Stability in Matching Markets with Sizes[*]

David Delacrétaz[†]

October 17, 2017

Abstract

Matching markets such as day care, tuition exchange, couples problems and refugee resettlement involve agents of different *sizes*. The size of an agent is the amount of capacity that he uses. In matching markets where all agents have the same size, there exists an agent-optimal stable matching. This structure disappears when agents have different sizes: the set of stable matchings may not contain an agent-optimal element and may even be empty. In this paper, we study a matching market where the size of an agent is either one or two. We propose a novel and constructive algorithm to find an agent-undominated stable matching whenever one exists. We introduce a novel relaxation of stability: *size-stability*. Size-stable matchings are non-wasteful but allow size-two agents to envy multiple size-one agents. We show that the set of size-stable matchings is nonempty. We adapt our algorithm to find a size-stable matching that is undominated from the point of view of size-two agents, thus compensating them for envying size-one agents. In the process, we outline a new trade-off between eliminating waste and bounding envy: at any non-wasteful matching, an agent may envy arbitrarily many other agents.

[*]I thank Georgy Artemov, Ivan Balbuzanov, Inácio Bó, Ingrid Burfurd, Yunho Cho, Sven Feldman, Edwin Ip, Bettina Klaus, Scott Duke Kominers, Maciej Kotowski, Bryn Lampe, Simon Loertscher, Claudio Mezzetti, Ellen Muir, Alex Nichifor, Andras Niedermayer, Peter Norman, Marek Pycia, José Rodrigues-Neto, Tayfun Sönmez, Alex Teytelboym, Neil Thakral, Utku Ünver, Cédric Wasser, Tom Wilkening, Jun Xiao as well as the audiences at the Australasian Economic Theory Workshop 2015 and 2016, the Australian Ph.D. Conference in Economics and Business 2016, MATCH-UP 2017, the Annual Congress of the Swiss Society of Economics and Statistics, Toulouse School of Economics and HEC Lausanne for their valuable comments. I gratefully acknowledge the support of the Faculty of Business and Economics, the Department of Economics, and the Centre for Market Design at the University of Melbourne. All errors are my own.

[†]Department of Economics, Level 4, FBE Building, University of Melbourne, 111 Barry St, Victoria 3010, Australia. Email: ddelacretaz@unimelb.edu.au.

1 Introduction

Centralized matching programs have been successfully implemented in various markets, including the National Resident Matching Program (Roth, 2003), school choice (Abdulkadiroglu and Sönmez, 2003) and kidney exchange (Roth, Sönmez, and Ünver, 2004). These involve a central clearing house that collects information on both sides of the market and uses an algorithm to match doctors to hospitals, students to schools, and patients to kidneys.

The matching of children to day care centers, in contrast, has remained decentralized around the world. Parents have to apply individually to multiple centers in the hope of getting a place and may have to decide whether or not to accept an offer before knowing whether or not a better one will become available. Che and Koh (2016) show that the outcome of such a decentralized matching market may be unfair and inefficient, even in the presence of waiting lists. Day care is an essential service for parents who wish to reconcile career and family, but places are often in short supply. Inefficiencies in the way these places are allocated can therefore have important consequences. In addition, these inefficiencies can often not be fully addressed by a price mechanism, as fees are regulated and access is prioritized by law.[1]

An important difference between the matching of children to day care centers and students to schools is that children often attend day care part-time while students always attend school full-time. A place at a day care center can therefore be shared among two or more children attending on different days. Mathematically, children have different *sizes* in the sense that they affect the capacity constraint of a day care center differently depending on whether they attend part-time or full-time. This seemingly small difference considerably complicates the search for a "good" matching rule, which may offer an explanation as to why the matching of children to day care centers has remained decentralized.[2]

Matching markets where agents have different sizes arise in a variety of contexts. Tuition exchange agreements between universities, for example those created by the Erasmus program, often contain this feature. These bilateral agreements allow students from one university to study at the other for a year or a semester. The home university is responsible for

[1]For example, families from a disadvantaged background or families who live in the neighborhood where the day care center is located often have a higher priority.

[2]Another potential difficulty with day care matching is that children may enter or exit at any time, giving the problem a dynamic aspect. The largest intake, however, takes place once a year when the older children start school or kindergarten. Optimizing this static problem has the potential to greatly improve the way the market operates.

selecting which students it will send to its partner. Places are often competitive and students may apply for several destinations, in order of preference. The selection process is a matching market: students have preferences over partner universities and have different priorities for each of them depending on the quality of their application. Capacities are determined by the exchange agreement, which can specify either a maximum number of students that can be selected or a maximum number of semesters that can be used. In the latter case, a student going for a year uses two units of capacity. Exchange has become an increasingly important part of university degrees and, in a globalized world, demand for it is likely to keep growing in the future. Designing mechanisms for universities to efficiently cope with this demand and match students and partner institutions in the best possible way requires to study models where agents have different sizes.

Agents with different sizes also present challenges in existing models. The National Resident Matching Program (NRMP) has matched medical graduates to residency hospitals in the United States since 1951. Since 1983, doctors have had the option to apply as couples and submit preferences over pairs of hospitals (Roth, 1984). A couple requires two positions, however they may not be at the same hospital. Refugee resettlement (Delacrétaz, Kominers, and Teytelboym, 2016) constitutes another extension. People accepted as refugees in a given country are generally resettled across various local areas that provide them with multiple services (e.g. housing, school places or training programs). Multidimensional constraints arise in this problem as families have different service requirements, depending for example on whether they have children or specific needs.

In order to pin down the implications of introducing sizes and develop suitable solutions, we study the simplest matching model with sizes. Agents (e.g. parents, students) have ordinal preferences over objects (e.g. day care centers, host universities) that are available in multiple identical units (e.g. part-time places in a specific day care center or semesters on exchange at a specific partner university). For each object, agents are ranked according to exogenous priorities. Agents can have a size of either one or two: there are *single-unit agents* who require one unit of an object (e.g. families placing their child part-time or students going on exchange for a semester) and *double-unit agents* who require two units of the same object (e.g. families placing their child full-time or students going on exchange for a year).

The model – as well as the insights gathered and techniques developed throughout the paper – can be extended in various ways to fit specific applications. Extensions can follow at least three directions. First, constraints may take place over several dimensions. Children going to day care part-time typically attend specific days, for example Monday-Wednesday-Friday or Tuesday-Thursday. A day care center has two capacity constraints: one for each part of the week. A part-time child uses one unit of capacity of either part and a full-time child uses a unit of capacity of both parts. Refugee resettlement also involves multidimensional constraints, each of which is a service. Families may also require multiple units of some services, for example a family with three school-age children requires three school places. Second, the model can be extended to the case where agents have preferences over both an object and a number of units. Students may prefer to go on exchange for a year but, if this is not possible, be willing to go for only a semester. The length of time they choose may also depend on the university they consider. Parents may likewise choose to settle for placing their child into day care part-time even though they would prefer a full-time slot. Third, agents may desire units of different objects. Doctors in a couple can for example work in two different hospitals located in the same city. Sizes take different forms in each of these applications, however they are present in all of them. Designing central clearing houses for these markets requires an understanding of how sizes affect matching problems. This paper sheds some light on the subject and constitutes a stepping stone towards developing new solutions for a wide-range of matching applications.

Stability – initially introduced by Gale and Shapley (1962) – is a central concept in matching theory. In our model, an agent and an object form a *blocking pair* of a given matching if the agent prefers the object to his own and at least the number of units he requires are either unassigned or assigned to agents with a lower priority. A matching is stable if it does not have any blocking pair. Stable matchings are desirable for a range of reasons that vary depending on the application. First, stability constitutes a natural equilibrium criterion. In the NRMP, a doctor and a hospital that form a blocking pair have an incentive to match with one another outside of the matching program, thus an unstable matching is not at equilibrium. Second, empirical evidence suggests that stable matchings prevent *unraveling*, that is clearing houses that implement stable matchings tend to achieve high participation rates (Roth, 1991). Third, stability constitutes an essential fairness criterion in some applications. This is the case in school choice, where stability ensures that a student may only miss out on a school he wants if all the students matched to that school have a higher priority.[3] Fairness is the main reason why stability is desirable in day care matching and tuition exchange. While one may be concerned about families and day care centers matching outside the centralized system, in practice, day care centers often do not freely choose which children to accept as some priorities are imposed by law and designed to ensure that places are allocated to those children who need them the most. In tuition exchange, the home university is in charge of selecting students, meaning that students and host universities do not have any possibility to re-match outside of the central

[3]Abdulkadiroglu and Sönmez (2003) refer to stability as the *elimination of justified envy* to highlight this fairness interpretation.

allocation. It is however both fair and in the home university's interest to send the students who have the strongest applications. Fourth, Delacrétaz, Kominers, and Teytelboym (2016) identify an additional reason why stability is desirable in refugee resettlement: it balances the preferences of refugee families and the priorities of local areas, giving the latter an incentive to participate in the resettlement program.

In a matching market where all agents have the same size (e.g. school choice), the set of stable matchings is nonempty and forms a lattice (Gale and Shapley, 1962; Roth and Sotomayor, 1990). This ensures the existence of an *agent-optimal stable matching*, that is a stable matching that makes all agents weakly better-off than all other stable matchings. Further, Gale and Shapley's (1962) agent-proposing Deferred Acceptance algorithm (Algorithm 1) finds this matching. This is no longer true in a matching market with sizes. The set of stable matchings may be empty (Example 1) and, if nonempty, it may contain multiple *agent-undominated stable matchings* instead of an agent-optimal stable matching (Example 2).[4] In addition, even when an agent-optimal stable matching exists, Gale and Shapley's (1962) agent-proposing Deferred Acceptance algorithm may fail to find it (Example 3). Matching markets with sizes call for different techniques that account for these differences.

Summary of the Results

The first part of the paper (Sections 3 and 4) focuses on stable matchings. The **Top-Down Bottom-Up (TDBU)** algorithm (Algorithm 2) iteratively identifies agents and objects that are not matched together in any stable matching. We combine it with a depth-first search in order to find an agent-undominated stable matching whenever the set of stable matchings is nonempty (Theorem 1). In the second part of the paper (Section 5), we show that eliminating waste and bounding instability to a given number of units per object are conflicting desiderata: at any non-wasteful matching, an agent who prefers an object to his own may have a higher priority than all agents matched to that object (Theorem 2). We introduce **size-stability**, a non-wasteful relaxation of stability that allows double-unit agents to envy single-unit agents. We argue that size-stable matchings that are undominated from the point of view of double-unit agents constitute a suitable solution concept because they are fair to double-unit agents despite the fact that their priorities may be violated. We show that such a matching exists (Corollary 5). The third and last part of the paper (Section 6) adapts our techniques in order to find a size-stable matching that is undominated from the point of view of double-unit agents (Theorem 3).

In the first part of the paper (Sections 3 and 4), we develop new techniques to find a stable

[4]An agent-undominated stable matching is such that, in every other stable matching, at least one agent is strictly worse-off.

matching. We first introduce the Top-Down Bottom-Up (TDBU) algorithm (Algorithm 2), which iteratively eliminates agent-object pairs that are not part of any stable matching. This principle lies at the heart of Gale and Shapley's (1962) Deferred Acceptance (DA) algorithm (Algorithm 1). In every round of the agent-proposing DA algorithm, each agent proposes to his favorite object that has not rejected him yet. Every object tentatively accepts proposing agents in order of priority up to its quota and rejects all remaining proposals. This process is repeated until a round occurs where all proposals are tentatively accepted. At that point, every agent is matched to the last object to which he proposed. If an object rejects an agent in any round, then the pair is not part of any stable matching as this would make a higher-priority agent worse-off and create a blocking pair. Therefore, the DA algorithm iteratively eliminates pairs that are not part of any stable matching.

The Top-Down part of the TDBU algorithm adapts the agent-proposing Deferred Acceptance algorithm. Objects *reject* agents whose priority is too low for the pair to be part of a stable matching. An agents no longer contests an object after a rejection. The Bottom-Up part of the algorithm adapts the object-proposing Deferred Acceptance algorithm.[5] Objects give a *guarantee* to agents whose priority is too high for them to be matched to a less preferred object in a stable matching. An agent who receives a guarantee no longer contests any of his less preferred objects. In the terminology of the Deferred Acceptance algorithm, this corresponds to agents' rejecting objects. The TDBU algorithm stops when it can no longer identify any guarantee or rejection. It is polynomial-time solvable and simplifies the matching market by reducing the number of agent-object pairs that need to be considered. At the end of the algorithm, it is possible to match every agent to his favorite object among the ones he is still contesting. That matching is the agent-optimal stable one if it is feasible, that is if every object is available in sufficiently many units to accommodate all agents matched to it. Since the capacity of each object may be violated by at most one unit, this matching may unfortunately not be feasible (Proposition 4).

The **Undominated Stable Matching (USM)** algorithm (Algorithm 3) first runs the TDBU algorithm. If the agent-optimal stable matching is not found in this way, it restricts its search to stable matchings that match specific agents and objects. In each round, one agent-object pair is *protected* in the sense that the search focuses on stable matchings that match the agent and object in that pair. The TDBU algorithm runs again in order to identify agent-object pairs that are not matched together in any such stable matching. This continues until either a stable matching that satisfies these restricted criteria is found or it is established

[5]In this version of the algorithm, objects propose – up to their quotas – to the agents with the highest priority. Each agent tentatively accepts the proposal of his favorite object and rejects all others. In a market where all agents have the same size, this algorithm produces the *agent-pessimal* (or object-optimal) stable matching, that is the one that makes all agents weakly worse-off than all other stable matchings.

that none exists, in which case the USM algorithm protects different pairs. Eventually, it either finds a stable matching – which is undominated because of the way pairs are protected – or establishes that the set of stable matchings is empty (Theorem 1).

In the second part of the paper (Section 5), we develop alternatives to stability. Stability is characterized in this model by the combination of three properties: 1-boundedness, size-consistency and non-wastefulness (Proposition 2). For any nonnegative integer K, a matching is ***K*-bounded** if, for any agent-object pair where the agent prefers the object to his own, at most K units of the object are either unassigned or assigned to agents with a lower priority. K-boundedness constitutes a fairness criterion in the sense that it bounds the number of units that an agent can claim given his priority. A matching is **size-consistent** if, for any agent-object pair where the agent prefers the object to his own, all agents matched to that object have either a lower priority or a larger size. Size-consistency also constitutes a fairness criterion as it ensures that any violation of priority is due to agents having different sizes. A matching is **non-wasteful** if, for any agent-object pair where the agent prefers the object to his own, the object has enough unassigned units for the agent to be matched to it without removing any other agent. Non-wastefulness constitutes both a fairness and an efficiency criterion as it ensures that units only remain unassigned if they cannot benefit any agent.

We show that, for any nonnegative integer K, the existence of a K-bounded and non-wasteful matching is not guaranteed (Theorem 2). In contrast, Delacrétaz, Kominers, and Teytelboym's (2016) Priority-Focused Deferred Acceptance (PFDA) algorithm (Algorithm 4) finds a matching that is 1-bounded and size-consistent (Proposition 8), which implies the existence of such a matching. The disadvantage of this solution is that one unit per object may be wasted. Size-stability constitutes a non-wasteful alternative. In a size-stable matching, double-unit agents may envy single-unit agents but all other priorities are respected and every unit is assigned unless it cannot benefit any agent. We show that size-consistency and non-wastefulness characterize size-stability (Proposition 8) and that the set of size-stable matchings is nonempty (Proposition 10). Although size-consistency and non-wastefulness are two fairness criteria, size-stable matchings may be unfair to double-unit agents because there does not exist any bound on how many single-unit agents may violate their priority. In order to alleviate this problem, we propose **d-undominated size-stable matchings** as a solution concept. A d-undominated size-stable matching is such that all other size-stable matchings make at least one double-unit agents worse-off. These matchings maximize the welfare of double-unit agents given the size-stability constraint.

The third and last part of the paper (Section 6) is devoted to finding a d-undominated size-stable matching. The **Permissive Top-Down Bottom-Up (p-TDBU)** iteratively eliminates agent-object pairs in a way that is adapted to size-stability. The **d-Undominated**

Size-Stable Matching (d-USSM) algorithms (Algorithm 6) conducts a depth-first search that ultimately produces a d-undominated size-stable matching (Theorem 3). In every round, a set of agent-object pairs is protected and the p-TDBU algorithm is used to reduce the number of agent-object pairs that need to be considered.

Related Literature

Various approaches have been proposed to find a stable matching in couples problems. These are valid in our model, which is a special case where both partners want to work in the same hospital and are ranked identically by every hospital. The latest redesign of the National Resident Matching Program's (NRMP) algorithm took place in 1998. An "engineering" solution was implemented for this specific market (Roth and Peranson, 1997, 1999; Roth, 2003). Singles are matched first with the Deferred Acceptance algorithm. Couples are then introduced one at a time, following a random order. Whenever a couple enters, blocking pairs (medical graduates and hospitals who would rather be matched together than with their current partner) are matched until none remains. This algorithm may fail to find a stable matching even when one exists (Kojima, 2015) and, should one be found, it is not clear how it compares to other stable matchings in terms of doctor and hospital welfare.[6] Biró, Fleiner, and Irving (2016) and Biró and Fleiner (2016) propose an alternative approach based on Scarf's (1967) lemma. They argue using experimental and simulation evidence that this approach outperforms Roth and Peranson's (1997, 1999) in the sense that it finds a stable matching in more instances, particularly when the proportion of couples in the market is high. This does not mean, however, that it finds a stable matching in all instances where one exists. The algorithm proposed in this paper has the advantage of finding an agent-undominated stable matching whenever the set of stable matchings is nonempty.

Echenique and Yenmez (2007) study a college admission model where students have preferences over colleges as well as over the set of colleagues with whom they attend. The authors propose an algorithm based on each stable matching being a fixed point of an increasing function. It either finds all stable matchings or indicates that none exists. Kojima (2015) adapts this approach to couples problems and, by extension, to our model. From a computational point of view, Echenique and Yenmez (2007) admit that the algorithm may, in some cases, need to check every possible matching, though they argue it is unlikely to happen. The Top-Down Bottom-Up (TDBU) algorithm reduces the number of agent-object pairs that need to be considered in polynomial time. The depth-first search part may be computationally

[6] A stable matching has however been found every year since the redesign, making the solution a satisfying one for this specific application. In addition, whether the algorithm uses a doctor- or hospital-proposing procedure appears to have little impact on the outcome (Roth and Peranson, 1997).

more intensive[7] but does not require looking through every possible matching since many are eliminated by the TDBU algorithm at the start of every round. Another computational advantage is that our algorithm only needs to find one stable matching and guarantees that the latter is undominated. Kojima's (2015) algorithm requires finding all stable matchings before some of them can be identified as undominated.

Biró, Manlove, and McBride (2014) propose to use integer programming techniques to find a stable matching in the couples problem. While not considered in their paper, it may be possible to add an additional objective function in order to ensure that an undominated stable matching is found. Comparing the computational properties of these techniques with our USM algorithm lies beyond the scope of this paper. We see the two approaches as complementary for two reasons. First, there does not exist any polynomial-time solvable algorithm that finds a stable matching in this model (McDermid and Manlove, 2010). Consequently, the size of the market is key to an algorithm's ability to find a stable matching in a timely manner. As the TDBU algorithm is polynomial-time solvable, it can be used first to reduce the size of the market, thus substantially reducing the running time of any algorithm that finds a stable matching or report that the set of stable matchings is empty. Second, an important concern when it comes to practical applications is that the participants be able to understand how the algorithm works. The general idea behind the TDBU algorithm – which is based on Deferred Acceptance – and the depth-first search of the USM algorithm are intuitive and can be explained with relative ease. This can prove useful even in the hypothetical case where an equivalent but more efficient algorithm is ultimately used to calculate the matching. Third and last, our algorithms are constructive in the sense that they shed light on the structure of the problem. For example, the TDBU algorithm allows precisely identifying the case where an object is matched beyond capacity (Corollary 2).

Refugee resettlement (Delacrétaz, Kominers, and Teytelboym, 2016) constitutes a different extension of the present model. Refugee families require several units of different services (e.g. housing, school places or training programs) that can be provided by local areas, leading to a matching market with multi-dimensional constraints. Delacrétaz, Kominers, and Teytelboym's (2016) *Top Choice* algorithm adapts our USM algorithm to this more general setup and finds an undominated stable matching whenever one exists. The multidimensional constraints however imply that even the TDBU algorithm is not polynomial-time solvable in this context. Additionally, our USM algorithm takes advantage of the relative structure offered by the present model in order to protect specific agent-object pairs. The Top Choice algorithm could be used in our model but would be less efficient as a result. Delacrétaz,

[7]McDermid and Manlove (2010) show that finding whether or not a stable matching exists in this model is an NP-complete problem.

Kominers, and Teytelboym (2016) also provide a wasteful relaxation of stability. We show that their Priority-Focused Deferred Acceptance (PFDA) algorithm (Algorithm 4) finds a 1-bounded and size-consistent matching in our model.

Biró and McDermid (2014) study an extension of our model where agents' sizes lie between 1 and $n \geq 2$. They show that, if the quota of each object is increased or decreased by at most $n-1$ units, then a polynomial-time solvable algorithm exists that finds a stable matching. In our model, $n = 2$ so the quotas need to change by at most one unit. Dean, Goemans, and Immorlica (2006) and Yenmez (2014) each propose a polynomial-time algorithm based on deferred acceptance that are achieves these bounds. The matching produced by Dean, Goemans, and Immorlica's (2006) algorithm is stable if the capacity of each object is increased by at most $n-1$ units[8] and the one produced by Yenmez' (2014) algorithm is stable if the capacity of each object is decreased by at most $n-1$ units.[9] Nguyen and Vohra (2017) derive a similar result for the couples problem. Increasing or decreasing the quota of each object (hospital) by at most two units allows finding a market where a stable matching exists. In addition, the sum of quotas does not decrease and increases by at most four units. Our approach differs in the fact that we consider fixed quotas and do not modify them. Some of the algorithms we introduce nevertheless relate to this literature. The TDBU algorithm (Algorithm 2) finds a matching that is stable if at most one unit of capacity is added to each object. Whenever an extra unit is added, it benefits a single-unit agent while in Dean, Goemans, and Immorlica's (2006) algorithm it benefits a double-unit agent. Delacrétaz, Kominers, and Teytelboym's (2016) PFDA algorithm (Algorithm 4) finds a matching that is stable if at most one unit of capacity is removed from each object. That matching dominates the one produced by Yenmez' (2014) algorithm because the PFDA allows single-unit agents to be assigned the last unit of an object in some cases while Yenmez's (2014) algorithm precludes it.

Kennes, Monte, and Tumennasan (2014) study the assignment of children to day care centers in a dynamic context, taking into account the fact that children may move from one center to another once they have secured a place. The authors do not consider the part-time feature of day care and effectively build a dynamic extension of the school choice model, framed in the context of day care. Considering the dynamic aspect of day care is certainly

[8]Cseh and Dean (2016) adapt that algorithm in order to find a matching that minimizes the total number of units that need to be added.

[9]Yenmez (2014) studies a model of college admissions with contracts. Sizes are introduced by the fact that students can take up either a full place or a fraction $1/n$ of a place. Stability obtains by assuming that colleges do not need to fill up their capacity, instead they reject students as soon as they have less than a full place available. This can be translated into the model of Dean, Goemans, and Immorlica (2006) and Biró and McDermid (2014) by letting a full place be equal to n units of capacity. Then a college stops filling up its quota once it has $n-1$ or less units available.

worthwhile, however the success encountered by the reforms of school choice systems across the world suggests that taking care of that static problem can already greatly improve the way the market operates. A dynamic mechanism that caters for agents of different sizes could then lead to further improvements.

Finally, Dur and Ünver (2017) consider a different aspect of tuition exchange. Their emphasis lies on the balance of students between the two partners and its impact on exchange agreements in a dynamic environment. We instead take the terms of the agreements as given and focus on one university selecting which students it will send to its partners.

The remainder of the paper is organized as follows. Section 2 describes the setup and presents some preliminary results about stability. Section 3 introduces the Top-Down Bottom-Up (TDBU) algorithm, which eliminates agent-object pairs that are not part of any stable matching. We combine the TDBU algorithm with a depth-first search in Section 4 in order to find an undominated stable matching whenever one exists. Section 5 shows that a K-bounded and non-wasteful matching may not exist and introduces d-undominated size-stable matchings as a non-wasteful relaxation of stability. In Section 6, we adapt the techniques from Sections 3 and 4 in order to find a d-undominated size-stable matching. Section 7 concludes and all proofs are in the appendix.

2 Preliminaries

2.1 Setup

There are a set A of **agents** and a set O of **objects**. Each object $o \in O$ is available in $q_o \geq 1$ identical units. We refer to q_o as the **quota** of object o and define the **quota vector q** to be the $|O|$-dimensional vector containing all quotas. The set of agents is partitioned into two subsets S and D. Agents in S are the **single-unit agents** and require one unit. Agents in D are the **double-unit agents** and require two units of the same object.[10] We define $w_a \in \{1, 2\}$ such that $w_a \equiv 1$ if $a \in S$ and $w_a \equiv 2$ if $a \in D$ to be the **size** of agent a. The **size vector w** is the $|A|$-dimensional vector containing the size of all agents. We assume the existence a **null object**, denoted $\emptyset$, with a large enough quota to accommodate all agents: $q_\emptyset = \sum_{a \in A} w_a = 2|D| + |S|$.

[10]In school choice, agents are students, objects are schools, and units are seats in a school. All students are single-unit agents. In day care, agents are children (or their parents), objects are day care centers and units are part-time places in a center. Single-unit agents are children who require a part-time place and double-unit agents are children who require a full-time place. In tuition exchange, agents are students, objects are host universities, and units are places to go on exchange for a semester. Single-unit agents are students who want to go on exchange for a semster and double-unit agents are students who want to go on exchange for a year.

Agents have strict, transitive and ordinal **preference relations** over all objects. The preference relation of agent a is denoted $\succ_a$ and $o \succ_a o'$ signifies that agent a prefers object o to object o'. We denote the weak preference relation by $o \succeq_a o'$, which means that either $o \succ_a o'$ or $o = o'$. The **preference profile** is a $|A|$-tuple of preference relations $\boldsymbol{\succ} \equiv (\succ_a)_{a \in A}$, which contains the preference relations of all agents. An object is **acceptable** to an agent if he prefers it to the null object. For every object other than the null, agents are ranked in order of priority. $\rhd_o$ represents the **priority relation** of object o and consists of a strict and transitive ranking of all agents. $a \rhd_o a'$ signifies that agent a has a higher priority than agent a' for object o. We denote the weak priority relation by $a \unrhd_o a'$, which means that either $a \rhd_o a'$ or $a = a'$. The **priority profile** is a $|O|$-tuple $\boldsymbol{\rhd} \equiv (\rhd_o)_{o \in O}$ containing the priority relations of all objects.[11] A **market** is a tuple $\langle A, O, \boldsymbol{\succ}, \boldsymbol{\rhd}, \mathbf{w}, \mathbf{q} \rangle$.

The only difference between this model and school choice (Abdulkadiroglu and Sönmez, 2003) is the existence of double-unit agents. We next illustrate the impact that this seemingly small difference has on the set of stable matchings.

2.2 Stable Matchings

Throughout the paper, we consider sets of agent-object pairs $X \in 2^{A \times O}$. Where there is no risk of confusion, we use the term **pair** to refer to an agent-object pair. We say that agent a **contests** object o at set of pairs X if $(a, o) \in X$. We denote by $O_a(X) \equiv \{o \in O \mid (a, o) \in X\}$ the set of objects contested by a. The **top choice** of a at X, denoted $\overline{o}_a(X)$, is the object that a prefers among those that he contests. Formally, $\overline{o}_a(X) \equiv o \in O_a(X)$ such that $o \succeq_a o'$ for all $o' \in O_a(X)$. An agent who does not contest any object does not have a top choice. An object that is contested by agent a but is not his top choice is referred to as one of a's **subsequent choices**. An agent who contests at most one object does not have any subsequent choice.

A set of pairs X is **complete** if every agent contests *at least* one object. We denote the set of complete sets of pairs by $\mathbb{X} \equiv \{X \in 2^{A \times O} \mid |O_a(X)| \geq 1 \text{ for all } a \in A\}$. A **matching** is a set of pairs where every agent contests *exactly* one object. We denote the set of matchings by $\mathbb{M} \equiv \{\mu \in 2^{A \times O} \mid |O_a(\mu)| = 1 \text{ for all } a \in A\}$. For any matching $\mu \in \mathbb{M}$, we say that agent a is **matched to** object o at μ if $(a, o) \in \mu$. In that case, we say that a is **assigned** w_a units of o at μ.

[11] In order to keep the model as simple as possible, it is assumed that even the agent with the lowest priority can get an object so long as its units are not claimed by other agents. This assumption is natural in a model where the focus lies on agent welfare and without loss of generality since an eligibility threshold under which an agent cannot be matched to an object can be introduced by ranking the objects after the null one in the agent's preference relation.

For any set of pairs $X \in 2^{A \times O}$, we denote the set of agents who contest o at X by $A_o(X) \equiv \{a \in A \mid (a,o) \in X\}$ and the set of agents who contest o at X as their top choice by $\overline{A}_o(X) \equiv \{a \in A \mid \overline{o}_a(X) = o\}$. For a matching μ, $A_o(\mu) = \overline{A}_o(\mu)$ denotes the set of agents who are matched to o at μ. A matching is **feasible** if for every object the number of units assigned does not exceed the quota. Formally, $\mu \in \mathbb{M}$ is feasible if $\sum_{a \in A_o(\mu)} w_a \leq q_o$ for all $o \in O$.

We are now in a position to formally define stable matchings. For any $n \geq 1$, an agent a has a ***n*-unit claim** to an object o at matching μ if he prefers o to his own object and at least n units of o are either unassigned or assigned to agents with a lower priority. Formally, a has a n-unit claim to o at μ if $o \succ_a \overline{o}_a(\mu)$ and $\sum_{a' \in \hat{A}_{(a,o)}(\mu)} w_{a'} \leq q_o - n$, where $\hat{A}_{(a,o)}(X) \equiv \{a' \in A_o(X) \mid a' \rhd_o a\}$ denotes the set of agents who contest o at X and have a higher priority for it than a. We say that an agent has a claim to n units of o if he has a n-unit claim to that object. The pair (a,o) is a **blocking pair** of matching μ if a has a w_a-unit claim to o at μ.

Definition 1. *A matching is **stable** if it is feasible and does not have any blocking pair.*

We denote by $\mathbb{S} \subseteq \mathbb{M}$ the set of stable matchings. Definition 1 naturally extends the concept of stability from Gale and Shapley (1962) to a setup with single- and double-unit agents and is equivalent to it if all agents require one unit. Double-unit agents may have a 1-unit claims to some objects without affecting stability. This does not create a blocking pair because double-unit agents need two units in order to be matched to an object.

Domination is a partial order on the set of matchings $\mathbb{M}$. We say that matching μ dominates matching $\mu' \neq \mu$ if $\overline{o}_a(\mu) \succeq_a \overline{o}_a(\mu')$ for all $a \in A$. In words, μ dominates μ' if it makes all agents weakly better-off and at least one agent strictly better-off. We are in particular interested in partially ordering stable matchings. A stable matching $\mu \in \mathbb{S}$ is an **undominated stable matching** if there does not exist any *stable* matching $\mu' \in \mathbb{S}$ such that μ' dominates μ. A stable matching μ is the **optimal stable matching** if, for every stable matching $\mu' \neq \mu$, μ dominates μ'. An undominated stable matching exists as long as the set of stable matchings is nonempty. If there exists a unique undominated stable matching, then it is the optimal stable matching. Otherwise, the market does not have an optimal stable matching.

2.3 Preliminary Results

If all agents require one unit ($w_a = 1$ for all $a \in A$) (as in the school choice model), an optimal stable matching exists and the agent-proposing deferred acceptance algorithm (Algorithm 1) finds it (Gale and Shapley, 1962; Abdulkadiroglu and Sönmez, 2003). This is no longer true in general if there are both single- and double-unit agents, as we illustrate next. The set of stable matchings may be empty (Example 1) or contain multiple undominated stable matchings (Example 2). Additionally, the agent-proposing Deferred Acceptance algorithm may produce an unstable matching, even when an optimal stable matching exists (Example 3).

Example 1 (No Stable Matching)**.** There are two single-unit agents s_1 and s_2, one double-unit agents d_1 and two non-null objects o_1 and o_2. The preferences, priorities and quotas are detailed below:

$$\begin{array}{llll} \succ_{s_1}: o_1, o_2, \emptyset & \succ_{d_1}: o_1, \emptyset, o_2 & \rhd_{o_1}: s_2, d_1, s_1 & q_{o_1} = 2 \\ \succ_{s_2}: o_2, o_1, \emptyset & & \rhd_{o_2}: s_1, s_2, d_1 & q_{o_2} = 1 \end{array}$$

We show that the market presented in Example 1 does not have any stable matching. If d_1 is matched to o_1, then s_1 is not by feasibility. As s_1 has the highest priority for o_2, a blocking pair is formed unless he is matched to it. In that case however, s_2 is unmatched and forms a blocking pair with o_1. If d_1 is not matched to o_1, s_2 is as otherwise (d_1, o_1) constitutes a blocking pair. Then s_2 and o_2 form a blocking pair unless s_1 is matched to o_2. In this case however, (s_1, o_1) constitutes a blocking pair.

