

Menu Selection: A Computational Approach to Minimizing Food Waste

Haris Aziz

UNSW Sydney, Australia
haris.aziz@unsw.edu.au

Nicholas Mattei

Tulane University, USA
nsmattei@tulane.edu

Shivika Narang

UNSW Sydney, Australia
s.narang@unsw.edu.au

Sanjukta Roy

Indian Statistical Institute, India
sanjukta@isical.ac.in

Abstract

We introduce a novel collective decision making problem that captures the ubiquitous issue of ordering food to cater for varied dietary preferences and requirements. Our settings involve agents with diverse dietary requirements over menu options with varied serving sizes. The goal is to select a menu where everyone has enough food they can consume and wastage of food is minimized. We introduce two different consumption models: optimistic and pessimistic. Optimistic consumption assumes a situation when a central planner can optimally allocate the food ordered among the agents to maximize the number of people who get enough to eat. Pessimistic considers the worst case guarantee on consumption when agents fill their own plates in an arbitrary order. Under either consumption model, we seek valid menus (under which all agents are sufficiently fed) of minimum size.

Our work provides two sets of characterizations: (1) we characterize valid menus under either consumption model and (2) we characterize the space of instances that admit polynomial-time algorithms to find minimum sized menus. Our results also help us design Integer Linear Programs to find minimum sized menus in general settings. Furthermore, we present polynomial-time algorithms for important special cases. We then consider the worst case discrepancy between the size of minimum sized optimistic and pessimistic menus. We call this the *waste of pessimism* captured by the ratio of the minimum sized pessimistic menu to that of the minimum sized optimistic menu. We show tight upper bounds on this ratio. Our results also provide additional insights on the problem of finding a minimum sized maximal matching, which may be of independent interest.

1 Introduction

Deciding on a suitable menu catering to varied dietary requirements is a ubiquitous problem when sharing food — a quintessential human activity that transcends borders and divisions. Everyday, millions of people debate on how to choose a non-wasteful menu order for a group to jointly consume at an eatery of some sort. It is important to decide on an order which ensures enough food for everyone but at the same time is not so large that it leads to wastage. The choice of food to be shared may be affected by the various dietary restrictions and requirements among the diners. Dietary restrictions can stem from religious, ethical, health or many other considerations. For example, some people only eat vegan. Others avoid nuts because of allergies. Some cannot tolerate spice. In the broader sense, menu composition also comes into play when designing humanitarian food packages as outlined by United Nations High Commissioner for Refugees (UNHCR) with different standards for children, the elderly, infirm, and other groups.¹ When ordering food, we want to ensure that people have *enough food* that they find feasible to eat but *not so much that it leads to unnecessary waste*. The need for small sized menus may be from a variety of factors such as the food waste or the storage requirements from leftovers being minimized. We refer to the problem of finding a minimum sized menu that feeds everybody as the *Menu Selection* problem.

For a given menu, what food is actually consumed, depends on the order in which people fill their plates and what they choose. If a central planner allocates the selected food among the agents, it is easier to ensure the consumption is maximized and wastage, in turn, is minimized. In contrast, if the agents fill their own plates coming in an *arbitrary order*, much more food may be wasted. A less restricted person may end up consuming food intended for someone who can only eat from a limited number of options. This may leave the person with more food restrictions with nothing to eat. When ordering food, one could potentially order a separate acceptable dish or menu option for each person. However, such an approach can be quite wasteful especially if each menu option serves multiple people. The central problem of this paper is how do we design a menu order that ensures that wastage is minimized and everyone is well-fed under pessimistic and/or optimistic perspectives. To this end, we introduce a formal, conceptual framework which captures how the ordered menu is consumed. The smallest menu that feeds all agents will have the least food waste. Consequently, we focus on finding minimum sized menus that feed all agents.

Example 1.1 (Motivating Example). *Consider an example where there are four agents 1, 2, 3, and 4 who want to order something to eat together. The first three agents are vegetarian whereas agent 4 can also eat meat. There are two food options o_1 (a vegetarian pizza) that serves 3 agents and o_2 (a chicken burger) that serves 1. Since agents 1, 2, and 3 are vegetarians, they can only eat the pizza and hence only find o_1 acceptable. On the other hand, agent 4 finds both food options acceptable. This acceptability relation can be represented as a bipartite graph (Figure 1). Note that s_{o_i} denote the serving size of option o_i , $i \in \{1, 2\}$. The vegetarian agents 1, 2, and 3 (denoted by a green leaf icon next to them) only have edges to option o_1 representing pizza (with a green pizza slice next to it), while agent 4 has edges to both food options.*

The goal is to order sufficient food for the agents. If the order is $\{o_1, o_2\}$, it provides sufficient food and guarantees no food wastage for the agents as long as agent 4 does not eat any part of option o_1 . There is risk that if agent 4 eats some of the pizza, then the other agents will not have sufficient amount of acceptable food. On the other hand, if two units of o_1 are ordered, then the agents have sufficient food to eat irrespective of who eats what. In this case, 2 servings of food are wasted. Ordering two units of o_1 is risk averse but it leads to food wastage.

¹<https://emergency.unhcr.org/emergency-assistance/food-security/emergency-food-assistance-standard>

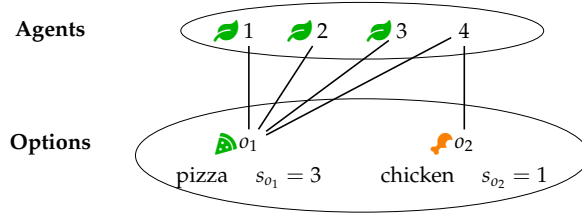


Figure 1: A graphical representation of agents and options described in [Example 1.1](#), with vegetarian agents indicated by a leaf icon on the left.

Ordering more food than necessary invariably leads to food waste. Minimizing food waste at the consumer end is a concern across the developed world. Whether due to food being quick to spoil, limited to no storage for leftovers, or the increased cost associated with ordering more food. When ordering food collectively, it is imperative to minimize food waste. On a global scale, halving the total amount of food wasted is one of the United Nations’ Sustainable Development Goals, specifically target 12.3.² Some prior work has gone into algorithms for selecting menus [28] and on computationally minimizing food waste by learning food preferences and restrictions [29, 46]. To the best of our knowledge, the objectives have not been studied together. We study a complementary setting where we are already equipped with the knowledge of individual food restrictions and aim to select a menu that can minimize food wastage, while everyone is well-fed.

Our Contributions

Conceptual Contribution. We detail a new formal model for a natural collective choice problem that concerns ordering food *with the goal of feeding a group while minimizing waste*. Our setting has a set of n agents, a set of m options each with associated serving size s , and each agent i has an associated acceptability set A_i consisting of all options i can consume. We develop two models of consumption to capture two extremes of how people may eat: *optimistic and pessimistic*.

Intuitively, these concepts are based on whether agents’ plates are filled in a centralized or a decentralized manner. We call the former optimistic consumption, where a central entity is able to serve each agent food they find acceptable. We call the latter as pessimistic consumptions, we consider the worst case where agents may fill their plates themselves, in an arbitrary order. This setting allows for some agents to “adversarially” fill their plates in order to minimize the food available to a specific set of agents. For either consumption model, we wish to ensure that there is enough food for all agents. We call a menu which ensures that every agent is fed as optimistically or pessimistically *valid*. These terms are defined in [Section 2](#). We defer an extended discussion on how our model relates to existing work to [Appendix A](#).

Further, this *introduces a new line of questioning*: given a fixed algorithm and partial control on the input instance, can we ensure optimal outcomes with minimum inputs? Specifically, optimistic consumption is captured by a maximum cardinality matching between agents and the menu. In turn, pessimistic consumption is captured by the minimum sized maximal matching. Consequently, the problem of finding a minimum size valid menu is the same as ensuring the respective matching algorithm produces an agent-saturating matching. In [Section 3](#), we show the equivalence between these matching based definitions and the motivating picking sequence approach. These characterizations help us design algorithms to find minimum sized menus. They also inform our subsequent results. Further, we adapt the well studied notion of price of anarchy to this setting and call it the *waste of pessimism*. Specifically, this is the ratio of the sizes of the minimum sized pessimistically valid menu to that of the minimum sized optimistically valid menu. Our

²<https://sdgs.un.org/goals/goal12>

	Min Sized Valid Menus				Waste of Pessimism
	Optimistic		Pessimistic		
General	NP-comp	(Prop. 4.2)	NP-comp	(Prop. 4.3)	$\geq \frac{n+4}{4}$
$s_o \leq 2$	$O^*(2^n)$	(Prop. 4.4)	NP-comp	(Prop. 4.3)	$\geq \frac{n+4}{4}$
Identical	in P	} (Prop. 5.5)	in P	} (Prop. 5.6)	$= 1$
Chained	in P		in P		≤ 2
Laminar	in P		in P		$\leq \frac{n+4}{4}$

Table 1: Summary of results on (1) computational complexity of finding minimum sized menus and (2) known bounds on waste of pessimism.

tools and algorithms could inform food delivery apps, humanitarian aid packaging, and even recommendations at restaurants in order to minimize waste and satisfy participating agents.

Our results can be classified along the following broad directions, and our algorithmic and waste of pessimism results are also summarized in Table 1.

1. Characterizing valid menus under optimistic and pessimistic consumption,
2. Classifying instances that admit polynomial-time algorithms for computing minimum sized valid menus, and
3. Obtaining tight bounds on the worst-case waste of pessimism.

Computational Results. We prove in Section 4.1 that checking if a given instance admits an optimistically or pessimistically valid menu of size at most B is NP-complete. For the optimistic case, this intractability result requires that serving sizes are at least 3. Meanwhile, for the pessimistic case, the intractability holds even when all options have a serving size of one. Based on the characterizations of valid menus, we design mixed integer linear programs (MILPs) for computing minimum sized valid menus in Section 4.2. While these MILPs gives us practical tools for computing minimum sized menus, they do not give good worst case guarantees. We develop a significantly faster algorithm, parameterized with respect to the number of agents n , for computing a minimum sized optimistically valid menu in Section 4.3. Note that we typically expect the number of options available m exceed the number of agents. Consequently, Algorithm 1 running in time $O^*(2^n)$ will be faster, in practice, than a naive algorithm which is exponential in the number of options.

