanism design approach," *The American Economic Review*, 93, 729–747.

Abizada, A. and S. Chen (2011): "The college admissions problem with entrance criterion," Mimeo, University of Rochester.

Alcalde, J. and S. Barberà (1994): "Top dominance and the possibility of strategy-proof stable solutions to matching problems," *Economic Theory*, 4, 417–435.

Barberà, S. (1983): "Strategy-proofness and pivotal voters: a direct proof of the Gibbard-Satterthwaite Theorem," *International Economic Review*, 24, 413–417.

Barberà, S., D. Berga, and B. Moreno (2010): "Individual versus group-strategy-proofness: When do they coincide?" *Journal of Economic Theory*, 145, 1648–1674.

——— (2014): "Group-strategy-proofness in private good economies without money: matching, division and house allocation," Mimeo.

Bird, C. G. (1984): "Group incentive compatibility in a market with indivisible goods," *Economics Letters*, 14, 309–313.

Dubins, L. and D. Freedman (1984): "Machiavelli and the Gale-Shapley algorithm," *American Mathematical Monthly*, 88, 485–494.

Ehlers, L., I. E. Hafalir, M. B. Yenmez, and M. A. Yildirim (2011): "School choice with controlled choice constraints: Hard bounds versus soft bounds," Mimeo.

Erdil, A. and H. Ergin (2008): "What's the matter with tie-breaking? Improving efficiency in school choice," *The American Economic Review*, 98, 669–689.

Ergin, H. I. (2002): "Efficient resource allocation on the basis of priorities," *Econometrica*, 70, 2489–2497.

Fragiadakis, D., A. Iwasaki, P. Troyan, S. Ueda, and M. Yokoo (2012): "Strategy-proof matching with minimum quotas," Mimeo.

Gale, D. and L. S. Shapley (1962): "College admissions and the stability of marriage," *American Mathematical Monthly*, 69, 9–15.

Gibbard, A. (1973): "Manipulation of voting schemes: A general result," *Econometrica*, 41, 587–601.

Hakimov, R. and O. Kesten (2014): "The equitable top trading cycles mechanism for school choice," WZB Dicussion Paper No. SP II 2014-210.

Hatfield, J. W. (2009): "Strategy-proof, efficient, and non-bossy quota allocations," *Social Choice and Welfare*, 33, 505–515.

Hatfield, J. W. and F. Kojima (2009): "Group incentive compatibility for matching with contracts," *Games and Economic Behavior*, 67, 745–749.

Hideo Konishi, T. Q. and J. Wako (2001): "On the Shapley-Scarf economy: the case of multiple types of indivisible goods," *Journal of Mathematical Economics*, 35, 1–15.

Hylland, A. and R. Zeckhauser (1979): "The efficient allocation of individuals to positions," *The Journal of Political Economy*, 87, 293–314.

Kesten, O. (2006): "On two competing mechanisms for priority-based allocation problems," *Journal of Economic Theory*, 127, 155–171.

Kojima, F. (2010): "Impossibility of stable and non-bossy matching mechanisms," *Economics Letters*, 107, 69–70.

Kojima, F. and M. Manea (2010): "Axioms for Deferred Acceptance," *Econometrica*, 78, 633–653.

le Breton, M. and V. Zaporozhets (2009): "On the equivalence of coalitional and individual strategy-proofness," *Social Choice and Welfare*, 33, 287–309.

Ma, J. (1994): "Strategy-proofness and the strict core in a market with indivisibilities," *International Journal of Game Theory*, 23, 75–83.

Maskin, E. (1999): "Nash Equilibrium and Welfare Optimality," *Review of Economic Studies*, 66, 23–38.

Morrill, T. (2014): "Two Simple Variations of Top Trading Cycles," Economic Theory (forthcoming).

Pápai, S. (2000): "Strategyproof assignment by hierarchical exchange," *Econometrica*, 68, 1403–1433.

——— (2001): "Strategyproof and non-bossy multiple assignments," *Journal of Public Economic Theory*, 3, 257–271.

——— (2003): "Strategyproof exchange of indivisible goods," *Journal of Mathematical Economics*, 39, 931–959.

Pycia, M. and M. U. Ünver (2013): "Incentive-compatible allocation and exchange of discrete resources," Mimeo.

Rhee, S. (2011): "Strategy-proof allocation of individual goods among couples," *Japanese Economic Review*, 62, 289–303.

Roth, A. (1982a): "The economics of matching: stability and incentives," *Mathematics of Operations Research*, 7, 617–628.

——— (1982b): "Incentive compatibility in a market with indivisible goods," *Economics Letters*, 9, 127–132.

Roth, A. and A. Postlewaite (1977): "Weak versus strong domination in a market with indivisible goods," *Journal of Mathematical Economics*, 4, 131–137.

Satterthwaite, M. (1975): "Strategy-proofness and Arrow's Conditions: Existence and correspondence theorems for voting procedures and social welfare functions," *Journal of Economic Theory*, 10, 187–216.

Satterthwaite, M. and H. Sonnenschein (1981): "Strategy-proof allocation mechanisms at differentiable points," *Review of Economic Studies*, 48, 587–597.

Sen, A. (2001): "Another direct proof of the Gibbard-Sattherthwaite Theorem," *Economics Letters*, 70, 381–385.

Shapley, L. and H. Scarf (1974): "On cores and indivisibility," *Journal of Mathematical Economics*, 1, 23–37.

Sönmez, T. and T. Switzer (2011): "Matching with branch-of-choice contracts at United States Military Academy," Mimeo, Boston University.

Svensson, L.-G. (1999): "Strategy-proof allocation of indivisible goods," *Social Choice and Welfare*, 16, 557–567.

Thomson, W. (2013): "Strategy-proof allocation rules," Mimeo, University of Rochester.

——— (2014): "On the axiomatics of resource allocation: Interpreting nonbossiness," Mimeo, University of Rochester.