We next illustrate the fact that, even when the set of stable matchings is nonempty, it may not have the structure it has in canonical models.

Example 2 (Multiple USMs)**.** There are two single-unit agents s_1 and s_2, two double-unit agents d_1 and d_2 and two non-null objects o_1 and o_2. The preferences, priorities and quotas are detailed below:

$$\begin{array}{llll} \succ_{s_1}: o_1, o_2, \emptyset & \succ_{d_1}: o_1, \emptyset, o_2 & \rhd_{o_1}: s_2, d_1, s_1, d_2 & q_{o_1} = 2 \\ \succ_{s_2}: o_2, o_1, \emptyset & \succ_{d_2}: o_2, \emptyset, o_1 & \rhd_{o_2}: s_1, d_2, s_2, d_1 & q_{o_2} = 2 \end{array}$$

We show that the market presented in Example 2 has two undominated stable matchings:

$$\mu \equiv \{(s_1, o_1), (s_2, o_1), (d_1, \emptyset), (d_2, o_2)\} \quad \text{and} \quad \mu' \equiv \{(s_1, o_2), (s_2, o_2), (d_1, o_1), (d_2, \emptyset)\}.$$

If s_1 is matched to o_1, then d_1 is not by feasibility. The latter is then matched to the null object as he finds o_2 unacceptable. Additionally, stability dictates that d_2 be matched to o_2 as he has the second highest priority for that object behind s_1, who is not matched to it. s_2 is consequently not matched to o_2 as this would violate feasibility. He is matched to o_1 as otherwise he would have a claim to an unassigned unit of that object. We obtain matching

μ. s_1 and d_2 do not have a claim since they are matched to their first preference. s_2 is matched to his second preference but does not have a claim to his first one, o_2, as both units of that object are assigned to the higher priority agent d_2. d_1 is also matched to his second preference, however he only has a 1-unit claim to his first preference, o_1, as one unit of that object is assigned to the higher priority agent s_2. μ is consequently stable. By an analogous reasoning, μ' is the only stable matching where s_2 is matched to o_2. If neither s_1 nor s_2 is matched to his first preference, stability dictates that they both be matched to their second one as they have the highest priorities for these objects. That is, s_1 is matched to o_2 and s_2 is matched to o_1. By feasibility, d_1 and d_2 are both matched to the null object. The matching obtained is not stable as it has two blocking pairs: (s_1, o_1) and (s_2, o_2). We conclude that μ and μ' are the only two stable matchings in this market. As μ favors s_1 and d_2 while μ' favors s_2 and d_1, they are both undominated stable matchings.

We have seen in the above two examples that an optimal stable matching may not exist, instead the set of stable matchings may be empty or contain multiple undominated stable matchings. An obvious implication is that, contrary to the school choice model, the Deferred Acceptance algorithm does not always find the optimal stable matching. A natural question is then whether the Deferred Acceptance algorithm finds the optimal stable matching whenever it exists. We answer in the negative by presenting a counterexample.

Example 3 (DA Fails)**.** There are three single-unit agents s_1, s_2 and s_3, two double-unit agent d_1 and d_2 and three non-null objects o_1, o_2 and o_3. The preferences, priorities and quotas are detailed below:

$$\begin{array}{llll}
\succ_{s_1}: o_1, o_3, o_2, \emptyset & \succ_{d_1}: o_1, o_3, o_2, \emptyset & \rhd_{o_1}: s_3, d_1, d_2, s_1, s_2 & q_{o_1} = 2 \\
\succ_{s_2}: o_3, o_2, o_1, \emptyset & \succ_{d_2}: o_3, \emptyset, o_2, o_1 & \rhd_{o_2}: s_1, s_2, d_2, s_3, d_1 & q_{o_2} = 1 \\
\succ_{s_3}: o_2, o_1, o_3, \emptyset & & \rhd_{o_3}: s_3, d_1, d_2, s_1, s_2 & q_{o_3} = 2
\end{array}$$

We show that

$$\mu = \{(s_1, o_1), (s_2, o_2), (s_3, o_1), (d_1, o_3), (d_2, \emptyset)\}$$

is the unique stable matching in this market, which directly implies that it is the optimal one. Consider first the case where d_2 is matched to o_3. By feasibility, no other agent is matched to that object. Then one of s_1 or s_2 is matched to o_2 as otherwise (s_2, o_2) constitutes a blocking pair. Again by feasibility, this means that s_3 is not matched to o_2. As s_3 has the highest priority for his second preference, o_1, he is matched to that object. Feasibility then prevents d_1 from being matched to o_1. d_1 is consequently matched to an object he ranks below o_3 so that (d_1, o_3) constitutes a blocking pair. As d_2 finds o_1 and o_2 unacceptable, we conclude that

Algorithm 1: Agent-Proposing Deferred Acceptance (Gale and Shapley, 1962)

```
Initialization:
Given a market, let $X_1 \equiv A \times O$.
Round $k \geq 1$:
    Given $X_k$, every agent $a \in A$ proposes to his top choice $\overline{o}_a(X_k)$.
    For each object $o$, agents are considered one at a time in order of priority. An
    agent is tentatively accepted if the total number of units required by himself
    and all tentatively accepted agents with a higher priority does not exceed $q_o$.
    The agent is rejected otherwise.
    If all agents are tentatively accepted, let $\mu^{\text{DA}} \equiv \overline{\mu}(X_k)$. Otherwise, construct
    $X_{k+1}$ by eliminating from $X_k$ all pairs involving an agent and an object such
    that the latter has rejected the former's proposal and continue to Round $k+1$.
```

Round 1		Round 2		Round 3		Round 4		Round 5	
$s_1 \to o_1$	✗	$s_1 \to o_3$	✗	$s_1 \to o_2$	✓	$s_1 \to o_2$	✓	$s_1 \to o_2$	✓
$s_2 \to o_3$	✗	$s_2 \to o_2$	✓	$s_2 \to o_2$	✗	$s_2 \to o_1$	✓	$s_2 \to o_1$	✓
$s_3 \to o_2$	✓	$s_3 \to o_2$	✗	$s_3 \to o_1$	✓	$s_3 \to o_1$	✓	$s_3 \to o_1$	✓
$d_1 \to o_1$	✓	$d_1 \to o_1$	✓	$d_1 \to o_1$	✗	$d_1 \to o_3$	✓	$d_1 \to o_3$	✓
$d_2 \to o_3$	✓	$d_2 \to o_3$	✓	$d_2 \to o_3$	✓	$d_2 \to o_3$	✗	$d_2 \to \emptyset$	✓

Table 1: DA algorithm on Example 3.

he is matched to the null object in all stable matchings. One of s_3 or d_1 is matched to o_3 as a result, otherwise (d_2, o_3) constitutes a blocking pair. If s_3 is matched to o_3, however, (s_3, o_1) constitutes a blocking pair since s_3 prefers o_1 to o_3 and has the highest priority for it. We conclude that d_1 is matched to o_3 in all stable matchings. This implies that s_3 is matched to o_1 as otherwise (d_1, o_1) constitutes a blocking pair. Additionally, s_1 and o_1 form a blocking pair if they are not matched together since o_1 is s_1's first preference and neither d_1 nor d_2 is matched to it. s_2 is not matched to o_3 by feasibility and therefore forms a blocking pair with o_2 unless he is matched to it. We conclude that μ is the unique stable matching in this market, which directly implies that it is the optimal one.

The agent-proposing Deferred Acceptance algorithm is presented in Algorithm 1. Table 1 displays its steps when applied to Example 3. In Round 1, every agent proposes to his first preference. o_1 rejects s_1 because it also has a proposal from d_1 who has a higher priority and it does not have enough units for both agents. o_3 similarly rejects s_2 because of d_2. It is already clear at this point that the Deferred Acceptance algorithm does not produce μ since s_1 no longer contests o_1. In Round 2, s_1 proposes to his second choice o_3 and, like s_2 in the previous round, is rejected because of d_2. s_2 proposes to o_2, leading to that object rejecting s_3. In Round 3, s_1 is down to his third preference: o_2. He is tentatively accepted this time as he has the highest priority for that object, however o_2 has a quota of one unit, meaning that s_2 is rejected. Meanwhile, s_3 proposes to o_1, his second preference and is likewise tentatively accepted as he has the highest priority for that object. d_1 is rejected as a result since only one unit of o_1 remains available. This is where a blocking pair is created. o_1 rejected s_1 in Round 1 because of d_1. Now that s_3 has replaced d_1, one unit has become available but s_1 can no longer benefit from it as he was rejected earlier on. In Round 4, s_2 benefits from that available unit as he proposes to o_1. The blocking pair remains however as $s_1 \rhd_{o_1} s_2$. d_1 proposes to o_3 after o_1 rejected him. He is tentatively accepted as he has a higher priority than d_2. The latter is rejected as o_3 has a quota of two. Finally, d_2 proposes to the null object in Round 5. None of the agents are rejected and the algorithm ends. It generates

$$\mu^{DA} = \{(s_1, o_2), (s_2, o_1), (s_3, o_1), (d_1, o_3), (d_2, \emptyset)\}.$$

This matching is not stable since (s_1, o_1) constitutes a blocking pair. Combining the information gathered through our three examples yields the following result.[12]

Proposition 1. *The set of stable matchings may be empty or contain multiple undominated stable matchings. The Deferred Acceptance algorithm may produce an unstable matching even if an optimal stable matching exists.*

2.4 Characterization of Stability

We introduce three properties that characterize stability and will prove crucial throughout the remainder of the paper. In Sections 3 and 4, we search for a matching that satisfies all three properties. That matching is then stable by our characterization result. In Sections 5 and 6, we propose relaxations of stabilitythat are characterized by subsets of these properties.

The first property bounds the size of claims that may exist at a given matching. For any nonnegative integer K, a matching $\mu \in \mathbb{M}$ is ***K*-bounded** if there does not exist any $K+1$-unit claim at μ. If a matching is K-bounded, an agent has the assurance that at most K units of an object he prefers to his own are either unassigned or assigned to agents with a

[12]McDermid and Manlove (2010) show that the set of stable matchings may be empty in this model. They also show that no polynomial-time solvable algorithm finds a stable matching whenever one exists, which implies the last part of the statement as Deferred Acceptance is polynomial-time solvable.

lower priority. This constitutes a fairness criterion as a matching may be seen as "reasonably fair" if only a small proportion of the units available are not assigned to the agents with the highest priority.

The second property ensures that priorities may only be violated when agents have different sizes. Given a matching $\mu \in \mathbb{M}$, we say that agent a **envies** agent a' at μ if $\overline{o}_{a'}(\mu) \succ_a \overline{o}_a(\mu)$ and $a \rhd_o a'$. That is, a envies a' if he would prefer to be matched to a''s object and has a higher priority for it. A matching μ is **size-consistent** if for all $a, a' \in A$ such that a envies a' at μ, $w_a > w_{a'}$. In words, double-unit agents only envy single-unit agents and single-unit agents do not envy anyone. This again constitutes a fairness criterion: if an agent's priority is violated, this can be justified by his larger size.

The third and last property is common in matching theory. A matching $\mu \in \mathbb{M}$ is **wasteful** if an agent a has a claim to at least w_a unassigned units of an object o at μ. Formally, μ is wasteful if there exists $a \in A$ and $o \in O$ such that $o \succ_a \overline{o}_a(\mu)$ and $\sum_{a' \in A_o(\mu)} w_{a'} \leq q_o - w_a$. Waste arguably constitutes the worst kind of blocking pair since the units to which an agent has a claim do not even benefit an agent with a lower priority. As such, non-wastefulness constitutes both a fairness and an efficiency criterion. We are now in a position to state our characterization result.

Proposition 2. *A feasible matching is stable if and only if it is* 1*-bounded, size-consistent and non-wasteful.*

3 Top-Down Bottom-Up (TDBU)

The Deferred Acceptance algorithm fails to produce a stable matching in Example 3 because s_1 proposes to o_1 "at the wrong time". That is, s_1 proposes to o_1 in Round 1 when the latter also receives a proposal from a double-unit agent with a higher priority. Had s_1 been able to wait for d_1 to be rejected, he would have been accepted as s_3 only requires one unit. A solution to improve the Deferred Acceptance algorithm emerges from these observations: o_1 could tentatively accept s_1 despite not currently having a unit available for him, in case one becomes available. This is in part the essence of the Top-Down Bottom-Up (TDBU) algorithm, which we introduce in this section. We introduce the algorithm in Section 3.1 and present its properties in Section 3.2. In Section 3.3, we show that the TDBU algorithm finds the unique stable matching of Example 3.

3.1 Description

The TDBU algorithm is formally presented in Algorithm 2. Any set of agent-object pairs $X \in 2^{A \times O}$ can enter and the algorithm produces $\varphi(X)$, which is either the empty set or a complete subset of X. $X^1 \equiv X$ enters the first step[13] and the algorithm identifies rejections and guarantees in order to eliminate pairs and obtain $X^2 \subseteq X^1$. X^2 then enters the second step, where pairs are again eliminated in order to obtain $X^3 \subseteq X^2$. This continues until some Step $K \geq 1$ where either an incomplete set of pairs enters ($X^K \notin \mathbb{X}$) or the algorithm is unable to eliminate any pair ($X^K = X^{K+1}$). In the former case, X^K does not include any stable matching. The algorithm ends and produces the empty set ($\varphi(X) = \varnothing$). In the former case, the algorithm ends and produces $\varphi(X) = X^K$.

Below, we define the rules according to which the TDBU algorithm eliminates pairs in any given step from the Top-Down and from the Bottom-Up. We begin with the latter.

Bottom-Up

Let $X \in \mathbb{X}$ be a complete set of agnet-object pairs. For every pair $(a, o) \in X$, the **guarantee number** of agent a for object o at X, denoted $\mathcal{G}_{(a,o)}(X)$, is obtained by adding up the size of a and the sizes of all agents who contest o at X and have a higher priority than him. That is,

$$\mathcal{G}_{(a,o)}(X) \equiv w_a + \sum_{a' \in \hat{A}_{(a,o)}(X)} w_{a'}.$$

We say that o gives a a **guarantee** at X if $\mathcal{G}_{(a,o)}(X) \leq q_o$. In that case, agents with a higher priority than a may be assigned at most $q_o - w_a$ units of o in all matchings included in X. Should a be matched to a less preferred object, (a, o) would constitute a blocking pair. This means that a is guaranteed to be matched to either o or an object he prefers in all stable matchings included in X. All pairs that involve a and a less preferred object to o can be eliminated as they are not an element of any stable matching included in X. We formalize this idea below.

Lemma 1. *If $\mathcal{G}_{(a,o)}(X) \leq q_o$, then for any object $o' \in O$ such that $o \succ_a o'$ and any stable matching included in X $\mu \in \mathbb{S} \cap 2^X$, $(a, o') \notin \mu$.*

Guarantees can be thought of as objects proposing to the agents with the highest priority up to its quota. These agents have the assurance to be matched to that object in the worst-

[13]We use the term "Round" for algorithms that start with the full set of pairs $A \times O$ and produce either a matching or the empty set (Algorithms 1, 3, 4, and 6). We use the term "Step" for algorithms that start with any set of pairs and produce either a complete subset (but not necessarily a matching) or the empty set (Algorithms 2 and 5). This is to avoid confusion as Algorithms 2 and 5 are used in some rounds of Algorithms 3 and 6, respectively.

Algorithm 2: Top-Down Bottom-Up (TDBU)

```
Initialization:
Given a market and a set of agent-object pairs $X$, let $X^1 \equiv X$.
Step $k \geq 1$:
If $X^k \notin \mathbb{X}$, let $\varphi(X) \equiv \varnothing$. Otherwise, construct $X^{k+1}$ by eliminating from $X^k$ all
agent-object pairs such that, at $X^k$, either the agent receives a guarantee from an
object he prefers or the object rejects the agent.
If $X^{k+1} = X^k$, let $\varphi(X) \equiv X^k$. Otherwise, continue to Step $k+1$.
```

case scenario and stop contesting any less preferred object. In the next step, this will reduce the guarantee numbers agents with a lower priority, potentially allowing some of them to receive a guarantee.

Top-Down

In contrast to its bottom-up counterpart, the top-down part of the algorithm identifies pairs such that the agent's priority is too low in order for him to get that object in any stable matching. Let again $X \in \mathbb{X}$ be a complete set of agent-object pairs and let $(a, o) \in X$ be one of those pairs. We define the **envy set** of agent a for object o at X, denoted $E_{(a,o)}(X)$, to be the set of agents who would envy a – as defined in Section 2.4 – should he be matched to o. That is,

$$E_{(a,o)}(X) \equiv \{a' \in A \mid o \succ_{a'} \overline{o}_a(X) \text{ and } a' \rhd_o a\}.$$

$E_{(a,o)}(X)$ contains all agents who have a higher priority than a and prefer o to their top choice at X. In any matching $\mu \subseteq X$ where $(a, o) \in \mu$, these agents envy a. Agents may be only be matched to an object for which they have a nonempty envy set in limited instances as shown in the following lemma.

Lemma 2. *For any stable matching $\mu \in \mathbb{S}$ and any object o, there exists at most one agent a such that $(a, o) \in \mu$ and $E_{(a,o)}(\mu) \neq \varnothing$. Furthermore, $a \in S$, a has the lowest priority among agents matched to o and $\sum_{a' \in A_o(\mu)} = q_o$.*

Lemma 2 derives from 1-boundedness. If a double-unit agent or multiple single-unit agents are matched to o and envied, a 2-unit claim exists and the matching is not 1-bounded. This inevitably occurs if another agent matched to o has a lower priority than a since that agent would be envied as well. If $\sum_{a' \in A_o(\mu)} w_{a'} < q_o$, at least one unit of o is unassigned. Any agent who envies a has a claim to the unit assigned to a as well as any unassigned unit, violating 1-boundedness.

As a counterpart to his guarantee number, we next define the **rejection number** of agent a for object o at X, which we denote by $\mathcal{R}_{(a,o)}(X)$. This number is calculated by adding up the size of a, the size of all agents who contest o as their top choice and have a higher priority than a, and the size of all agents in the envy set of (a, o) at X:

$$\mathcal{R}_{(a,o)}(X) \equiv w_a + \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'} + \sum_{a' \in E_{(a,o)}(X)} w_{a'}.$$

Before we turn to the connection between envy sets, rejection numbers and whether an agent-object pair is an element of any stable matching, let us outline a property that follows almost directly from the definition of guarantee and rejection numbers.

Lemma 3. *Let $(a, o) \in X \subseteq X'$ with $X, X' \in \mathbb{X}$. Then*

$$\mathcal{G}_{(a,o)}(X) \leq \mathcal{G}_{(a,o)}(X') \quad \text{and} \quad \mathcal{R}_{(a,o)}(X) \geq \mathcal{R}_{(a,o)}(X').$$

Lemma 3 means that, as the TDBU algorithm advances and eliminates more agent-object pairs, guarantee numbers decrease and rejection numbers increase. The former may allow finding additional guarantees and, as we see next, the latter may allow finding additional rejections. Finally, rejection numbers and envy sets are closely related in the following sense. Suppose that the rejection number of a for o at X exceeds the object's quota, that is $\mathcal{R}_{(a,o)}(X) > q_o$. Then in any feasible matching $\mu \subseteq X$ such that $(a, o) \in \mu$, an agent with a higher priority than a is matched to a less preferred object, meaning he envies a. We formalize this below.

Lemma 4. *If $\mathcal{R}_{(a,o)}(X) > q_o$, then for any feasible matching $\mu \subseteq X$, $E_{(a,o)}(\mu) \neq \varnothing$.*

It will prove convenient throughout the remainder of the paper to define $S_o(X) \equiv A_o(X) \cap S$ to be the set of single-unit agents who contest object o at X. Similarly, let $\overline{S}_o(X) \equiv \overline{A}_o(X) \cap S$ be the set of single-unit agents who contest o at X as their top choice and $\hat{S}_{(a,o)}(X) \equiv \hat{A}_{(a,o)}(X) \cap S$ be the set of single-unit agents that contest o at X and have a higher priority than a. We analogously define $D_o(X) \equiv A_o(X) \cap D$, $\overline{D}_o(X) \equiv \overline{A}_o(X) \cap D$ and $\hat{D}_{(a,o)}(X) \equiv \hat{A}_{(a,o)}(X) \cap D$. We are now in a position to state conditions under which it can be established that (a, o) is not an element of any stable matching included in X.

Lemma 5. *For any $\mu \in \mathbb{S} \cap 2^X$, $(a, o) \notin \mu$ if any of the following three conditions is satisfied:*

(i) There exists $a' \in E_{(a,o)}(X)$ such that $w_a \geq w_{a'}$ or there exists $s \in \hat{S}_{(a,o)}(X) \cap \overline{S}_o(X)$ such that $E_{(s,o)}(X) \neq \varnothing$;

(ii) $a \in D$ and $\mathcal{R}_{(a,o)}(X) > q_o$;

(iii) $a \in S$, $\mathcal{R}_{(a,o)}(X) > q_o$ and either

- *There exists $s \in \overline{S}_o(X)$ such that $\mathcal{R}_{(a,o)}(X) > \mathcal{R}_{(s,o)}(X) \geq q_o$, or*
- *$\mathcal{R}_{(a,o)}(X) - q_o$ is odd and for all $s \in S_o(X) \setminus \overline{S}_o(X)$, $\mathcal{R}_{(s,o)}(X) \geq q_o$.*

Throughout the paper, we refer to the different parts of Lemma 5 as Conditions (i), (ii) and (iii). If any of these conditions holds, we say that object o **rejects** agent a at X. Condition (i) identifies cases where an object rejects an agent independently of his rejection number. This occurs when another agent who does not contest the object envies him, making it impossible for the pair to be an element of any stable matching.[14] The first part of Condition (i) derives directly from size-consistency. If a pair (a, o) satisfied it, then in any matching $\mu \subseteq X$ that contains (a, o), an agent no larger than a envies him. μ is, as a result, not size-consistent, hence not stable. The second part derives directly from size-consistency and Lemma 2. If (a, o) satisfies it, then any stable matching $\mu \in \mathbb{S}$ that contains (a, o) contains (s, o) by size-consistency. As $s \triangleright a$ and $E_{(s,o)}(X) \neq \varnothing$, $E_{(a,o)}(X) \neq \varnothing$, which implies that μ is not stable by Lemma 2. Condition (ii) derives from Lemma 4. If a is a double-unit agent and his rejection number exceeds q_o, he is envied in any feasible matching $\mu \subseteq X$ that contains (a, o). This violates size-consistency as well as 1-boundedness, therefore μ is not stable.

Single-unit agents may in contrast be envied by double-unit agents in a stable matching, therefore a rejection number above the quota is not enough to draw the same conclusion. Condition (iii) identifies those cases where it can be inferred that (a, o) is not an element of any stable matching included in X. The formal proof is left for the appendix, instead we illustrate this condition in Table 2. Each of the three panels represents an object o that is available in four units ($q_o = 4$) and the agents that contest it at some set of pairs X. Agents are displayed in the first column in order of priority (the agent in the first row has the highest priority and so on). A "T" is displayed in the second column if o is the agent's top choice. The third column is devoted to each agent's rejection number and a cross appears in the fourth column if the object rejects the agent, that is if the pair satisfies Lemma 5 at X. We assume for simplicity that all agents' envy sets are empty at X, therefore each agent's

[14]Condition (i) does not play a role when all agent-object pairs enter the TDBU algorithm. If the algorithm starts with a set $X \subset A \times O$, however Condition (i) may allow identifying pairs that are not an element of any stable matching included in X.

rejection number is found by adding up his own size and the sizes of all agents with a higher priority who contest o as their top choice.

In Panel (a), the rejection number of s_1 and d_2 are their own sizes, respectively 1 and 2. As o is d_2's top choice, his size counts towards d_3's rejection number, which is 4. s_2's rejection number is calaculated by adding his size to d_2 and d_3's, leading to a rejection number of $2+2+1=5$. s_3's rejection number is similarly $2+2+1+1=6$. s_2's rejection number exceeds the quota, however o does not reject him as (s_2, o) does not satisfy Condition (iii). The first part of Condition (iii) does not hold since s_2 has the highest priority among agents who contest o as their top choice and have a rejection number larger than or equal to the quota. The second part does not apply either since s_1 contests o as a subsequent choice and his rejection number is less than the quota. The rationale for o not rejecting s_2 is that it could become s_1's top choice latter on. In that case, d_3 would be rejected and a unit would become available to s_2. On the other hand, o rejects s_3. This illustrates the first part of Condition (iii): $s_2 \in \overline{S}_o(X)$, $s_2 \rhd_o s_3$ and $\mathcal{R}_{(s_2,o)}(X) = 5 > 4 = q_o$. o rejects s_3 because even if a unit were to become available as a result of a double-unit agent's rejection, it would benefit s_2 and not s_3.

Panel (b) illustrates the case where the first part of Condition (iii) is satisfied because a single-unit agent contests the object as his top choice and has a rejection number equal to the quota. s_4 cannot obtain the unit of o currently assigned to s_3 even if o becomes s_1's top choice, therefore o rejects s_4. Panel (c) illustrates another situation where an object does not reject a single-unit agent with a rejection number above the quota. (s_2, o) clearly does not satisfy the first part of Condition (iii). It does not satisfy the second part either as $\mathcal{R}_{(s_2,o)}(X) - q_o = 6 - 4 = 2$ is even. The reason why o does not reject s_2 is that it rejects d_3 as the latter's rejection number exceeds the quota (Condition (ii)). Therefore, the last unit of o remains available for s_2.

Finally, Panel (d) illustrates a situation where a single-unit agent is rejected because of the second part of Condition (iii). s_1 is rejected as $R_{(s_1,o)}(X) - q_o = 5 - 4 = 1$ is odd and there does not exist any single-unit agent who contests o as a subsequent choice and has a rejection number below the quota. The rationale for s_1's rejection is that if d_4 is rejected, this will be because o has become the top choice of another double-unit agent, as a result no unit is made available to s_1.

3.2 Results and Discussion

The TDBU algorithm produces a subset of the set of pairs that entered. As our following proposition formalizes, it however preserves all stable matchings included in that set.

Panel (a)

o	Pref.	$\mathcal{R}$	(4)
s_1		1	
d_2	T	2	
d_3	T	4	
s_2	T	5	
s_3		6	✗

Panel (b)

o	Pref.	$\mathcal{R}$	(4)
s_1		1	
s_2	T	1	
d_1	T	3	
s_3	T	4	
s_4		5	✗

Panel (c)

o	Pref.	$\mathcal{R}$	(4)
d_1		2	
d_2	T	2	
s_1	T	3	
d_3	T	5	✗
s_2	T	6	

Panel (d)

o	Pref.	$\mathcal{R}$	(4)
d_1		2	
d_2		2	
d_3	T	2	
d_4	T	4	
s_1		5	✗

Table 2: Illustration of Condition (iii) from Lemma 5.

Proposition 3. *For any $X \in 2^{A\times O}$ and any stable matching $\mu \in \mathbb{S}$, $\mu \subseteq X$ implies $\mu \subseteq \varphi(X)$.*

Proposition 3 almost directly follows from Lemmas 1 and 5, which ensure that none of the agent-object pairs eliminated throughout are an element of any stable matching included in X.

The first step towards finding a stable matching is to run the TDBU algorithm over the full set of pairs $A \times O$ in order to narrow down the problem to a smaller set of agent-object pairs. Our next result states the main properties of $\varphi(A\times O)$. It is useful to define, for any complete set of agent-object pairs $X \in \mathbb{X}$, the **top matching** of X, denoted $\overline{\mu}(X) \equiv \{(a, \overline{o}_a(X)\}_{a\in A}$. In words, the top matching of any complete set of pairs is the one where all agents are matched to their top choice.

Proposition 4. *$\varphi(A \times O)$ is complete and includes all stable matchings. $\overline{\mu}(\varphi(A \times O))$ is 1-bounded, size-consistent and non-wasteful. It is the optimal stable matching if and only if it is feasible and dominates all stable matchings otherwise.*

If all agent-object pairs enter the TDBU algorithm, it yields a complete set of pairs because all agents receive a guarantee from the null object, therefore, an agent contests the null object. By Proposition 3, that subset includes all stable matchings. Consequently, any matching that makes an agent better-off compared to $\overline{\mu}(\varphi(A \times O))$ is unstable, hence $\overline{\mu}(\varphi(A \times O))$ is either the optimal stable matching or it is unstable and dominates all stable matchings.

Running the TDBU algorithm on $A \times O$ and matching all agents to their top choice may allow finding the market's optimal stable matching, in fact we show in Section 3.3 that this is the case in Example 3. Unfortunately, the matching obtained in this way may not be stable. In that case, more agent-object pairs need to be eliminated in order to find a stable matching or establish that none exists. This is the purpose of Section 4.

The Top-Down Bottom-Up algorithm can be thought of as simultaneously running both versions of the Deferred Acceptance algorithm, with suitable modifications to fit a setting where some agents require two units. It creates an upper and a lower bound on the set of stable matchings that move towards one another. By the time the algorithm ends, the top choice of each agent represents an upper bound: the agent cannot get an object he prefers in any stable matching. The object the agent's least preferred object among those he still contests provides a lower bound: the agent cannot be matched to a less preferred object in any stable matching. In Example 3, those bounds coincide for all agents and the unique stable matching is found. In matching markets where all agents require one unit ($w_a = 1$ for all $a \in A$), the bounds correspond to the extreme stable matchings: $\overline{\mu}(\varphi(A \times O))$ is the optimal stable matching while the pessimal (or object-optimal) stable matching is found by giving all agents the object they like least among those they contest at $\varphi(A \times O)$.

If the objective is to find the optimal stable matching (or an undominated one), one might question the purpose of the Bottom-Up part of the algorithm, that is the guarantees. Determining the worst object with which agents can be matched does not appear directly relevant to finding an undominated stable matching. The reason comes from the second part of Lemma 5's Condition (iii). The existence of a single-unit agent who contests an object as a subsequent choice may be key to determining whether or not the object rejects a single-unit agent with a rejection number greater than the quota. If single-unit agents stop contesting objects below their lower bound, objects may be able to reject single-unit agents that they could not reject otherwise. Panel (a) of Table 2 illustrates such a situation. o does not reject s_2 because a unit would become available to him should s_1 contest o as his top choice later. If s_1 stops contesting o because it can be established that he is matched to an object he prefers in all stable matchings, then o rejects s_2. In Section 3.3, we further illustrate the importance of guarantees as we show that the TDBU algorithm would not find the optimal stable matching of Example 3 if it only consisted of its Top-Down part.

We conclude with a remark on computational complexity. In every step, the TDBU algorithm considers each agent-object pair exactly once to give it a guarantee and a rejection number and, depending on these numbers, determines whether the agent receives a guarantee, a rejection or neither. Therefore, the number of substeps required within each step is bounded by $|A| \cdot |O|$. In all steps but the last one, at least one pair is eliminated, therefore the number of steps is also bounded by $|A| \cdot |O|$. The TDBU algorithm is consequently polynomial-time solvable. Finding a stable matching is on the other hand an NP-hard problem (McDermid and Manlove, 2010), which becomes apparent in the next section. The size of the market that is considered is therefore key to the running time of any algorithm that computes a stable matching. An advantage of the TDBU algorithm is that it reduces the number of pairs that need to be considered in polynomial time. It is then possible to compute a stable matching in a smaller market.

3.3 Example

We illustrate the TDBU algorithm with Example 3. As we have seen, the DA algorithm fails to find a stable matching in that market. We show that, in contrast, the TDBU algorithm finds the unique (and thus optimal) stable matching.

All computation steps are displayed in Table 3. Each panel is devoted to a non-null object and contains six columns. Agents who contest the object are displayed in the first column in order of priority. Each row within a panel is therefore devoted to an agent-object pair. The agent's ordinal preference for the object is displayed in the second column. Given the important role of top choices in determining rejection numbers, we denote them by "T". We use "U" ("unacceptable") if the agent prefers the null to the object. These pairs are eliminated in the first step as the null object gives by definition a guarantee to all agents. The guarantee and rejection numbers associated with the pair are respectively displayed in the third and fourth columns. The fifth column displays a check mark if the object gives the agent a guarantee and a cross if it rejects him. We write "El." in the sixth and last column if the pair is eliminated. This occurs when either the agent receives a guarantee for an object he prefers or the object rejects the agent. Eliminated pairs do not enter the next step. They may however still impact rejection numbers as the agent may join the envy set of an agent with a lower priority who still contests the object.