In Section 5 we complement the intractability results by providing polynomial-time algorithms for computing minimum sized valid menus under several natural restrictions. We find that the efficient algorithms apply to several common realistic settings. For the case of optimistic consumption, we find polynomial-time algorithms for the following cases: (a) each option has a serving size of 1 or 2 (presented in Section 5.1) and (b) Acceptability sets are *laminar*: for any two agents, their acceptability sets must be either disjoint or one is contained in the other (provided in Section 5.2). Laminar food restrictions seem quite natural: e.g., those who eat meat can eat vegetarian food as well, similarly, those who prefer vegetarian food, can have vegan food as well – satisfying laminarity. For pessimistic consumption, we find a polynomial-time algorithm when acceptability sets are laminar. To this end, we refine the characterization for pessimistically valid menus for the laminar setting. These results help us in analyzing the waste of pessimism.

Waste of Pessimism. In Section 6, we study the worst case discrepancy between the size of minimum sized optimistic and pessimistic menus. We call this the *waste of pessimism*, captured by the ratio of the size of the minimum sized pessimistic menu to that of the minimum sized optimistic menu. We show tight upper bounds on this ratio for structured settings. Our main result is a tight

bound for the case of laminar acceptability sets. Specifically, we show that given an instance with laminar acceptability relations, waste of pessimism is at most $\frac{n+4}{4}$, where n is the number of agents. Interestingly, there is an instance where this bound is tight and the serving sizes are at most 2. In contrast, we prove that the worst case bound to be 2 when we assume *chained* acceptability sets, i.e., for any two agents, the acceptability set of one must be a subset of the other. Observe that the laminar setting is a generalization of the chained setting. This shows that the worst case blowup in menu size is not very significant when agents do not have restrictive requirements.

Consequences For Computing Minimum Sized Maximal Matchings. Throughout the paper we use matchings as a tool to both describe consumption models as well as find solutions. Specifically, we use minimum sized maximal matchings to define pessimistic consumption. As a result, our characterization for pessimistically valid menus also provides insights on deciding given a bipartite graph, whether the minimum sized maximal matching saturates one side of the bipartition. Further, a consequence of our algorithm for laminar acceptability sets shows that a minimum sized maximal matching can be computed when neighborhoods of vertices are laminar.

Any omitted proofs are deferred to the appendix.

2 Model

We consider a set of agents $N = \{1, 2, \dots, n\}$ and a set of options $O = \{o_1, \dots, o_m\}$. Each option can be viewed as a dish that could potentially feed multiple people. We assume all agents need the same amount of food and will not eat more than what they need. The term $s_o \in \mathbb{Z}_+ \setminus \{0\}$ indicates the serving size of $o \in O$, that is, how many people can be fed by one quantity of o . For example, a pasta dish may serve one person, but a large pizza is intended to be shared by multiple people.

Each agent $i \in N$ has a set of acceptable options $A_i \subseteq O$ that do not violate i 's personal dietary requirements. We assume that all options in $O \setminus A_i$ are not consumable by/unacceptable to i . Each agent has a requirement of getting a serving of one unit of food from any of the options that are acceptable to them. Note that agents do not express preferences over dishes beyond the dichotomous preferences of acceptable vs unacceptable. Without loss of generality, we assume that for each $o \in O$, there exists some $i \in N$ s.t. $o \in A_i$; otherwise we could discard the option.

Our aim is to select a minimum sized menu that can feed all agents something they find acceptable, i.e., a menu order or, more simply, menu M is a multiset of elements in O along with the quantity selected for each option. We use the notation $q_o(M)$ to denote the number of occurrences of $o \in O$ within M . That is, $q_o(M)$ is the quantity of o ordered under M . We use $\text{size}(M) = \sum_{o \in O} q_o s_o$ to denote the size of menu M . Given a set $S \subseteq O$, we shall use $M|_S$ to denote the menu M' where $q_o(M') = q_o(M)$ for each $o \in S$ and $q_o(M') = 0$ for each $o \in O \setminus S$. We call our menu planning problem as the MENU SELECTION problem and denote an instance of it by $I = \langle N, O, (s_o)_{o \in O}, (A_i)_{i \in N} \rangle$.

Example 2.1 (MENU SELECTION instance). *We now revisit the informal Example 1.1 and reframe it in the notation introduced. Consider a problem with 4 agents $N = \{1, 2, 3, 4\}$ and two options $O = \{o_1, o_2\}$ where $s_{o_1} = 3$ and $s_{o_2} = 1$, and $A_1 = A_2 = A_3 = \{o_1\}$ and $A_4 = \{o_1, o_2\}$.*

Consumption Models

When ordering food, we typically have some natural constraints. Our primary constraint is that the selected menu should ensure that *each agent has enough food to eat*. Further, it is important to note that if agents are allowed to fill their plates, however they wish, one at a time in some order, then a menu that might suffice for everyone under one ordering of the agents may not be sufficient under another. To this end, we study two consumption models: optimistic and pessimistic. We say that a menu is valid under a specific model, if under that model, there is enough acceptable

food for each agent. In order to capture the ways in which agents may consume from a menu, we introduce the following tool:

Definition 2.1 (Consumption Graph). *Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and a menu M , the consumption graph of M , denoted by $G_M(I) = (N, O_M, E_M)$, is a bipartite graph where there is a vertex for each agent i and $q_o(M)s_o$ vertices for each option $o \in O$. There is an edge from i to a copy of o s.t. $q_o(M) \geq 1$ if and only if $o \in A_i$.*

We shall capture how agents fill their plates from the selected menu by a matching on this consumption graph. Given a menu M , x is a *feasible matching* if $x : N \times O \rightarrow [0, 1]$, s.t. $\sum_{o \in A_i} x_{i,o} \leq 1$, $\sum_{i \in N} x_{i,o} \leq q_o(M)s_o$ and $x_{i,o} > 0$ only if $o \in A_i$. That is, $x_{i,o}$ denotes the amount of option o that i consumes, where i only consumes acceptable options and the maximum amount of o consumable is $q_o(M)s_o$, where $q_o(M)$ is the quantity of o ordered. Note that we use a definition of matchings consistent with linear programs, in order to simplify our discussion of ILPs later. Furthermore, it is well known that the linear polytope is integral [49]. Consequently, we shall assume, without loss of generality, that any feasible matching x that we compute is integral.

Optimistic Consumption

Under the optimistic consumption model, we effectively assume that a central planner fills everyone's plates such that all agents have enough acceptable food to eat. This can be seen as a setting where agents' plates are being prepared beforehand and being served to the agent at the table.

Definition 2.2 (Optimistically Consumed Amount λ). *Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and a menu M , let x be the maximum sized matching of the consumption graph $G_M(I)$. The optimistically consumed amount of M is $\lambda(M) = \sum_{(i,o) \in N \times O} x_{i,o}$, that is, the size of x .*

Optimistic Consumption setting can be thought of as a generalization of set cover. In set cover, each set covers all its constituent elements but in our setup, one option can only cover the corresponding serving size. As we need to feed all agents, we look for optimistically valid menus.

Definition 2.3 (Optimistically Valid Menus). *Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$, a menu M is valid under optimistic consumption, or is optimistically valid, if $\lambda(M) = n$. We use $\Lambda(I)$ to denote the space of all optimistically valid menus.*

That is, M is optimistically valid if there exists a feasible matching $x : N \times O \rightarrow [0, 1]$, s.t. $\sum_{o \in O} x_{i,o} = 1$. Observe that if there is such a matching, we can divide the food among the n agents that each agent gets a sufficient amount of something they find acceptable. We shall call this the *optimistic consumption matching*.

Observe that optimistic consumption is not always feasible in practice, for instance at children's birthday parties or in airplanes. Consider when there are two passengers on an airplane and one of them is vegan. Optimistically, it would suffice to have only one vegan meal and one meat based meal. If the non-vegan passenger is asked for their meal choice first and they take the vegan meal, then the vegan passenger is left with nothing to eat. To this end, we also study pessimistic consumption.

Pessimistic Consumption

Under pessimistic consumption, we allow agents to fill their plate in any manner, so long as they only eat from acceptable options and do not purposefully waste food. That is, we only consider those settings, where if an agent has a plate that is not full, there should be nothing left for them to eat. Allowing for complete autonomy in this manner, we wish to ensure that there should be enough food for each agent. We capture such consumption using *maximal matchings*.

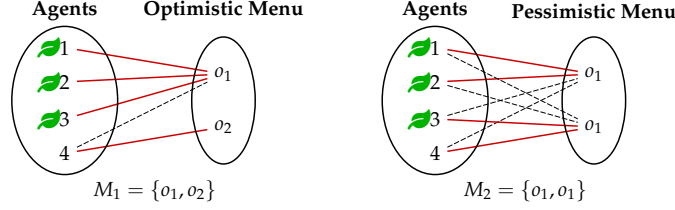


Figure 2: Consumption graphs of the minimum sized menus in [Example 2.2](#). The corresponding optimistic and pessimistic matchings are drawn in red. Edges not in the matching are dashed.

Formally, we call a matching x for the consumption graph G_M *maximal* if for each $i \in N$ s.t. $\sum_{o \in O} x_{i,o} < 1$, we have that for each option $o \in \{o \in A_i | q_o(M) > 0\}$, it holds $\sum_{i' \in N} x_{i',o} = q_o(M)s_o$.

Definition 2.4 (Pessimistically Consumed Amount γ). *Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and a menu M , let x be the minimum sized maximal matching of the consumption graph $G_M(I)$. The pessimistically consumed amount of M is $\gamma(M) = \sum_{(i,o) \in N \times O} x_{i,o}$, that is, the size of x .*

Definition 2.5 (Pessimistically Valid Menu). *Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$, a menu M is valid under pessimistic consumption, or is pessimistically valid, if $\gamma(M) = n$. We use $\Gamma(I)$ to denote the space of all optimistically valid menus.*

Analogous to the optimistic setting, we shall call the minimum sized maximal matching $x : N \times O \rightarrow [0, 1]$ as the *pessimistic consumption matching*.