Step 1

All agent-object pairs enter the first step: $X^1 \equiv A \times O$. In the first panel, devoted to o_1, the agents are displayed in order of priority. o_1 is d_1 and s_1's top choice, s_3's second choice, s_2's third choice and is unacceptable to d_2. As s_3's is a single-unit agent and lies at the top of the priority list, both his guarantee and rejection numbers are 1. d_1's guarantee number is 3 (=1+2) but his rejection number is only 2 since o_1 is not s_3's top choice. d_2's guarantee number is 5 (=1+2+2). His rejection number is 4 (=2+2) as d_1's size counts towards it but s_3's does not. Analogously, s_1's guarantee and rejection numbers are respectively 6 (=1+2+2+1) and 3 (=2+1). Notice that s_1's rejection number is smaller than d_2's despite the latter having a higher priority. This is due to s_1's smaller size and the fact that d_2 contests o_1 as a subsequent choice, hence his size does not count towards s_1's rejection number. Finally, s_2's guarantee number is 7 (=1+2+2+1+1) and his rejection number, which takes the sizes of d_1 and s_1 into account, is 4 (=2+1+1).

o_1 gives s_3 a guarantee as his guarantee number is 1 and $q_{o_1} = 2$. A check mark is consequently displayed in the fifth column. o_1 on the other hand rejects d_2 since the latter's rejection number is larger than the quota (Condition (ii) in Lemma 5), resulting in the fifth column displaying a cross and the sixth column displaying "El.". The pair (d_2, o_1) is eliminated

Step 1

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.	o_3	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
s_3	2	1	1	✓		s_1	3	1	1	✓		s_3	3	1	1	✓	El.
d_1	T	3	2			s_2	2	2	1			d_1	2	3	2		
d_2	U	5	4	✗	El.	d_2	U	4	2	✗	El.	d_2	T	5	2		
s_1	T	6	3			s_3	T	5	1			s_1	2	6	3		
s_2	3	7	4	✗	El.	d_1	3	7	3	✗	El.	s_2	T	7	3		

Step 2

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.	o_3	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
s_3	2	1	1	✓		s_1	3	1	1	✓		d_1	2	2	2	✓	
d_1	T	3	2			s_2	2	2	1			d_2	T	4	2		
s_1	T	4	3			s_3	T	3	1			s_1	2	5	3	✗	El.
												s_2	T	6	3	✗	El.

Step 3

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.	o_3	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
s_3	2	1	1	✓		s_1	2	1	1	✓		d_1	2	2	2	✓	
d_1	T	3	2			s_2	T	2	1			d_2	T	4	2		
s_1	T	4	3			s_3	T	3	2	✗	El.						

Step 4

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.	o_3	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
s_3	T	1	1	✓		s_1	2	1	1	✓		d_1	2	2	2	✓	
d_1	T	3	3	✗	El.	s_2	T	2	1			d_2	T	4	2		
s_1	T	4	4														

Step 5

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.	o_3	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
s_3	T	1	1	✓		s_1	2	1	1	✓	El.	d_1	T	2	2	✓	
s_1	T	2	4	✓		s_2	T	2	1			d_2	T	4	4	✗	El.

Step 6

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.	o_3	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
s_3	T	1	1	✓		s_2	T	1	1	✓		d_1	T	2	2	✓	
s_1	T	2	4	✓													

Table 3: TDBU algorithm on Example 3.

and will not enter Step 2. Note that the elimination would have taken place even without the rejection because o_1 is unacceptable to d_2. s_1 and s_2 both have a rejection number above the quota. o_1 does not reject s_1 as the pair does not satisfy Condition (iii) of Lemma 5. The first part of Condition (iii) is not satisfied as s_1 is the only single-unit agent contesting o_1 as his top choice and the second part is violated by the presence of s_3 who contests o_1 as a subsequent choice and has a rejection number smaller than the quota. The rationale for o_1 not rejecting s_1 is that, should o_1 become s_3's top choice, d_1 would be rejected and a unit would become available to s_1. The ability to allow agents to continue contesting an object despite having a rejection number above the quota is a key difference between Top-Down Bottom-Up and Deferred Acceptance. In the latter, o_1 rejects s_1, which ultimately creates a blocking pair. In the former, an object only rejects an agent when it can be established that the pair is not an element of any stable matching. This happens to s_2, who is rejected by o_1. The pair satisfies the first part of Condition (iii) due to the presence of s_1. The rationale for the rejection is that if o_1 becomes s_3's top choice, s_1 takes advantage of the unit that has become available, not s_2. The fifth and sixth columns respectively display a cross and "El.".

The second panel is devoted to o_2. Guarantee and rejection numbers are calculated in the same way. s_1 receives a guarantee as his guarantee number is equal to the quota. The double-unit agents are naturally rejected since o_2 only has one unit available. In the third panel, o_3 gives s_3 a guarantee. The pair is nevertheless eliminated as s_3 also receives a guarantee from o_1, which he prefers. s_1 and s_2 both have a rejection number above the quota. The presence of s_3 implies that they do not satisfy the second part of Condition (iii) of Lemma 5. s_1 does not satisfy the first part as the only single-unit agent with a higher priority, s_3, contests o_3 as a subsequent choice. As s_1 also contests o_3 as a subsequent choice, o_3 does not reject s_2 either.

Overall, seven agent-object pairs are eliminated in Step 1. s_1 stops contesting the null object as he has received a guarantee from o_3. s_2 stops contesting o_1 as the latter has rejected him. s_3 has received a guarantee from his second choice, o_1, and stops contesting his last two, o_3 and $\emptyset$, as a result. d_1 has been rejected by o_2 and stops contesting it and d_2 stops contesting both o_1 and o_2 for the same reason. We obtain X^2 by removing these seven pairs from $X^1 \equiv A \times O$, that is

$$X^2 = (A \times O) \setminus \{(s_1, \emptyset), (s_2, o_1), (s_3, o_3), (s_3, \emptyset), (d_1, o_2), (d_2, o_1), (d_2, o_2)\}.$$

Subsequent Steps

Neither o_1 nor o_2 gives any new guarantee or makes any new rejection in Step 2. The guarantee number of d_1 for o_3 has however fallen to 2 as s_3 no longer contests that object. o_3

gives d_1 a guarantee as a result, which means that the latter will not contest the null object in Step 3. The elimination of the pair (s_3, o_3) has important consequences for s_1 and s_2. The latter now satisfy the second part of Condition (iii) as they are the only single-unit agents to contest o_3 while the difference between their rejection number and the quota is 1. Both agents are rejected as a result. The rationale for these rejections is that o_3's two units are assigned to one of d_1 or d_2 in any stable matching. Three additional agent-object pairs are eliminated in Step 2 so that ten remain:

$$X^3 = \{(s_1, o_1), (s_1, o_2), (s_2, o_2), (s_2, \emptyset), (s_3, o_1), (s_3, o_2), (d_1, o_1), (d_1, o_3), (d_2, o_3), (d_2, \emptyset)\}.$$

This illustrates the importance of the TDBU algorithm's Bottom-Up parts. o_1 gave s_3 a guarantee in Step 1, which allows the latter to stop contesting o_3. This in turn allows o_3 to reject s_1 and s_2. If the TDBU algorithm only consisted of its Top-Down part, it would end in the next step as it would not have any more agent-object pair to eliminate.

In Step 3, o_2 has become s_2's top choice after he was rejected by o_3. As s_2's rejection number for o_2 is equal to the quota, o_2 rejects s_3 by the first part of Condition (iii). This is the only agent-object pair eliminated in this step. In turn, o_1 is s_3's top choice in Step 4 as the latter was rejected by o_2. d_1's rejection number for that object rises to $3 > 2 = q_{o_1}$ as a result, leading to a rejection. Then

$$X^5 = \{(s_1, o_1), (s_1, o_2), (s_2, o_2), (s_2, \emptyset), (s_3, o_1), (d_1, o_3), (d_2, o_3), (d_2, \emptyset)\}.$$

In Step 5, d_1 has joined the envy set of s_1 for o_1 – that is $E_{(s_1,o_1)}(X^5) = \{d_1\}$ – as $o_1 \succ_{d_1} o_3 = \overline{o}_{d_1}(X^5)$ and $d_1 \rhd_{o_1} s_1$. d_1's size therefore continues to count towards s_1's rejection number for o_1, which remains 4. In contrast, d_1's size no longer counts towards s_1's guarantee number for o_1, which falls to $q_{o_1} = 2$ as a result. Then o_1 gives s_1 a guarantee and consequently s_1 stops contesting o_2. On the other hand, d_2's rejection number for o_3 is now 4 as a result of o_3 becoming d_1's top choice so o_3 rejects d_2. In Step 6, s_2 has risen to the top of o_2's priority list (as s_1 no longer contests that object). Therefore, s_2 receives a guarantee abd stops contesting the null. Step 7 is not displayed as no additional agent-object pair is eliminated. The algorithm ends and yields

$$\varphi(A \times O) = X^8 = X^7 = \{(s_1, o_1), (s_2, o_2), (s_3, o_1), (d_1, o_3), (d_2, \emptyset)\}.$$

As each agent contests exactly one object, the above set of pairs is a matching, hence $\overline{\mu}(\varphi(A \times O)) = \varphi(A \times O)$. In addition, that matching is feasible. which by Proposition 4 means that it is the only stable matching in this market. This is consistent with our findings from Section

2.3.

4 Undominated Stable Matching

The Top-Down Bottom-Up algorithm allows finding a matching $\overline{\mu}(\varphi(A \times O))$, which is the optimal stable matching if it is feasible (Proposition 4). In this section, we study the case where $\overline{\mu}(\varphi(A \times O))$ is not feasible. The Undominated Stable Matching (USM) algorithm starts with the set of all agent-object pairs $X_1 \equiv A \times O$ and runs the TDBU algorithm to calculate $\varphi(A \times O)$. If that set's top matching is not feasible, it serves as a starting point towards a search for an undominated stable matching. The USM algorithm *protects* agent-object pairs one at a time and uses the TDBU algorithm in between in order to narrow down the search to stable matchings that contain all protected pairs. This continues until either a stable matching is found, or it can be established that there does not exist any stable matching containing all the protected pairs. In the former case, the matching found is an undominated stable matching as any other stable matching makes at least one of the agents involved in a protected pair worse-off. In the latter case, the algorithm backtracks and eliminates the last pair it has protected. If there is no such pair, the algorithm ends and reports that the set of stable matchings is empty.

4.1 Bottlenecks, Protections and Excess Supply

We begin by introducing concepts that are key to determining when our algorithm protects a new agent-object pair, backtracks or ends. We build upon these rules in Section 4.2 to formally introduce the Undominated Stable Matching (USM) algorithm. We call a complete set of agent-object pairs **irreducible** if the TDBU algorithm does not change it. We denote the set of irreducible sets of agent-object pairs by $\mathbb{I}$. Formally, $\mathbb{I} \equiv \{X \in \mathbb{X} \mid \varphi(X) = X\}$. For any $X \in \mathbb{X}$, $\varphi(X)$ is by construction irreducible if and only if it is complete. In each Round, the USM algorithm makes use of the TDBU algorithm to obtain either the empty set or an irreducible set of agent-object pairs. The empty set trivially does not include any stable matching, therefore the USM algorithm has to backtrack if this is the case. In this section, given an irreducible set of agent-object pairs, we determine which pairs are eligible to be protected. We also derive conditions under which it can be established that an irreducible set of agent-object pairs has a stable top matching or does not include any stable matching. These conditions will guide the USM algorithm, which will end in the former case and backtrack in the latter. If neither condition is satisfied, it protects an additional pair.

Irreducible sets of agent-object pairs have a large amount of structure, which will prove useful throughout our analysis. The next three results detail the properties of these sets.

Lemma 6. *For any $X \in \mathbb{I}$ and any $o \in O$, there exists at most one agent $a \in \overline{A}_o(X)$ such that $E_{(a,o)}(X) \neq \varnothing$. Furthermore, $a \in S$ and $E_{(a,o)}(X) \subseteq D$.*

Lemma 6 derives almost directly from Condition (i) of Lemma 5. If an agent has a nonempty envy set for a given object at X, then the object rejects him unless he is a single-unit agent, the envy set contains exclusively double-unit agents and he has a higher priority than all other single-unit agents with a nonempty envy set for the same object. These conditions hold whenever the TDBU algorithm ends, hence they hold at any irreducible set of agent-object pairs. An immediate consequence of Lemma 6 is that at the top matching of an irreducible set of pairs, an agent may only envy another agent of smaller size.

Corollary 1 (to Lemma 6). *For any $X \in \mathbb{I}$, $\overline{\mu}(X)$ is size-consistent.*

Our last result on irreducible sets of pairs relates to feasibility. It shows that the top matching of an irreducible set of pairs violates feasibility by at most one unit per object and only in a specific case.

Lemma 7. *If $\sum_{a \in \overline{A}_o(X)} w_a > q_o$ for some $o \in O$, then $\sum_{a \in \overline{A}_o(X)} w_a = q_o + 1$ and there exist $d \in \overline{D}_o(X)$, $s \in \overline{S}_o(X)$ and $s' \in S_o(X) \setminus \overline{S}_o(X)$ such that $\mathcal{R}_{(d,o)}(X) = q_o$ and $s' \rhd_o d \rhd_o s$.*

Lemma 7 constitutes a rather intuitive result. In every step of the TDBU algorithm and for every object o, at most one single-unit agent with a rejection number larger than or equal to the quota continues to contest o as his top choice (first part of Condition (iii)). This is only possible when a single-unit agent with a rejection number strictly smaller than the quota contests o as a subsequent choice. These conditions are then met when the TDBU algorithm ends. Lemma 7 allows introducing an important concept.

Definition 2. *Given $X \in \mathbb{I}$, a* ***bottleneck*** *is a pair of agent-object pairs $((d,o),(s,o))$ with $o \in O$, $d \in \overline{D}_o(X)$ and $s \in \overline{S}_o(X)$ such that $R_{(d,o)}(X) = q_o$ and $d \rhd_o s$.*

We denote the set of pairs that involve a double-unit agent and appear in a bottleneck of X by

$$B(X) \equiv \{(d,o) \in D \times O \mid d \in \overline{D}_o(X), \mathcal{R}_{(d,o)}(X) = q_o \text{ and } \sum_{a \in \overline{A}_o(X)} w_a = q_o + 1\}.$$

The next result follows directly from Lemma 7 and Definition 2.

Corollary 2 (to Lemma 7). *For any $X \in \mathbb{I}$ and any $o \in O$, there exist $d \in D$ and $s \in S$ such that $((d,o),(s,o))$ constitutes a bottleneck of X if and only if $\sum_{a \in \overline{A}_o(X)} w_a > q_o$.*

Bottlenecks are what prevent the TDBU algorithm from moving forward because both (d,o) and (s,o) may be an element of a stable matching, but keeping both of them violates the quota of o. The number of bottlenecks in X, $|B(X)|$, is then equal to the number of objects whose capacity is violated and to the number of units that would need to be created in order for $\overline{\mu}(X)$ to be feasible. An immediate consequence is that $\overline{\mu}(X)$ is feasible if and only if $B(X) = \varnothing$.[15]

While $\overline{\mu}(\varphi(A \times O))$ is 1-bounded, size-consistent and non-wasteful (Proposition 4), this may not be true for all irreducible sets of agent-object pairs, therefore the top matchings of such set may be feasible but not stable. A similar concept to bottlenecks is useful to determine cases where an irreducible set of pairs does not include any stable matching.

Definition 3. *Given $X \in \mathbb{I}$, a **phantom bottleneck** is a pair $(d,o) \in \overline{D}_o(X) \times O$ such that $\mathcal{R}_{(d,o)}(X) = q_o$, for any $s \in S$ such that $d \rhd_o s$, $\overline{o}_s(X) \succ_s o$ and there exists $s' \in S_o(X) \setminus \overline{S}_o(X)$ with $s' \rhd_o d$.*

We denote the set of pairs between a double-unit agent and his top choice involved in either a bottleneck or a phantom bottleneck of $X \in \mathbb{I}$ by

$$B^*(X) \equiv \{(d,o) \in D \times O \mid d \in \overline{D}_o(X), R_{(d,o)}(X) = q_o \text{ and } \exists s \in \hat{S}_{(d,o)}(X) \setminus \overline{S}_o(X)\}.$$

A phantom bottleneck resembles a bottleneck in many ways: the last two units of the object are assigned to a double-unit agent and there is a single-unit agent with a higher priority who contests o as a subsequent choice. Should there be a single-unit agent s with a lower priority than d contesting o as his top choice, $((d,o),(s,o))$ would constitute a bottleneck. There just happens not to be any such agent. One can therefore think of a phantom bottleneck as one that involves a "phantom" single-unit agent. Phantom bottlenecks do not cause any problem when it comes to feasibility since the sizes of agents whose top choice is o add up to exactly q_o. They however matter when it comes to determining whether or not X includes any stable matching.

Consider again $X \in \mathbb{I}$ and suppose that its top matching $\overline{\mu}(X)$ is not feasible. Then $B(X)$ is nonempty by Lemma 7 and so is $B^*(X)$ since $B(X) \subseteq B^*(X)$ by definition. Let $(d,o) \in B^*(X)$ and suppose we want to **protect** (d,o), that is we want to restrict our attention to stable matchings that contain (d,o). Letting $\mu \in \mathbb{S}$ be any such matching (i.e.

[15] Biró and McDermid (2014) show that a polynomial-time solvable algorithm exists in this model that finds a matching that is stable if quotas are allowed to vary by at most one unit. As $\overline{\mu}(\varphi(A \times O))$ is size-consistent, 1-bounded and non-wasteful (Proposition 4), Corollary 1 implies that the TDBU algorithm constitutes such approximation: $\overline{\mu}(\varphi(A \times O))$ is stable in the market where the objects involved in a bottleneck each have one extra unit of capacity. The algorithm proposed by Dean, Goemans, and Immorlica (2006) possesses the same properties, however it does not produce the same matching as the extra units benefit double-unit agents.

$(d,o) \in \mu$), we can identify agent-object pairs that are not an element of μ. By definition, d contests o as his top choice and his rejection number is q_o. As X is irreducible, $E_{(d,o)}(X)$ is empty. Therefore

$$\mathcal{R}_{(d,o)}(X) = w_d + \sum_{a \in \hat{A}_{(d,o)}(X) \cap \overline{A}_o(X)} w_a = q_o.$$

By size-consistency, all agents who have a higher priority than d and contest o as their top choice are matched to o at μ. All pairs involving these agents and their less preferred objects can be eliminated as a result. Agents who contest o as a subsequent choice and have a higher priority than d cannot be matched to o at μ as this would violate feasibility. They cannot either be matched at μ to an object they rank below o as they would envy d, violating size-consistency. All pairs involving these agents and objects they like weakly less than o can be eliminated as a result. We formalize this below.

Definition 4. *The **protection set** of $(d,o) \in B^*(X)$ at $X \in \mathbb{I}$ is*

$$P_{(d,o)}(X) \equiv \{(a,o') \in X \mid a \trianglerighteq_o d, o \succeq_a o' \text{ and } \overline{o}_a(X) \succ_a o'\}$$

Lemma 8. *For any $X \in \mathbb{I}$, any pair $(d,o) \in B^*(X)$ and any stable matching $\mu \in \mathbb{S}$, $(d,o) \in \mu \subseteq X$ implies $\mu \subseteq X \setminus P_{(d,o)}(X)$.*

The protection set of (d,o) in X contains agent-object pairs that are not an element of any stable matching included in X that contains (d,o). These pairs can be eliminated to obtain $X \setminus P_{(d,o)}(X)$, which includes all stable matchings that are a subset of X and contain (d,o). That set is not necessarily irreducible and can enter the TDBU algorithm in order to further reduce the set of agent-object pairs to consider. By Proposition 3, $\varphi(X \setminus P_{(d,o)}(X))$ includes all stable matchings that contain (d,o) and are a subset of X.

A particularly interesting case occurs if $(d,o) \in B(X)$, that is if there exists a bottleneck $((d,o),(s,o))$ at X. By definition, (s,o) is not an element of (d,o)'s protection set at X, however, when $X \setminus P_{(d,o)}(X)$ enters the TDBU algorithm, that pair is eliminated in the first step. The reason is that all agents with a higher priority than d who contest o at $X \setminus P_{(d,o)}(X)$ contest it as their top choice. As $\mathcal{R}_{(d,o)}(X \setminus P_{(d,o)}(X)) = q_o$, $\mathcal{R}_{(s,o)}(X \setminus P_{(d,o)}(X)) > q_o$ and there does not exist any single-unit agent who contests o as a subsequent choice and has a rejection number lower than the quota. (s,o) satisfies Condition (iii) of Lemma 5, as a result o rejects s at $X \setminus P_{(d,o)}(X)$.

If $(d,o) \in B^*(X) \setminus B(X)$, that is if (d,o) constitutes a phantom bottleneck of X, the absence of any agent contesting o as a subsequent choice at $\varphi(X \setminus P_{(d,o)}(X))$ means that (d,o) no longer constitutes a phantom bottleneck of that new set of agent-object pairs. Proposition 5 summarizes the properties of $\varphi(X \setminus P_{(d,o)}(X))$.

Proposition 5. *For any $X \in \mathbb{I}$, any pair $(d,o) \in B^*(X)$ and any stable matching $\mu \in \mathbb{S}$, $(d,o) \in \mu \subseteq X$ implies $\mu \subseteq \varphi(X \setminus P_{(d,o)}(X))$. Further, there does not exist any $d' \in D$ such that $(d,o) \in B^*(\varphi(X \setminus P_{(d,o)}(X)))$.*

The first part of the statement derives directly from Lemma 8 and Proposition 3: $X \setminus P_{(d,o)}(X)$ includes all stable matchings included in X that contain (d,o) and $\varphi(X \setminus P_{(d,o)}(X))$ includes all stable matchings included in $X \setminus P_{(d,o)}(X)$. The second part of the statement comes from the fact that all agents who contest o at $\varphi(X \setminus P_{(d,o)}(X))$ do so as their top choice, thus making it impossible for o to be involved in a bottleneck or phantom bottleneck by Definitions 2 and 3.

Proposition 5 shows that it is possible to narrow down the search for a stable matching by focusing on those stable matchings that contain specific agent-object pairs. The USM algorithm does exactly this, protecting one pair at a time until either a stable matching is found or it can be established that the set of pairs found does not include any stable matching. Eliminating pairs in a protection set means that some agents stop contesting objects without having a guarantee for an object they prefer. As a result, an agent may be rejected by all the objects he contests throughout the TDBU algorithm. Recall that when this occurs, the TDBU algorithm ends and returns the empty set. In that case, it can trivially be concluded that the set of pairs encountered does not include any stable matching. Otherwise, the USM algorithm considers an irreducible set of pairs (recall that by definition the empty set is not irreducible as it is not complete). We now derive conditions under which it can be established that this set of pairs has a stable top matching or does not include any stable matching.

Given an irreducible set of agent-object pairs $X \in \mathbb{I}$ and a pair $(a,o) \in X$, we define the **size-adjusted magnitude** of a's claim to o at $\overline{\mu}(X)$, denoted $\mathcal{M}_{(a,o)}(X)$, to be the largest claim that a has to o, adjusted to his size. That is, $\mathcal{M}_{(a,o)}(X) = 0$ if a does not have a w_a-unit claim to o at $\overline{\mu}(X)$ and $\mathcal{M}_{(a,o)}(X) = n > 0$ if a has a $n + (w_a - 1)$-unit claim to o but does not have a $n + w_a$-unit claim to o at $\overline{\mu}(X)$. Formally,

$$\mathcal{M}_{(a,o)}(X) \equiv \begin{cases} \max\{0, q_o - (w_a - 1) - \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'}\} & \text{if } o \succ_a \overline{o}_a(X) \\ 0 & \text{otherwise.} \end{cases}$$

The relevance of $\mathcal{M}_{(a,o)}(X)$ is that (a,o) constitutes a blocking pair of $\overline{\mu}(X)$ if and only if $\mathcal{M}_{(a,o)}(X) > 0$. In order for that blocking pair to disappear, at least $\mathcal{M}_{(a,o)}(X)$ units of o must be assigned to agents with a higher priority than a. We define the **largest size-adjusted claim** to o, denoted, $\mathcal{L}_o(X)$, to be the one with the largest size-adjusted magnitude:

$$\mathcal{L}_o(X) \equiv \max_{a \in A} \mathcal{M}_{(a,o)}(X).$$

We denote the sum of all objects' largest size-adjusted claims by $\mathcal{L}(X) \equiv \sum_{o\in O} \mathcal{L}_o(X)$. The properties of largest size-adjusted claims are detailed in the following Lemma.

Lemma 9. *For any $X \in \mathbb{I}$, $\overline{\mu}(X)$ does not have any blocking pair if and only if $\mathcal{L}(X) = 0$. For every $o \in O$, $\mathcal{L}_o(X) > 0$ implies $q_o - \sum_{a\in A_o(\overline{\mu}(X))} w_a \geq \mathcal{L}_o(X)$.*

Lemma 9 outlines the importance of $\mathcal{L}_o(X)$ when it comes to finding stable matchings included in X. The first part of the statement comes from the fact that $\mathcal{L}_o(X) > 0$ implies that some agent a has a w_a-unit claim to o at $\overline{\mu}(X)$ but $\mathcal{L}_o(X) = 0$ implies that this is not the case. Therefore, calculating $\mathcal{L}_o(X)$ is enough to determine whether or not $\overline{\mu}(X)$ contains any blocking pair. Additionally, this result implies that for any $\mu \in \mathbb{S} \cap 2^X$, that is for any stable matching included in X, $\sum_{o\in O} \mathcal{L}_o(\mu) = 0$. Therefore, at least $\mathcal{L}_o(X)$ units of o need to be assigned to agents with a higher priority than the ones to whom they are assigned at $\overline{\mu}(X)$ (if any). The second part of the statement derives from Lemma 6 and the fact that at most one unit of o is assigned to an agent who is envied at $\overline{\mu}(X)$. It implies that all claims at $\overline{\mu}(X)$ involves unassigned units. Thus, a matching $\mu \in \mathbb{S} \cap 2^X$ is such that for any object o with $\mathcal{L}_o(X) > 0$, the number of units assigned at μ is at least $\mathcal{L}_o(X)$ over the number of units assigned at $\overline{\mu}(X)$. Mathematically, $\sum_{a\in A(\mu)} w_a - \sum_{a\in A(\overline{\mu}(X))} w_a \geq \mathcal{L}_o(X)$. If that condition is not met, agents who have a claim to unassigned units at $\overline{\mu}(X)$ also do at μ and the latter is not stable.

For any $\mu \in \mathbb{S} \cap 2^X$, the total size of agents matched to objects with a positive largest size-adjusted claim at μ is larger than at $\overline{\mu}(X)$. As the total size of all agents is fixed at $\sum_{a\in A} w_a = 2|D| + |S|$, the total size of agents matched to objects that have a largest size-adjusted claim of zero (o such that $\mathcal{L}_o(X) = 0$) is smaller. Mathematically,

$$\sum_{o\in\{o'|\mathcal{L}_{o'}(X)>0\}} \left(\sum_{a\in A_o(\mu)} w_a\right) - \left(\sum_{a\in \overline{A}_o(X)} w_a\right) = \sum_{o\in\{o'|\mathcal{L}_{o'}(X)=0\}} \left(\sum_{a\in \overline{A}_o(X)} w_a\right) - \left(\sum_{a\in A_o(\mu)} w_a\right) > 0.$$

Consider an object $o \in O$ such that $\mathcal{L}_o(X) = 0$. Intuitively, it seems that reducing the total size of agents who contest o as their top choice would increase $\mathcal{L}_o(X)$, in which case it is impossible to reach $\mu \subseteq X$ such that $\mathcal{L}(\mu) = 0$ if $\mathcal{L}(X) > 0$. However, if the object is involved in a bottleneck or phantom bottleneck, the total size of agents matched to that object may decrease by one unit without affecting its largest size-adjusted claim. To see this, suppose that $o \in O$ is involved in a bottleneck of X, that is there exists $d \in D$ such that $(d, o) \in B(X)$. By definition, the total size of agents contesting o at X as their top choice is $\sum_{a\in\overline{A}_o(X)} w_a = q_o + 1$. By Lemma 7, the agent with the lowest priority is a single-unit agent and by Lemma 6 no agent other than him is envied at $\overline{\mu}(X)$. If that agent stops contesting o, the total size of agents contesting o as their top choice falls to q_o but no agent has a claim to o. Consider next an object o that is involved in a phantom bottleneck of X. That is, there exists $d \in D$ such that $(d, o) \in B^*(X) \setminus B(X)$. Then $\sum_{a \in \overline{A}_o(X)} w_a = q_o$ and $\mathcal{R}_{(d,o)}(X) = q_o$. Additionally, there exists $s \in S_o(X) \setminus \overline{S}_o(X)$ but there does not exist $s' \in S$ such that $o \succ_{s'} \overline{o}_{s'}(X)$. Consider a matching $\mu \subseteq X$ such that s is matched to o but d is not and all other agents matched to o at μ are those who contest it at X as their top choice. Formally, $A_o(\mu) = (\overline{A}_o(X) \setminus \{d\}) \cup \{s\}$. Then the total size of agents matched to o is $q_o - 1$, however no single-unit agent has a claim to the unassigned unit of o, therefore $\mathcal{L}_o(\mu) = 0$.

The above considerations imply that X may include a stable matching even when $\mathcal{L}(X) > 0$, however there needs to be enough objects involved in a bottleneck or a phantom bottleneck in order to increase the total size of agents matched to objects with a positive largest size-adjusted claim. Lemma 10 formalizes these conditions.

Lemma 10. *Let $X \in \mathbb{I}$, $\mu \in \mathbb{S} \cap 2^X$, and $o \in O$. If there exists $d \in D$ such that $(d, o) \in B^*(X)$, then $\mathcal{L}_o(X) = 0$ and*

$$\sum_{a \in A_o(\mu)} w_a - \sum_{a \in \overline{A}_o(X)} w_a \geq \mathcal{L}_o(X) - 1 = -1.$$

Otherwise,

$$\sum_{a \in A_o(\mu)} w_a - \sum_{a \in \overline{A}_o(X)} w_a \geq \mathcal{L}_o(X).$$

Corollary 3 (to Lemma 10)**.** *Given $X \in \mathbb{I}$, if there exists $o \in O$ such that $|A_o(X)| - |\overline{A}_o(X)| < \mathcal{L}_o(X)$, then $\mathbb{S} \cap 2^X = \varnothing$.*

Lemma 10 shows that the total size of agents matched to an object may decrease by at most one unit if the object is involved in a bottleneck or phantom bottleneck and may not decrease otherwise. In fact, if an object o has an positive largest size-adjusted claim, the total size of agents matched to it must increase by at least $\mathcal{L}_o(X)$. This requires the presence of enough agents who contest o as a subsequent choice, hence Corollary 3 is an immediate consequence of Lemma 10. We finally make use of these result to determine when X has a stable top matching or does not include any stable matching.

Definition 5. *The **excess supply** of $o \in O$ at $X \in \mathbb{I}$ is*

$$\mathcal{E}_o(X) \equiv \begin{cases} -1 & \text{if there exists } d \in D \text{ such that } (d, o) \in B^*(X) \\ \mathcal{L}_o(X) & \text{if } (d, o) \notin B^*(X) \text{ for all } d \in D \text{ and } |A_o(X)| - |\overline{A}_o(X)| \geq \mathcal{L}_o(X) \\ \infty & \text{if } |A_o(X)| - |\overline{A}_o(X)| < \mathcal{L}_o(X). \end{cases}$$

We denote the sum of all objects' excess supply at X by $\mathcal{E}(X) \equiv \sum_{o \in O} \mathcal{E}_o(X)$. The term "excess supply" refers to units that are unassigned at X and cause $\overline{\mu}(X)$ to have blocking

Algorithm 3: Undominated Stable Matching (USM)

```
Initialization:
Given a market and an index of agent-object pairs $\theta$, let $X_1 \equiv A \times O$ and $Z_1 \equiv \varnothing$.
Round $k \geq 1$:
Use the TDBU algorithm to calculate $\varphi(X_k)$.
Case 1: $\mathcal{L}(\varphi(X_k)) = |B(\varphi(X_k))| = 0$.
    Let $\mu^*_\theta \equiv \overline{\mu}(\varphi(X_k))$.
Case 2: $\mathcal{L}(\varphi(X_k)) + |B(\varphi(X_k))| > 0$ and $\mathcal{E}(\varphi(X_k)) \leq 0$.
    Let $z_k$ be the element of $B^*(\varphi(X_k))$ with the smallest index $\theta$. Construct $X_{k+1}$ by protecting $z_k$ at $\varphi(X_k)$ and $Z_{k+1}$ by adding $z_k$ to $Z_k$ and continue to Round $k+1$.
Case 3: $\varphi(X_k) = \varnothing$ or $\mathcal{E}(\varphi(X_k)) > 0$.
    If $Z_k = \varnothing$, let $\mu^*_\theta \equiv \varnothing$. Otherwise, let $z_i$ be the last pair that was added to $Z_k$. Construct $X_{k+1}$ by eliminating $z_i$ from $X_i$, let $Z_{k+1} \equiv Z_i$ and continue to Round $k+1$.
```

pair. If an object has more such units than the total size of the agents who contest it as a subsequent choice, its excess supply is ∞ as it is involved in a blocking pair in all matchings included in X. If on the other hand the object is involved in a bottleneck or a phantom bottleneck, its excess supply is -1 as the total size of agents contesting it as its top choice can decrease by one unit without creating a blocking pair. We are now in a position to formally state the conditions under which it can be established that an irreducible set of agent-object pairs has a stable top matching or does not include any stable matching.

Proposition 6. $\overline{\mu}(X) \in \mathbb{S}$ *if and only if* $\mathcal{L}(X) = |B(X)| = 0$. *If* $\mathcal{E}(X) > 0$, *then* $\mathbb{S} \cap 2^X = \varnothing$.