Example 2.2 (Optimistically and Pessimistically Valid Menus). *Consider [Example 1.1](#) with 4 agents $N = \{1, 2, 3, 4\}$ and two options $O = \{o_1, o_2\}$ where $s_{o_1} = 3$ and $s_{o_2} = 1$, and $A_1 = A_2 = A_3 = \{o_1\}$ and $A_4 = \{o_1, o_2\}$. Then $\{o_1, o_2\}$ is an optimistically value menu. However, it is not pessimistically valid. The menu $\{o_1, o_1\}$ is pessimistically valid. We illustrate these menus in [Figure 2](#).*

Remark 2.1. *Consider an instance where for each agent $|A_i| = 1$. In this case, we need to order the one item this agent finds acceptable or they will not be able to eat. Specifically, for each option $o \in O$, we need to order $\left\lceil \frac{|\{i \in N | o \in A_i\}|}{s_o} \right\rceil$ quantities of o . Recall that for each option, we assume at least one agent finds it acceptable. In the case where $|A_i| = 1$, each option must be ordered. This is a minimum sized optimistically and pessimistically valid menu.*

3 Characterizations of Valid Menus

In this section, we provide characterizations of optimistically and pessimistically valid menus. These characterizations also help connect our graph theoretic definitions of consumption to the motivation: ensure that each agent is fed one unit of acceptable food. To this end, we first define a consumption sequence to denote the order in which the options in a given menu are consumed. Recall that given $S \subset O$ and a menu M , $M|_S$ denotes the multiset that contains as many copies of $o \in S$ as M and no copies of options not in S . For the purposes of this section and its proofs, we shall use the notation \widehat{M} to denote the multiset containing $q_o(M)s_o$ quantities of each option $o \in O$. We say that π is an ordering over N or a *picking sequence* if $\pi : [n] \rightarrow N$ and π is a bijection.

Definition 3.1 (Possible Outcome). *Given a MENU SELECTION instance I and a menu M and a picking sequence π , we say that $\omega : [n] \rightarrow O \cup \emptyset$ is a possible outcome of π on M if for each $k \in [n]$, we have that $\omega(k) \in \widehat{M}^k|_{A_{\pi(k)}}$ where $\widehat{M}^k = \widehat{M} \setminus (\bigcup_{k' < k} \omega(k'))$, i.e., \widehat{M}^k is the set of available options after the first $k - 1$ agents have picked and $\widehat{M}^k|_{A_{\pi(k)}}$ is the set of options in \widehat{M}^k that are acceptable by the agent $\pi(k)$. Further, it*

must hold that $\omega(k) = \emptyset \Leftrightarrow \widehat{M}^k|_{A_{\pi(k)}} = \emptyset$. We use $\Omega(\pi, M)$ to denote the space of all possible outcomes of π .

In other words, ω captures one possible outcome of agents picking food according to picking sequence π s.t. at the k^{th} turn agent $\pi(k)$ picks one unit of an acceptable option from M , if any are left. Recall that our motivation for the optimistic case was to ensure that there was some way to ensure that each agent finds something acceptable to eat. The following characterization shows that our definition does precisely this.

Theorem 3.1. [Characterization of optimistically valid menus] Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and a menu M , the following conditions are equivalent:

1. M is optimistically valid;
2. The consumption graph G_M admits matching of size n ;
3. There exists a picking sequence π and a possible outcome ω of π on M s.t. for each $k \in [n]$, $\omega(k) \neq \emptyset$.

Remark 3.1. Note that in the proof of the above theorem, when constructing a consumption sequence for an optimistic menu, the exact ordering of agents is irrelevant, as long as their chosen option is consistent with a maximum cardinality matching of the consumption graph G_M .

We now turn to pessimistic menus. Recall that a menu M is pessimistically valid if every minimum sized maximal matching of G_M has size n . Note that, the exact computation of a minimum sized maximal matching is not essential to check for pessimistic validity. It's only necessary to check its size is n . Specifically, even those maximal matchings that capture adversarial behavior towards any one agent need to have size n . To this end, we need that for each agent $i \in N$, there should be more food ordered from A_i than the other agents $N \setminus \{i\}$ combined can eat. We capture this risk to pessimistic validity as follows:

Definition 3.2 (Pessimistic Risk ρ). Given a MENU SELECTION instance I and a menu M , for an agent $i \in N$, the pessimistic risk under M denoted by $\rho_i(M)$ is the size of the maximum cardinality matching between $N \setminus \{i\}$ and $M|_{A_i}$.

We find that this definition helps us succinctly capture a necessary and sufficient condition for pessimistically valid menus.

Theorem 3.2. [Characterization of pessimistically valid menus] Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and a menu M , the following conditions are equivalent:

1. M is pessimistically valid;
2. For every agent $i \in N$, we have that $\text{size}(M|_{A_i}) = \sum_{o \in A_i} q_o(M) s_o \geq 1 + \rho_i(M)$;
3. For every picking sequence π and every $\omega \in \Omega(\pi, M)$, we have that for each $k \in [n]$, $\omega(k) \neq \emptyset$.

Partial Proof. We defer the proof for the equivalence of (1) and (2) to [Appendix B](#). Assuming this equivalence holds, we now prove the remaining case.

Equivalence of statement (3) to pessimistic validity. Let M be pessimistically valid. Consider a consumption sequence ω of M with a corresponding ordering π . Recall that for any $k \in [n]$, each preceding agent, under π , can consume at most one unit of an option acceptable to agent i . Consequently, we have that the difference between the number of units of acceptable options for

agent $\pi(k)$ under M and that under \widehat{M}^k is $size(\widehat{M}|_{A_{\pi(k)}}) - size(\widehat{M}^k|_{A_{\pi(k)}}) \leq \rho_i$. As M is pessimistically valid, from (2) we have that $size(M|_{A_i}) = \sum_{o \in A_i} q_o(M)s_o \geq 1 + \rho_i(M)$. Consequently, $\widehat{M}^k|_{A_{\pi(k)}}$ must be non-empty. As a result, $\omega(k) \neq \emptyset$ for each $k \in [n]$.

Finally, for the converse, let M be a menu s.t. for every consumption sequence ω of M , we have that for each $k \in [n]$, $\omega(k) \neq \emptyset$. Fix agent $i \in N$. Let x be maximum cardinality matching between $N \setminus \{i\}$ and $M|_{A_i}$. Consider an arbitrary ordering π under which $\pi(n) = i$. That is, i is the last agent to pick from M . Let $N^i = \{j \in N \setminus \{i\} \mid \sum_{o \in A_j} x_{j,o} = 1\}$. Consider a choice of ω s.t. for $\pi(k) \in N^i$ and the option o for which $x_{\pi(k),o} = 1$, we set $\omega(k) = o$. For all $k \in [n]$ s.t. $k \notin N^i$, choose $\omega(k)$ arbitrarily from \widehat{M}^k , whenever \widehat{M}^k is non-empty. It is straightforward to see that ω is indeed a consumption sequence for M . Consequently, it must hold that $\omega(n) \neq \emptyset$, by assumption. As a result, we must have that $\widehat{M}^n|_{A_i}$ must be non-empty. This gives us that the size of x if $\rho_i = \sum_{j \neq i} \sum_{o \in A_j} x_{j,o} < \sum_{o \in A_i} q_o(M)s_o$.

Hence, we have that for each $i \in N$, $size(M|_{A_i}) > \rho_i$ and thus, M is pessimistically valid. \square

4 Computing Minimum Sized Valid Menus

We now explore the computational aspects of finding minimum sized menus that are optimistically or pessimistically valid. As mentioned before, this problem is a generalization of the SETCOVER problem so it is unsurprising that it proves to be computationally intractable. We first discuss the complexity of finding minimum sized menus, then give exact algorithms for finding minimum sized optimistically valid menus.

4.1 Computational Complexity

Before settling the complexity of computing a minimum sized menu, we first discuss the complexity of verification of validity. Observe that one way of checking if a menu is optimistically or pessimistically valid is to compute the optimistically and pessimistically consumed amount.

Proposition 4.1. *Given a MENU SELECTION instance I and a menu M*

1. *the amount of M that is optimistically consumed $\lambda(M)$ can be computed in polynomial time,*
2. *it is NP-hard to compute the amount of M that is pessimistically consumed $\gamma(M)$, and*
3. *it can be checked in polynomial time if M is optimistically and/or pessimistically valid.*

Remark 4.1. *Given a graph $G = (V, E)$ using [Proposition 4.1](#), we can check in polynomial time if the minimum sized maximal matching has size $|V|$.*

We now show that finding a minimum sized optimistically valid menu is NP-hard even when each agent finds at most two options acceptable and each option has serving size three. The proof is via a reduction from the 3DIMENSIONALMATCHING (3DM) problem.

Proposition 4.2. *Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and size $B \geq n$ where $s_o \geq 3$, it is NP-complete to decide if there exists an optimistically valid menu of size at most B .*

We show the NP-hardness of finding minimum sized pessimistically valid menus via a reduction from the Exact Cover (X3C) problem, which is known to be NP-complete. [30].

Proposition 4.3. *Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and size $B \geq n$ where $s_o = 1$, it is NP-complete to decide if there exists a pessimistically valid menu of size at most B .*

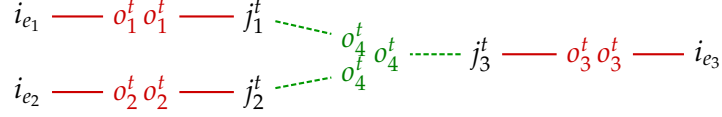


Figure 3: Consumption graph for set gadget for a set $S_t = \{e_1, e_2, e_3\}$ with the case for S_t in X3C shown in **solid red** edges and not in S_t shown in **dashed green** edges.

Proof Sketch. We give a reduction from the EXACT3COVER (X3C) problem, which is known to be NP-hard [30]. It is formally defined as follows:

EXACT3COVER (X3C)

Given: A universe \mathcal{U} of $3k$ elements and a family of ℓ subsets \mathcal{S} , where for each $S_t \in \mathcal{S}$, we have that $|S_t| = 3$.

Question: Do there exist k mutually disjoint subsets $\{S_{i_1}, \dots, S_{i_k}\}$ which cover \mathcal{U} , that is, $\cup_{j \in [k]} S_{i_j} = \mathcal{U}$?

Given an instance of X3C $\langle \mathcal{U}, \mathcal{S} \rangle$, we create a MENU SELECTION instance as follows: For each element $e \in \mathcal{U}$, create element agent i_e . For each subset $S_t \in \mathcal{S}$, create three set agents j_1^t, j_2^t and j_3^t and options o_1^t, \dots, o_4^t . For each $o \in O$, we set $s_o = 1$. For each $S_t \in \mathcal{S}$, let $S_t = \{e_1^t, e_2^t, e_3^t\}$. For each $z \in [3]$, $A_{j_z^t} = \{o_z^t, o_4^t\}$. Furthermore, add o_z^t to $A_{i_{e_z}}$ for each $z \in [3]$.

Consequently, we create an instance with $n = 3(k + \ell)$ agents and $m = 4\ell$ options. For each $e \in \mathcal{U}$, A_{i_e} contains exactly one option corresponding to each subset containing e . We illustrate a set gadget for set $S_t = \{e_1, e_2, e_3\}$ in Figure 3. We show that an X3C exists if and only if there is a pessimistically valid menu of size $n = 3(k + \ell)$. Note that if an X3C exists, say $\{S_{i_1}, \dots, S_{i_k}\}$, we can create a pessimistically valid menu M of size n . For each set $S_t \in \{S_{i_1}, \dots, S_{i_k}\}$, we set $q_{o_1^t}(M) = q_{o_2^t}(M) = q_{o_3^t}(M) = 2$ and $q_{o_4^t}(M) = 0$. For each set $S_t \notin \{S_{i_1}, \dots, S_{i_k}\}$, we set $q_{o_1^t}(M) = q_{o_2^t}(M) = q_{o_3^t}(M) = 0$ and $q_{o_4^t}(M) = 3$. \square

4.2 Mixed Integer Linear Programs for Minimum Sized Valid Menus

We now provide Mixed Integer Linear Program (MILP) formulations for finding minimum sized valid menus. MILP solvers allow us to find solutions quickly, in practice. Throughout this discussion, we shall slightly overload our notation and use q_o to denote the chosen amount of option o . Observe that every choice of $(q_o)_{o \in O}$ has a bijection to the space of menus. Consequently, we shall talk about a choice of $(q_o)_{o \in O}$ and its corresponding menu interchangeably. Further, variables x_o, x_o^i and y_j shall be binary variables denoting whether the corresponding option/agent is selected.

Optimistic Consumption.

We now present an MILP formulation for computing a minimum sized optimistic menu. Here, we take a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ as input. We use m variables q_o for each $o \in O$ to denote the quantities of each option o . We further denote x_o^i to denote a maximum cardinality matching between agents and their respective acceptable options. The MILP below is essentially adding minimum sized quantity objective function on top of a classical bipartite matching linear program.

$$\begin{aligned}
& \text{minimize} && \sum_{o \in O} q_o s_o \\
& \text{subject to} && \sum_{i \in N} x_o^i \leq q_o s_o, && o \in O \\
& && \sum_{o \in A_i} x_o^i \geq 1, && i \in N \\
& && x_o^i \in [0, 1], && i \in N, o \in O \\
& && q_o \in \{0, 1, \dots, \lceil \frac{n}{s_o} \rceil\} && o \in O
\end{aligned}$$

The MILP above has $m + nm$ variables and $2m + 2n + mn$ constraints. Observe that for a menu that is not optimistically valid, by definition, no matching can ensure that $\sum_{o \in A_i} x_o^i \geq 1$. Consequently, the polytope defined in the MILP above is feasible for a choice of $(q_o)_{o \in O}$ if and only if the corresponding menu is optimistically valid. Further, as optimistically valid menus are guaranteed to exist, there is always a solution for which the MILP is feasible. As a result, given a MENU SELECTION instance I , the MILP finds a minimum sized optimistically valid menu.

Pessimistic Consumption.

We now look for an MILP for pessimistic consumption. We first give an ILP to check if a given menu is pessimistically valid. Consider [Theorem 3.2](#), where, to check for pessimistic validity for each agent $i \in N$, we compute a maximum cardinality matching from $N \setminus \{i\}$ to options in A_i . Finding such a matching with an LP can be done by setting the objective function to maximize the size of the matching. However, a construction which puts the matching size in the objective is not useful when building minimum sized pessimistically valid menus. Consequently, we require a polytope which is feasible if and only if the menu is pessimistically valid. Unlike the case for optimistic consumption, we do not want to be able to match all agents in $N \setminus \{i\}$. Ideally, we would like to find menus that minimize the size of this maximum cardinality matching.

There seems to be no straightforward way to encode a maximum cardinality matching, where all agents need not be matched, solely through linear constraints. However, recall that we do not actually need to compute a maximum cardinality matching. From [Theorem 3.2](#), we only require that each maximum cardinality matching from $N \setminus \{i\}$ to $M|_{A_i}$ should leave some acceptable food for agent i . In other words, we have that M is pessimistically valid for i if and only if there is no option saturating matching for the graph between $N \setminus \{i\}$ and $M|_{A_i}$. From Hall's theorem [[34](#)], we can certify this without computing a maximum cardinality matching. Specifically, there must be some set of options in $M|_{A_i}$ for which the condition in Hall's theorem is violated. That is, there must exist $S \subseteq A_i$ s.t. $\sum_{o \in S} q_o s_o > |\{j \in N \setminus \{i\} : A_j \cap S \neq \emptyset\}|$. We capture this in the following ILP which will be feasible if and only if such a set S exists. In this way, we combine our characterization in [Theorem 3.2](#) with Hall's theorem to check pessimistic validity through an ILP.

ILP Setup Given a menu M , we use the following ILP to check pessimistic validity for a fixed $i \in N$. Let $N^i = \{j \in N \setminus \{i\} \mid \exists o \in A_i \cap A_j\}$. Observe that only the agents in N^i can be matched in a maximum cardinality matching from $N \setminus \{i\}$ to $M|_{A_i}$. We use x_o to denote where o is in S and we set y_j to 1 iff for some $o \in A_j \cap A_i$ we have that $x_o = 1$. We use variables of type x_o to select options and y_j to select agents in $N \setminus \{i\}$. Consequently, we always choose $x_o, y_j \in \{0, 1\}$. Thus, the amount of food within S is captured by $\sum_{o \in A_i} x_o q_o s_o$, where $x_o = 1 \Leftrightarrow o \in S$. For S to violate Hall's theorem, we have that the size of its neighborhood set, which is captured by $\sum_{j \in N \setminus \{i\}} y_j < \sum_{o \in A_i} x_o q_o s_o$.

$$\begin{aligned}
& \text{minimize} && 1 \\
& \text{subject to} && \\
& \sum_{o \in A_i} x_o q_o s_o \geq 1 + \sum_{j \in N^i} y_j && \text{(Hall's theorem violation certificate)} \quad (1) \\
& y_j \geq x_o && j \in N^i, o \in A_i \cap A_j \neq \emptyset \quad (2) \\
& y_j \leq \sum_{o \in A_i \cap A_j} x_o && j \in N^i \quad (3) \\
& x_o \in \{0, 1\} && o \in A_i \\
& y_j \in \{0, 1\} && j \in N^i
\end{aligned}$$

Observe that the ILP above has at most $n + m$ variables and at most $nm + 2n + m + 1$ constraints.

Correctness of ILP We now show that this ILP is feasible if and only if the given menu M is pessimistically valid for i . Let the above ILP be feasible. Let $((x_o)_{o \in A_i}, (y_j)_{j \in N^i})$ be a feasible solution of the ILP. Choose the set of options $S = \{o \in A_i | x_o = 1\}$. Observe that this must be non-empty under any feasible solution. The amount of capacity across these options is $\sum_{o \in S} q_o s_o = \sum_{o \in A_i} x_o q_o s_o$.

Observe that for each $j \in N^i$ s.t. there exists $o \in S \cap A_j$. For each of these agents, from constraint 2, we need $y_j \geq x_o = 1$. Consequently, it must hold that $y_j = 1$, as each $y_j \in \{0, 1\}$. Further, for all agents $j \in N \setminus \{i\}$ s.t. $A_j \cap S = \emptyset$, by definition of S , we have that for each option $o \in A_j \cap A_i$, it must hold that $x_o = 0$. Thus, by constraint 3, it must hold that $y_j = 0$. Consequently, we have that $\sum_{j \in N^i} y_j = |\{j \in N^i | \exists o \in S \cap A_j\}|$ which is precisely the size of neighborhood of S .

As $((x_o)_{o \in A_i}, (y_j)_{j \in N^i})$ is a feasible solution of the ILP, we have that $\sum_{o \in A_i} x_o q_o s_o > \sum_{j \in N^i} y_j$. Thus, by Hall's theorem, no matching in the consumption graph of $M|_{A_i}$ can saturate the options in $M|_{A_i}$. Consequently, M must be pessimistically valid for i .

For the converse, let M be pessimistically valid for i . Thus, there must exist a set of options $S \subseteq A_i$ s.t. $\sum_{o \in S} q_o s_o > |\{j \in N^i | \exists o \in S \cap A_j\}|$. For each option $o \in A_i$, set $x_o = 1 \Leftrightarrow o \in S$. For each $j \in N^i$, set $y_j = \min\{1, \sum_{o \in A_j \cap A_i} x_o\}$. It is straightforward to verify that this is a feasible solution for the ILP above.

Constructing Pessimistic Menus. We now extend the ideas in the previous ILP to get a minimum sized pessimistically valid menu for a given instance M . To this end, we need to construct a polytope that is feasible iff a menu is pessimistically valid for each $i \in N$. Consequently, for each $i \in N$, we introduce variables x_o^i, w_o^i and y_j^i where $o \in A_i$ and $j \in N^i$. As before, we have that $N^i = \{j \in N \setminus \{i\} | \exists o \in A_i \cap A_j\}$. We also now have q_o as a variable for every $o \in O$. Further, we simulate the amount of food within the set from option o (for agent i) $x_o^i q_o s_o$ by the variable w_o^i . We shall introduce constraints that ensure $w_o^i = x_o^i q_o s_o$.