4.2 Undominated Stable Matching Algorithm

The USM algorithm is is presented in Algorithm 3. The full set of agent-object pairs $A \times O$ enters the algorithm. The latter returns μ^*_θ, which is either the empty set or a matching. In Round 1, it uses the TDBU algorithm to calculate $\varphi(A \times O)$. By Proposition 4, $\overline{\mu}(\varphi(A \times O))$ is the optimal stable matching if and only if $B(X) = \varnothing$.[16] Otherwise, an element of

[16]This is consistent with Proposition 6 as $\overline{\mu}(\varphi(A \times O))$ is 1-bounded, size-consistent and non-wasteful and therefore does not have any blocking pair, meaning by Lemma 9 that $\mathcal{L}(X) = 0$.

$B^*(\varphi(A \times O))$ is protected. $B^*(X)$ may contain multiple elements, in which case the one with the lowest **index** is protected. This index is a function $\theta : D \times O \to \{1, 2, \ldots, |D \times O|\}$ such that $(d, o) \neq (d', o')$ implies $\theta((d, o)) \neq \theta((d', o'))$. In practice, θ could be determined through a lottery or reflect a general priority over double-unit agents. As we show in Theorem 1, the index does not impact the outcome if an optimal stable matching exists or if the set of stable matchings is empty, however it may determine which undominated stable matching is selected if multiple ones exist. We label the protected agent-object pair z_1. The set of pairs that enters Round 2 is calculated by eliminating the protection set of z_1 from $\varphi(A \times O)$: $X_2 \equiv \varphi(A \times O) \setminus P_{z_1}(\varphi(X_1))$. We also let $Z_2 \equiv \{z_1\}$ be the set of pairs that have been protected thus far. ($Z_1 = \varnothing$ since no pair is protected at the beginning of the algorithm.)

In general, in Round k, a set of pairs X_k enters, a subset Z_k of which has been protected. The TDBU algorithm is used to calculate $\varphi(X_k)$, a set that by construction is irreducible if and only if it is nonempty. Depending on the properties of $\varphi(X_k)$, the algorithm finds itself in one of three possible cases. Case 1 is the simplest one, it occurs when $\mathcal{L}(\varphi(X_k)) = |B(\varphi(X_k))| = 0$. By Proposition 6, $\overline{\mu}(\varphi(X_k))$ is stable. The algorithm ends as it has found a stable matching. In fact, it has found an undominated stable matching because all agent-object pairs that are not an element of $\varphi(X_k)$ are not an element of any stable matching that includes Z_k. Therefore, any other stable matching makes at least one of the agents involved in a protected pair worse-off. Case 2 occurs when $\mathcal{L}(\varphi(X_k)) + |B(\varphi(X_k))| > 0$ but $\mathcal{E}(\varphi(X_k)) \leq 0$. By Proposition 6, $\overline{\mu}(\varphi(X_k))$ is not stable, however the second part of the statement does not apply, therefore it may be possible to find a stable matching by eliminating more agent-object pairs. The algorithm protects the element of $B^*(\varphi(X))$ with the lowest index. Such pair, which we label z_k, exists since $|B(\varphi(X_k))| > 0$ directly implies $B^*(\varphi(X)) \neq \varnothing$ and the combination of $\mathcal{L}(\varphi(X_k)) > 0$ and $\mathcal{E}(\varphi(X_k)) \leq 0$ also implies $B^*(\varphi(X)) \neq \varnothing$. The algorithm generates Z_{k+1} by adding the newly protected pair z_k to Z_k and X_{k+1} by removing its protection set $P_{z_k}(\varphi(X_k))$ from $\varphi(X_k)$. By Lemma 8, X_{k+1} includes all stable matchings that themselves include Z_{k+1}. Case 3 arises when neither of the above two conditions is satisfied, which occurs when either $\varphi(X_k) = \varnothing$ or $\mathcal{E}(\varphi(X_k)) > 0$.[17] In that case, $\varphi(X_k)$ does not include any stable matching. This is trivial when $\varphi(X_k) = \varnothing$ and implied by Proposition 6 when $\mathcal{E}(\varphi(X_k)) > 0$. It implies that there does not exist any stable matching that includes Z_k. In turn, if $Z_k = \varnothing$, this means that there does not exist any stable matching in this market. The algorithm ends and produces the empty set to indicate that fact. If $Z_k \neq \varnothing$, the algorithm backtracks and eliminates the last protected pair, which we label z_i. That pair is not an element of any stable matching that includes $Z_i = Z_k \setminus \{z_i\}$. ($Z_i = Z_k \setminus \{z_i\}$

[17]Cases 1 and 2 implicitly imply that $\varphi(X_k) \neq \varnothing$ since $\mathcal{L}(\cdot)$, $B(\cdot)$ and $\mathcal{E}(\cdot)$ are only defined for irreducible sets of agent-object pairs, which are complete (hence nonempty) by definition.

since z_i was protected in Round i.) X_{k+1} and Z_{k+1} are obtained by removing z_i from X_k, respectively Z_k (then $Z_{k+1} = Z_i$) and the algorithm continues to Round $k+1$. X_{k+1} includes all stable matchings that include Z_i. The USM algorithm goes on until it either finds an undominated stable matching or establishes that the set of stable matchings is empty. This result is formalized below.

Theorem 1. *For any index θ, μ^*_θ is an undominated stable matching if the set of stable matchings is nonempty and $\mu^*_\theta = \varnothing$ otherwise.*

An immediate implication of Theorem 1 is that the index θ does not affect its outcome if the set of stable matchings is empty or contains an optimal stable matching. In the former case, the algorithm produces an empty set and in the latter case, it finds the optimal stable matching since all other stable matchings are dominated. The index determines what the algorithm tries first and may for that reason impact its running time. If multiple undominated stable matchings exist, the algorithm stops as soon as one is found, and which one that is may depend on the index.

4.3 d-Undominated Stable Matching

The USM algorithm is constructed in such a way that only pairs involving a double-unit agent are eligible for protection. In the case where multiple undominated stable matchings exist, this may have an incidence on the outcome. Proposition 7 shows that this construction confers to the USM algorithm an additional property. We argue (Example 4) that this property is desirable.

d-domination is a partial order on the set of matchings $\mathbb{M}$. We say that matching μ d-dominates matching μ' if $\overline{o}_d(\mu) \succeq_d \overline{o}_d(\mu')$ for all $d \in D$ and there exists $a \in A$ such that $\overline{o}_a(\mu) \succ_a \overline{o}_a(\mu')$. In words, μ d-dominates μ' if it makes all *double-unit* agents weakly better-off and at least one agent strictly better-off. A stable matching $\mu \in \mathbb{S}$ is a **d-undominated stable matching** if there does not exist any *stable* matching $\mu' \in \mathbb{S}$ such that μ' *d-dominates* μ.

Proposition 7. *For any index θ, μ^*_θ is a d-undominated stable matching if the set of stable matchings is nonempty and $\mu^*_\theta = \varnothing$ otherwise.*

d-domination constitutes a relaxation of domination as it only requires that double-unit agents be made weakly worse-off. All d-undominated stable matchings are undominated but the converse is not necessarily true, hence Proposition 7 strengthens Theorem 1. It derives directly from the fact that protected pairs involve double-unit agents, therefore any stable matching other than μ^*_θ (if the latter is a stable matching) makes at least one double-unit

Round 1

$X_1 = A \times O$, $Z_1 = \varnothing$

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
s_2	2	1	1	✓		d_1	2	2	2	✓	
d_1	T	3	2			s_2	T	3	1		
s_1	T	4	3			s_1	2	4	2		

$\varphi(X_1) = (A \times O) \setminus \{(s_2, \emptyset), (d_1, \emptyset)\}$

Round 2

$X_2 = \{(s_1, o_1), (s_1, o_2), (s_1, \emptyset), (s_2, o_2), (d_1, o_1)\}$
$Z_2 = \{(d_1, o_1)\}$

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
d_1	T	2	2	✓		s_2	T	1	1	✓	
s_1	T	3	3	✗	El.	s_1	2	2	2	✓	
o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
d_1	T	2	2	✓		s_2	T	1	1	✓	
						s_1	T	2	2	✓	

$\varphi(X_2) = \{(s_1, o_2), (s_2, o_2), (d_1, o_1)\}$

Table 4: USM algorithm on Example 4.

agent worse-off. This feature of the USM algorithm may appear unfair at first sight, however as we illustrate next it prevents awkward situations where a double-unit agent envies a single-unit agent and the only reason why the matching is stable is that another single-unit agent was made worse-off in the process.

Example 4 (Unequal USMs)**.** There are two single-unit agents s_1 and s_2, one double-unit agent d_1 and two non-null objects o_1 and o_2. The preferences, priorities and quotas are detailed below:

$$\begin{array}{llll} \succ_{s_1}: o_1, o_2, \emptyset & \succ_{d_1}: o_1, o_2, \emptyset & \rhd_{o_1}: s_2, d_1, s_1 & q_{o_1} = 2 \\ \succ_{s_2}: o_2, o_1, \emptyset & & \rhd_{o_2}: d_1, s_2, s_1 & q_{o_2} = 2 \end{array}$$

The market presented in Example 4 has two stable matchings: $\mu \equiv \{(s_1, o_2), (s_2, o_2), (d_1, o_1)\}$ and $\mu' \equiv \{(s_1, o_1), (s_2, o_1), (d_1, o_2)\}$. Neither matching dominates the other as s_2 and d_1 are better-off at μ while s_1 is better-off at μ'. Table 4 displays the work of the USM algorithm in this market. In the first round, the TDBU algorithm only eliminates $(s_2, \emptyset)$ and $(d_1, \emptyset)$ since s_2 and d_1 receive a guarantee from their respective second choices. $B(\varphi(X_1)) = B^*(\varphi(X_1)) = \{(d_1, o_1)\}$ therefore (d_1, o_1) is protected no matter θ. The protection set of (d_1, o_1) contains (d_1, o_2) and (s_2, o_1) (Definition 4). When $\varphi(X_1)$ enters the TDBU algorithm,

o_1 rejects s_1 since the pair satisfies Condition (iii) of Lemma 5 as a result of s_2 no longer contesting o_1. The TDBU algorithm yields $\varphi(X_2)\{(s_1, o_2), (s_2, o_2), (d_1, o_1)\}$ as all agents obtain a guarantee for their top choice. The USM algorithm stops at the end of Round 2 and yields $\mu^*_\theta = \mu$.

The USM algorithm produces μ for any θ. The reason why it never produces μ' is that μ d-dominates it as $\overline{o}_{d_1}(\mu') \succ_{d_1} \overline{o}_{d_1}(\mu')$. We argue that missing μ' is a virtue of the USM algorithm because μ constitutes a more desirable solution. s_1 has the lowest priority for both objects, therefore it appears fair that s_2 and d_1 first be matched to the object they want and that s_1 then obtain whatever remains. Following that logic, s_2 and d_1 should be respectively assigned one unit of o_2 and two units of o_1 and s_1 should receive the remaining unit of o_2, leading to matching μ. μ' is constructed by inefficiently matching s_2 to o_1 and d_2 to o_2. The fact that s_2 and d_1 have different sizes benefits s_1 because one unit of o_1 remains available.

4.4 Examples

We use our examples from Section 2.3 in order to illustrate the USM algorithm. Example 3 has an optimal stable matching and we showed in Section 3.3 that the TDBU algorithm finds it. An immediate consequence is that the USM algorithm finds it in its first round. Example 1 does not have any stable matching and we show in Section 4.4.1 that the USM algorithm returns the empty set. In contrast, Example 2 has two undominated stable matchings and we show in Section 4.4.2 that the USM algorithm returns either one of them, depending on its index θ. In Section 4.4.3, we illustrate the depth-first search conducted by the USM algorithm with two examples.

4.4.1 Example 1

Table 5 displays the computations of the USM algorithm on Example 1. As d_1 is the only double-unit agent, the index does not impact the outcome, therefore we consider any arbitrary index θ. The USM algorithm lasts three rounds, which are displayed in Table 5. Round 1 begins by running the TDBU algorithm on $X_1 \equiv A \times O$. s_1 and s_2 each receive a guarantee for their second choice so that $(s_1, \emptyset)$ and $(s_2, \emptyset)$ are eliminated. (d_1, o_2) is also eliminated since d_1 finds o_2 unacceptable and o_2 only has one unit available. o_1 does not reject s_1 despite the latter having a rejection number above the quota, because a unit of o_1 would free up should that object become s_2's top choice. The TDBU algorithm stops after two rounds and yields $\varphi(X_1) = \{(s_1, o_1),\ (s_1, o_2),\ (s_2, o_1),\ (s_2, o_2),\ (d_1, o_1),\ (d_1, \emptyset)\}$. $\varphi(X_1)$ contains one bottleneck: $((d_1, o_1), (s_1, o_1))$ so X_2 is constructed by protecting (d_1, o_1). The protection set contains all pairs involving d_1 and a less preferred object to o_1 as well as all pairs involving

Round 1

$X_1 = A \times O$, $Z_1 = \varnothing$

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.
s_2	2	1	1	✓		s_1	2	1	1	✓	
d_1	T	3	2			s_2	T	2	1		
s_1	T	4	3			d_1	U	4	3	✗	El.
o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.
s_2	2	1	1	✓		s_1	2	1	1	✓	
d_1	T	3	2			s_2	T	2	1		
s_1	T	4	3								

$\varphi(X_1) = (A \times O) \setminus \{(s_1, \emptyset), (s_2, \emptyset), (d_1, o_2)\}$

Round 2

$X_2 = \{(s_1, o_1), (s_1, o_2), (s_2, o_2), (d_1, o_1)\}$
$Z_2 = \{(d_1, o_1)\}$

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.
d_1	T	2	2	✓		s_1	2	1	1	✓	
s_1	T	3	3	✗	El.	s_2	T	2	1		
o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.
d_1	T	2	2	✓		s_1	T	1	1	✓	
						s_2	T	2	2	✗	El.

$\varphi(X_2) = \varnothing$

Round 3

$X_3 = \{(s_1, o_1), (s_1, o_2), (s_2, o_1), (s_2, o_2), (d_1, \emptyset)\}$
$Z_3 = \varnothing$

o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.
s_2	2	1	1	✓		s_1	2	1	1	✓	El.
s_1	T	2	3	✓		s_2	T	2	1		
o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.
s_2	2	1	1	✓	El.	s_2	T	1	1	✓	
s_1	T	2	3	✓							
o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(1)	El.
s_1	T	1	3	✓		s_2	T	1	1	✓	

$\varphi(X_3) = \{(s_1, o_1), (s_2, o_2), (d_1, \emptyset)\}$

Table 5: USM algorithm on Example 1.

s_2 and an object he ranks weakly below o_1, therefore: $P_{(d_1,o_1)}(\varphi(X_1)) = \{(d_1,\emptyset),(s_2,o_1)\}$ and $X_2 = \{(s_1,o_1),(s_1,o_2),(s_2,o_2),(d_1,o_1)\}$. In Round 2, o_1 rejects s_1 in the first step of the TDBU algorithm. This results from s_2 no longer contesting o_1, which implies that d_1 is now assigned both units of that object in any stable matching. s_1 contests o_2 as his top choice in the second step of the TDBU algorithm and o_2 rejects s_2 as a consequence. As s_2 no longer contests any object, the TDBU algorithm ends and yields the empty set. We conclude that there does not exist any stable matching that contains (d_1,o_1). X_3 is constructed by removing (d_1,o_1) from $\varphi(X_1) = \varphi(A \times O)$, meaning that d_1 only contests the null object. When that set enters the TDBU algorithm, s_1 and s_2 receive a guarantee from their respective top choices, resulting in each agent contesting exactly one object: $\varphi(X_3) = \{(s_1,o_1),(s_2,o_2),(d_1,\varnothing)\}$. As $d_1 \rhd_{o_1} s_1$, d_1 has a 2-unit claim to o_1 so $\mathcal{L}_{o_1}(\varphi(X_3)) = 1$. As none of the agents contests o_1 as a subsequent choice, $\mathcal{E}_{o_1}(\varphi(X_3)) = \infty$ and X_3 does not include any stable matching.[18] The USM algorithm cannot backtrack since $Z_3 = \varnothing$, therefore it stops and produces $\mu^*_\theta = \varnothing$. This outcome was to be expected as we had established in Section 2.3 that this market does not have any stable matching.

4.4.2 Example 2

As was shown in Section 2.3, Example 2 has two undominated stable matchings

$$\mu \equiv \{(s_1,o_1),(s_2,o_1),(d_1,\emptyset),(d_2,o_2)\} \quad \text{and} \quad \mu' \equiv \{(s_1,o_2),(s_2,o_2),(d_1,o_1),(d_2,\emptyset)\}.$$

We consider an index θ is such that $\theta((d_1,o_1)) < \theta((d_2,o_2))$ and show that the USM algorithm produces μ'. Table 6 details the computations. Irrespective of θ, Round 1 starts by running the TDBU algorithm on $A \times O$. d_1 and d_2 respectively find o_2 and o_1 unacceptable, therefore (d_1,o_2) and (d_2,o_1) are eliminated (o_2 and o_1 also respectively reject d_1 and d_2). In contrast, o_2 and o_1 give a guarantee to s_1 and s_2, respectively, as a result $(s_1,\emptyset)$ and $(s_2,\emptyset)$ are eliminated. The TDBU algorithm ends after its second step as it does not eliminate any additional pair. $\varphi(X_1) = \varphi(A \times O)$ contains two bottlenecks: $B(\varphi(X_1)) = B^*(\varphi(X_1)) = \{(d_1,o_1),(d_2,o_2)\}$. As $\theta((d_1,o_1)) < \theta((d_2,o_2))$, (d_1,o_1) is protected. The protection set is $P_{(d_1,o_1)}(\varphi(X_1)) = \{(s_2,o_1),(d_1,\emptyset)\}$ and the search is now restricted to stable matchings that contain (d_1,o_1).

Round 2 starts with

$$X_2 = \varphi(X_1) \setminus P_{(d_1,o_1)}(\varphi(X_1)) = \{(s_1,o_2),(s_2,o_2),(d_1,o_1),(d_2,o_2),(d_2,\emptyset)\}$$

[18]As $\varphi(X_3)$ is itself a matching, this conclusion can be inferred directly from the fact that it is not stable as (d_1,o_1) is a blocking pair.

Round 1

$X_1 = A \times O$, $Z_1 = \varnothing$											
o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
s_2	2	1	1	✓		s_1	2	1	1	✓	
d_1	T	3	2			d_2	T	3	2		
s_1	T	4	3			s_2	T	4	3		
d_2	U	6	5	✗	El.	d_1	U	6	5	✗	El.
o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
s_2	2	1	1	✓		s_1	2	1	1	✓	
d_1	T	3	2			d_2	T	3	2		
s_1	T	4	3			s_2	T	4	3		
$\varphi(X_1) = \{(s_1, o_1), (s_1, o_2), (s_2, o_1), (s_2, o_2), (d_1, o_1), (d_1, \emptyset), (d_2, o_2), (d_2, \emptyset)\}$											

Round 2

$X_2 = \{(s_1, o_1), (s_1, o_2), (s_2, o_2), (d_1, o_1), (d_2, o_2), (d_2, \emptyset)\}$ $Z_2 = \{(d_1, o_1)\}$											
o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
d_1	T	2	2	✓		s_1	2	1	1	✓	
s_1	T	3	3	✗	El.	d_2	T	3	2		
						s_2	T	4	3		
o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
d_1	T	2	2	✓		s_1	T	1	1	✓	
						d_2	T	3	3	✗	El.
						s_2	T	4	4		
o_1	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.	o_2	Pref.	$\mathcal{G}$	$\mathcal{R}$	(2)	El.
d_1	T	2	2	✓		s_1	T	1	1	✓	
						s_2	T	2	4	✓	
$\varphi(X_2) = \{(s_1, o_2), (s_2, o_2), (d_1, o_1), (d_2, \emptyset)\}$											

Table 6: USM algorithm on Example 2 ($\theta(d_1, o_1) < \theta(d_2, o_2)$).

and $Z_2 = \{(d_1, o_1)\}$. o_1 rejects s_1 in the first step of the TDBU algorithm as s_2 no longer contests that object. In the second step, o_2 has become s_1's top choice, resulting in d_2's rejection number rising to $3 > 2 = q_{o_2}$. o_2 rejects d_1. In the third step, s_2's guarantee number for o_2 has fallen to $2 = q_{o_2}$ following d_2's rejection. $(s_2, \emptyset)$ is eliminated and the TDBU algorithm yields $\varphi(X_2) = \{(s_1, o_2), (s_2, o_2), (d_1, o_1), (d_2, \emptyset)\} = \mu'$. Since $\mathcal{L}(\varphi(X_2)) = |B(\varphi(X_2))| = 0$, the USM algorithm ends and produces $\mu^*_\theta \equiv \overline{\mu}(\varphi(X_2)) = \varphi(X_2) = \mu'$. Through an analogous reasoning, it is possible to verify that the USM algorithm produces μ if $\theta(d_1, o_1) > \theta(d_2, o_2)$.

4.4.3 Depth-First Search

Our last example does not arise from a specific market but presents a hypothetical depth-first search that further illustrates the USM algorithm. Figure 1 displays a hypothetical depth-first search where the USM algorithm finds an undominated stable matching after seventeen rounds. The algorithm starts with some market and some index θ over pairs involving a double-unit agent.

In Round 1, the TDBU algorithm runs over $X_1 = A \times O$ and produces $\varphi(X_1)$. It is such that $B(\varphi(X_1) \neq \varnothing$, therefore the agent-object pair with the lowest index θ in $B^*(\varphi(X_1))$, say z_1, is protected. X_2 is calculated by removing the protection set of z_1 from $\varphi(X_1)$ and Z_2 is obtained by adding z_1 to Z_1. As $Z_1 = \varnothing$, $Z_2 = \{z_1\}$. In Round 2, the TDBU algorithm generates $\varphi(X_2) \in \mathbb{I}$. It is such that $\mathcal{E}(\varphi(X_2)) \leq 0$ and $\mathcal{L}(X) + |B(X)| > 0$ therefore the algorithm constructs X_3 and Z_3 by protecting another pair z_2. Round 3 works in an analogous way, this time z_3 is protected. Then $Z_4 = \{z_1, z_2, z_3\}$.

In Round 4, the TDBU algorithm produces $\varphi(X_4)$ and this time either $\varphi(X_4) = \varnothing$ or $\mathcal{E}(\varphi(X_4)) > 0$, meaning that X_4 does not include any stable matching ($\mathbb{S} \cap 2^{X_4} = \varnothing$). We in turn infer that $Z_4 = \{z_1, z_2.z_3\}$ is not a subset of any stable matching. The algorithm proceeds to reject the last protected agent-object pair, in that case z_3. X_5 and Z_5 are constructed by removing z_3 from $\varphi(X_3)$ and Z_3, respectively.

In Rounds 5 and 6, the pairs z_5 and z_6 are protected so that $Z_7 = \{z_1, z_2, z_5, z_6\}$. It is found in Round 7 that X_7 does not include any stable matching. The last protected pair, z_6, is rejected so that $X_8 = \varphi(X_7) \backslash \{z_6\}$ and $Z_8 = \{z_1, z_2, z_5\}$. Again, running the TDBU algorithm allows inferring that X_8 does not include any stable matching. Then $Z_8 = Z_6 = \{z_1, z_2, z_5\}$ is not a subset of any stable matching, hence X_6 does not include any stable matching. The algorithm goes back to the last set of agent-object pairs that could potentially include a stable matching, $\varphi(X_5)$, and eliminates the last protected pair, z_5. It follows that $X_9 = \varphi(X_5) \backslash \{z_5\}$ and $Z_9 = \{z_1, z_2\}$.

The next two rounds are similar. In Round 9, it is found that $\{z_1, z_2\}$ is not a subset of any stable matching, hence $X_{10} = X_2 \setminus \{z_2\}$ and $Z_{10} = \{z_1\}$. In Round 10, it is established that z_1 is not an element of any stable matching so $Z_{11} = \varnothing$ and $X_{11} = \varphi(X_1) \setminus \{z_1\}$. The algorithm is almost back to where it started, however it has established in the process that none of the stable matchings contain z_1.

Round 11 calculates $\varphi(X_{11})$ and protects another agent-object pair, labeled z_{11}. Note that z_{11} could very well be one of the pairs that was previously eliminated, for example it is possible that $z_{11} = z_2$. z_2 was eliminated because it is not an element of any stable matching containing z_1, however now that z_1 has been eliminated it is again possible to

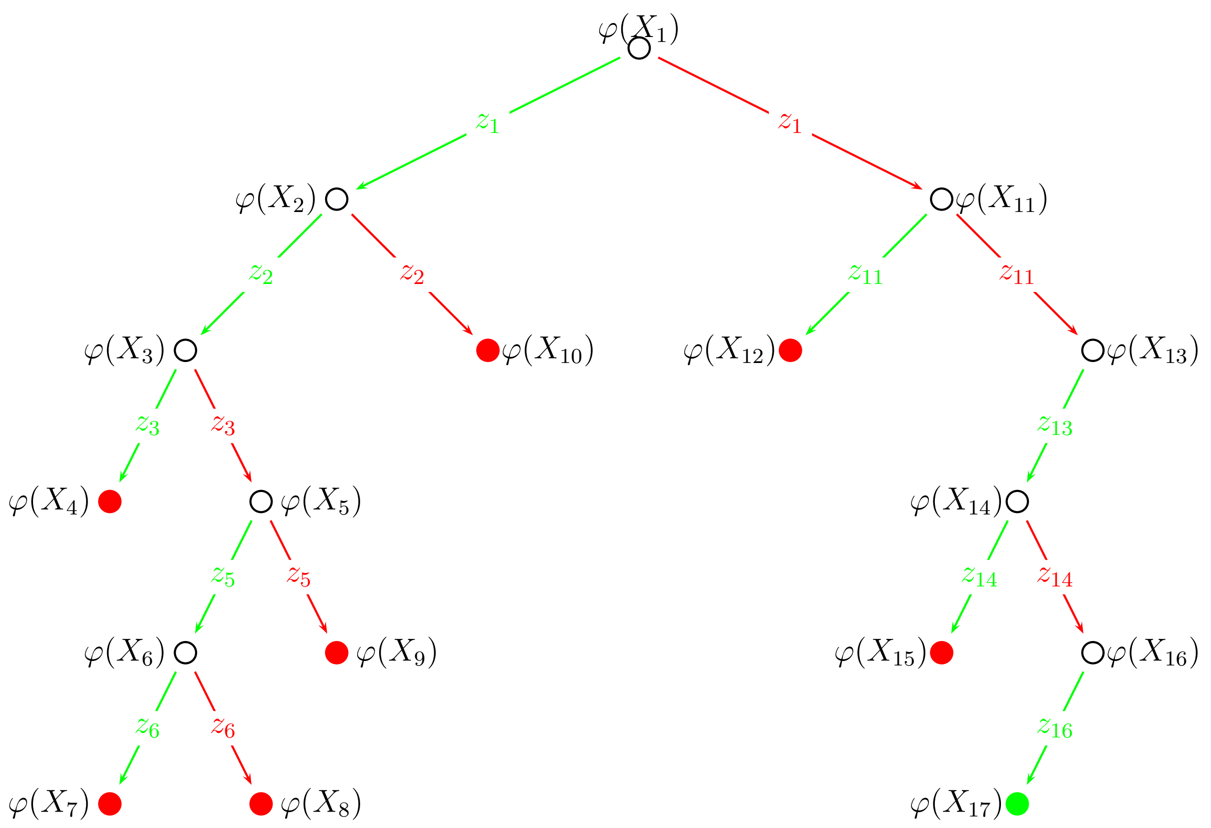

Figure 1: Depth-First Search of the USM algorithm (USM found).

find a stable matching that contains z_2. Round 12 establishes however that X_{12} does not include any stable matching, hence z_{11} is not an element of any stable matching either. X_{13} is constructed by removing z_{11} from X_{11} and once again the set of protected agent-object pairs is empty: $Z_{13} = \varnothing$.

Agent-object pairs z_{13} and z_{14} are protected in Rounds 13 and 14. z_{14} is subsequently eliminated in Round 15 as either $\varphi(X_{15}) = \varnothing$ or $\mathcal{E}(\varphi(X_{15}) > 0$. Another pair, z_{16}, is protected in Round 16 and, finally in Round 17, $\mathcal{L}(\varphi(X_{17})) = |B(\varphi(X_{17}))| = 0$. The algorithm ends and yields the undominated stable matching $\mu^*_\theta \equiv \overline{\mu}(\varphi(X_{17}))$.

Figure 2 presents a slightly different scenario. Rounds 1 to 16 are identical so that $Z_{17} = \{z_{13}, z_{16}\}$. In contrast to the previous case, $\varphi(X_{17})$ now either is the empty set or has a positive excess supply. Then z_{16} is eliminated in Round 17: $X_{18} = \varphi(X_{17}) \setminus \{z_{16}\}$ and $Z_{18} = \{z_{13}\}$. The same holds for $\varphi(X_{18})$, hence z_{13} is eliminated in Round 18: $X_{19} = \varphi(X_{13}) \setminus \{z_{13}\}$ and $Z_{19} = \varnothing$.

In Round 19, it is again established that $\varphi(X_{19}) = \varnothing$ or $\mathcal{E}(\varphi(X_{19})) > 0$ so X_{19} is does not include any stable matching. Since $Z_{19} = \varnothing$, the algorithm ends and reports that the set of stable matchings is empty. To see this, recall that $\varphi(X_1)$ includes all stable matchings and observe that $X_{19} = \varphi(X_1) \setminus \{z_1, z_{11}, z_{13}\}$. As protecting any of z_1, z_{11} or z_{13} is incompatible with stability, these pairs are not an element of any stable matching. X_{19} consequently

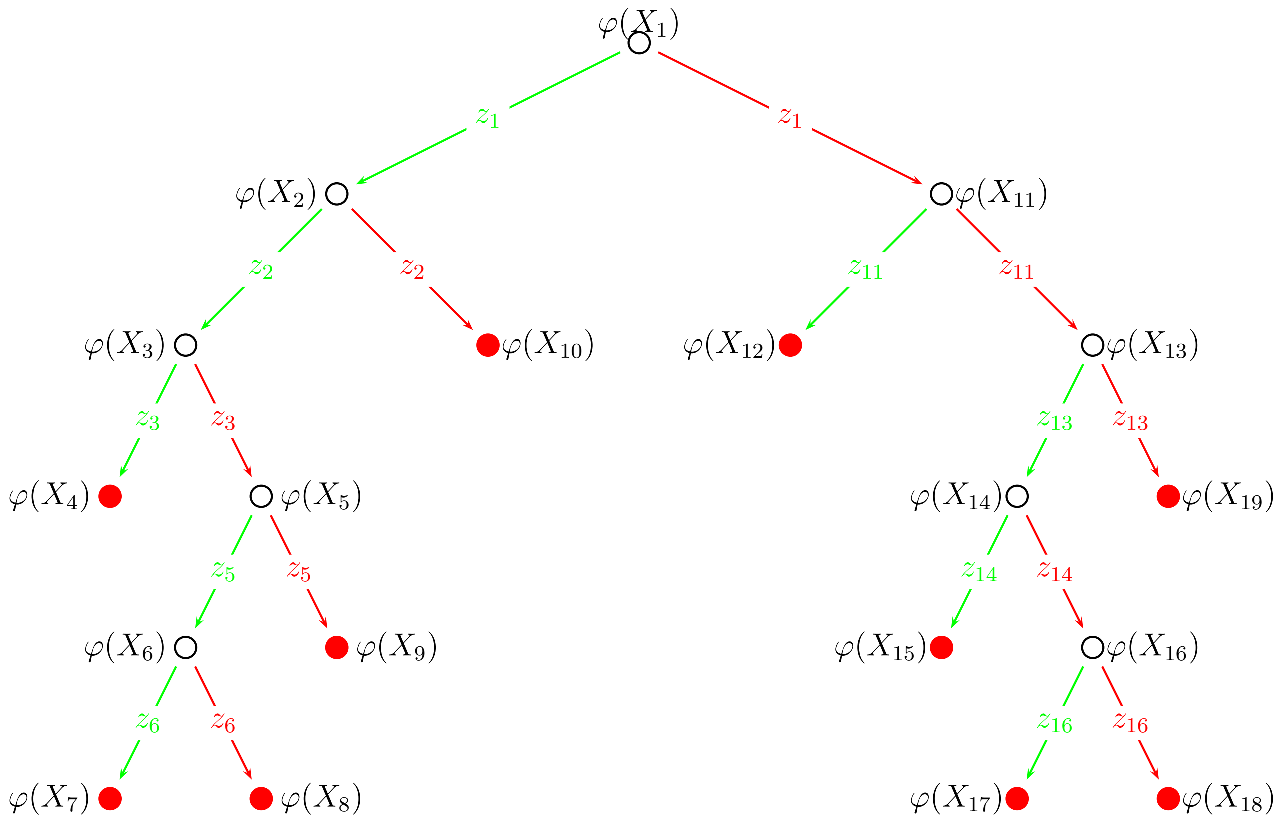

Figure 2: Depth-First Search of the USM algorithm (empty set of stable matchings).

includes all stable matchings. The fact that it does not include any stable matching implies that there does not exist any stable matching. The algorithm ends and produces $\mu_\theta^* = \varnothing$.

5 Relaxing Stability

The USM algorithm finds an undominated stable matching whenever one exists. Additionally, it has the double computational advantage to reduce the size of the market in polynomial time in every round and to guarantee that the first stable matching it finds is not dominated, thus allowing to find such solution without having to compute the whole set of stable matchings. In applications where a stable matching exists with a high probability, the USM algorithm is likely to constitute a satisfying solution. The experience of the National Resident Matching Program (NRMP) – a stable matching has been found every year since the program's inception – suggests that this is the case in large markets where agents tend to like the same objects and objects tend to assign a high priority to the same agents. In the case where the set of stable matchings is empty however, the USM algorithm simply reports that fact and does not provide any alternate solution. It consequently is unlikely to prove successful in applications where the non-existence of stable matchings constitutes a common occurrence.

In this section, we return to our three properties – K-boundedness, size-consistency and non-wastefulness – that characterize stability (Proposition 2) and consider relaxations of stability that involve any two of these properties. We show (Theorem 2) that for any nonnegative integer K, there exists a market in which the set of feasible, K-bounded and non-wasteful matchings is empty. Thus, any relaxation of stability that includes these two properties does not guarantee existence and a market designer must choose between bounding the size of claims and eliminating waste. We show on the other hand that a feasible, 1-bounded and size-consistent matching as well as a feasible, size-consistent and non-wasteful matching exist in any market. In Section 6, we build upon the latter result as well as the techniques developed in Sections 3 and 4 to propose a suitable relaxation of stability for applications where eliminating waste is a priority.

5.1 Impossibility result

A direct consequence of the possible non-existence of stable matchings in this market (Proposition 1) and our characterization result (Proposition 2) is that the set of feasible, K-bounded, size-consistent and non-wasteful matchings may be empty. We strengthen this result by showing that removing size-consistency as a requirement does not guarantee existence.

Theorem 2. *For any nonnegative integer K, the set of feasible, K-bounded and non-wasteful matchings may be empty.*

In order to provide intuition for this result, we present an example below where any non-wasteful matching has a 2-unit claim, proving that a 1-bounded and non-wasteful matching may not exist. That example can be extended to cater for larger numbers (Example 7 in the appendix) in order to prove the full theorem. The mechanism for the proof is the same as in the simple case presented below.

Example 5 (No 1-bounded non-wasteful Matching)**.** There are three single-unit agents s_1, $\hat{s}_1$ and $\hat{s}_2$, one double-unit agent d_1 and two non-null objects o_1 and o_2. The preferences, priorities and quotas are

$$\begin{array}{llll} \succ_{s_1}: o_1, o_2, \emptyset & \succ_{\hat{s}_1}: o_2, o_1, \emptyset & \rhd_{o_1}: \hat{s}_1, \hat{s}_2, d_1, s_1 & q_{o_1} = 2 \\ \succ_{d_1}: o_1, \emptyset, o_2 & \succ_{\hat{s}_2}: o_2, o_1, \emptyset & \rhd_{o_2}: s_1, \hat{s}_1, \hat{s}_2, d_1 & q_{o_2} = 2 \end{array}$$

If d_1 is matched to o_1, then feasibility dictates that none of the single-unit agents be matched to o_1 and at least one of them be matched to the null object. If s_1 is matched to the null object, he has a 2-unit claim to o_2 as he has the highest priority for that object. If either $\hat{s}_1$ or $\hat{s}_2$ is matched to the null object, he has a 2-unit claim to o_1 as both units of that object are assigned to d_1 who has a lower priority. If d_1 is matched to o_2, he has a claim to at least two unassigned units of the null object as the latter is available in enough units to accommodate all agents. It follows that d_1 is matched to the null object in any 1-bounded and non-wasteful matching. In that case, d_1 has a 2-unit claim to o_1 unless at least one of $\hat{s}_1$ or $\hat{s}_2$ is matched to that object. That agent has a claim to an unassigned unit of o_2 unless the other two single-unit agents are matched to o_2. The matching obtained is wasteful since one unit of o_1 is unassigned and s_1 prefers o_1 to o_2. We conclude that there does not exist any 1-bounded and non-wasteful matching in Example 5.

Theorem 2 sheds some light on the non-existence of stable matchings in markets with sizes, which arises from a tension between two necessary conditions for stability.[19] Theorem 2 also has important consequences from a market design point of view as any relaxation of stability that guarantees existence may limit the size of claims or eliminate waste but cannot achieve both in general.

We show in Section 5.2 that the Priority Focused Deferred Acceptance (PFDA) algorithm (Delacrétaz, Kominers, and Teytelboym, 2016) finds a 1-bounded and size-consistent matching, which directly implies the existence of such a matching. In Section 5.3, we define **size-stability** as a novel relaxation of stability for matching markets with sizes. We show that size-stability is characterized by size-consistency and non-wastefulness (Proposition 9) and that the set of size-stable matchings is nonempty (Proposition 10). We discuss the trade-offs between the two approaches in light of several applications in Section 5.4.

5.2 Wasteful Solution

In applications where waste can be tolerated, a sensible relaxation of stability consists in the combination of feasibility, K-boundedness and size-consistency. As we discuss in Section 5.4, Delacrétaz, Kominers, and Teytelboym (2016) argue that refugee resettlement constitutes such an application. We show that their Priority-Focused Deferred Acceptance (PFDA) algorithm (Algorithm 4) produces a feasible, 1-bounded and size-consistent matching. Its structure is reminiscent of the Deferred Acceptance algorithm with the difference that an agent stops contesting an object as soon as that object has rejected an agent with a higher priority. This ensures that agents never envy each other: an agent is assured that all agents

[19]While removing size-stability as a requirement does not guarantee existence in general, it has an impact on some markets. Example 1 shows that the set of stable matchings may be empty even in the case where a feasible, 1-bounded and non-wasteful matching exists. $\{(s_1, \emptyset), (s_2, o_2), (d_1, o_1)\}$ satisfies those properties as s_1 is the only agent to have a claim, that claim is to o_2 and the unique unit of that object is assigned to s_2. That matching is however not size-consistent as s_1 envies s_2.

Algorithm 4: Priority-Focused DA (Delacrétaz, Kominers, and Teytelboym, 2016)

```
Initialization:
Given a market, let $X_1 \equiv A \times O$.
Round $k \geq 1$:
    Given $X_k$, every agent $a \in A$ proposes to his top choice $\overline{o}_a(X_k)$.
    For each object $o$, agents are considered one at a time in order of priority. An
    agent is tentatively accepted if all proposing agents with a higher priority have
    been tentatively accepted and the total number of units required by himself
    and all proposing agents with a higher priority does not exceed $q_o$. The agent
    is rejected otherwise.
    If all agents are tentatively accepted, let $\mu^{\text{DA}} \equiv \overline{\mu}(X_k)$. Otherwise, construct
    $X_{k+1}$ by eliminating from $X_k$ all pairs involving an agent and an object such
    that the object has rejected the proposal of either the agent or an agent with
    a higher priority and continue to Round $k+1$.
```

matched to an object he prefers to his own have a higher priority than him for that object.[20]

The PFDA algorithm may create waste when a double-unit agent is rejected and leaves one unit unassigned, however all other units are assigned. As the algorithm ensures that no agent envies another, that unassigned unit is the only one to which agents may have a claim. These considerations naturally lead to the following result.

Proposition 8. *μ^{PFDA} is feasible, 1-bounded and size-consistent.*

Corollary 4 (to Proposition 8)**.** *There exists a feasible, 1-bounded and size-consistent matching in all markets.*

Proposition 8 means that, whenever a small amount of waste can be tolerated, claims can be limited to just one unit while preserving size-consistency.[21] The PFDA algorithm also has the advantage to be polynomial-time solvable since at least one agent-object pair is definitively eliminated in each round.[22]

[20]Delacrétaz, Kominers, and Teytelboym (2016) refer to this property as *quasi-stability*.

[21]The DA algorithm also produces a 1-bounded matching, however the latter is not necessarily size-consistent. This is illustrated by Example 3, where the DA algorithm matches s_2 to o_1 and s_1 to o_2 although $s_1 \rhd_{o_1} s_2$ and $o_1 \succ_{s_1} o_2$.

[22]Proposition 8 implies that the PFDA algorithm algorithm also satisfies Biró and McDermid's (2014)

Round 1		Round 2		Round 3		Round 4	
$s_1 \to o_1$	✗	$s_1 \to o_2$	✓	$s_1 \to o_2$	✓	$s_1 \to o_2$	✓
$\hat{s}_1 \to o_2$	✓	$\hat{s}_1 \to o_2$	✓	$\hat{s}_1 \to o_2$	✓	$\hat{s}_1 \to o_2$	✓
$\hat{s}_2 \to o_2$	✓	$\hat{s}_2 \to o_2$	✗	$\hat{s}_2 \to o_1$	✓	$\hat{s}_2 \to o_1$	✓
$d_1 \to o_1$	✓	$d_1 \to o_1$	✓	$d_1 \to o_1$	✗	$d_1 \to \emptyset$	✓

Table 7: PFDA algorithm on Example 5.

Table 7 presents the computations of the PFDA algorithm on Example 5. In the first Round, o_1 is available in enough units to satisfy the requirement of either one of s_1 and d_1, but not both. The agent with the lowest priority – in this case s_1 – is rejected. In Round 2, s_1 proposes to o_2, his second choice. Only two of s_1, $\hat{s}_1$ and $\hat{s}_2$ may be matched to that object. $\hat{s}_2$ is rejected has he has the lowest priority. He then proposes to o_1 in Round 3 and is tentatively accepted as he has a higher priority than both d_1 and s_1. Then d_1 is rejected since the first unit of o_1 is now tentatively assigned to $\hat{s}_2$ and only one remains. d_1 proposes to the null object in Round 4. The algorithm ends as none of the agents are rejected. It yields

$$\mu^{PDFA} = \{(s_1, o_2), (\hat{s}_1, o_2), (\hat{s}_2, o_1), (d_1, \emptyset)\}.$$

$\hat{s}_1$ does not have a claim at μ^{PDFA} since he is matched to his first preference. $\hat{s}_2$ does not have a claim either since he is matched to his second preference and both units of his first preference, o_2, are assigned to agents with a higher priority (s_1 and $\hat{s}_1$). d_1 and s_1 are also matched to their second preference, respectively $\emptyset$ and o_2. They both have a one-unit claim to their first preference, o_1, as one unit of that object is assigned to the higher priority agent $\hat{s}_2$ and the other is unmatched. μ^{PFDA} is consequently 1-bounded. It is also size-consistent as agents only have a claim to an unassigned unit.[23]

relaxation of stability (see footnote 15). The difference with the TDBU algorithm is that μ^{PFDA} is stable in a market where the capacity of some objects (those with a wasted unit) is reduced by one unit while $\overline{\mu}(\varphi(A \times O))$ is stable in a market where the capacity of some objects (those involved in a bottlneck) is increased by one unit. The algorithm proposed by Yenmez (2014) also constitutes such an approximation in our model, however it entirely precludes single-unit agents from being assigned the last unit of an object. Consequently, the matching it produces is dominated by the outcome of the PFDA algorithm.

[23] Yenmez (2014) introduces a version of the agent-proposing deferred acceptance algorithm which can be adapted to our model. In every round, objects accept agents one at a time in order of priority as long as they have at least two units available and reject the remaining proposals after that. One unit of each object may therefore remain unassigned at the end of the algorithm even though some single-unit agents have been rejected. As priorities are otherwise respected, this algorithm produces a 1-bounded and size-stable matching. The only difference between this algorithm and the PFDA algorithm is that the latter assigns the last unit of a single-unit agent as long as no double-unit agent with a higher priority has been rejected. The matching produced by the PFDA algorithm consequently weakly dominates the outcome of Yenmez' (2014) algorithm.

5.3 Size-Stable Matchings

While waste is tolerable in some applications, in many cases it is preferable to assign units to a lower priority agent than not at all. We argue in Section 5.4 that this is the case in tuition exchange. We know from Theorem 2 that the size of claims cannot be bounded if existence and non-wastefulness are to be guaranteed. We show that on the other hand size-consistency is compatible with these properties. We argue, however, that not all matchings satisfying these properties are desirable and that double-unit agents need to be compensated for the possible priority violations.

We say that an agent-object pair (a, o) is a **strong blocking pair** of a matching μ if a has a claim at μ to at least w_a units of o that are either unassigned or assigned to agents with a lower priority and a weakly larger size. Formally, $(s, o) \in S \times O$ is a strong blocking pair of $\mu \in \mathbb{M}$ if $\sum_{a \in \hat{A}_{(a,o)}(\mu)} w_a \leq q_o - 1$ and $(d, o) \in D \times O$ is a strong blocking pair of $\mu \in \mathbb{M}$ if $\sum_{a \in \hat{A}_{(a,o)}(\mu) \cup S_o(\mu)} w_a \leq q_o - 2$. The definition of a strong blocking pair is equivalent to that of a blocking pair for single-unit agents. For double-unit agents, it is a stronger concept as a strong blocking pair only occurs if two or more units are either unassigned or assigned to *double-unit* agents with a lower priority. Double-unit agents can therefore have a claim to arbitrarily many units assigned to single-unit agents with a lower priority without creating a strong blocking pair.

Definition 6. *A matching is **size-stable** if it is feasible and does not have any strong blocking pair.*

We denote by $\tilde{\mathbb{S}}$ the set of size-stable matchings. Size-stability constitutes a relaxation of stability as it allows blocking pairs that involve a double-unit agent with a claim to units assigned to single-unit agent. In fact, it is characterized by all but one of the properties that characterize stability.

Proposition 9. *A feasible matching μ is size-stable if and only if it is size-consistent and non-wasteful.*

Size stability constitutes an essential fairness criterion. While it does not completely eliminate blocking pairs, it ensures that no unit is wasted and that an agent a only envies an agent a' if $w_a > w_{a'}$. These properties have important practical implications as they offer a justification to agents whose priority may have been violated.

A matching $\mu \in \mathbb{M}$ is an **undominated size-stable matching** if it is size-stable and not dominated by any size-stable matching. Similarly, μ is a **d-undominated size-stable matching** if it is size-stable and not d-dominated by any size-stable matching.

Round 1		Round 2	
$s_1 \to o_1$	✓	$s_1 \to o_1$	✓
$\hat{s}_1 \to o_2$	✓	$\hat{s}_1 \to o_2$	✓
$\hat{s}_2 \to o_2$	✓	$\hat{s}_2 \to o_2$	✓
$d_1 \to o_1$	✗	$d_1 \to \emptyset$	✓

Table 8: SDDA algorithm on Example 5.

The set of size-stable matchings is nonempty in all markets. In fact, a simple algorithm finds one such matching. We define the **size-disjoint priority relation** of object o, denoted $\rhd^*_o$, such that for any $a, a' \in A$ with $a \rhd_o a'$, $a' \rhd^*_o a$ if $w_{a'} < w_a$ and $a \rhd^*_o a'$ otherwise. In words, the size-disjoint priority relation is constructed by ranking all single-unit agents above all double-unit agents and ranking agents of same size in order of priority. The **size-disjoint priority profile** is the $|O|$-tuple of size-disjoint priority relations $\boldsymbol{\rhd}^* \equiv (\rhd^*_o)_{o \in O}$. We call Size Disjoint Deferred Acceptance (SDDA) the algorithm obtained by running the Deferred Acceptance algorithm on the market $\langle A, O, \succ, \boldsymbol{\rhd}^*, \mathbf{w}, \mathbf{q} \rangle$ and denote by μ^{SDDA} the matching produced in this way.

Proposition 10. *μ^{SDDA} is an undominated size-stable matching.*

μ^{SDDA} is size-consistent by construction of $\boldsymbol{\rhd}^*$. It is also non-wasteful and not dominated by any other size-stable matching as objects only reject agents when its units have been assigned to agents with a higher size-disjoint priority. We illustrate the algorithm with Example 5. Computations are displayed in Table 8. In Round 1, all agents propose to their first preference. o_2 tentatively accepts both $\hat{s}_1$ and $\hat{s}_2$ as their total size is $2 = q_{o_2}$. In contrast, o_1 cannot tentatively accept both s_1 and d_1 as their total size is $3 > 2 = q_{o_1}$. Although d_1 has a higher priority, $s_1 \rhd^*_o d_1$ since $w_{s_1} < w_{d_1}$, therefore o_1 tentatively accepts s_1 and rejects d_1, illustrating how the algorithm can violate priorities. In Round 2, d_1 proposes to his second preference, which is the null object. As all agents are tentatively accepted, the algorithm ends and produces

$$\mu^{\text{SDDA}} = \{(s_1, o_1), (\hat{s}_1, o_2), (\hat{s}_2, o_3), (d_1, o_1)\}.$$

μ^{SDDA} is size-stable as its only blocking pair is (d_1, o_1) and one of o_1's two units is assigned to a single-unit agent. It is undominated as all single-unit agents are matched to their first preference and matching d_1 to his first preference, o_1, violates feasibility unless s_1 is made worse-off. In fact, it can be verified that μ^{SDDA} is the unique size-stable matching in this market since making any agent either better- or worse-off creates a strong blocking pair.

The SDDA algorithm provides a sensible solution to Example 5. Compared to the PFDA algorithm, it eliminates waste by matching s_1 to o_1. The trade-off is that it allows d_1 to have

Round 1		Round 2	
$s_1 \to o_1$	✓	$s_1 \to o_1$	✓
$s_2 \to o_1$	✓	$s_2 \to o_1$	✓
$s_3 \to o_1$	✓	$s_3 \to o_1$	✓
$s_4 \to o_2$	✓	$s_4 \to o_2$	✓
$d_1 \to o_1$	✗	$d_1 \to \emptyset$	✓
$d_2 \to o_1$	✗	$d_2 \to \emptyset$	✓

Table 9: SDDA algorithm on Example 6.

a 2-unit claim to o_1. In general, however, the SDDA algorithm may produce an arguably unfair outcome as it favors single-unit agents over double-unit ones and only takes priorities into account for same-size agents. We illustrate this with another example.

Example 6 (Unfair SDDA Outcome)**.** There are four single-unit agents s_1, s_2, s_3 and s_4, two double-unit agents d_1 and d_2 and two non-null objects o_1 and o_2. The preferences, priorities and quotas are

$\succ_{s_1}: o_1, o_2, \emptyset$	$\succ_{s_3}: o_1, o_2, \emptyset$	$\succ_{d_1}: o_1, \emptyset, o_2$	$\rhd_{o_1} : d_1, s_4, d_2, s_1, s_2, s_3$	$q_{o_1} = 4$
$\succ_{s_2}: o_1, o_2, \emptyset$	$\succ_{s_4}: o_2, o_1, \emptyset$	$\succ_{d_2}: o_1, \emptyset, o_2$	$\rhd_{o_2} : s_1, s_2, s_4, s_3, d_1, d_2$	$q_{o_2} = 2$

Table 9 presents the computations of the SDDA algorithm on Example 6. In Round 1, all agents propose to their first preference. s_4 proposes to o_2 and is tentatively accepted as he is the only proposers. All other agents propose to o_1. Despite having a higher priority, the double unit agents d_1 and d_2 are considered after the single-unit agents s_1, s_2 and s_3. The latter are tentatively accepted and, as only one unit remains available, d_1 and d_2 are rejected. In Round 2, d_1 and d_2 propose to the null object. All agents are tentatively accepted so the algorithm ends and produces

$$\mu^{\text{SDDA}} = \{(s_1, o_1), (s_2, o_1), (s_3, o_1), (s_4, o_2), (d_1, \emptyset), (d_2, \emptyset)\}.$$

This matching is size-stable as it has only two blocking pairs – (d_1, o_1) and (d_2, o_1) – and three units of o_1 are assigned to single-unit agents. It is an undominated size-stable matchings as all singe-unit agents are matched to their first preference and matching a double-unit agent to his first preference would make at least one single-unit agent worse-off.

In spite of these desirable properties, μ^{SDDA} is arguably unfair to the double-unit agents, who were rejected by o_1 because of single-unit agents with a lower priority. In some instances, this is inevitable. We know from Theorem 2 that the size of claims cannot be bounded for non-wasteful matchings, a result that directly carries over to size-stable matchings. In Example

5, d_1 is not matched to o_1 in any size-stable matchings. Not fulfilling his claim to that object can be justified as it would violate size-stability. This is not the case in Example 6. The matching

$$\mu' = \{(s_1, o_1), (s_2, o_1), (s_3, o_2), (s_4, o_2), (d_1, o_1), (d_2, \emptyset)\}$$

is size-stable. The only blocking pair is (d_2, o_1), which is not a strong blocking pair since the four units of o_1 are assigned to d_1, who has a higher priority, and s_1 and s_2, who have a smaller size. Compared to μ^{SDDA}, μ' does not violate d_1's priority as he is matched to o_1. d_2's priority is still violated, however this cannot be prevented. If d_2 is matched to o_1, feasibility dictates that either d_1 or both s_1 and s_2 be matched to another object. In the former case, d_1 envies d_2 and the matching is not size-stable. In the latter case, size-stability dictates that both s_1 and s_2 be matched to o_2. In turn, s_4 is matched to the null object by feasibility and envies d_2, violating size-stability. μ' is fairer than μ^{SDDA} as it respects the priorities of double-unit agents as much as possible. Formally, μ' d-dominates μ^{SDDA} because it makes d_1 better-off and d_2 is indifferent between the two matchings. In fact, μ' is d-undominated because any d_2 cannot be matched to o_1 without violating size-stability.[24]

We propose *d-undominated* size-stable matchings as a solution concept. Such a matching alleviates the disadvantage that double-unit agents suffer as their priority may be violated. Every blocking pair is justifiable in the sense any size-stable matching that contains the pair makes at least one other double-unit agent worse-off. In Section 4.2 (Example 4), we argued that finding a d-undominated stable matching rather than any undominated one is desirable from a fairness point of view as it avoids creating 1-unit claims whenever possible. This fairness concern becomes crucial when it comes to size-stability and the size of claims that double-unit agents may have is unbounded.

As d-domination is a partial order, the existence of a size-stable matching implies the existence of a d-undominated size-stable matching. The next result then follows directly from Proposition 10.

Corollary 5 (to Proposition 10)**.** *The set of d-undominated size-stable matchings is nonempty.*

Beyond existence, d-undominated size-stable matchings have a welfare advantage over stable matchings. By Propositions 2 and 9, all stable matchings are size-stable. Consequently, a d-undominated size-stable matching may dominate some size-stable matchings – including some d-undominated stable matchings – but none of the stable matchings dominate any d-undominated size-stable matching. In various applications (e.g. day care or tuition exchange), stability is desirable as a fairness criterion. d-undominated size-stable matchings

[24]It can be verified that μ^{SDDA} and μ' are the only two size-stable matchings in this market.

may constitute a strong enough fairness requirement in many of these applications as blocking pairs can be justified by size differences and the fact that any other size-stable matching makes a double-unit agent worse-off. If so, their welfare and existence advantages over stability make d-undominated size-stable matchings a natural solution concept. The question that remains is how to compute these matchings. We provide an answer in Section 6, where we adapt our techniques from Sections 3 and 4 in order to find such a matching.

5.4 Discussion

Theorem 2 and Proposition 8 show that eliminating waste comes at a high cost when it comes to respecting priorities as agents may have a claim to an arbitrarily large number of units in all non-wasteful matchings, but it is possible to limit all claims to one unit by wasting at most one unit of each object. These results outline an important trade-off that a market designer faces when confronted to a matching market with sizes. Whether it is preferable to eliminate waste or bound the size of claims depends on the specificities of the application at stake. As Delacrétaz, Kominers, and Teytelboym (2016) argue, waste is tolerable in refugee resettlement as groups of refugees (agents) arrive regularly and any unused capacity can benefit the next group. Goodwill from local areas (objects) is on the other hand of the utmost importance as they are the one providing service capacities (quotas). Respecting the local areas' priorities and not fully using the capacity they make available constitute two market design tools that can help generate that goodwill. The PFDA algorithm appears to be a natural solution in that case.[25] At the other end of the spectrum, tuition exchange is a "one-shot" market in the sense that the number of students that a university can send to its partners does not increase if some places were not used in previous years. A university exchange office may for then prefer to send a lower quality applicant rather than no one at all.[26] Eliminating waste may also constitute a priority in a market with agents on both sides, such as the National Resident Matching Program (NRMP). Participants may rapidly lose trust in the matching system if a hospital does not fill a position that eligible applicants would be willing to take. The problem may be less severe if the hospital has been able to fill the position with a lower-quality applicant. Day care lies somewhere in between. While children may join at any point, the largest intake takes place once a year when older children

[25]Delacrétaz, Kominers, and Teytelboym (2016) also show that the PFDA algorithm can be made strategy-proof at the cost of making families worse-off and creating more waste.

[26]This may of course depend on a university's specific circumstances. Dur and Ünver (2017) argue that some agreements specify that each partner must send approximately the same number of students over a moving time window. Such a feature could motivate an exchange office to send fewer students one year in order to send more of them later on. In practice, this requirement appears in the United States but is uncommon outside. For example, it does not play a role in Erasmus agreements.

```
Algorithm 5: Permissive Top-Down Bottom-Up (p-TDBU)

Initialization:
Given a market, a set of agent-object pairs $X$, let $X^1 \equiv X$.
Step $k \geq 1$:
If $X^k \notin \mathbb{X}$, let $\phi(X) \equiv \varnothing$. Otherwise, construct $X^{k+1}$ by eliminating from $X^k$ all
agent-object pairs such that the object p-rejects the agent at $X^k$ and, if $X = A \times O$,
all agent-object pairs such that the agent receives a guarantee at $X^k$ from an object
he prefers.
If $X^{k+1} = X^k$, let $\phi(X) \equiv X^k$. Otherwise, continue to Step $k+1$.
```

go to school. At this point, it may be tolerable to underuse the capacity of day care centers in the hope that these places be filled with children who apply later. The cost of doing so may however be large as demand typically greatly exceeds supply. Whether it outweighs the benefit of limiting the size of claims depends on specific circumstances that are beyond the scope of this paper.

6 d-Undominated Size-Stable Matching

d-undominated size-stability constitutes a natural solution concept when eliminating waste is an important requirement. Double-unit agents may have large claims to some objects but are as well-off as size-stability permits. In this section, we show that a suitable modification of our Undominated Stable Matching (USM) algorithm (Algorithm 3) produces a d-undominated size-stable matching. We present the Permissive Top-Down Bottom-Up (p-TDBU) algorithm in Section 6.1 and combine it with a depth-first search in Section 6.2. We illustrate the full algorithm in Section 6.3 with Example 6.

6.1 Permissive Top-Down Bottom-Up

The Permissive Top-Down Bottom-Up (p-TDBU) algorithm is presented in Algorithm 5. It follows the general architecture of the TDBU algorithm (Algorithm 2) but adapts it to size-stability. Any set of agent-object pairs $X \subseteq A \times O$ can enter the algorithm. The latter returns $\phi(X)$, which is either the empty set or a complete subset of X. In each round, a set of agent-object pairs $X \in 2^{A\times O}$ enters. If X is complete ($X \in \mathbb{X}$), the algorithm eliminates some of the pairs and considers a smaller set in the next round, until such round where it does not find any pair to eliminate or an incomplete set enters. If X is incomplete ($X \notin \mathbb{X}$), the algorithm ends and returns the empty set.

The Bottom-Up part is identical to that of the TDBU algorithm. An object o gives an agent a a guarantee at X if a's guarantee number for o at X does not exceed o's quota. In the next round, an agent with a guarantee stops contesting all less preferred objects.

The Top-Down part is modified to allow more single-unit agents to continue contesting an object for which their rejection number (as defined in Section 3.1) exceeds the quota, hence it is more "permissive". This allows single-unit agents to take advantage of units that become available after a double-unit agent with a higher priority stops contesting an object.

Specifically, given a pair $(a, o) \in X$, the algorithm calculates the **p-rejection number** of agent a for object o at X, which we denote $\tilde{\mathcal{R}}_{(a,o)}(X)$. For double-unit agents, p-rejection numbers are calculated almost identically to rejection numbers by adding up the size of the agent as well as all agents with a higher priority who contest the object as their top choice. For single-unit agents, the size of all single-unit agents with a higher priority who contest the object as their top choice is taken into account but the size of double-unit agents is only taken into account if they have a higher priority and do not contest any other object. For notational convenience, let us define $\tilde{D}_{(a,o)}(X) \equiv \{d' \in \hat{D}_{(a,o)}(X) \mid O_{d'}(X) = \{o\}\}$ to be the set of double-unit agents who *only* contest o at X and have a higher priority than d. Then the p-rejection number of agent a for object o at $X \in \mathbb{X}$ is

$$\tilde{\mathcal{R}}_{(a,o)}(X) \equiv \begin{cases} w_a + \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'} & \text{if } a \in D \\ w_a + \sum_{s \in \hat{S}_{(a,o)}(X) \cap \overline{S}_o(X)} w_s + \sum_{d \in \tilde{D}_{(a,o)}(X)} w_d & \text{if } a \in S. \end{cases}$$

We say that object o **p-rejects** agent a at X if $\tilde{\mathcal{R}}_{(a,o)}(X) > q_o$. If a double-unit agent d's p-rejection number for an object o exceeds the quota, then an agent envies d in any feasible matching that contains (d, o). In any size-stable matching included in X, d is not matched to o. This condition is identical to Lemma 5's Condition (ii). If a single-unit agent's p-rejection number for an object o exceeds the quota, then the q_o units of o are assigned to agents with a higher priority that are either a single-unit agent who contests o as his top choice or a double-unit agent that only contests o. If s is matched to o, by feasibility, at least one of these agents is not. If a single-unit agent who contests o as his top choice at X is matched to a different object, he prefers o to that object and size-consistency is violated. If a double-unit agent who only contests o at X is not matched to it, an incomplete set of pairs obtains. We conclude that s is not matched to o in any size-stable matching. If an object o p-rejects an agent a at X, the pair is not an element of any size-stable matching included in X. In the next step of the p-TDBU algorithm, a no longer contests o.

The p-rejection number differs from the rejection number in two important ways. First, the size of double-unit agents who contest more than one object do not count towards single-unit agents' p-rejection number. Second, the size of agents in the pair's envy set is not taken into account. Both differences arise from the definition of size-stability. Whether or not a single-unit agent is envied by double-unit agents does not affect his ability to be matched to the object in a size-stable matching, therefore his p-rejection number is not affected by these agents.

Similarly to the TDBU algorithm, the p-TDBU algorithm eliminates agent-object pairs from the top-down and the bottom-up. A pair is eliminated if the agent has received a guarantee form an object he prefers or if the object p-rejects the agent. It does so iteratively until such step where it obtains an incomplete set of pairs or it cannot eliminate any pair. We show below that this does not affect the search for a d-undominated stable matching. The Top-Down part only eliminates agent-object pairs that are not an element of any size-stable matching, as the following lemma formalizes.

Lemma 11. *For any $X \in \mathbb{X}$ and any $(a, o) \in X$, if o p-rejects a at X, then for any $\mu \in \tilde{\mathbb{S}} \cap 2^X$, $(a, o) \notin \mu$.*

The Bottom-Up part does not have the same properties. As discussed in Section 5.3, a size-stable matching can be constructed by favoring single-unit agents over double-unit ones. If the guarantee number of a double-unit agent d for an object o at X does not exceed q_o, this does not mean that d is matched to o or an object he prefers in any size-stable matching. As long as enough single-unit agents are matched to o, d can be matched to a less preferred object without (d, o) being a strong blocking pair. In fact, other double-unit agents may benefit from d not being matched to o, as a result eliminating all pairs involving d and his less preferred objects may preclude finding some d-undominating size-stable matchings. The following Lemma shows however that the p-TDBU algorithm does not eliminate all d-undominated size-stable matchings when it is used on the full set of agent-object pairs.

Lemma 12. *$\phi(A \times O)$ includes at least one d-undominated size-stable matching.*

Lemma 12 means that the p-TDBU algorithm can be used initially on the full set of agent-object pairs. This allows searching for a d-undominated size-stable matching within a smaller set of pairs. That set, $\phi(A \times O)$, may not contain all d-undominated size-stable matchings but it contains at least one. Unfortunately, Lemma 12 does not generalize to smaller sets of pairs so the d-USSM cannot use the Bottom-Up part of the p-TDBU algorithm after the

```
Algorithm 6: d-Undominated Size-Stable Matching (d-USSM)

Initialization:
Given a market and an index $\theta$.
Round $k \geq 1$:
Use the p-TDBU algorithm to calculate $\phi(X_k)$
Case 1: $\phi(X_k) \neq \varnothing$ and $C(\phi(X_k)) = \varnothing$.
Let $\tilde{\mu}^*_\theta \equiv \overline{\mu}(\phi(X_k))$.
Case 2: $\phi(X_k) \neq \varnothing$ and $C(\phi(X_k)) \neq \varnothing$.
Let $z_k$ be the candidate of $\phi(X_k)$ with the smallest index $\theta$. Construct $X_{k+1}$ by
protecting $z_k$ at $\phi(X_k)$ and $Z_{k+1}$ by adding $z_k$ to $Z_k$ and continue to Round $k+1$.
Case 3: $\phi(X_k) = \varnothing$.
Let $z_i$ be the last pair that was added to $Z_k$. Construct $X_{k+1}$ and $Z_{k+1}$ by eliminating
$z_i$ from $\phi(X_i)$, let $Z_{k+1} = Z_i$ and continue to Round $k+1$.
```

first round as this runs the risk of producing a set of pairs that does not include any d-undominated size-stable matching. For this reason, the Bottom-Up part is only used when the algorithm is applied to the full set of agent-object pairs $A \times O$. When the algorithm is applied to a smaller set, objects p-rejects agents but do not give any guarantee.[27] Similarly to the TDBU algorithm, the p-TDBU algorithm produces a smaller set of agent-object pairs. Mirroring Proposition 4, we show that the top matching of that set has important properties.