As before, we set the range of the values of x_o^i and y_j^i to be only $\{0, 1\}$, as they are selection variables. In order to choose the range of q_o and subsequently w_o we make the following observation.

Observation 4.1. *Given a MENU SELECTION instance I , under a minimum sized optimistically or pessimistically valid menu M , we have that $q_o(M) \leq \lceil \frac{n}{s_o} \rceil$. As a result, under M , the amount of food from any fixed option $o \in O$ is $q_o(M) s_o < n + s_o$.*

Consequently, it is sufficient to set the range of values for q_o as $\{0, 1, \dots, \lceil \frac{n}{s_o} \rceil\}$ and for w_o to be $\{0, 1, \dots, n + s_o\}$. Both of these values are determined by the instance. Unfortunately, we cannot

now simply use the previous ILP as is. Firstly, the previous ILP is focused on one agent. To this end, we now replace variables of type x_o and y_j by x_o^i and y_j^i for each $i \in N$. Further, observe that the term $x_o q_o s_o$ in the previous ILP is no longer linear now as q_o is a variable. We need a way to set w_o to $q_o s_o$ if $x_o = 1$ and set $w_o = 0$ otherwise. To this end, we introduce three new linear constraints (5)-(7). If $x_o = 1$, then (5) and (7) together ensure that $w_o = q_o s_o$, and (6) puts no restriction of w_o as $(n + s_o)$ is a large number. Otherwise if $x_o = 0$, then $w_o = 0$ due to (6).

$$\begin{aligned}
& \text{minimize} && \sum_{o \in O} q_o s_o \\
& \text{subject to} && \\
& \sum_{o \in A_i} w_o^i \geq 1 + \sum_{j \in N^i} y_j^i && i \in N && (4) \\
& w_o^i \leq q_o s_o && i \in N, o \in A_i && (5) \\
& w_o^i \leq x_o^i (n + s_o) && i \in N, o \in A_i && (6) \\
& w_o^i \geq q_o s_o - (1 - x_o^i)(n + s_o) && i \in N, o \in A_i && (7) \\
& y_j^i \geq x_o^i && i \in N, j \in N^i, o \in A_i \cap A_j && (8) \\
& y_j^i \leq \sum_{o \in A_i \cap A_j} x_o^i && i \in N, j \in N^i && (9) \\
& x_o^i \in \{0, 1\} && i \in N, o \in A_i && \\
& w_o^i \in \{0, 1, \dots, (n + s_o)\} && i \in N, o \in A_i && \\
& y_j^i \in \{0, 1\} && i \in N, j \in N^i && \\
& q_o \in \left\{0, 1, \dots, \left\lceil \frac{n}{s_o} \right\rceil\right\} && o \in O &&
\end{aligned}$$

We defer a discussion of the correctness and the size of the ILP to [Appendix C](#). The correctness is shown in two parts: we first show that w_o^i is always equal $x_o^i q_o s_o$ for every agent $i \in N$ and every option $o \in A_i$. We then show that ILP above is always feasible for every menu planning instance.

4.3 Parameterized Algorithm for Minimum Sized Optimistic Menus

In this section we design a FPT algorithm to find a minimum size menu that is optimistically valid parameterized by the number of agents. Recall from [Observation 4.1](#) that under any valid menu for any option $q_o(M) \leq n$. Thus, we need to check at most n^m valid menus to find a minimum size one. Consequently, there is a trivial $O(n^m)$ time algorithm. We present an algorithm that significantly improves this running time by reducing our problem to the problem of finding maximum weight partition. Towards this first we define maximum weighted partition. A tuple of subsets (S_1, S_2, \dots, S_k) of a set N is a partition of the set N if $\cup_{i \in [k]} S_i = N$ and $S_i \cap S_j = \emptyset$ for each pair $i, j \in [k]$, and $i \neq j$. The MAXIMUM WEIGHTED PARTITION problem [14] is defined as follows:

MAXIMUM WEIGHTED PARTITION

Given: A universe \mathcal{N} , a set family \mathcal{F} over \mathcal{N} , a non-negative integer $k \in \mathbb{Z}_+$, and functions f_1, f_2, \dots, f_k , and a target value $B \in \mathbb{R}$

Question: Does there exist a partition (S_1, S_2, \dots, S_k) of \mathcal{N} in to k parts such that each $S_i \in \mathcal{F}$ and the sum $\sum_{i \in [k]} f_i(S_i) \geq B$?

Algorithm Overview. We use the variable k to denote how many distinct options are in a menu. For each integer $k \in [m]$, we reduce the given instance of MENU SELECTION to an instance of

Algorithm 1: Parameterized Algorithm via Maximum-Weight Partition

Input: Menu Planning instance $\langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$

Output: An optimistically valid menu M

```
1 Let  $size = mn$ ;  
2 for  $k \leftarrow 1$  to  $m$  do  
3   foreach  $o_j \in O$  do  
4      $\lfloor$  Let  $N_j \leftarrow \{i \in N : i \text{ accepts } o_j\}$ ; // Define feasible types of agents  
5     Define  $\mathcal{F} \leftarrow \{N' \subseteq N_j : N' \neq \emptyset, j \in O\}$ ;  
6     foreach  $N' \in \mathcal{F}$  do  
7       Let  $O' \leftarrow \{o_j \in O : N' \subseteq N_j\}$  and choose  $o' \in \arg \min_{j \in O'} \{|N'| \bmod s_j\}$ ;  
8       Set  $f(N') \leftarrow -(|N'| \bmod s_{o'})$ ; // Define weight for each type  
       // Solve maximum weight partition  
9       Compute a partition  $\mathcal{X} = (X_1, \dots, X_k)$  of  $N$  with  $X_i \in \mathcal{F}$  maximizing  $\sum_{i=1}^k f(X_i)$ ;  
10      if  $size >$  the weight of  $\mathcal{X}$ , then  $size =$  weight of  $\mathcal{X}$ ;  
11 Let  $(X_1, \dots, X_{k^*})$  be a maximum weight partition with weight  $size$ ;  
12 foreach  $z \in [k^*]$  do  
13   Let  $O_z \leftarrow \{o_j \in O : X_z \subseteq N_j\}$  and choose  $o_z \in \arg \min_{j \in O_z} \{|X_z| \bmod s_j\}$ ;  
14   Set  $q_{o_z}(M) \leftarrow \lfloor \frac{|X_z|}{s_{o_z}} \rfloor$ ; // Define the menu  
15 foreach  $o \in O \setminus \{o_z : z \in [k^*]\}$  do  
16   Set  $q_o(M) \leftarrow 0$ ;  
17 return  $M$ ;
```

MAXIMUM WEIGHT PARTITION. We use the set of agents as the set of elements. We generate the family \mathcal{F} to be the non-empty subsets of agents that all find an option to be acceptable. For each such set $N' \in \mathcal{F}$, we define the weights to be $f(N') = -(|N'| \bmod s_{o'})$ where o' is an option that minimizes this value among $\bigcap_{i \in N'} A_i$. We find a maximum weight partition of the agents. For each part (a subset of agents) of it, we order enough of an option that is acceptable to this subset of agents and that has minimum waste. Finally, we output the minimum size menu over all $k \in [m]$.

Proposition 4.4. *Given a MENU SELECTION instance I , let s_{max} denote the maximum serving size of any option, i.e., $s_{max} = \max_{o \in O} s_o$ and $m = |O|$. Algorithm 1 computes a minimum sized optimistically valid menu in $O^*(2^n m^3 s_{max})$ time.*

5 Polynomial-time Algorithms for Structured Instances

We now complement our complexity results by showing that whenever the conditions for them do not hold, we can, in fact, find minimum sized valid menus in polynomial time. Our results throughout this section can be combined to give the following theorems.

Theorem 5.1 (Classifying Tractable Instances for Optimistic Consumption). *Finding a minimum sized optimistically valid menu is*

- a. NP-complete even for instances with $s_o = 3$ for each $o \in O$ and
- b. in P for instances with $s_o \leq 2$ for each $o \in O$ or those with laminar acceptability relations.

Theorem 5.2 (Classifying Tractable Instances for Pessimistic Consumption). *Finding a minimum sized pessimistically valid menu is*

- i. NP-complete even for instances with $s_o = 1$ for each $o \in O$ and
- ii. in P for instances with laminar acceptability relations

Remark 5.1 (Tightness of Intractability). *Both our hardness reductions Proposition 4.2 and Proposition 4.3 create instances where for any two agents $i, j \in N$, all three of the following sets may be non-empty: $A_i \setminus A_j$, $A_j \setminus A_i$ and $A_i \cap A_j$. When at most two are non-empty, we show that the acceptability sets must be laminar (formally defined in Section 5.2). Further, the reduction in the proof of Proposition 4.2 for optimistic consumption, constructs an instance with $s_o = 3$ for each option $o \in O$. In contrast, when $s_o \leq 2$ a minimum sized optimistically valid menu can be found in polytime via Algorithm 2. Consequently, our computational results are tight in this respect.*

5.1 Limited Serving Sizes

We now present our algorithm for $s_o \leq 2$. Note that this is a fairly realistic assumption as many restaurants only offer options for a single person or to be split between two. The algorithm constructs a graph with weights on edges to decide how to choose which agents should share an option with $s_o = 2$ and which agents require an individual option.

Algorithm 2: Min Sized Optimistic Menus when $s_o \leq 2$

Input: Menu Planning instance $\langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ where $s_o \in \{1, 2\}$ for all $o \in O$

Output: An optimistically valid menu M

- 1 Initialize $M \leftarrow \emptyset$;
 - 2 Create graph $G = (V, E)$ where for each $i \in N$, add vertex x_i ;
 - 3 For each $o \in O$ and $i \in N$ s.t. $s_o = 1$ and $o \in A_i$, add node y_o^i ;
 - 4 For each i, j s.t. there exists $o \in A_i \cap A_j$ with $s_o = 2$, add edge (x_i, x_j) with weight $w(x_i, x_j) = 2$;
 - 5 For each i and $o \in A_i$ with $s_o = 1$, add edge (x_i, y_o^i) with weight $w(x_i, y_o^i) = 1$;
 - 6 Let μ be a max weight matching on G under w ;
 - 7 **for each** $(x_i, x_j) \in \mu$ **do**
 - 8 Choose $o \in A_i \cap A_j$ such that $s_o = 2$;
 - 9 $M \leftarrow M \cup \{o\}$;
 - 10 **for each** $(x_i, y_o^i) \in \mu$ **do**
 - 11 $M \leftarrow M \cup \{o\}$;
 - 12 **for each unmatched** x_i **under** μ **do**
 - 13 Choose $o \in A_i$;
 - 14 $M \leftarrow M \cup \{o\}$;
 - 15 Return M ;
-

Proposition 5.3. *Given a MENU SELECTION $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ where for each $o \in O$, $s(o) \in \{1, 2\}$, Algorithm 2 finds a minimum sized optimistically valid menu in polynomial time.*

Proof Sketch. Recall that under optimistic consumption, it is enough to find a menu such that there is at least one way of partitioning the food order among the agents s.t. every agent finds something acceptable to eat. If each option has $s_o = 1$, a minimum sized optimistically valid menu can be

constructed by adding any one acceptable option for each agent $i \in N$. However, when some options have a serving size of 2 (or more), this simplistic approach may lead menus of much larger size than the minimum, as two people may be able to share the same option of larger size. To this end, we use a matching based approach in [Algorithm 2](#) to find minimum sized valid menus. \square

5.2 Structured Acceptability Relations

In [Section 5.1](#), we show intractability of computing min sized menus under both optimistic and pessimistic consumption for instances where given there exist pairs of agents $i, j \in N$, s.t. all three of $A_i \setminus A_j$, $A_j \setminus A_i$ and $A_i \cap A_j$ are non-empty. We show that if at most two of these sets are non-empty, we can in fact find min sized valid menu under either consumption model in polynomial time. We now discuss three different structural assumptions which emerge when we only allow at most two of the three aforementioned sets to be non-empty for any pair of agents.

Laminar Acceptability Relations. This is the most general structured setting that allows for tractability. Here, we assume that for any $i, j \in N$ where $|A_i| \leq |A_j|$, it must be that $A_i \cap A_j$ is either A_i or \emptyset . Further, we assume that there exists some $i^* \in N$ who finds all options acceptable, that is, $A_{i^*} = O$. Without this assumption, we can break the given instance into multiple independent instances. We can choose each maximal acceptability set A^v , where for each $i \in N$ either $A_i \subseteq A^v$ or $A_i \cap A^v = \emptyset$, and all its subsets separately. As a result, we assume without loss of generality that there exists an agent who finds all options acceptable.

Lemma 5.4. *Given an MENU SELECTION instance I , for every $i, j \in N$ at most two of $A_i \cap A_j$, $A_i \setminus A_j$ and $A_j \setminus A_i$ are non-empty if and only if I has laminar acceptability relations.*

With the assumption that there is a unique maximal set, laminar sets can be represented as a forest. Given a collection of distinct acceptability sets A^1, \dots, A^k that are laminar, there exists a tree with a vertex for each distinct set. We shall assume that this tree is rooted at the vertex corresponding to the acceptability set $A^v = O$. We call this the laminar containment tree where A^v is an ancestor of $A^{v'}$ if and only if $A^{v'} \subsetneq A^v$. In order to find min sized menus, we use an algorithm identical acceptability relations as a subroutine.

Identical Acceptability Relations. The simplest way of having a structure is to have all agents have the same acceptability relations. That is, for all $i, j \in N$, we have that $A_i = A_j$. Consequently, we denote an instance of Identical MENU SELECTION by $\langle N, A, (s_o)_{o \in A} \rangle$. Observe that we do not impose any restriction on the serving sizes of options. Despite this, we find positive results for finding both min sizes optimistically and pessimistically valid menus under identical acceptability relations. Identical acceptability relations capture settings where food needs to be ordered for a homogeneous group, such as at a religious gathering or a child's birthday party. We can generalize this setting further, while still retaining some structure.

Chained Acceptability Relations. Here, we assume that the acceptability relations of the agents form a sequence of subsets. That is, for any $i, j \in N$ where $|A_i| \leq |A_j|$, it must be that $A_i \subseteq A_j$. Chained acceptability relations capture dietary restrictions based on specific types of ingredients such as vegan, vegetarian, pescetarian, no red meat and unrestricted. Note that the laminar setting is a generalization of the chained setting.

We provide [Algorithm 3](#), in [Appendix D](#), to find a minimum sized valid menu under identical acceptability relations. Note that under identical acceptability relations every optimistically valid menu is pessimistically valid. We use this as a subroutine to find minimum sized valid menus. Our main results in the section are as follows. It is achieved via dynamic programming based solutions to finding minimum sized menus. As is typical with dynamic programming, the algorithms return minimum possible sizes of valid menus, but the actual menus can be found analogously.

Proposition 5.5. *Given a MENU SELECTION instance I with laminar acceptability relations, there exists a polynomial-time algorithm that finds the minimum size of a optimistically valid menu M .*

Proof Sketch. Laminar sets can be represented by a containment tree where if $A^i \subsetneq A^j$, then A^i must be an ancestor of A^j in the containment tree. Let A^1, \dots, A^k be the distinct acceptability sets labeled in a pre-order traversal of the containment tree. That is, if $A_i \subseteq A_j$ then $i \geq j$. The agents with acceptability set A^i can be fed from a combination of items across A^i or a combination of one child alone. Consequently, to find min sized menus, we need to determine the best combination of agents being fed from each acceptability set. To this end, for each internal node A^i , we need to decide how many of the corresponding agents eat from A^i and how many eat from each child of A^i .

To avoid going over all possible combinations here, we use the tree structure to build these combinations sequentially. We give a dynamic program in [Algorithm 5](#) that uses the pre-order traversal and is filled starting from leaves to root. Each node v other than the root inherits a “debt” from either its previous sibling (according to the label) or its parent (if no sibling comes earlier in the labeling), but not both. Now v needs to decide how much of this debt it will keep for the subtree rooted at v , and how much to pass off onto the next sibling, if any. This is done by going over all the values of the debt passed on and selecting the minimum. For one specific value, the debt retained plus n_v is the number of agents that need to be fed from the subtree rooted at v . Of this number, v now decides how many to feed from the set A^v and passes off the remaining to the first child (according to the labelling), if any. This is again done by going over all the values of the debt passed on and selecting the minimum. \square

We now give an overview of our algorithm to find minimum sized pessimistically valid menus for the laminar setting.

Proposition 5.6. *Given a MENU SELECTION instance I with laminar acceptability relations, there exists a polynomial-time algorithm that finds the minimum size of a pessimistically valid menu M .*

Proof Sketch. We refine the characterization of pessimistically valid menus for laminar acceptability relations in [Lemma D.2](#). Here, we show that a menu is pessimistically valid, if and only if the amount of food from each leaf of the laminar containment tree is enough to feed all agents whose acceptability set contains this leaf vertex. As a result, it is sufficient to only feed agents from these *minimal acceptability sets*. Based on this, we present [Algorithm 6](#) (in [Appendix D](#)) to compute a minimum sized pessimistically valid menu. The algorithm proceeds by considering the laminar containment tree and the t leaves A^{v_1}, \dots, A^{v_t} . For each leaf v_j it adds the minimum size pessimistic menu for feeding $n_{v_j} + \sum_{v: A^{v_j} \subsetneq A^v} n_v$ from A^{v_j} using the algorithm for identical acceptability relations as a subroutine. The combined sizes for each leaf is returned. \square

Example 5.1 (Min sized valid menus for Laminar Acceptability Relations). *Consider the following instance where $n = 12$ and $m = 4$. Let $A_1 = \{o_1\}$, $A_2 = A_3 = A_4 = \{o_2\}$, $A_5 = \{o_1, o_2\}$, $A_6 = \{o_3\}$ and $A_7 = \dots = A_{12} = \{o_1, o_2, o_3, o_4\}$. Further, set serving sizes as follows: $s_{o_1} = 2$, $s_{o_2} = s_{o_3} = 3$ and $s_{o_4} = 1$. Observe that any menu that does not contain $\{o_1, o_2, o_3\}$ will not be valid. Here the minimum sized optimistically valid menus will have size $n = 12$. Some examples are: $M_1 = \{o_1, o_2, o_3, o_4, o_4, o_4, o_4\}$, $M_2 = \{o_1, o_1, o_1, o_2, o_3\}$ and $M_3 = \{o_1, o_2, o_2, o_3, o_4\}$. Further, menu $M_4 = \{o_1, o_2, o_3, o_3, o_3\}$ will also be optimistically valid but has size 14.*

Recall that we need enough food from each minimal acceptability set to be pessimistically valid. Observe that exactly 8 agents find o_1 acceptable, 10 find o_2 acceptable and 7 find o_3 acceptable. Consequently, the unique pessimistically valid menu is $M_5 = \{o_1, o_1, o_1, o_1, o_2, o_2, o_2, o_2, o_3, o_3, o_3\}$ which has size 29.

Remark 5.2. *Given a graph $G = (V, E)$ where the neighborhoods of vertices are laminar, can find a minimum sized maximal matching in polynomial time.*

6 Waste of Pessimism

We now try to measure the difference in the menu sizes needed if pessimistic consumption is assumed but optimistic consumption is practiced. Informally, we are trying to measure how much extra food needs to be ordered to have a pessimistically valid menu vs an optimistically valid menu. We call this the *waste of pessimism* (WOP). Recall that $\Lambda(I)$ and $\Gamma(I)$ denote the spaces of optimistically and pessimistically valid menus, respectively.