Proposition 11. $\overline{\mu}(\phi(A \times O))$ *is size-consistent and non-wasteful. It is a d-undominated size-stable matching if and only if it is feasible.*

6.2 Full Algorithm

The d-Undominated Size-Stable Matching (d-USSM) algorithm is presented in Algorithm 6. It adapts the USM algorithm in order to find a d-undominated size-stable matching. The full set of agent-object pairs $A \times O$ enters the algorithm. The latter returns $\tilde{\mu}^*_\theta$, which is a

[27]One may question the purpose of having the Bottom-Up part at all. Indeed, all results of Section 6 hold if the p-TDBU algorithm only consists of its Top-Down part even when applied to $A \times O$. The advantage of the Bottom-Up part is to further reduce the set of pairs that need to be considered. As the depth-first search of the d-USSM algorithm is not polynomial-time solvable, starting in Round 1 with a smaller set of pairs can have a large impact on the algorithm's running time.

matching. The p-TDBU algorithm runs first and produces $\phi(A \times O)$. If the top matching of that set is feasible, the algorithm ends and produces that matching. Otherwise, the remainder of the d-USSM algorithm searches for (and finds) a d-undominated size-stable matching included in $\phi(A \times O)$. (Lemma 12 guarantees the existence of such a matching.) It begins by protecting a pair (d, o) involving a double-unit agent to construct X_2. The p-TDBU algorithm runs at the start of Round 2. As a result of the protection, d only contests o and his size counts towards the rejection number of single-unit agents who contest o and have a lower priority. The p-TDBU algorithm may be able to eliminate more pairs as a result. The d-USSM algorithm continues protecting pairs one at a time until the p-TDBU algorithm produces either a set of pairs with a feasible top matching or the empty set. In the former case, it ends as it has found a d-undominated size-stable matching. In the latter case, it has found that there does not exist any size-stable matching included in $\phi(A \times O)$ that contains all pairs that have been protected thus far. The algorithm returns to the last protected pair and eliminates it. Such pair exists as otherwise it would have been established that $\phi(A \times O)$ does not include any size-stable matching, a contradiction of Lemma 12. Therefore, the d-USSM algorithm continues until it finds a d-undominated size-stable matching.

Mirroring Section 4, we say that a complete set of agent-object pairs is **p-irreducible** if it does not change when it enters the p-TDBU algorithm. Formally, the set of p-irreducible sets of agent-object pairs $\tilde{\mathbb{I}}$ is defined such that for any $X \in \mathbb{X}$, $X \in \tilde{\mathbb{I}}$ if $\phi(X) = X$. Given a p-irreducible set of pairs $X \in \tilde{\mathbb{I}}$, the following definition determines what pairs are eligible for protection.

Definition 7. *Given a p-irreducible set of agent-object pairs $X \in \tilde{\mathbb{I}}$, $(a, o) \in X$ is a **candidate** of X if (i) $\sum_{a \in \overline{A}_o(X)} w_a > q_o$, (ii) $d \in \overline{D}_o(X)$ and $|O_d| > 1$, and (iii) for all $d' \in \overline{D}_o(X)$ with $|O_{d'}| > 1$, $d' \trianglerighteq_o d$.*

We denote the set of candidates of X by $C(X)$. In words, a candidate of X is an agent-object pair $(a, o) \in X$ such that (i) the combined size of all agents who contest o as their top choice exceeds the quota, (ii) a is a double-unit agent who contests o as his top choice but also contests at least one other object, and (iii) all other agents in the same situation have a higher priority. Candidates involve double-unit agents who contest an object as their top choice but whose size does not count towards the rejection number of single-unit agents with a lower priority because they contest other objects, as a result the total size of agents contesting the object as their top choice exceeds the quota. If there are multiple agents in that situation for a given object, the candidate is the pair involving the agent with the lowest priority. These pairs, at most one per object, are eligible for protection and the one with the smallest index θ is protected.[28]

Protecting the pair $(d, o) \in C(X)$ allows identifying other pairs that can be eliminated. This obviously includes all pairs involving d and any object other than o. A matching $\mu \subseteq X$ that includes (d, o) is not size-consistent, hence not size-stable, if an agent with a higher priority than d for o is matched to a less preferred object. Any pair involving an agent with a higher priority than d for o and one of d's less preferred objects can be eliminated as a consequence. We formalize this below.

Definition 8. *Given an p-irreducible set of agent-object pairs $X \in \tilde{\mathbb{I}}$, the **p-protection set** of a pair $(d, o) \in C(X)$ is*

$$\tilde{P}_{(d,o)}(X) \equiv \{(a, o') \in X \mid a \trianglerighteq_o d \text{ and } o \succ_a o'\}.$$

Lemma 13. *For any $\mu \in \tilde{\mathbb{S}} \cap 2^X$ such that $(d, o) \in \mu$, $\mu \subseteq \phi(X \setminus \tilde{P}_{(d,o)}(X))$.*

Protecting (d, o) allows focusing on size-stable matchings that contain it. The set of pairs constructed by protecting (d, o) is not necessarily p-irreducible. d as well as double-unit agents with a higher priority exclusively contest o, which means that their sizes count towards the rejection numbers of single-unit agents with a lower priority. The latter may be rejected as a result. Lemma 13 means that the set obtained by successively protecting (d, o) and running the p-TDBU algorithm contains all size-stable matchings included in X that contain (d, o).

We now have all the ingredients of the d-USSM algorithm. In each Round, the d-USSM considers a set that includes all size-stable matchings that themselves include all the protected pairs. If that set is empty, then there does not exist any size-stable matching that includes all the protected pairs. The algorithm backtracks by eliminating the last protected pair. By Lemma 12, such pair exists as if no pair has been protected the set considered includes a d-undominated size-stable matching. In addition, if the set considered is complete, its top matching is size-consistent and non-wasteful because pairs are protected and eliminated in a way that preserves these properties. Eventually, enough pairs have been eliminated so that a set without any candidate is considered. The top matching of that set is size-stable by Proposition 9 since it is feasible, size-consistent and non-wasteful. This section's main result naturally follows.

[28]Alternative rules could be used to determine which pairs are eligible for protection. For example, one might want to start by protecting pairs involving double-unit agents with the highest priority or consider one double-unit agent at a time, no matter where he ranks in the object's priority. While various approaches would allow finding a d-undominated size-stable matching, an advantage of the chosen approach is that in each round, the top matching of the set of pairs considered is size-consistent and non-wasteful if it exists. Therefore, the algorithm stops whenever a feasible top matching is found without having to verify whether some objects have excess supply as is the case in Section 4.

Theorem 3. *For any index θ, $\tilde{\mu}^*_\theta$ is a d-undominated size-stable matching.*

6.3 Example

We showed in Section 5.3 that Example 6 has two size-stable matchings:

$$\mu^{\text{SDDA}} = \{(s_1, o_1), (s_2, o_1), (s_3, o_1), (s_4, o_2), (d_1, \emptyset), (d_2, \emptyset)\}$$
$$\text{and} \quad \mu' = \{(s_1, o_1), (s_2, o_1), (s_3, o_2), (s_4, o_2), (d_1, o_1), (d_2, \emptyset)\}.$$

μ' is a d-undominated size-stable matching, but μ^{SDDA} is not as μ' d-dominates it. We next illustrate the d-USSM algorithm with Example 6 and show that it produces $\tilde{\mu}^*_\theta = \mu'$ for any index θ. Computations are displayed in Table 10.

In Round 1, the p-TDBU algorithm runs on the full set of pairs $X_1 = A \times O$. Both double-unit agents contest all three objects, therefore their sizes do not count towards the p-rejection numbers of single-unit agents with a lower priority. The p-rejection number of s_4 is 1, his own size, since d_1's size does not count towards it. The p-rejection number of s_1 is also 1 as the sizes of d_1 and d_2 do not count towards it and s_4 contests o_1 as a subsequent choice. The p-rejection numbers of s_2 and s_3 are respectively 2 $(= 1 + 1)$ and 3 $(= 1 + 1 + 1)$ since these agents as well as s_1 contest o_1 as their top choice. The p-rejection number of d_1 is 2, his own size, since he has the highest priority. The p-rejection number of d_2 is 4 $(= 2 + 2)$ as d_1's size is taken into account. None of the p-rejection numbers exceed $q_{o_1} = 4$, therefore o_1 does not p-reject any agent. s_4 is the only agent to contest o_2 as his top choice. It causes d_1 and d_2's p-rejection numbers to exceed the quota, therefore o_2 p-rejects these two agents. Even without p-rejection, these pairs would have been eliminated by the Top-Down part since all agents receive a guarantee from the null object and d_1 and d_2 find o_2 unacceptable. s_1, s_2, and s_4 receive a guarantee from their respective second preference and, as a result, no longer contest the null object. Importantly, d_1 receives a guarantee from o_1, his top choice. In the second step of the p-TDBU algorithm, d_1 *only* contests o_1. This is denoted by "$\tilde{\text{T}}$" in the second column of the row devoted to that pair. The two units that d_1 requires are now taken into account when calculating the p-rejection numbers of single-unit agents with a lower priority. The p-rejection numbers for o_1 of all four single-unit agents each rise by 2. This causes o_1 to p-reject s_3 as his p-rejection number has risen to $5 > 4 = q_{o_1}$. The third step is not displayed as the p-TDBU algorithm does not eliminate any additional pair. In total, seven pairs have been eliminated and eleven remain. $\overline{\mu}(\phi(X_1))$ is not feasible since the combined size of s_1, s_2, d_1, and d_2 is $6 > 4 = q_{o_1}$. The only candidate in $\phi(X_1)$ is (d_2, o_1) since d_1 only contests o_1. That pair is protected, which means that d_2 stops contesting the

Round 1

$X_1 = A \times O$, $Z_1 = \varnothing$											
o_1	Pref.	$\mathcal{G}$	$\tilde{\mathcal{R}}$	(4)	El.	o_2	Pref.	$\mathcal{G}$	$\tilde{\mathcal{R}}$	(2)	El.
d_1	T	2	2	✓		s_1	2	1	1	✓	
s_4	2	3	1	✓		s_2	2	2	1	✓	
d_2	T	5	4			s_4	T	3	1		
s_1	T	6	1			s_3	2	4	2		
s_2	T	7	2			d_1	U	6	3	✗	El.
s_3	T	8	3			d_2	U	8	3	✗	El.
o_1	Pref.	$\mathcal{G}$	$\tilde{\mathcal{R}}$	(4)	El.	o_2	Pref.	$\mathcal{G}$	$\tilde{\mathcal{R}}$	(2)	El.
d_1	$\tilde{\text{T}}$	2	2	✓		s_1	2	1	1	✓	
s_4	2	3	3	✓		s_2	2	2	1	✓	
d_2	T	5	4			s_4	T	3	1		
s_1	T	6	3			s_3	2	4	2		
s_2	T	7	4								
s_3	T	8	5	✗	El.						
$\phi(X_1) = \{(s_1, o_1), (s_1, o_2), (s_2, o_1), (s_2, o_2), (s_3, o_2), (s_3, \emptyset), (s_4, o_1), (s_4, o_2), (d_1, o_1), (d_2, o_1), (d_2, \emptyset)\}$											

Round 2

$X_2 = \{(s_1, o_1), (s_1, o_2), (s_2, o_1), (s_2, o_2), (s_3, o_2), (s_3, \emptyset), (s_4, o_1), (s_4, o_2), (d_1, o_1), (d_2, o_1)\}$ $Z_2 = \{(d_2, o_1)\}$									
o_1	Pref.	$\tilde{\mathcal{R}}$	(4)	El.	o_2	Pref.	$\tilde{\mathcal{R}}$	(2)	El.
d_1	$\tilde{\text{T}}$	2			s_1	2	1		
s_4	2	3			s_2	2	1		
d_2	$\tilde{\text{T}}$	4			s_4	T	1		
s_1	T	5	✗	El.	s_3	T	2		
s_2	T	6	✗	El.					
o_1	Pref.	$\tilde{\mathcal{R}}$	(4)	El.	o_2	Pref.	$\tilde{\mathcal{R}}$	(2)	El.
d_1	$\tilde{\text{T}}$	2			s_1	T	1		
s_4	2	3			s_2	T	2		
d_2	$\tilde{\text{T}}$	4			s_4	T	3	✗	El.
					s_3	T	4	✗	El.
o_1	Pref.	$\tilde{\mathcal{R}}$	(4)	El.	o_2	Pref.	$\tilde{\mathcal{R}}$	(2)	El.
d_1	$\tilde{\text{T}}$	2			s_1	T	1		
s_4	T	3			s_2	T	2		
d_2	$\tilde{\text{T}}$	5	✗	El.					
$\phi(X_2) = \varnothing$									
$X_3 = \{(s_1, o_1), (s_1, o_2), (s_2, o_1), (s_2, o_2), (s_3, o_2), (s_3, \emptyset), (s_4, o_1), (s_4, o_2), (d_1, o_1), (d_2, \emptyset)\}$									

Table 10: d-USSM algorithm on Example 6.

null object. (No other pair is eliminated as d_1 and s_4 have already stopped contesting any object they rank below o_1.)

In Round 2, the p-TDBU algorithm no longer calculates guarantee numbers since the set of pairs that enters does not contain all pairs in the market ($X_2 \subset (A \times O)$). d_2 only contests o_1, therefore his size counts towards the p-rejection numbers of single-unit agents with a lower priority. The p-rejection numbers of s_1 and s_2 for o_1 are respectively 5 and 6 and exceed the quota. o_1 p-rejects both agents. In the second step of the p-TDBU algorithm, s_1 and s_2 contest o_2 as their top choice, as a result that object p-rejects s_3 and s_4. In turn, s_4 contests o_1 as his top choice in the third step and that object p-rejects d_2. The set of pairs considered is no longer complete as d_2 does not contest any object. The p-TDBU algorithm ends after this third step and produces the empty set. This means that no size-stable matching included in $\phi(X_1) = \phi(A \times O)$ contains (d_2, o_1). X_3 is constructed by removing that pair from $\phi(X_1) = \phi(A \times O)$.

Round 3 is not displayed in Table 10 because the p-TDBU algorithm does not eliminate any additional pair. The algorithm ends and produces

$$\tilde{\mu}^*_\theta = \overline{\mu}(\phi(X_3)) = \overline{\mu}(X_3) = \mu' = \{(s_1, o_1), (s_2, o_1), (s_3, o_2), (s_4, o_2), (d_1, o_1), (d_2, \emptyset)\}.$$

7 Conclusion

In this paper, we consider an extension of the canonical school choice model where agents may require either one or two units of an object. This seemingly small difference has important consequences on the set of stable matchings, which does not necessarily contain an agent-optimal stable matching and may even be empty. We propose the Top-Down Bottom-Up (TDBU) algorithm – which iteratively identifies and eliminates agent-object pairs that are not part of any stable matching – to reduce the number of agent-object pairs that need to be considered in order to find a stable matching. We identify the existence of *bottlenecks* as the reason why the TDBU algorithm does not necessarily find the agent-optimal stable matching, and by extension the reason why such a matching may not exist. The Undominated Stable Matching (USM) algorithm uses these bottlenecks to conduct a depth-first search that finds an undominated stable matching whenever one exists and reports that the set of stable matchings is empty otherwise (Theorem 1). Our algorithms have several advantages over existing techniques. First, they find an undominated stable matching without needing to compute the whole set of stable matchings. Second, the TDBU algorithm is polynomial-time solvable and reduces the number of agent-object pairs that the USM algorithm has to consider in each round, allowing it to run quicker. The TDBU algorithm may also be used in conjunction with other techniques in order to reduce the size of the market that needs to be considered. Third, the basic ideas behind the TDBU and the USM algorithms are relatively intuitive and can be explained to participants.

We characterize stability to be the combination of three properties: K-boundedness, size-consistency and non-wastefulness. We show that any relaxation of stability that combines K-boundedness and non-wastefulness does not guarantee existence (Theorem 2). This result identifies an important trade-off that a market designer faces when confronted to a matching market with sizes: the size of claims can be bounded or waste can be eliminated but not both. If waste is tolerable, we show that the size of all claims can be limited to one unit while preserving size-consistency. As a non-wasteful relaxation of stability, we propose size-stability, which we characterize as the combination of size-consistency and non-wastefulness. We show that the set of size-stable matchings is nonempty, however it may contain matchings that are unfair to double-unit agents who may envy single-unit agents. We propose to compensate double-unit agents by selecting a size-stable matching that is undominated for them. We adapt the TDBU and USM algorithms in order to find such a matching (Theorem 3). We argue that the market designer's choice between eliminating waste and limiting the size of claims should depend on the application at stake and in particular on how tolerable waste and priority violations are relative to one another. On the one hand, refugee resettlement provides an example where waste is tolerable but the respect of priority is essential. On the other hand, tuition exchange is an application where wasting places may be worse than violating priorities.

The present paper opens several avenues for future research. First, the model can be extended to fit several real-world matching problems. Agents may have multidimensional requirements, have preferences over both objects and a number of units, or desire units of different objects. In all cases, it is possible to iteratively eliminate agent-object pairs by, for each object, giving a guarantee to agents with a high enough priority and rejecting those with a low enough priority. Protections can then be used to find an agent-undominated stable matching or establish that the set of stable matchings is empty. Size-stability can also be extended so long as it is possible to rank agents in terms of sizes. This is not possible when there are multiple services and an agent requires more units of a service than another agent but fewer units of another service. The trade-off between eliminating waste and bounding the size of claims naturally remains in all extensions.

Second, McDermid and Manlove (2010) have established that it is not possible in general to find a stable matching in polynomial time, but whether this extends to d-undominated size-stable matchings remains an open question. Such algorithm, if it exists, would constitute an improvement over the d-USSM algorithm presented in this paper. Otherwise, an interesting question is how to reach such a matching as efficiently as possible in order to make the concept applicable to markets as large as possible. Similarly, it would appear worthwhile to compare different techniques – and possible combinations of them – in order to minimize the amount of computations required to find a stable matching.

Finally, this paper has remained silent on the strategic properties of the proposed algorithms. While it is clear that neither the USM nor the d-USSM algorithm is strategy-proof, successfully misrepresenting one's preferences appears difficult given the amount of information required about priorities and other agents' preferences. In addition, the success encountered by the NRMP despite the fact that it uses a manipulable algorithm is reassuring. Nevertheless, a formal result on how close to strategy-proof these algorithms are would be of great help to implement these techniques in real matching markets. The structure provided by our algorithms may prove useful to such investigations.

References

Abdulkadiroglu, A., and T. Sönmez (2003): "School Choice: A Mechanism Design Approach," *American Economic Review*, 93(3), 729–747.

Biró, P., and T. Fleiner (2016): "Fractional Solutions for Capacitated NTU-Games, with Applications to Stable Matchings," *Discrete Optimization*, 22(A), 241–254.

Biró, P., T. Fleiner, and R. Irving (2016): "Matching Couples with Scarf's Algorithm," *Annals of Mathematics and Artificial Intelligence*, 77(3), 303–316.

Biró, P., D. F. Manlove, and I. McBride (2014): "The Hospitals/Residents Problem with Couples: Complexity and Integer Programming Models," in *Experimental Algorithms. SEA 2014. Lecture Notes in Computer Science*, ed. by J. Gudmundsoon, and J. Katajainen, vol. 8504, pp. 10–21. Springer, Cham.

Biró, P., and E. J. McDermid (2014): "Matching with Sizes (or Scheduling with Processing Set Restrictions," *Discrete Applied Mathematics*, 164(1), 61–67.

Che, Y.-K., and Y. Koh (2016): "Decentralized College Admissions," *Journal of Political Economy*, 124(5), 1295–1338.

Cseh, A., and B. C. Dean (2016): "Improved Algorithmic Results for Unsplittable Stable Allocation Problems," *Journal of Combinatorial Optimization*, 32(3), 657–671.

Dean, B. C., M. X. Goemans, and N. Immorlica (2006): "The Unsplittable Stable Marriage Problem," *IFIP International Conference on Theoretical Computer Science (IFIP TCS)*, pp. 65–75.

Delacrétaz, D., S. D. Kominers, and A. Teytelboym (2016): "Refugee Resettlement," Working Paper.

Dur, U. M., and M. U. Ünver (2017): "Two-Sided Matching via Balanced Exchange," Working Paper.

Echenique, F., and M. B. . Yenmez (2007): "A solution to matching with preferences over colleagues," *Games and Economic Behavior*, 59, 46–71.

Gale, D., and L. Shapley (1962): "College Admissions and the Stability of Marriage," *American Mathematical Monthly*, 69, 9–14.

Kennes, J., D. Monte, and N. Tumennasan (2014): “The Day Care Assignment: A Dynamic matching market,” *American Economic Journal: Microeconomics*, 6(4), 362–406.

Kojima, F. (2015): “Finding All Stable Matchings with Couples,” *Journal of Dynamics and Games*, 2(3/4), 321–330.

McDermid, E. J., and D. F. Manlove (2010): “Keeping partners together: algorithmic results for the hospitals/residents problem with couples,” *Journal of Combinatorial Optimization*, 19(3), 279–303.

Nguyen, T., and R. Vohra (2017): “Near-Feasible Stable Matchings with Couples,” Working paper.

Roth, A. E. (1984): “The Evolution of the Labor Market for Medical Interns and Residents: A Case Study in Game Theory,” *Journal of Political Economy*, 92, 991–1016.

Roth, A. E. (1991): “Natural Experiment in the Organization of Entry-Level Labor Markets: Regional Markets for New Physicians and Surgeons in the United Kingdom,” *American Economic Review*, 81(3), 415–440.

Roth, A. E. (2003): “The Origins, History, and Design of the Resident Match,” *Journal of the American Medical Association*, 289(7), 1585–1632.

Roth, A. E., and E. Peranson (1997): “The Effects of the Change in the NRMP Matching Algorithm,” *Journal of the American Medical Association*, 278(9), 729–732.

——— (1999): “The Redesign of the Matching Market for American Physicians: Some Engineering Aspects of Economic Design,” *American Economic Review*, 89, 748–780.

Roth, A. E., T. Sönmez, and M. U. Ünver (2004): “Kidney Exchange,” *Quarterly Journal of Economics*, 119, 457–488.

Roth, A. E., and M. Sotomayor (1990): *Two-sided matching: A study in game-theoretic modeling and analysis.* Cambridge University Press.

Scarf, H. (1967): “The core of an N person game,” *Econometrica*, 35, 50–69.

Yenmez, B. M. (2014): “College Admissions,” GSIA Working paper #2014-E24.

Appendix: Proofs

Proposition 1 is proved in the main text.

Proof of Proposition 2: (If) Let μ be a feasible, 1-bounded, size-consistent and non-wasteful matching. By definition, a double-unit agent and an object form a blocking pair if the former has a 2-unit claim to the latter. As μ is 1-bounded, there does not exist any blocking pair at μ that involves a double-unit agent. Because μ is size-consistent, single-unit agents do not envy any agent, therefore they do not have a claim to any assigned unit. Because μ is non-wasteful, single-unit agents do not have a claim to any unassigned unit. It follows that single-unit agents do not have any claim at all and are therefore not involved in any blocking pair. μ is stable as a result. $\square$

(Only If) Let μ be a feasible matching that is not 1-bounded. Then there exists an agent a who has a 2-unit claim to an object o. As $w_a \leq 2$, a and o form a blocking pair and μ is not stable. Next, let μ be a feasible and size-inconsistent matching. Then there exist two agents a and a' such that a envies a' and $w_a \leq w_{a'}$. By definition, the fact that a envies a' implies $\overline{o}_{a'}(\mu) \succ_a \overline{o}_a(\mu)$ and $a \rhd_{\overline{o}_{a'}(\mu)} a'$, therefore a has a $w_{a'}$-unit claim to $\overline{o}_{a'}(\mu)$. As $w_a \leq w_{a'}$, a has a w_a-unit claim to $\overline{o}_{a'}(\mu)$, which means that a and $\overline{o}_{a'}(\mu)$ form a blocking pair, hence μ is not stable. Finally, let μ be a feasible and wasteful matching. Then there exists an agent a who has a claim to w_a unassigned units of an object o. This directly implies that a has a w_a-unit claim to o, as a result (a, o) constitutes a blocking pair and μ is not stable. ■

Proof of Lemma 1: Suppose that $\mathcal{G}_{(a,o)}(X) \leq q_o$ and that, at some matching $\mu \subseteq X$, a is matched to a less preferred object o' (that is, $o \succ_a o'$). As $\mu \subseteq X$, $\hat{A}_{(a,o)}(\mu) \subseteq \hat{A}_{(a,o)}(X)$ so

$$\sum_{a' \in \hat{A}_{(a,o)}(\mu)} w_{a'} \leq \sum_{a' \in \hat{A}_{(a,o)}(X)} w_{a'} \leq q_o - w_a.$$

Then (a, o) constitutes a blocking pair and μ is not stable. ■

Lemma 2 is proved in the main text.

Proof of Lemma 3: (G) As $X \subseteq X'$, $\hat{A}_{(a,o)}(X) \subseteq \hat{A}_{(a,o)}(X')$. Therefore

$$\mathcal{G}_{(a,o)}(X) = w_a + \sum_{a' \in \hat{A}_{(a,o)}(X)} w_{a'} \leq w_a + \sum_{a' \in \hat{A}_{(a,o)}(X')} w_{a'} = \mathcal{G}_{(a,o)}(X'). \quad \square$$

(R) For any a' such that $a' \rhd_o a$, a' is an element of $E_{(a,o)}(X) \cup (\hat{A}_{(a,o)}(X) \cap \overline{A}_o(X))$ if and only if he does not contest any object at X that he prefers to o. As $X \subseteq X'$, a contests at X' all the objects that he contests at X, therefore

$$E_{(a,o)}(X') \cup (\hat{A}_{(a,o)}(X') \cap \overline{A}_o(X')) \subseteq E_{(a,o)}(X) \cup (\hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)).$$

If follows that

$$\begin{aligned} \mathcal{R}_{(a,o)}(X') &= w_a + \sum_{a' \in \hat{A}_{(a,o)}(X') \cap \overline{A}_o(X')} w_{a'} + \sum_{a' \in E_{(a,o)}(X')} w_{a'} \\ \leq \quad & w_a + \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'} + \sum_{a' \in E_{(a,o)}(X)} w_{a'} = \mathcal{R}_{(a,o)}(X). \quad \blacksquare \end{aligned}$$

Lemma 4 is proved in the main text.

Proof of Lemma 5: We have shown in the main text that (a, o) is not an element of any stable matching contained in X if it satisfies Condition (i) or (ii). It remains to show that this is also true if it satisfies Condition (iii). Suppose that $a \in S$, $\mathcal{R}_{(a,o)}(X) > q_o$ and there exists a stable matching $\mu \in \mathbb{S}$ that contains (a, o). We show in turn that each part of Condition (iii) yields a contradiction.

Suppose towards a contradiction the existence of a single-unit agent $s \in \overline{S}_o(X)$ such that $\mathcal{R}_{(a,o)}(X) > \mathcal{R}_{(s,o)}(X) \geq q_o$. By Lemma 4, a is envied at μ: $E_{(a,o)}(\mu) \neq \varnothing$. By size-consistency, μ also contains (s, o) and by Lemma 2 s is not envied at μ: $E_{(s,o)}(\mu) = \varnothing$. By Lemma 3, $\mathcal{R}_{(s,o)}(\mu) \geq \mathcal{R}_{(s,o)}(X) \geq q_o$. Combining the last two results with the definition of a rejection number implies

$$w_s + \sum_{a' \in \hat{A}_{(s,o)}(X) \cap \overline{A}_o(X)} w_{a'} \geq q_o.$$

Therefore all q_o units of o are assigned at μ to agent with a higher priority than a. It follows that μ is not feasible, a contradiction.

Suppose now, also towards a contradiction, that $\mathcal{R}_{(a,o)}(X) - q_o$ is odd and for all $s \in S_o(X) \setminus \overline{S}_o(X)$, $\mathcal{R}_{(s,o)}(X) \geq q_o$. The difference between $\mathcal{R}_{(a,o)}(\mu)$ and $\mathcal{R}_{(a,o)}(X)$ is obtained by adding up the size of all agents with a higher priority than a who contest o as a subsequent choice at X and are matched to o or a less preferred object at μ. Suppose that an agent $s \in S_o(X) \setminus \overline{S}_o(X)$ – that is a single-unit agent who contests o as a subsequent choice at X – is matched to o or a less preferred object at μ. By assumption, $\mathcal{R}_{(s,o)}(X) \geq q_o$, therefore μ is not stable by our first argument. The difference between $\mathcal{R}_{(a,o)}(\mu)$ and $\mathcal{R}_{(a,o)}(X)$ is therefore even as it is obtained by adding up the size of double-unit agents. Then $\mathcal{R}_{(a,o)}(\mu) - q_o$ is odd by assumption. By Lemma 2, $\sum_{a' \in A_o(\mu)} w_{a'} = q_o$ and a has the lowest priority among agents matched to o. Then $w_a + \sum_{a' \in \hat{A}_{(a,o)}(\mu)} w_{a'} = q_o$ so $\mathcal{R}_{(a,o)}(\mu) = q_o + \sum_{a' \in E_{(a,o)}(\mu)} w_{a'}$. By size-consistency, $E_{(a,o)}(\mu) \subseteq D$ so $\mathcal{R}_{(a,o)}(\mu) - q_o$ is even, a contradiction. ■

Proof of Proposition 3: We proceed by induction. Let $K = 1, 2, \ldots$ be the last step of the algorithm. If X_K is complete, then $\varphi(X) = X_K$. If X_k is incomplete, it does not include any (stable) matching and $\varphi(X) = \varnothing$. In both cases $\varphi(X)$ includes all stable matchings included in X_K. It remains to show that X_K includes all stable matchings included in X. The result is trivial if $K = 1$. If $K > 1$, suppose towards an inductive argument that for some $k = 1, 2, \ldots, K-1$, X_k includes all stable matchings included in X_k. Our induction hypothesis holds trivially for $k = 1$ as $X_1 = X$. By Lemmas 1 and 5, X_{k+1} includes all stable matchings that are included in X_k, hence it includes all stable matchings included in X. By induction, X_K includes all stable matchings included in X. ■

Proof of Proposition 4: Let K be the number of steps that the TDBU algorithm lasts when applied to $A \times O$ and, for all $k = 1, \ldots, K$, let X^k denote the set of agent-object pairs that enters Step k. Then $X^1 = A \times O$ and X^k is complete for all $k = 1, \ldots, K-1$. If X^K is incomplete, then an object o rejects an agent a even though $\mathcal{G}_{(a,o)}(X^{K-1}) \leq q_o$, a contradiction.

We prove the remaining properties by induction. Suppose that for some $k = 1, \ldots, K-1$, X^k includes all stable matchings and $\overline{\mu}(\varphi(X^k))$ is 1-bounded, size-consistent and non-wasteful. $X^1 = A \times O$ trivially satisfies these properties. We show that X^{k+1} satisfies them as well. By Condition (i) of Lemma 5, if an object rejects an agent it also rejects all agents with a lower priority and a weakly larger size, therefore $\overline{\mu}(\varphi(X^{k+1}))$ is size-consistent. An object does not reject an agent unless the latter's rejection number exceeds the quota. By the induction hypothesis, at X^k, at most one single-unit agent has a nonempty envy set for o and that set only contains double-unit agents. Additionally, all double-unit agents have an empty envy set for o at X^k. Therefore, if o rejects a, then at least $q_o - w_a$ units of o are tentatively assigned to agents with a higher priority, which means that $\overline{\mu}(X^{k+1})$ is non-wasteful. By Condition (iii) of Lemma 5, for any object o, all but one single-units agent with a nonempty envy set for o are rejected and by Condition (i) all double-unit agents with a nonempty envy-set are rejected. As $\overline{\mu}(\varphi(X^{k+1}))$ is non-wasteful, it is 1-bounded. Finally, by Lemmas 1 and 5, any pair that is eliminated in Round k is not an element of any stable matching included in X^k, therefore X^{k+1} includes all stable matchings by the induction hypothesis.

By induction, we conclude that X^K includes all stable matchings and $\overline{\mu}(X^k)$ is 1-bounded, size-consistent and non-wasteful. As X^K is also complete, $\varphi(A \times O) = X^k$. If $\overline{\mu}(\varphi(A \times O))$ is feasible, it is stable by Proposition 2. As $\varphi(A \times O)$ includes all stable matchings, $\overline{\mu}(\varphi(A \times O))$ dominates all other stable matchings and is consequently the optimal stable matching. If $\overline{\mu}(\varphi(A \times O))$ is not feasible, then by definition it is not stable. However, as $\varphi(A \times O)$ includes all stable matchings, $\overline{\mu}(\varphi(A \times O))$ dominates all stable matchings. ■

Proof of Lemma 6: Suppose towards a contradiction the existence of an agent two agents $a \in \overline{A}_o(X)$ and $a' \in E_{(a,o)}(X)$ with $w_a \geq w_{a'}$. (a, o) satisfies Condition (i) of Lemma 5 so o rejects a at X, which contradicts the assumption that X is irreducible. Then $a \in \overline{A}_o(X)$ and $E_{(a,o)}(X) \neq \varnothing$ implies $a \in S$ and $E_{(a,o)}(X) \subseteq D$.