Definition 6.1 (Waste of Pessimism). *Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$, the waste of pessimism is the ratio of the minimum sized pessimistically valid menu under I and that of the minimum sized optimistic menu*

$$\text{WoP}(I) = \frac{\min_{M \in \Gamma(I)} \sum_o q_o(M) s_o}{\min_{M \in \Lambda(I)} \sum_o q_o(M) s_o}.$$

Example 6.1 (Waste of Pessimism). *We now compute the waste of pessimism of instances discussed in previous examples.*

1. Consider [Example 1.1](#) with 4 agents $N = \{1, 2, 3, 4\}$ and two options $O = \{o_1, o_2\}$ where $s_{o_1} = 3$ and $s_{o_2} = 1$, and $A_1 = A_2 = A_3 = \{o_1\}$ and $A_4 = \{o_1, o_2\}$. Here, $\{o_1, o_2\}$ is an minimum sized optimistically valid menu. On the other hand $\{o_1, o_1\}$ is the min-sized pessimistically valid. The waste of pessimism for this instance is $6/4 = 3/2$.
2. Consider [Example 5.1](#). Recall that the minimum sized optimistically valid menu has size $n = 12$ and the minimum sized pessimistically valid menu has size 29. Here, the waste of pessimism is $29/12$.

Observe that the instance described in [Example 1.1](#) is one with chained acceptability relations, whereas the one in [Example 5.1](#) is one with laminar acceptability relations. We now prove tight bounds on the Waste of Pessimism for structured acceptability relations. Our bounds show that this stark contrast in waste of pessimism between laminar and chained scales as n increases.

Theorem 6.1. *Given a MENU SELECTION instance I , we have that*

1. *if I has laminar acceptability relations, $\text{WoP}(I) \leq \frac{n+4}{4}$,*
2. *if I has chained acceptability relations, $\text{WoP}(I) \leq 2$, and*
3. *if I has identical acceptability relations, $\text{WoP}(I) = 1$.*

Proof Overview. Fix a MENU SELECTION instance $I = \langle N, O, (s_o)_{o \in O}, (A_i)_{i \in N} \rangle$ with laminar acceptability relations. We first consider the laminar case. Given instance I , we shall create an alternate instance I' , also with laminar acceptability relations with the same number of agents and minimal acceptability sets, but which will always satisfy $\text{WoP}(I) \leq \text{WoP}(I')$ and then reason about its maximum possible waste of pessimism value. We will prove this bound as follows:

- i. Create an alternate instance I' with laminar acceptability relations, with the same number of agents and the same number of minimal acceptability sets.
- ii. Prove that $\text{WoP}(I) \leq \text{WoP}(I')$.
- iii. Find a worst case choice of serving sizes and number of minimal acceptability sets for WoP
- iv. Demonstrate an instance where this bound is tight.

Essentially, the alternate instance created is such that its corresponding laminar containment tree is a star graph that has the same number of leaves as that of I . Under I' , the root and the leaves of the two tree correspond to the same sets as those of the root and leaves under I . Further, under I' , exactly one agent finds has an acceptability set corresponding to each leaf. All other agents have acceptability set corresponding to the root. From [Lemma D.2](#), the amount of food needed from each leaf under I' can only increase now. We use this to show that $WoP(I) \leq WoP(I')$. As I' significantly structured, we can argue about its worst case bounds. We show that the worst case WoP happens when the number of leaves is $n/2$ and the minimum serving size of any option in a minimal acceptability set has size at least two. \square

7 Conclusions and Discussion

We have introduce the problem of menu selection and study its computational complexity. We characterize the space of instances that allow for polynomial-time algorithms for computing minimum sized optimistically valid menus. For the pessimistic case, we show that for some realistic settings, minimum sized pessimistically valid menus can in fact be computed efficiently. We introduce a price of anarchy analog for this setting, *the waste of pessimism*. We find tight upper bounds on the price of anarchy for the case of laminar, chained and identical acceptability relations.

Our paper opens many interesting avenues for future work. Firstly, there is no known tight bound on waste of pessimism outside of laminar instances. We conjecture that the upper bound for laminar should in fact extend to *all* instances. Additionally, there are many other directions that can be pursued within this model. One important direction is to develop approximation algorithms which find valid menus of size at most $\alpha \geq 1$ times the minimum sized valid menu. Similarly, it would be interesting to explore how to build “minimal” valid menus and understand what guarantee they achieve on the minimum sized valid menu.

Our paper offers a new perspective on the literature on online matching problems (e.g., [\[24, 35, 39\]](#)). In online bipartite matching, one side of the graph is known in advance, while vertices on the other side arrive sequentially with their incident edges, and the algorithm must irrevocably decide whether to match each arrival to an available neighbor. The objective is to maximize the size or weight of the matching, a model that captures many real-time allocation settings and has been widely studied under adversarial, random-order, and stochastic arrivals. Our problem can be seen as a preceding decision problem: selecting a subset of the known side under capacity or cost constraints. While we adopt optimistic and pessimistic perspectives, it would be interesting to incorporate ideas from online matching that exploit stochastic information about arrivals or consumption, e.g., kidney matching [\[21, 22\]](#) or online recommender systems settings [\[1\]](#).

Another direction would be to introduce additional objectives or constraints such as costs over options. Alternately, there may be complementary options where a protein and/or vegetable based option, such as a burger, steak or salad, needs to be paired with a carbohydrate based option like rice or bread. Another constraint may be that different agents have different hunger levels and need different amounts of food. For instance, adults need one portion but kids may only need half a portion. Our DP based algorithms extend to this case in a straightforward manner, as we simply need to adjust the target goal.

Finally, it would also be interesting to introduce agent preferences over acceptable options. This allows us to pursue objectives such as Pareto optimality, maximum welfare and even fairness in our choice of menu. Potentially, this may allow for lower pessimistic risk, if agents only eat their most preferred options from a given menu. As a result, our work opens many intriguing possibilities of further work on menu selection with applications to other collective choice settings [\[2, 6\]](#).

References

- [1] Amanda Aird, Paresha Farastu, Joshua Sun, Elena Stefancová, Cassidy All, Amy Voida, Nicholas Mattei, and Robin Burke. 2024. Dynamic fairness-aware recommendation through multi-agent social choice. *ACM Transactions on Recommender Systems* 3, 2 (2024), 1–35. (Cited on page 18)
- [2] Martin Aleksandrov and Toby Walsh. 2020. Online fair division: A survey. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 13557–13562. (Cited on page 18)
- [3] Georgios Amanatidis, Haris Aziz, Georgios Birmpas, Aris Filos-Ratsikas, Bo Li, Hervé Moulin, Alexandros A Voudouris, and Xiaowei Wu. 2023. Fair division of indivisible goods: Recent progress and open questions. *Artificial Intelligence* 322 (2023), 103965. (Cited on page 23)
- [4] Udo Arend, Klaus-Peter Muthig, and Jens Wandmacher. 1987. Evidence for global feature superiority in menu selection by icons. *Behaviour & Information Technology* 6, 4 (1987), 411–426. (Cited on page 24)
- [5] H. Aziz, I. Schlotter, and T. Walsh. 2016. Control of Fair Division. In *Proceedings of the 25th IJCAI conference*. (Cited on page 23)
- [6] Haris Aziz and Nisarg Shah. 2020. Participatory budgeting: Models and approaches. In *Pathways Between Social Science and Computational Social Science: Theories, Methods, and Interpretations*. Springer, 215–236. (Cited on page 18)
- [7] Ashwinkumar Badanidiyuru, Shahar Dobzinski, Hu Fu, Robert Kleinberg, Noam Nisan, and Tim Roughgarden. 2012. Sketching valuation functions. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*. SIAM, 1025–1035. (Cited on page 24)
- [8] Ashwinkumar Badanidiyuru, Shahar Dobzinski, and Sigal Oren. 2012. Optimization with demand oracles. In *Proceedings of the 13th ACM conference on electronic commerce*. 110–127. (Cited on page 24)
- [9] Siddhartha Banerjee, Vasilis Gkatzelis, Safwan Hossain, Billy Jin, Evi Micha, and Nisarg Shah. 2023. Proportionally Fair Online Allocation of Public Goods with Predictions. In *32nd International Joint Conference on Artificial Intelligence*. 32nd International Joint Conference on Artificial Intelligence. (Cited on page 23)
- [10] Siddharth Barman, Anand Krishna, Pooja Kulkarni, and Shivika Narang. 2024. Sublinear Approximation Algorithm for Nash Social Welfare with XOS Valuations. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 8–1. (Cited on page 24)
- [11] Siddharth Barman and Paritosh Verma. 2021. Approximating Nash social welfare under binary XOS and binary subadditive valuations. In *International Conference on Web and Internet Economics*. Springer, 373–390. (Cited on page 24)
- [12] Matthias Bentert, Robert Brederick, Eva Deltl, Pallavi Jain, and Leon Kellerhals. 2025. How to Resolve Envy by Adding Goods. In *34th International Joint Conference on Artificial Intelligence, IJCAI 2025*. International Joint Conferences on Artificial Intelligence, 3753–3761. (Cited on page 23)