To complete the proof, suppose – again towards a contradiction – the existence of two or more distinct single-unit agents in that situation. That is, suppose there exist $s, s' \in \overline{S}_o(X)$ ($s \neq s'$) such that $E_{(s,o)}(X) \neq \varnothing$ and $E_{(s',o)}(X) \neq \varnothing$. Without loss of generality, let $s \rhd_o s'$. Then (s', o) satisfies the second part of Condition (i), contradicting again the irreducibility of X. ■

Proof of Lemma 7: If $\sum_{a \in \overline{A}_o(X)} w_a > q_o$, then by definition there exists $s \in \overline{A}_o(X)$ such that $\mathcal{R}_{(s,o)}(X) > q_o$. As X is irreducible, o does not reject s at X, therefore (s, o) does not satisfy any of the three conditions of Lemma 5. The fact that it the pair does not satisfy Condition (ii) directly implies $s \in S$, hence $s \in \overline{S}_o(X)$. The fact that the pair does not satisfy Condition (iii) implies that s is the only agent who contests o as his top choice and has a rejection number larger than the quota. Therefore, $\sum_{a \in \overline{A}_o(X)} w_a = q_o + 1$, which implies the first part of the statement.

We also conclude from our above observation that s has the lowest priority for o among the elements of $\overline{A}_o(X)$, therefore

$$w_s + \sum_{a \in \hat{A}_{(s,o)}(X) \cap \overline{A}_o(X)} w_a = q_o + 1.$$

Let $d \in \overline{A}_o(X)$ be the agent with the second lowest priority among the elements of $\overline{A}_o$. Such agent exists as $\sum_{a \in \overline{A}_o(X)} w_a = q_o + 1$, $q_o \geq 1$ and $s \in S$ imply that s is not the only element of $\overline{A}_o(X)$. Then

$$w_d + \sum_{a \in \hat{A}_{(d,o)}(X) \cap \overline{A}_o(X)} w_a = q_o.$$

so $\mathcal{R}_{(d,o)}(X) \geq q_o$. Then $d \in D$ since (s, o) does not satisfy Condition (iii) of Lemma 5. In turn, o does not reject d at X so that pair does not satisfy Condition (ii). We conclude that $E_{(d,o)}(X) = \varnothing$, hence $\mathcal{R}_{(d,o)}(X) = q_o$. Finally,

$$\mathcal{R}_{(s,o)}(X) = w_s + \sum_{a \in \hat{A}_{(s,o)}(X) \cap \overline{A}_o(X)} w_a + \sum_{a \in E_{(s,o)}(X)} w_a = q_o + 1 + \sum_{a \in E_{(s,o)}(X)} w_a.$$

As (s,o) does not satisfy Condition (i), $E_{(s,o)}(X) \subseteq D$, hence $\mathcal{R}_{(s,o)}(X) - q_o$ is odd. As a consequence, (s,o) satisfies Condition (iii) unless there exists $s' \in S_o(X) \setminus \overline{S}_o(X)$ with $R_{(s',o)}(X) < q_o$. Then $s' \rhd_o d \rhd_o s$, which implies the second and last part of the statement. ■

Proof of Lemma 8: Let $\mu \in \mathbb{S}$ be a stable matching such that $(d,o) \in \mu \subseteq X$. We begin by showing that $(\hat{A}_{(d,o)}(X) \cap \overline{A}_o(X)) \cup \{d\} = A_o(\mu)$, that is the agents matched to o at μ are exactly those who contest o as their top choice at X and have a weakly higher priority than d. Suppose towards a contradiction the existence of an agent $a \in \hat{A}_{(d,o)}(X) \cap \overline{A}_o(X)$ such that $a \notin A_o(\mu)$. As $\mu \subseteq X$, $o \succ_a \overline{o}_a(\mu)$. Additionally, $(d,o) \in \mu$ and $a \rhd_o d$, therefore a envies d at μ. μ is consequently not size-consistent, hence not stable since $d \in D$, a contradiction. Then $(\hat{A}_{(d,o)}(X) \cap \overline{A}_o(X)) \cup \{d\} \subseteq A_o(\mu)$. As $(d,o) \in B^*(X)$, $\mathcal{R}_{(d,o)}(X) = q_o$ by definition and as X is irreducible, $E_{(d,o)}(X) = \varnothing$. (Otherwise (d,o) would satisfy Condition (i) of Lemma 5 and o would reject d at X.) Therefore

$$\mathcal{R}_{(d,o)}(X) = w_d + \sum_{\hat{A}_{(d,o)}(X) \cap \overline{A}_o(X)} w_a = q_o.$$

If $(\hat{A}_{(d,o)}(X) \cap \overline{A}_o(X)) \cup \{d\} \subset A_o(\mu)$, then $\sum_{a \in A_o(\mu)} w_a > q_o$, violating feasibility. We conclude that $(\hat{A}_{(d,o)}(X) \cap \overline{A}_o(X)) \cup \{d\} = A_o(\mu)$.

Consider now $(a,o') \in P_{(d,o)}(X)$. If $\overline{o}_a(X) \succ_a o$, then $a \notin (\hat{A}_{(d,o)}(X) \cap \overline{A}_o(X)) \cup \{d\} = A_o(\mu)$. If on the other hand $\overline{o}_a(X) = o$, then $o \succ_a o'$. $(a,o') \in \mu$ implies $o \succ_a o' = \overline{o}_a(\mu)$. As $(d,o) \in \mu$ and $a \rhd_o d$, a envies d, which violates size-consistency. ■

Proposition 5 is proved in the main text.

Proof of Lemma 9: We prove each part of the statement in turn. Consider a pair $(a,o) \in X$ such that $\mathcal{M}_{(a,o)}(X) = 0$. Then either $\overline{o}_a(X) \succeq_a o$, in which case (a,o) does not constitute a blocking pair at $\overline{\mu}(X)$, or

$$q_o - (w_a - 1) - \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'} \leq 0.$$

Notice that $\hat{A}_{(a,o)}(X) \cap \overline{A}_o(X) = \hat{A}_{(a,o)}(\overline{\mu}(X))$. Therefore

$$\sum_{a' \in \hat{A}_{(a,o)}(\overline{\mu}(X))} w_{a'} \geq q_o - (w_a - 1),$$

which is equivalent to

$$\sum_{a' \in \hat{A}_{(a,o)}(\overline{\mu}(X))} w_{a'} > q_o - w_a.$$

It follows that a does not have a w_a-unit claim to o at $\overline{\mu}(X)$, hence (a, o) does not constitute a blocking pair of $\overline{\mu}(X)$. Suppose now in contrast that $\mathcal{M}_{(a,o)}(X) > 0$. Then $\overline{o}_a(X) \succ_a o$ and

$$q_o - (w_a - 1) - \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'} > 0.$$

By an analogous reasoning to the one above, the latter inequality is equivalent to

$$\sum_{a' \in \hat{A}_{(a,o)}(\overline{\mu}(X))} w_{a'} \leq q_o - w_a.$$

It follows that a has a w_a-unit claim to o at $\overline{\mu}(X)$, hence (a, o) constitutes a blocking pair of $\overline{\mu}(X)$. Combining our two results, we conclude that $(a, o) \in X$ is a blocking pair of $\overline{\mu}(X)$ if and only if $\mathcal{M}_{(a,o)}(X) > 0$. By definition, there exists $(a, o) \in X$ such that $\mathcal{M}_{(a,o)}(X) > 0$ if and only if $\sum_{o \in O} \mathcal{L}_o > 0$, therefore $\overline{\mu}(X)$ has at least one blocking pair if and only if $\sum_{o \in O} \mathcal{L}_o > 0$, which proves the first part of the statement.

In order to prove the second part of the statement, suppose that $\mathcal{L}_o(X) > 0$ for some $o \in O$. Then there exists $a \in A$ such that $\mathcal{M}_{(a,o)}(X) = \mathcal{L}_o(X) > 0$. Then

$$\mathcal{L}_o(X) = q_o - (w_a - 1) - \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'},$$

which is equivalent to

$$q_o - \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'} = \mathcal{L}_o(X) + (w_a - 1). \tag{1}$$

If $\overline{A}_o(X) \subseteq \hat{A}_{(a,o)}(X)$, that is if all agents who contest o as their top choice have a higher priority than a, then

$$q_o - \sum_{a' \in \overline{A}_o(X)} w_{a'} = q_o - \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'} = \mathcal{L}_o(X) + (w_a - 1) \geq \mathcal{L}_o(X)$$

and the second part of the statement is satisfied since $\overline{A}_o(X) = A_o(\overline{\mu}(X))$ by definition. Otherwise, there exists $\hat{a} \in \overline{A}_o(X)$ such that $a \rhd_o \hat{a}$. As $o \succ_a \overline{o}_a(X)$, $a \in E_{(\hat{a},o)}(X)$. We first consider the left side of (1). By Lemma 6, $\hat{a} \in S$ and $\hat{a}$ is the only agent in $\overline{A}_o(X)$ such that the envy set at X of the pair he forms with o is nonempty. This directly implies that $\hat{a}$ is the only agent in $\overline{A}_o(X)$ who has a lower priority than a for o, therefore $A_o(\overline{\mu}(X)) = \overline{A}_o(X) = (\hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)) \cup \{\hat{a}\}$. Then

$$\sum_{a' \in A_o(\overline{\mu}(X))} w_{a'} = w_{\hat{a}} + \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'} = 1 + \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'}$$

so

$$q_o - \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'} = q_o + 1 - \sum_{a' \in A_o(\overline{\mu}(X))} w_{a'}.$$

We now turn to the right side of (1). Lemma 6 implies that $E_{(\hat{a},o)}(X) \subseteq D$, therefore $a \in D$. This in turn implies that

$$\mathcal{L}_o(X) + (w_a - 1) = \mathcal{L}_o(X) + 1.$$

Combining our last two equations with (1) yields

$$q_o + 1 - \sum_{a' \in A_o(\overline{\mu}(X))} w_{a'} = \mathcal{L}_o(X) + 1,$$

hence

$$q_o - \sum_{a' \in A_o(\overline{\mu}(X))} w_{a'} = \mathcal{L}_o(X)$$

and the second part of the statement also holds in this case. ■

Proof of Lemma 10: Consider a pair $(d,o) \in D \times O$ such that $(d,o) \in B(X)$. Then there exists $s \in S$ such that $((d,o),(s,o))$ is a bottleneck of X and $\sum_{a \in \overline{A}_o(X)} w_a = q_o + 1$ and no agent has a claim to o so $\mathcal{L}_o(X) = 0$. Suppose towards a contradiction that $\sum_{a \in A_o(\mu)} w_a \leq q_o - 1$. If $s \in S_o(\mu)$, then d has a 2-unit claim to o, contradicting the stability of μ. If $s \notin S_o(\mu)$, then s has a claim to the unassigned unit of o, again a contradiction.

Consider next a pair $(d,o) \in D \times O$ such that $(d,o) \in B^*(X) \setminus B(X)$, that is (d,o) is a phantom bottleneck of X. Then $\sum_{a \in \overline{A}_o(X)} w_a = q_o$. If $\sum_{a \in A_o(\mu)} w_a \leq q_o - 2$, the set

$\overline{A}_o(X) \setminus A_o(\mu)$ is nonempty and all agents in it have a claim to the two unassigned units of o so μ is not stable.

Consider next an object $o \in O$ such that $\mathcal{L}_o(X) > 0$. Then by Lemma 9, $\sum_{a \in \overline{A}_o(X)} w_a \leq q_o - \mathcal{L}_o(X)$. Then if

$$\sum_{a \in A_o(\mu)} w_a - \sum_{a \in \overline{A}_o(X)} w_a < \mathcal{L}_o(X),$$

$\sum_{a \in A_o(\mu)} w_a < q_o$ and an agent has a claim to o, hence μ is not stable.

Finally, consider an object $o \in O$ such that $\mathcal{L}_o(X) = 0$ and o is not involved in a bottleneck or phantom bottleneck. If $\sum_{a \in A_o(\mu)} w_a < \sum_{a \in \overline{A}_o(X)} w_a$, then there exists an agent in $\overline{A}_o(X) \setminus A_o(\mu)$ who has a claim to o and μ is not stable. ■

Proof of Proposition 6: By Lemma 9, $\overline{\mu}(X)$ does not have any blocking pair if and only if $\mathcal{L}(X) = 0$. By Lemma 7, $\overline{\mu}(X)$ is feasible if and only if it X does not have any bottleneck. The first part of the statement immediately follows.

If $\mathcal{E}(X) > 0$ then either there exists $o \in O$ such that $\mathcal{E}_o(X) = \infty$ or, for all $o \in O$,

$$\mathcal{E}_o(X) = \begin{cases} -1 & \text{if there exists } d \in D \text{ such that } (d, o) \in B^*(X) \\ \mathcal{L}_o(X) & \text{otherwise.} \end{cases}$$

In the former case, $\mathbb{S} \cap 2^X = \varnothing$ by Corollary 3. In the latter case,

$$\mathcal{E}(X) = \sum_{o \in O} \mathcal{E}_o(X) = \left(\sum_{o \in O} \mathcal{L}_o(X) \right) - |B^*(X)| = \mathcal{L}(X) - |B^*(X)|,$$

where the second equality makes use of the fact that $\mathcal{L}_o(X) = 0$ whenever an object o is involved in either a bottleneck or a phantom bottleneck (Lemma 10). Then by assumption,

$$\mathcal{L}(X) - |B^*(X)| > 0.$$

Towards a contradiction, suppose the existence of a matching $\mu \in \mathbb{S} \cap 2^X$. On the one hand,

$$\sum_{o \in O} \left(\sum_{a \in A_o(\mu)} w_a - \sum_{a \in \overline{A}_o(X)} w_a \right) \geq \left(\sum_{o \in O} \mathcal{L}_o \right) - |B^*(X)| = \mathcal{L}(X) - |B^*(X)|$$

by Lemma 10. On the other hand, all agents are by definition matched to exactly one object at any matching, therefore

$$\sum_{o\in O}\sum_{a\in A_o(\mu)} w_a = \sum_{o\in O}\sum_{a\in \overline{A}_o(X)} w_a = \sum_{a\in A} w_a.$$

This implies

$$\sum_{o\in O}\left(\sum_{a\in A_o(\mu)} w_a - \sum_{a\in \overline{A}_o(X)} w_a\right) = 0.$$

Combining our last two results yields $\mathcal{L}(X) - |B^*(X)| \leq 0$, a contradiction. ■

Theorem 1 is directly implied by Proposition 7.

Proof of Proposition 7: Let K be the number of rounds that the USM algorithm lasts and for all $k = 1, \ldots K$, let X_k be the set of pairs that enters Round k. Then $X_1 = A \times O$.

If $\varphi(X_K) = \varnothing$ or $\mathcal{E}(\varphi(X_K)) > 0$, then the algorithm is in Case 3 in Round K so $\mu^*_\theta = \varnothing$. $\varphi(X_K)$ does not include any stable matching, therefore there does not exist any stable matching containing all pairs in Z_K. As the algorithm ends in Round K, $Z_K = \varnothing$, which implies that there does not exist any stable matching.

If $\varphi(X_K) \neq \varnothing$ and $\mathcal{E}(\varphi(X_k)) \leq 0$, the algorithm is in Case 1 in Round K since it ends in that round. Therefore, $\mathcal{L}(\varphi(X_K)) = |B(\varphi(X_K)| = 0$. By Lemma 7, $\overline{\mu}(\varphi(X_K)$ is feasible and by Lemma 9 it does not contain any blocking pair. Combining these two results, $\overline{\mu}(\varphi(X_K))$ is stable. $\varphi(X_K)$ includes all stable matchings that themselves include Z_K, therefore $\overline{\mu}(\varphi(X_K))$ dominates all other stable matchings that include Z_k. In addition, any stable matching that does not include Z_k makes at least one agent with a protected pair worse-off, which means that $\overline{\mu}(\varphi(X_K))$ is not d-dominated by any stable matching. We conclude that $\mu^*_\theta = \overline{\mu}(\varphi(X_K))$ is a d-undominated stable matching. ■

Proof of Theorem 2: For any $K = 1, 2, \ldots,$ if there exists a matching market where the set of K-bounded and non-wasteful matchings is empty, then by definition the set of $K-1$-bounded and non-wasteful matchings that is also empty in that market. It is therefore sufficient to show that for any odd positive integer K, there exists a matching market where the set of K-bounded and non-wasteful matchings is empty. Let K be an odd positive integer and consider the following example.

Example 7 (No K-bounded non-wasteful Matching)**.** There are $3K$ single-unit agents $s_1, \ldots, s_K, \hat{s}_1, \ldots, \hat{s}_{2K}$, $(K+1)/2$ double-unit agent $d_1, \ldots, d_{(K+1)/2}$ and two non-null objects o_1 and o_2. The preferences, priorities and quotas are

$$\begin{array}{llllll}
\succ_{s_1}: & o_1, o_2, \emptyset & \succ_{\hat{s}_1}: & o_2, o_1, \emptyset & \succ_{d_1}: & o_1, \emptyset, o_2 \\
\vdots & & \vdots & & \vdots & \\
\succ_{s_K}: & o_1, o_2, \emptyset & \succ_{\hat{s}_{2K}}: & o_2, o_1, \emptyset & \succ_{d_{(K+1)/2}}: & o_1, \emptyset, o_2
\end{array}$$

$$\begin{array}{ll}
\rhd_{o_1} : \hat{s}_1, \ldots, \hat{s}_{2K}, d_1, \ldots, d_{(K+1)/2}, s_1, \ldots, s_K & q_{o_1} = 2K \\
\rhd_{o_2} : s_1, \ldots, s_K, \hat{s}_1, \ldots, \hat{s}_{2K}, d_1, \ldots, d_{(K+1)/2} & q_{o_2} = 2K
\end{array}$$

Towards a contradiction, suppose the existence of a K-bounded and non-wasteful matching μ in this market. First, consider the case where all double-unit agents are matched to o_1: (d_1, o_1), $\ldots$, $(d_{(K+1)/2}, o_1) \in \mu$. Towards an inductive argument, suppose that

$$\{(d_1, o_1), \ldots, (d_{(K+1)/2}, o_1), (\hat{s}^1, o_1), \ldots, (\hat{s}^K, o_1), (s^1, o_2), \ldots, (s^K, o_2)\} \subseteq \mu,$$

where $0 \leq n \leq K-1$, $\hat{s}^1, \ldots, \hat{s}^K$ are n distinct elements of $\{\hat{s}_1, \ldots, \hat{s}_{2K}\}$ and $s^1, \ldots, s^K$ are n distinct elements of $\{s_1, \ldots, s_K\}$. In words, at least n agents in $\{\hat{s}_1, \ldots, \hat{s}_{2K}\}$ are matched to o_1 (their second preference) and at least n agents in $\{s_1, \ldots, s_K\}$ are matched to o_2 (their second preference). Then $K+1$ units of o_1 are assigned to double-unit agents and n units are assigned to $\hat{s}^1, \ldots, \hat{s}^K$. As μ is feasible by assumption, at most $K-(n+1)$ units of o_1 may be assigned to the $K-n$ agents in $\{s_1, \ldots, s_K\} \setminus \{s^1, \ldots, s^K\}$. Consequently, there exists an agent $s^{K+1} \in \{s_1, \ldots, s_K\} \setminus \{s^1, \ldots, s^K\}$ who is not matched to o_1. At most $K-1$ units of o_2 may be assigned to agents who have a higher priority than s^{K+1} for o_2, therefore if s^{K+1} is matched to the null object, he envies at least $K+1$ units of o_2, contradicting the assumption that μ is K-bounded. s^{K+1} is then necessarily matched to o_2 ($(s^{K+1}, o_2) \in \mu$). In turn, $n+1$ units of o_2 are assigned to $s^1, \ldots, s^K, s^{K+1}$ and by feasibility at most $2K-(n+1)$ units may be assigned to the $2K-n$ agents in $\{\hat{s}_1, \ldots, \hat{s}_{2K}\} \setminus \{\hat{s}^1, \ldots, \hat{s}^K\}$. Consequently, there exists an agent $\hat{s}^{K+1} \in \{\hat{s}_1, \ldots, \hat{s}_{2K}\} \setminus \{\hat{s}^1, \ldots, \hat{s}^K\}$ who is not matched to o_2. If $\hat{s}^{K+1}$ is matched to the null object, he envies the $K+1$ units of o_1 assigned to the double-unit agents and μ is not K-bounded. $\hat{s}^{K+1}$ is therefore matched to o_1 ($(\hat{s}^{K+1}, o_1) \in \mu$). We conclude that

$$\{(d_1, o_1), \ldots, (d_{(K+1)/2}, o_1), (\hat{s}^1, o_1), \ldots, (\hat{s}^K, o_1), (\hat{s}^{K+1}, o_1), (s^1, o_2), \ldots, (s^K, o_2), (s^{K+1}, o_2)\} \subseteq \mu.$$

By induction, at least K agents in $\{\hat{s}_1, \ldots, \hat{s}_{2K}\}$ are matched to o_1. Then K units of o_1 are assigned to these agents and $K+1$ units are assigned to the double-unit agents. μ is not feasible as a consequence, a contradiction.

It remains to show that a contradiction also arises in the case where for some $i = 1, \ldots, (K+1)/2$, d_i is not matched to o_1. If d_i is matched to o_2, he envies at least two unassigned units of the null object, contradicting the assumption that μ is non-wasteful. Therefore, d_i is matched to the null object: $(d_i, \emptyset) \in \mu$. If none of the agents $\hat{s}_1, \ldots, \hat{s}_{2K}$ are matched to o_1, then only double-unit agents may be matched to o_1 and have a higher priority than d_i. As there are $(K-1)/2$ double-unit agents other than d_i, at most $K-1$ units of o_1 are assigned to agents with a higher priority than d_i. Consequently, d_i envies at least $K+1$ units of o_1, contradicting the assumption than μ is K-bounded. Then there exists an agent $\hat{s}^0 \in \{\hat{s}_1, \ldots, \hat{s}_{2K}\}$ who is matched to o_1: $(\hat{s}^0, o_1) \in \mu$. Towards an inductive argument, suppose that

$$\{(d_i, \emptyset), (\hat{s}^0, o_1), (\hat{s}^1, o_1), \ldots, (\hat{s}^K, o_1), (s^1, o_2), \ldots, (s^K, o_2)\} \in \mu,$$

where $0 \leq n \leq K-1$, $\hat{s}^0, \hat{s}^1, \ldots, \hat{s}^K$ are $n+1$ distinct elements of $\{\hat{s}_1, \ldots, \hat{s}_{2K}\}$ and $s^1, \ldots, s^K$ are n distinct elements of $\{s_1, \ldots, s_K\}$. As at least $n+1$ agents out of $\hat{s}_1, \ldots, s_{2K}$ are matched to o_1, at most $2K-(n+1)$ of them are matched to o_2. Additionally, no double-unit agent is matched to o_2 as he would envy at least two units of the null object, contradicting the assumption that μ is non-wasteful. If exactly n agents out of $s_1, \ldots, s_K$ are matched to o_2, then at most $2K-(n+1)+n = 2K-1$ units of o_2 are assigned altogether. It directly follows that at least one unit of o_2 is unassigned. As $\hat{s}^0, \hat{s}^1, \ldots, \hat{s}^K$ envy that unit, μ is wasteful, a contradiction. There consequently exists an agent $s^{K+1} \in \{s_1, \ldots, s_K\} \setminus \{s^1, \ldots, s^K\}$ who is matched to o_2: $((s^{K+1}, o_2) \in \mu)$. In turn, the fact that at least $n+1$ agents out of $s_1, \ldots, s_K$ are matched to o_2 implies that at most $K-(n+1)$ of them are matched to o_1. Additionally, at most $(K-1)/2$ double-unit agents are matched to o_1 since d_i is not, therefore at most $K-1$ units of o_1 are assigned to them. If exactly $n+1$ agents out of $\hat{s}_1, \ldots, \hat{s}_{2K}$ are matched to o_1, then at most $K-(n+1)+K-1+n+1 = 2K-1$ units of o_1 are assigned altogether. This directly implies that at least one unit of o_1 is unassigned. As $s^1, \ldots, s^K, s^{K+1}$ envy that unit, μ is wasteful, a contradiction. There consequently exists an agent $\hat{s}^{K+1} \in \{\hat{s}_1, \ldots, \hat{s}_{2K}\} \setminus \{\hat{s}^0, \hat{s}^1, \ldots, \hat{s}^K\}$ who is matched to o_1: $((\hat{s}^{K+1}, o_1) \in \mu)$. We conclude that

$$\{(d_i, \emptyset), (\hat{s}^0, o_1), (\hat{s}^1, o_1), \ldots, (\hat{s}^K, o_1), (\hat{s}^{K+1}, o_1), (s^1, o_2), \ldots, (s^K, o_2), (s^{K+1}, o_2)\} \in \mu.$$

By induction, at least $K+1$ agents out of $\hat{s}_1, \ldots, \hat{s}_{2K}$ are matched to o_1, which directly implies that at most $K-1$ of them are matched to o_2. As none of the double-unit agents are matched to o_2, it follows that at most $2K-1$ units of o_2 are assigned altogether, hence at least one is unassigned. Those of $\hat{s}_1, \ldots, \hat{s}_{2K}$ who are matched to o_1 envy that unit, hence

μ is wasteful, a contradiction. ■

Proof of Proposition 8: Let K be the number of rounds that the PFDA algorithm lasts and for all $k = 1, \ldots, K$, let X_k be the set of pairs that enters Round k. Then $X_1 = A \times O$ and $\overline{\mu}(X_K) = \mu^{\text{PFDA}}$. Trivially, $\overline{\mu}(X_1) = \overline{\mu}(A \times O)$ is 1-bounded and such that $E_{(a,o)}(X_1) = \varnothing$ for all $(a, o) \in X_1 = A \times O$. Towards an inductive argument, suppose that this is also the case for some $k = 1, \ldots, K-1$. That is, $\overline{\mu}(X_k)$ is 1-bounded and $E_{(a,o)}(X_k) = \varnothing$. By construction, if an object has rejected an agent in some Round $j \leq k$, then all agents with a lower priority who propose to that object in Round k are also rejected, therefore $E_{(a,o)}(X_{k+1}) = \varnothing$ for all $(a, o) \in X_{k+1}$ such that $a \in \overline{A}_o(X_{k+1})$. Then any claim at $\overline{\mu}_{k+1}$ involves unassigned units. By the induction hypothesis, if o has rejected at least one agent up to Round $k-1$, then $\sum_{a \in \overline{A}_o(X_k)} w_a \geq q_o - 1$. Then either o has not rejected any object up to Round $k-1$ or $\sum_{a \in \overline{A}_o(X_k)} w_a \geq q_o - 1$. In either case, o does not reject any agent in Round k unless the total size of agents with a higher priority is at least $q_o - 1$. It follows that $\sum_{a \in \overline{A}_o(X_{k+1})} w_a \geq q_o - 1$, hence agents have a claim to at most one unit of any object at $\overline{\mu}(X_{k+1})$, which means that the latter is 1-bounded.

By induction, $\mu^{\text{PFDA}} = \overline{\mu}(X_K)$ is 1-bounded and such that $E_{(a,o)}(X_K) = \varnothing$ for all $(a, o) \in X_K$. The latter implies that μ^{PFDA} is size-consistent. As the algorithm ends in Round K, no agent is rejected so μ^{PFDA} is feasible. ■

Proof of Proposition 9: (If) Let μ be a feasible, size-consistent and non-wasteful matching. Suppose towards a contradiction the existence of a strong blocking pair $(a, o) \in A \times O$ of μ. Then $o \succ_a \overline{o}_a(\mu)$. If $a \in S$, then $\sum_{a' \in \hat{A}_{(a,o)}(\mu)} w_{a'} \leq q_o - 1$. If in addition $\sum_{a' \in A_o(\mu)} w_{a'} \leq q_o - 1$, then μ is wasteful. Otherwise, there exists $a' \in A_o(\mu)$ such that a envies a' at μ, hence μ is not size-consistent. If $a \in D$, then $\sum_{a' \in \hat{A}_{(a,o)}(\mu) \cup S_o(\mu)} w_{a'} \leq q_o - 2$. If in addition $\sum_{a' \in A_o(\mu)} w_{a'} \leq q_o - 2$, then μ is wasteful. Otherwise, there exists $d \in D_o(\mu)$ such that a envies d at μ, hence μ is not size-consistent. We conclude that in all cases μ is either not size-consistent or wasteful, a contradiction, □

(Only If) Let μ be a size-stable matching. For any $(s, o) \in S \times O$, (s, o) is not a strong blocking pair so either $\overline{o}_s(\mu) \succeq_s o$ or $\sum_{a \in \hat{A}_{(s,o)}(\mu)} w_a = q_o$. In both cases, s does not have a claim to an unassigned unit at o and does not envy any agent matched to o at μ. For any $(d, o) \in D \times O$, (d, o) is not a strong blocking pair of μ so either $\overline{o}_d(\mu) \succeq_d o$ and d does not have any claim to o at μ or $\sum_{a \in \hat{A}_{(s,o)}(\mu) \cup S_o(\mu)} w_a \geq q_o - 1$. In the latter case, d does not have a claim to two unassigned units of o since at most one unit of this object is unassigned. In addition, $\sum_{a \in A_o(\mu)} w_a \leq q_o$ as μ is feasible by assumption. Consequently, there does not exists $d' \in D_o(\mu)$ such that $d \rhd d'$, that is d does not envy any double-unit agent at μ. We conclude that μ is size-consistent and non-wasteful in all cases. ■

Proof of Proposition 10: Let K be the number of rounds that the SDDA algorithm lasts and for all $k = 1, \dots, K$, let X_k be the set of pairs that enters Round k. Then $X_1 = A \times O$ and $\overline{\mu}(X_K) = \mu^{\text{SDDA}}$. Trivially, $\overline{\mu}(X_1) = \overline{\mu}(A \times O)$ is size-consistent, non-wasteful and not dominated by any size-stable matching. Towards an inductive argument, suppose that for some $k = 1, \dots, K-1$ and for all $j = 1, \dots, k$ $\overline{\mu}(X_j)$ is size-consistent, non-wasteful and not dominated by any size-stable matching.

For any $X \in 2^{A \times O}$ and any $(a, o) \in X$, let us denote by $\hat{A}^*_{(a,o)}(X) \equiv \{a' \in A_o(X) \mid a' \rhd^*_o a\}$ the set of agents who contest o at X and have a higher size-disjoint priority for it than a. Consider now a pair $(a, o) \in X_k$ such that o rejects a in Round k. Such pair exists since $k < K$. Without loss of generality, let a be the agent rejected by o in Round k with the highest size-disjoint priority. Then $\sum_{a' \in \hat{A}^*_{(a,o)} \cap \overline{A}_o(X_k)} w_{a'} > q_o - w_a$. If $a \in S$, all agents in $\hat{A}^*_{(a,o)} \cap \overline{A}_o(X_k)$ are single-unit agents since by definition a does not have a lower size-disjoint priority than any double-unit agent. Then all units of o are tentatively assigned to single-unit agents with a higher priority than a so the latter does not have a claim to o at X_{k+1}. The same is true for any agent that o has rejected beforehand. If $a \in D$, at least $q_o - 1$ units of o are tentatively assigned to agents with a higher priority, therefore a has at most a 1-unit claim to o at X_{k+1}. As all single-unit agents have a higher size-disjoint priority than a, o has not rejected any of them so no single-unit agent has a claim to o. It follows that $\overline{\mu}(X_{k+1})$ is size-consistent and non-wasteful. In addition, all rejections are dictated by feasibility, therefore any size-stable matching that makes some agent better-off than $\overline{\mu}(X_{k+1})$ also makes an agent worse-off. $\overline{\mu}(X_{k+1})$ is as a consequence not dominated by any size-stable matching.

By induction, we conclude that $\overline{\mu}(X_K)$ is size-consistent, non-wasteful and not dominated by any size-stable matching. As the algorithm ends in Round K, all proposals are accepted, which means that $\overline{\mu}(X_K)$ is also feasible, hence it is an undominated size-stable matching. ■

Proof of Lemma 11: Let $(a, o) \in X$ be a pair such that o rejects a at X and let $\mu \subseteq X$ be a feasible matching that contains (a, o). Consider first the case where $a \in D$. Then

$$\tilde{\mathcal{R}}_{(a,o)}(X) = w_a + \sum_{a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)} w_{a'} > q_o.$$

As μ is feasible, there exists $a' \in \hat{A}_{(a,o)}(X) \cap \overline{A}_o(X)$ such that $o \succ_{a'} \overline{o}_{a'}(\mu)$. Then a' envies a at μ so μ is not size-consistent. By Proposition 9, it is not size-stable.

Consider next the case where $a \in S$. Then

$$\tilde{\mathcal{R}}_{(a,o)}(X) = w_a + \sum_{s \in \hat{S}_{(a,o)}(X) \cap \overline{S}_o(X)} w_s + \sum_{d \in \tilde{D}_{(a,o)}(X)} w_d > q_o.$$

As agents in $\tilde{D}_{(a,o)}(X)$ only contest o at X, they are matched to o at μ. Therefore, as μ is feasible, there exists $s \in \hat{S}_{(a,o)}(X) \cap \overline{S}_o(X)$ such that $o \succ_s \overline{o}_s(\mu)$. Then s' envies s at μ so μ is not size-consistent. By Proposition 9, it is not size-stable. ■

Proof of Lemma 12: Lemma 12 arises as the combination of several results that are omitted in the main text. We present them one by one and then derive the desired result. We first introduce some useful notation and terminology. For any agent a and set of pairs X, let $\underline{o}_a(X) \equiv o \in O_a(X)$ such that $o' \succeq_a o$ for all $o' \in O_a(X)$ be a's **bottom choice** at X, that is a's least favorite object among those he contests. An agent does not have a bottom choice if he does not contest any object. The concept is completely analogous to the agent's top choice. We let K be the number of steps of the p-TDBU algorithm applied to the full set of pairs $A \times O$ and denote by X^k the set of pairs considered in Step $k = 1, \dots, K$. Then $X^k \subseteq X^j$ for any $k \geq j$, $X^1 = A \times O$, X^k is complete for all $k < K$ and $\phi(A \times O) = X^K$ if X^k is complete and $\phi(A \times O) = \varnothing$ otherwise.

Claim 1. *$\phi(A \times O)$ is complete and for every agent a, $\underline{o}_a(\phi(A \times O))$ is the object a prefers among those that give him a guarantee in the p-TDBU algorithm.*

Proof of Claim 1: Let a be any agent and let o be the object he prefers among those that give him a guarantee throughout the p-TDBU algorithm (such object exists as each agent receives at least a guarantee from the null object). Let $k = 1, \dots, K$ be the first step where o gives a a guarantee. For any $o' \in O$ such that $o \succ_a o'$, $(a, o') \notin X^{k+1}$, hence $(a, o') \notin X^K$. In addition, $\mathcal{G}_{(a,o)}(X^k) \leq q_o$. By Lemma 3, for any $j = k, \dots, K$, $\mathcal{G}_{(a,o)}(X^j) \leq \mathcal{G}_{(a,o)}(X^k) \leq q_o$ and by construction $\tilde{\mathcal{R}}_{(a,o)}(X^j) \leq \mathcal{G}_{(a,o)}(X^j) \leq q_o$ so o does not reject a at X^j. By assumption, a does not get a guarantee from any object he prefers to o, therefore $(a, o) \in X^K$ and $o = \underline{o}(X^K)$. This implies the first part of the statement as a contests at least one object. □

Claim 1 provides some basic properties of $\phi(A \times O)$. Our next step is to show that this set contains a size-stable matching. For this purpose, we define an alternative priority profile $\rhd^{\#}$ as follows. For all $o \in O$, the alternative priority relation $\rhd^{\#}$ is such that for any $a, a' \in A$ with $a \rhd_o a'$:

$$\begin{array}{ll} a' \rhd^{\#}_o a & \text{if } w_a > w_{a'} \text{ and } \sum_{\hat{a} \in \hat{A}_{(a,o)}(\phi(A \times O))} w_{\hat{a}} > q_o. \\ a' \rhd^{\#}_o a & \text{otherwise.} \end{array}$$

The alternative priority profile is the $|O|$-tuple $\rhd^{\#} = \{\rhd^{\#}_o\}_{o\in O}$ containing the alternative priority relation of all objects. Double-unit agent get to keep their priority over single-unit one for all objects that they contest and give them a guarantee or that they do not contest but would give them a guarantee if they did. For all other objects, they lose their priority to single-unit agents. We let $\mu^{\#}$ be the matching produced by the Deferred Acceptance algorithm when applied to the alternative market $\langle A, O, \succ, \rhd^{\#}, \mathbf{w}, \mathbf{q}\rangle$. Let N be the number of rounds of the DA algorithm applied to this alternative market and denote by X_k the set of pairs considered in Round $k = 1, \ldots, N$. Then $X^k \subseteq X^j$ for any $k \geq j$ and $X^1 = A \times O$. As the null object never rejects any proposal, X^k is complete for all $k = 1, \ldots N$ and $\mu^{\#} = \overline{\mu}(X^N)$. We denote by $\hat{A}^{\#}_{(a,o)}(X) \equiv \{a' \in A_o(X) \mid a' \rhd^{\#}_o a\}$ the set of agents who contest o at X and have a higher alternative priority than a. We now present three important properties of $\mu^{\#}$.

Claim 2. *For all $a \in A$, $\overline{o}_a(\mu^{\#}) \succeq_a \underline{o}_a(\phi(A \times O))$.*

Proof of Claim 2: Suppose towards a contradiction the existence of an agent a such that that $o \equiv \underline{o}_a(\phi(A \times O)) \succ_a \overline{o}_a$. For notational convenience, let $o \equiv \underline{o}_a(\phi(A \times O))$. There is a Round $k = 1, \ldots K - 1$ of the Deferred Acceptance algorithm where o rejects a, therefore $\sum_{a' \in \hat{A}^{\#}_{(a,o)}(X_k) \cap \overline{A}_o(X_k)} w_{a'} > q_o - w_a$. By Claim 1, o gives a a guarantee in the p-TDBU algorithm, therefore

$$\sum_{a' \in \hat{A}_{(a,o)}(\phi(A \times O))} w_{a'} \leq q_o - w_a.$$

It follows that for any $a' \in A$, $a \rhd^{\#}_o a'$ if and only if $a \rhd_o a'$, which means that $\hat{A}^{\#}_{(a,o)}(X_k) = \hat{A}_{(a,o)}(X_k)$. Then

$$\sum_{a' \in \hat{A}_{(a,o)}(X_k) \cap \overline{A}_o(X_k)} w_{a'} > q_o - w_a.$$

Combining our last two results implies the existence of an agent

$$\hat{a} \in (\hat{A}_{(a,o)}(X_k) \cap \overline{A}_o(X_k)) \setminus \hat{A}_{(a,o)}(\phi(A \times O)).$$

Then $\hat{a} \rhd_o a$ and $\hat{a}$ does not contest o at $\phi(A \times O)$. This means that he has received a guarantee for an object he prefers to o in the p-TDBU algorithm, hence $\underline{o}_{\hat{a}}(\phi(A \times O)) \succ_{\hat{a}} o$ by Claim 1. As $\hat{a}$ proposes to o in Round k, $ulo_{\hat{a}}(\phi(A \times O))$ rejects $\hat{a}$ in some Round $j < k$. By induction, there exists an agent who proposes in Round 1 to an object he ranks lower than all objects he contests at $\phi(A \times O)$, a contradiction since all agents propose to their first preference in Round 1. $\square$

Claim 2 establishes that all agents are matched at $\mu^{\#}$ to an object they like at least as much as their bottom choice at $\phi(A \times O)$. We next show that $\mu^{\#}$ is a stable matching of the alternative market.

Claim 3. *For any $a \in A$ and $o \in O$, if $o \succ_a \overline{o}_a(\mu^{\#})$, then $\sum_{a' \in \hat{A}^{\#}_{(a,o)}(\mu^{\#})} w_{a'} > q_o - w_a$.*

Proof of Claim 3: Suppose towards a contradiction the existence of an agent a and an object o such that $o \succ_a \overline{o}_a(\mu^{\#})$ and $\sum_{a' \in \hat{A}^{\#}_{(a,o)}(\mu^{\#})} w_{a'} \leq q_o - w_a$. Without loss of generality, let a be the agent in that situation with the highest priority for o. As $\mu^{\#} = \overline{\mu}(X_N)$ by definition,

$$\sum_{a' \in \hat{A}^{\#}_{(a,o)}(X_N) \cap \overline{A}_o(X_N)} w_{a'} \leq q_o - w_a.$$

As $o \succ_a \overline{o}_a(\mu^{\#})$, there exists a Round $k = 1, \ldots, N-1$ of the Deferred Acceptance algorithm where o rejects a. Then

$$\sum_{a' \in \hat{A}^{\#}_{(a,o)}(X_k) \cap \overline{A}_o(X_k)} w_{a'} > q_o - w_a.$$

Consequently, there exists a Round $j = k, \ldots, N-1$ such that

$$\sum_{a' \in \hat{A}^{\#}_{(a,o)}(X_j) \cap \overline{A}_o(X_j)} w_{a'} > q_o - w_a \quad \text{and} \quad \sum_{a' \in \hat{A}^{\#}_{(a,o)}(X_{j+1}) \cap \overline{A}_o(X_{j+1})} w_{a'} \leq q_o - w_a.$$

Then there exists an agent

$$\hat{a} \in (\hat{A}^{\#}_{(a,o)}(X_j) \cap \overline{A}_o(X_j)) \setminus (\hat{A}^{\#}_{(a,o)}(X_{j+1}) \cap \overline{A}_o(X_{j+1}).$$

$\hat{a}$ is such that $\hat{a} \rhd^{\#}_o a$ and $o \succ_{\hat{a}} \overline{o}_a(\mu^{\#})$. In addition,

$$\sum_{a' \in \hat{A}^{\#}_{(\hat{a},o)}(\mu^{\#})} w_{a'} \leq \sum_{a' \in \hat{A}^{\#}_{(a,o)}(\mu^{\#})} w_{a'} \leq q_o - w_a.$$

If $w_a \geq w_{\hat{a}}$, this directly implies $\sum_{a' \in \hat{A}^{\#}_{(\hat{a},o)}(\mu^{\#})} w_{a'} \leq q_o - w_{\hat{a}}$, a contradiction. Otherwise, $a \in S$ and $\hat{a} \in D$. As $\hat{a} \rhd^{\#}_o a$, $\underline{o}_{\hat{a}}(\phi(A \times O)) \succeq_{\hat{a}} o$. Then by Claim 2, $\overline{o}_a(\mu^{\#}) \succeq_{\hat{a}} o$, a contradiction. $\square$

Claim 3 shows that $\mu^{\#}$ does not have any blocking pair in the alternative market $\langle A, O, \succ, \rhd^{\#}, \mathbf{w}, \mathbf{q}\rangle$, which means that it is stable since the Deferred Acceptance produces by construction a feasible matching. In the original market $\langle A, O, \succ, \rhd, \mathbf{w}, \mathbf{q}\rangle$, $\mu^{\#}$ may have blocking pairs, however we show next that it does not have any strong one.

Claim 4. *$\mu^{\#}$ is size-stable.*

Proof of Claim 4: Consider an agent a and an object o such that $o \succ_a \overline{o}_a(\mu^{\#})$. By Claim 3, $\sum_{a' \in \hat{A}^{\#}_{(a,o)}(\mu^{\#})} w_{a'} > q_o - w_a$.

Consider first the case where $a \in D$. For any $\hat{a} \in A$ such that $\hat{a} \triangleright^{\#}_o a$, either $\hat{\triangleright}_o d$ or $\hat{\in} S$, which implies that $\hat{A}^{\#}_{(a,o)}(\mu^{\#}) \subseteq (\hat{A}_{(a,o)}(\mu^{\#}) \cup S_o(\mu^{\#}))$. Then

$$\sum_{a' \in \hat{A}_{(a,o)}(\mu^{\#}) \cup S_o(\mu^{\#})} w_{a'} \geq \sum_{a' \in \hat{A}^{\#}_{(a,o)}(\mu^{\#})} w_{a'} > q_o - w_a$$

so (a, o) is not a strong blocking pair of $\mu^{\#}$.

Consider next the case where $a \in S$. For any $\hat{a} \in A$ such that $\hat{a} \triangleright^{\#}_o a$, $w_{\hat{a}} \geq w_a$, hence $\hat{a} \triangleright_o a$. Therefore, $\hat{A}^{\#}_{(a,o)}(\mu^{\#}) \subseteq \hat{A}_{(a,o)}(\mu^{\#})$ and

$$\sum_{a' \in \hat{A}_{(a,o)}(\mu^{\#})} w_{a'} \geq \sum_{a' \in \hat{A}^{\#}_{(a,o)}(\mu^{\#})} w_{a'} > q_o - w_a.$$

(a, o) is not a (strong) blocking pair of $\mu^{\#}$.

We conclude that $\mu^{\#}$ does not have any strong blocking pair. As the Deferred Acceptance algorithm stops when none of the agents are rejected, $\mu^{\#}$ is feasible, which means it is size-stable. $\square$

We have now established that $\mu^{\#}$ is size-stable. The next claim shows that $\mu^{\#}$ is included in $\phi(A \times O)$, which implies that the latter set includes at least one size-stable matching.

Claim 5. $\mu^{\#} \subseteq \phi(A \times O)$.

Proof of Claim 5: Recall that $X^1, \ldots, X^K$ are the set of pairs considered throughout the p-TDBU algorithm. Trivially, $\mu^{\#} \subseteq X^1 = A \times O$. Towards an inductive argument, suppose that $\mu^{\#} \subseteq X^k$ for some $k = 1, \ldots, K-1$. Let $(a, o) \in X^{k+1} \setminus X^k$. Then either there exists o' such that $o' \succ_a o$ and o' gives a a guarantee at X^k or o rejects a at X^k.

In the former case, $\underline{o}_a(\phi(A \times O)) \succeq_a o' \succ_a o$ by Claim 1 so $(a, o) \notin \mu^{\#}$ by Claim 2. In the latter case, (a, o) is not an element of any size-stable matching included in X^k by Lemma 11. As $\mu^{\#}$ is size-stable by Claim 4 and $\mu^{\#} \subseteq X^k$ by the induction hypothesis, $(a, o) \notin \mu^{\#}$.

We conclude that $\mu^{\#} \subseteq X^{k+1}$. By induction, $\mu^{\#} \subseteq X^K$. By Claim 1, $\phi(A \times O)$ is complete, therefore $\mu^{\#} \subseteq X^K = \phi(A \times O)$. $\square$

The conjunction of Claims 4 and 5 implies that $\phi(A \times O)$ contains at least one size-stable matching. Our next two results allow inferring that it also contains a d-undominated one.

Claim 6. *If an object o gives a single-unit agent s a guarantee at $X \in \mathbb{X}$, then for any $\mu \in \tilde{\mathbb{S}} \cap 2^X$, $\overline{o}_s(\mu) \succeq_s o$.*

Proof of Claim 6: Let $s \in S$ and $o \in O$ be such that o gives s a guarantee in Round $k = 1, \ldots, K$ of the p-TDBU algorithm. Then $\mathcal{G}_{(s,o)}(X^k) \leq q_o$, therefore $\sum_{a \in \hat{A}_{(s,o)}(X^k)} w_a \leq q_o - 1$. For any matching $\mu \subseteq X_i$, $\sum_{a \in \hat{A}_{(a,o)}(\mu)} w_a \leq q_o - 1$, therefore $o \succ_s \overline{o}_s(\mu)$ implies that (s, o) is a strong blocking pair and μ is not size-stable. □

Claim 6 shows that when it comes to single-unit agents, guarantees allow eliminating agent-object pairs that are not in any size-stable matching included in the current set of pairs. It is only when guarantees are given to double-unit agents that some size-stable matchings may be lost. Our next and last result builds upon Claim 6 to show that $\mu^{\#}$ is only d-dominated by size-stable matchings included in $\phi(A \times O)$.

Claim 7. *Let $\mu, \mu' \in \tilde{\mathbb{S}}$ be two size-stable matchings such that μ dominates μ' and $\mu' \in \phi(A \times O)$. Then $\mu \subseteq \phi(A \times O)$.*

Proof of Claim 7: Let $\mu, \mu' \in \tilde{\mathbb{S}}$ be two size-stable matchings such that μ dominates μ' and $\mu' \in \phi(A \times O)$. Trivially, $\mu \subseteq X^1 = A \times O$. Suppose towards an inductive argument that $\mu \subseteq X^k$ for some $k = 1, \ldots, K-1$. Consider an agent-object pair $(a, o) \in X^{k+1} \setminus X^k$. Either a receives a guarantee at X^k from an object o' that he prefers to o or o rejects a at X^k.

In the former case, if $a \in S$ then by Claim 6 (a, o) is not an element of any size-stable matching included in X^k, hence $(a, o) \notin \mu$. If $a \in D$, then $\underline{o}_a(\phi(A \times O)) \succeq_a o' \succ_a o$ by Claim 1. Since $\mu' \subseteq \phi(A \times O)$, $\overline{o}_a(\mu') \succeq_a \underline{o}_a(\phi(A \times O))$ so $\overline{o}_a(\mu') \succ_a o$. As μ d-dominates μ', $\overline{o}_a(\mu) \succeq_a \overline{o}_a(\mu')$ so $\overline{o}_a(\mu) \succ_a o$ and $(a, o) \notin \mu$.

In the latter case, by Lemma 11 (a, o) is not an element of any size-stable matching included in X^k, hence $(a, o) \notin \mu$. We conclude that $\mu \subseteq X^{k+1}$, hence by induction $\mu \subseteq X^K = \phi(A \times O)$. □

As $\mu^{\#}$ a size-stable matching included in $\phi(A \times O)$ by Claims 4 and 5, a direct consequence of Claim 6 is that for all size-stable matchings $\mu \in \tilde{\mathbb{S}}$ that d-dominate $\mu^{\#}$, $\mu \subseteq \phi(A \times O)$. We are now in a position to prove the main statement. By Claims 4 and 5, $\mu^{\#}$ is a size-stable matching included in $\phi(A \times O)$. If it is d-undominated, then the proof is complete. Otherwise, there exists a d-undominated size-stable matching μ that d-dominates $\mu^{\#}$. By Claim 7, $\mu \subseteq \phi(A \times O)$. ■

Proof of Proposition 11: We begin with the first part of the statement and show in turn that $\overline{\mu}(\phi(A \times O))$ is size-consistent and non-wasteful.

(Size-Stable) By Lemma 12, $\phi(A \times O)$ is complete since it includes a size-stable matching. Let K be the number of steps that the p-TDBU lasts when applied to $A \times O$ and for all $k = 1, \dots, K$, let X^k be the set of agent-object pairs that enters step k. Then $X^1 = A \times O$ and $X^K = \phi(A \times O)$. Suppose towards a contradiction the existence of two agents $a, \tilde{a} \in A$ such that $w_a \leq w_{\tilde{a}}$ and a envies $\tilde{a}$ at $\overline{\mu}(X^K) = \overline{\mu}(\phi(A \times O))$. For notational convenience, let $o \equiv \overline{o}_{\tilde{a}}(X^K)$ and without loss of generaility, let a be the agent with the highest priority for o among those who have a weakly smaller size than $\tilde{a}$ and envy him at $\overline{\mu}(X^K)$. Then $a \triangleright_o \tilde{a}$ and $o \succ_a \overline{o}_a(X^K)$. By Claim 1, a does not receive a guarantee from any object he prefers to $\overline{o}_a(X^K)$ throughout the p-TDBU algorithm. Therefore, the fact that a does not contest o at X^K implies the existence of a Round $k = 1, \dots, K-1$ where o p-rejects a. It follows that $\tilde{\mathcal{R}}_{(a,o)}(X^k) > q_o$. Whether a is a single- or a double-unit agent, this implies

$$\sum_{a' \in \hat{A}_{(a,o)}(X^k) \cap \overline{A}_o(X^k)} w_{a'} > q_o - w_a.$$

Consider first the case where $\tilde{a} \in D$. In Round K, o does not reject $\tilde{a}$, therefore

$$\tilde{\mathcal{R}}_{(\tilde{a},o)}(X^K) = w_{\tilde{a}} + \sum_{a' \in \hat{A}_{(\tilde{a},o)}(X^K) \cap \overline{A}_o(X^K)} w_{a'} \leq q_o.$$

As $a \triangleright_o \tilde{a}$ and $w_a \leq w_{\tilde{a}}$, this implies

$$\sum_{a' \in \hat{A}_{(a,o)}(X^K) \cap \overline{A}_o(X^K)} w_{a'} \leq q_o - w_a.$$

Then there exists an agent $\hat{a} \in (\hat{A}_{(a,o)}(X^k) \cap \overline{A}_o(X^k)) \setminus (\hat{A}_{(a,o)}(X^K) \cap \overline{A}_o(X^K))$. As $\hat{a} \in \hat{A}_{(a,o)}(X^k)$, $\hat{a} \triangleright_o a \triangleright_o \tilde{a}$. In addition, $\hat{a}$ contests o at X^k as his top choice but does not contest o at X^K, thus $o \succ_{\hat{a}} \overline{o}_{\hat{a}}(X^K)$. We conclude that $\hat{a}$ envies $\tilde{a}$ at $\overline{\mu}(X^K) = \overline{\mu}(\phi(A \times O))$ and has a weakly smaller size than him, contradicting the assumption that a is the agent in that situation with the highest priority for o.

Consider next the case where $\tilde{a} \in S$. Since $w_a \leq w_{\tilde{a}}$, $a \in S$. In that case, $\tilde{\mathcal{R}}_{(a,o)}(X^k) > q_o$ implies

$$\sum_{s \in \hat{S}_{(a,o)}(X^k) \cap \overline{S}_o(X^k)} w_s + \sum_{d \in \tilde{D}_{(a,o)}(X^k)} w_d > q_o - w_a = q_o - 1.$$

In Round K, o does not reject $\tilde{a}$, therefore $\tilde{\mathcal{R}}_{(\tilde{a},o)}(X^k) \leq q_o$, which implies

$$\sum_{s \in \hat{S}_{(\tilde{a},o)}(X^K) \cap \overline{S}_o(X^K)} w_s + \sum_{d \in \tilde{D}_{(\tilde{a},o)}(X^K)} w_d \leq q_o - w_{\tilde{a}} = q_o - 1.$$

As $a \rhd_o \tilde{a}$, this in turn implies

$$\sum_{s \in \hat{S}_{(a,o)}(X^K) \cap \overline{S}_o(X^K)} w_s + \sum_{d \in \tilde{D}_{(a,o)}(X^K)} w_d \leq q_o - 1.$$

Agents in $\tilde{D}_{(a,o)}(X^k)$ exclusively contest o at X^k. As $X^K \subseteq X^k$ and X^K is complete, these agents also exclusively contest o at X^K, therefore $\tilde{D}_{(a,o)}(X^k) \subseteq \tilde{D}_{(a,o)}(X^K)$. This implies the existence of an agent $\hat{a} \in (\hat{S}_{(a,o)}(X^k) \cap \overline{S}_o(X^k)) \setminus (\hat{S}_{(a,o)}(X^K) \cap \overline{S}_o(X^K))$. As $\hat{a} \in \hat{S}_{(a,o)}(X^k)$, $\hat{a} \rhd_o a \rhd_o \tilde{a}$ and $\hat{a} \in S$. In addition, $\hat{a}$ contests o at X^k as his top choice but does not contest o at X^K, thus $o \succ_{\hat{a}} \overline{o}_{\hat{a}}(X^K)$. We conclude that $\hat{a}$ envies $\tilde{a}$ at $\overline{\mu}(X^K) = \overline{\mu}(\phi(A \times O))$ and has a weakly smaller size than him, contradicting the assumption that a is the agent in that situation with the highest priority for o. $\square$

(Non-Wasteful) The proof is almost analogous to the one above. Suppose towards a contradiction the existence of an agent $a \in A$ and an object $o \in O$ such that $o \succ_a \overline{o}_a(X^K)$ and $\sum_{a' \in \overline{A}_o(X^K)} w_{a'} \leq q_o - w_a$. That is, a has a claim to w_a unassigned units of o. Without loss of generality, let a be the agent in that situation with the highest priority. Formally, for any $\hat{a} \in A$, $o \succ_{\hat{a}} \overline{o}_{\hat{a}}(X^K)$ and $\sum_{a' \in \overline{A}_o(X^K)} w_{a'} \leq q_o - w_{\hat{a}}$ implies $a \trianglerighteq \hat{a}$. By Claim 1, a does not receive a guarantee from any object he prefers to $\overline{o}_a(X^K)$ throughout the p-TDBU algorithm. Therefore, the fact that a does not contest o at X^K implies the existence of a Round $k = 1, \ldots, K-1$ where o p-rejects a. It follows that $\tilde{\mathcal{R}}_{(a,o)}(X^k) > q_o$.

Consider first the case where $a \in D$. Then $\tilde{\mathcal{R}}_{(a,o)}(X^k) > q_o$ implies

$$\sum_{a' \in \hat{A}_{(a,o)}(X^k) \cap \overline{A}_o(X^k)} w_{a'} > q_o - w_a.$$

Then there exists $\hat{a} \in (\hat{A}_{(a,o)}(X^k) \cap \overline{A}_o(X^k)) \setminus \overline{A}_o(X^K)$. Since $\hat{a} \in \hat{A}_{(a,o)}(X^k)$, $\hat{a} \rhd_o a$. In addition, $a \in D$ implies $w_{\hat{a}} \leq w_a$, therefore $\sum_{a' \in \overline{A}_o(X^K)} w_{a'} \leq q_o - w_{\hat{a}}$. $\hat{a}$ has a higher priority than a for o and a claim to $w_{\hat{a}}$ unassigned units of that object, a contradiction.

Consider next the case where $a \in S$. In that case, $\tilde{\mathcal{R}}_{(a,o)}(X^k) > q_o$ implies

$$\sum_{s \in \hat{S}_{(a,o)}(X^k) \cap \overline{S}_o(X^k)} w_s + \sum_{d \in \tilde{D}_{(a,o)}(X^k)} w_d > q_o - w_a.$$

As $X^K \subseteq X^k$ and X^K is complete, agents in $\tilde{D}_{(a,o)}(X^k)$ exclusively contest o at X^K, therefore $\tilde{D}_{(a,o)}(X^k) \subseteq \overline{A}_o(X^K)$. Then there exists $\hat{a} \in (\hat{S}_{(a,o)}(X^k) \cap \overline{S}_o(X^k)) \setminus \overline{A}_o(X^K)$. Since $\hat{a} \in \hat{S}_{(a,o)}(X^k)$, $\hat{a} \rhd_o a$ and $w_{\hat{a}} = w_a = 1$ so $\sum_{a' \in \overline{A}_o(X^K)} w_{a'} \leq q_o - w_{\hat{a}}$. $\hat{a}$ has a higher priority than a for o and a claim to $w_{\hat{a}}$ unassigned units of that object, a contradiction. $\square$

It remains to show that the second part of the statement holds. The "only if" part is trivial as size-stable matchings are by definition feasible. To prove the "if" part, suppose that $\overline{\mu}(\phi(A \times O))$ is feasible. By construction, $\overline{\mu}(\phi(A \times O))$ dominates – hence it d-dominates – all other matchings included in $\phi(A \times O)$. Therefore, no matching other than $\overline{\mu}(\phi(A \times O))$ is a d-undominated size-stable matching included in $\phi(A \times O)$. Then $\overline{\mu}(\phi(A \times O))$ is a d-undominated size-stable matching by Lemma 12. ■

Proof of Lemma 13: The result is trivial if there does not exist any size-stable matching included in X that contains (d, o), therefore we focus on the case where such a matching exists. Let μ be such matching, that is $\mu \in \tilde{\mathbb{S}} \cap 2^X$ and $(d, o) \in \mu$.

Suppose towards a contradiction the existence of a pair $(a, o') \in P_{(d,o)} \cap \mu$. By Definition 8, $a \trianglerighteq_o d$ and $o \succ_a o'$. As $(d, o) \in \mu$, $a \neq d$ so $a \triangleright_o d$. Then a envies d at μ since $\overline{o}_d(\mu) = o \succ_a o' = \overline{o}_a(\mu)$. As $d \in D$, $w_a \leq w_d$ and μ is not size-consistent, a contradiction. We conclude that $\mu \subseteq X \setminus \tilde{P}_{(d,o)}(X)$, which directly implies that $X \setminus \tilde{P}_{(d,o)}(X)$ is complete.

We next let $X \setminus \tilde{P}_{(d,o)}(X)$ enter the p-TDBU algorithm. Let K be the number of steps of the algorithm and let X^k denote the set of agent-object pairs that enters Step $k = 1, \ldots, K$. Then $X^1 = X \setminus \tilde{P}_{(d,o)}(X)$, X^k is complete for all $k = 1, \ldots, K-1$ and

$$\phi(X \setminus \tilde{P}_{(d,o)}(X)) = \begin{cases} X^K & \text{if } X^K \text{ is complete} \\ \varnothing & \text{otherwise.} \end{cases}$$

We have established that $\mu \subseteq X^1$. Towards an inductive argument, suppose that $\mu \subseteq X^k$ for some $k = 1, \ldots, K-1$. Consider a pair $(a, o) \in X^{k+1} \setminus X^k$. Then o p-rejects a at X^k. By Lemma 11, $(a, o) \notin \mu$ since $\mu \in \tilde{\mathbb{S}} \cap 2^{X^k}$. It follows that $\mu \subseteq X^{k+1}$, hence by induction $\mu \subseteq X^K$. This in turn implies that X^K is complete, hence $\mu \subseteq X^K = \phi(X \setminus \tilde{P}_{(d,o)}(X))$. ■

Proof of Theorem 3: Let K be the number of rounds that the d-USSM algorithm lasts and for all $k = 1, \ldots, K$, let X_k be the set of pairs that enters Round k. Then $X_1 = A \times O$ and $\tilde{\mu}^*_\theta = \overline{\mu}(\phi(X_K))$. We proceed by induction with the following hypothesis. For some $k = 1, \ldots, K-1$ and for some $j = 1, \ldots, k$, if $\phi(X_j) \neq \varnothing$, then $\overline{\mu}(\phi(X_j))$ is size-consistent, non-wasteful and not d-dominated by any size-stable matching. $\overline{\mu}(\phi(X_1)) = \overline{\mu}(\phi(A \times O))$ is clearly size-consistent and non-wasteful. By Lemma 12, it is not d-dominated by any size-stable matching so our hypothesis holds for X_1. It remains to show that it holds for X_{k+1}. In Round k, the algorithm is in either Case 2 or Case 3 (as $k < K$) so a set X_{k+1} is constructed. If $\phi(X_{k+1}) = \varnothing$, the induction hypothesis holds trivially, hence we focus on the case where $\phi(X_{k+1}) \neq \varnothing$, which implies $\phi(X_{k+1}) \in \mathbb{I}$.

If the algorithm is in Case 2 in Round k, then X_{k+1} is constructed by protecting the pair z_k. Let $d \in D$ and $o \in O$ be the agent and object in this pair, that is $(d, o) = z_k$. If By Lemma 13, $\phi(X_{k+1})$ contains all size stable matchings included in $\phi(X_k)$ that contain (d, o). Any other matching included in $\phi(X_k)$ makes d worse-off, hence $\overline{\mu}(\phi(X_{k+1}))$ is not d-dominated by any size-stable matching. By definition, $\tilde{P}_{(d,o)}(\phi(X_k))$ does not contain any pair where the object is the agent's top choice, hence $\overline{\mu}(X_{k+1}) = \overline{\mu}(\phi(X_k))$ is size-consistent and non-wasteful by the induction hypothesis.

If the algorithm is in Case 3 in Round k, then Z_{k+1} is constructed by removing the pair z_i, which is the element of Z_k that was protected, from Z_k. Equivalently, $Z_{k+1} = Z_i$. This pair exists since $Z_k \neq \varnothing$ by Lemma 12. We let $d \in D$ and $o \in O$ be the agent and the object in this pair, that is $(d, o) = z_i$. X_{k+1} is constructed by removing (d, o) from $\phi(X_i)$. ($\phi(X_i) \neq \varnothing$ since a pair was protected in Round i.) By construction, (d, o) is a candidate of $\phi(X_i)$, therefore d contests o as his top choice but also contests other objects. His demand does not count towards single-unit agents with a lower priority, therefore there exists at least one single-unit agent who is not tentatively assigned any unit of o. If a single-unit agent prefers o to his top choice at $\phi(X_i)$, there exist at least two such agents. Then (d, o) does not have a claim to any unassigned unit at $\overline{\mu}(X_{k+1})$ and, as by definition he has a lower priority than all other double-unit agent contesting o as their to choice at $\phi(X_i)$. It follows that $\overline{\mu}(X_{k+1})$ is size-consistent and non-wasteful since $\phi(X_i)$ is by the induction hypothesis.

We have established that $\overline{\mu}(\phi(X_{k+1}))$ is not d-dominated by any size-stable matching and that $\overline{\mu}(X_{k+1})$ is size-consistent and non-wasteful. It remains to show that $\overline{\mu}(\phi(X_{k+1}))$ is also size-consistent and non-wasteful. Throughout the p-TDBU algorithm, an object only rejects an agent if it also reject all agents with a lower priority and a weakly larger and all of its units are tentatively assigned to agents with a higher priority, therefore $\overline{\mu}(\phi(X_{k+1}))$ is size-consistent and non-wasteful.

By induction, this means that $\overline{\mu}(\phi(X_K))$ is size-consistent, non-wasteful and not dominated by any size-stable matching. $\overline{\mu}(\phi(X_K))$ is also feasible since $C(\phi(X_K)) = \varnothing$, therefore it is a d-undominated size-stable matching. ■