- [13] Arpita Biswas and Siddharth Barman. 2018. Fair Division Under Cardinality Constraints.. In *IJCAI*. 91–97. (Cited on page 24)
- [14] Andreas Björklund, Thore Husfeldt, and Mikko Koivisto. 2009. Set partitioning via inclusion-exclusion. *SIAM J. Comput.* 39, 2 (2009), 546–563. (Cited on pages 12, 29, 30, and 31)
- [15] Federico Bobbio, Margarida Carvalho, Andrea Lodi, Ignacio Rios, and Alfredo Torrico. 2023. Capacity Planning in Stable Matching: An Application to School Choice. In *Proceedings of the 24th ACM Conference on Economics and Computation*. 295. (Cited on page 23)
- [16] Federico Bobbio, Margarida Carvalho, Andrea Lodi, and Alfredo Torrico. 2026. Capacity variation in the many-to-one stable matching. *Operations Research Letters* (2026), 107416. (Cited on page 23)
- [17] Niclas Boehmer, Robert Brederbeck, Klaus Heeger, and Rolf Niedermeier. 2021. Bribery and Control in Stable Marriage. *Journal of Artificial Intelligence Research* 71 (2021), 993–1048. (Cited on page 23)
- [18] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. 2011. Submodular function maximization via the multilinear relaxation and contention resolution schemes. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*. 783–792. (Cited on page 24)
- [19] Jiehua Chen and Gergely Csáji. 2023. Optimal Capacity Modification for Many-To-One Matching Problems. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. 2880–2882. (Cited on page 23)
- [20] Jiehua Chen, Joanna Kaczmarek, Paul Nüsken, Jörg Rothe, Ildikó Schlotter, and Tessa Seeger. 2025. Control in computational social choice. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*. 10391–10399. (Cited on page 23)
- [21] John P Dickerson, Ariel D Procaccia, and Tuomas Sandholm. 2012. Optimizing kidney exchange with transplant chains: Theory and reality. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*. 711–718. (Cited on page 18)
- [22] John P Dickerson, Karthik A Sankararaman, Aravind Srinivasan, and Pan Xu. 2021. Allocation problems in ride-sharing platforms: Online matching with offline reusable resources. *ACM Transactions on Economics and Computation (TEAC)* 9, 3 (2021), 1–17. (Cited on page 18)
- [23] Shahar Dobzinski, Wenzheng Li, Aviad Rubinfeld, and Jan Vondrák. 2024. A constant-factor approximation for nash social welfare with subadditive valuations. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. 467–478. (Cited on page 24)
- [24] F. Echenique, N. Immorlica, and V. V. Vazirani. 2023. *Online and Matching-Based Market Design*. Cambridge University Press. (Cited on pages 18 and 23)
- [25] Jack Edmonds. 1965. Paths, trees, and flowers. *Canadian Journal of mathematics* 17 (1965), 449–467. (Cited on page 32)
- [26] Brandon Fain, Kamesh Munagala, and Nisarg Shah. 2018. Fair allocation of indivisible public goods. In *Proceedings of the 2018 ACM Conference on Economics and Computation*. 575–592. (Cited on page 23)

- [27] Uriel Feige. 2006. On maximizing welfare when utility functions are subadditive. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*. 41–50. (Cited on page 24)
- [28] Sara Fish, Yannai A Gonczarowski, and Sergiu Hart. 2025. Stable Menus of Public Goods: A Matching Problem. In *Proceedings of the 26th ACM Conference on Economics and Computation*. 348–348. (Cited on pages 2 and 23)
- [29] Annika N Flynn, Taro Takahashi, Alex Sim, and Jeffrey M Brunstrom. 2025. Dish swap across a weekly menu can deliver health and sustainability gains. *Nature Food* 6, 9 (2025), 843–847. (Cited on page 2)
- [30] Michael R Garey and David S Johnson. 1979. *Computers and intractability*. Vol. 174. freeman San Francisco. (Cited on pages 8, 9, 23, 26, and 27)
- [31] Jugal Garg, Pooja Kulkarni, and Aniket Murhekar. 2021. On Fair and Efficient Allocations of Indivisible Public Goods. In *41st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2021)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 22–1. (Cited on page 23)
- [32] Salil Gokhale, Samarth Singla, Shivika Narang, and Rohit Vaish. 2024. Capacity Modification in the Stable Matching Problem. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*. 697–705. (Cited on page 23)
- [33] Balakrishna Grandhi and Jyothisna Appaiah Singh. 2016. What a waste! A study of food wastage behavior in Singapore. *Journal of Food Products Marketing* 22, 4 (2016), 471–485. (Cited on page 23)
- [34] P Hall. 1935. On Representatives of Subsets. *Journal of the London Mathematical Society* 1, 1 (1935), 26–30. (Cited on page 10)
- [35] Zhiyi Huang, Zhihao Gavin Tang, and David Wajc. 2024. Online Matching: A Brief Survey. *SIGecom Exch.* 22, 1 (2024), 135–158. <https://doi.org/10.1145/3699824.3699837> (Cited on pages 18 and 23)
- [36] Jin-Woo Jung. 2014. A study on the influence of menu selection attributes and design of western restaurants on the customer value and customer satisfaction. *Culinary science and hospitality research* 20, 6 (2014), 69–79. (Cited on page 24)
- [37] Yeon-Sook Kang, Hun-Jin Park, and Jin-Woo Jung. 2011. A study on the effects of the selection attributes of Korean restaurant menu on customer satisfaction and revisit intention. *Culinary science and hospitality research* 17, 2 (2011), 1–17. (Cited on page 24)
- [38] John Karat, James E McDonald, and Matt Anderson. 1986. A comparison of menu selection techniques: touch panel, mouse and keyboard. *International Journal of Man-Machine Studies* 25, 1 (1986), 73–88. (Cited on page 24)
- [39] Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. 1990. An Optimal Algorithm for On-line Bipartite Matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, Harriet Ortiz (Ed.). ACM, 352–358. <https://doi.org/10.1145/100216.100262> (Cited on pages 18 and 23)

- [40] Christian Kroer and Dominik Peters. 2025. Computing Lindahl Equilibrium for Public Goods with and without Funding Caps. In *Proceedings of the 26th ACM Conference on Economics and Computation*. 129–129. (Cited on page 23)
- [41] Martin Lackner and Piotr Skowron. 2023. *Multi-winner voting with approval preferences*. Springer Nature. (Cited on page 23)
- [42] Hendrik W Lenstra Jr. 1983. Integer programming with a fixed number of variables. *Mathematics of operations research* 8, 4 (1983), 538–548. (Cited on page 29)
- [43] Manita Matharu, Neha Gupta, and Vikas Swarnakar. 2022. Efforts are made but food wastage is still going on: a study of motivation factors for food waste reduction among household consumers. *Asia-Pacific Journal of Business Administration* 14, 2 (2022), 244–264. (Cited on page 23)
- [44] Natália Valmorbida Moraes, Fernando Henrique Lermen, and Márcia Elisa Soares Echeveste. 2021. A systematic literature review on food waste/loss prevention and minimization methods. *Journal of Environmental Management* 286 (2021), 112268. (Cited on page 23)
- [45] Bahattin Ozdemir and Osman Caliskan. 2015. Menu design: A review of literature. *Journal of Foodservice Business Research* 18, 3 (2015), 189–206. (Cited on page 24)
- [46] Sandeep Kumar Panda and Madhukar Dwivedi. 2019. Minimizing food wastage using machine learning: A novel approach. In *Smart Intelligent Computing and Applications: Proceedings of the Third International Conference on Smart Computing and Informatics, Volume 1*. Springer, 465–473. (Cited on pages 2 and 23)
- [47] Aviyan Pandey. 2021. Food wastage: Causes, impacts and solutions. *Sci. Herit. J* 5, 1 (2021), 17–20. (Cited on page 23)
- [48] Kimberley Peters and Pr Hervé Remaud. 2020. Factors influencing consumer menu-item selection in a restaurant context. *Food Quality and Preference* 82 (2020), 103887. (Cited on page 24)
- [49] Michael D Plummer and László Lovász. 1986. *Matching theory*. Vol. 29. Elsevier. (Cited on page 5)
- [50] Ben Shneiderman. 1986. Designing menu selection systems. *Journal of the American Society for information science* 37, 2 (1986), 57–70. (Cited on page 24)
- [51] Douglas Brent West et al. 2001. *Introduction to graph theory*. Vol. 2. Prentice hall Upper Saddle River. (Cited on page 26)
- [52] Graham Wilson, Craig Stewart, and Stephen A Brewster. 2010. Pressure-based menu selection for mobile devices. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*. 181–190. (Cited on page 24)

A Related Work

The problem of minimizing food waste and its various causes have been studied from a variety of systemic angles [33, 43, 44, 46, 47]. Poor planning and overpurchasing has been consistently cited as a cause of food waste. In our paper, we take a complementary angle to the previous work, where we look at an algorithmic approach to choosing the most efficient menu to feed all agents. We introduce a novel framework, and to the best of our knowledge, no other work has studied an algorithmic model where every agent must find something to eat from a collective, waste minimizing menu.

Our model has connections with two problems in computational social choice: multi-winner voting (see e.g., Lackner and Skowron [41]) and fair division (see Amanatidis et al. [3] for a survey). Similar to multi-winner voting, we need to select one menu from a given set of options which will then be shared among the agents. Unlike voting problems, we assume an upper bound (via serving sizes) on how much of an option can serve an agent. Once a menu has been selected, one can consider ways to allocate the food among the agents. Unlike standard allocation settings, we can choose the options that will then be implicitly shared across agents.

Recently, Fish et al. [28] studied a related problem within a public good setting. They look for a stable selection of public goods such that each item is used by at least s agents and the menu is not blocked by more than t agents. In contrast to our model, as Fish et al. [28] consider public goods, where each option effectively has unlimited serving sizes as opposed to limited serving sizes in our model. Further, they do not require all agents to have at least one acceptable option in the menu, and some agents may go hungry. These assumptions are common when studying public goods [9, 26, 31, 40]. However, they need not hold in many day-to-day settings of ordering food.

Our problem, under the pessimistic consumption model, can be viewed as a natural decision problem that *precedes* online matching problems (see, e.g., [24, 35, 39]). Online matching has been well studied since the work of Karp et al. [39]. We discuss this connection in more detail in the Conclusions. Bipartite one-one matchings are also equivalent to house allocation with approval preferences. This perspective also connects our work to control problems.

Relations To Control and Optimization Our work also extends existing work on control problems within social choice (see Chen et al. [20] for a survey). Within social choice, control problems typically explore computational complexity of adding or removing agents/items to ensure the existence of solutions that satisfy certain properties [5, 17, 19, 32]. Our work differs from existing research in this space on two fronts. Firstly, the work in this space has been agnostic to the choice of mechanism or algorithm. In contrast, we introduce the question of controlling with a specific algorithm in mind. Secondly, prior work on control and social choice has not considered such “house allocation” type problems where agents have to each be matched to one item, but rather either looked at fair division [5, 12] or two-sided matchings [15, 16, 17, 19, 32]. The majority of the work in this space is also an *optimization problem*: find the minimum number of items/agents to add or remove so desirable outcomes can exist.

Covering problems are another family of optimization problems that relates to our problem under optimistic consumption. Problems like set cover, vertex cover, domination number are well known to be NP-hard [30]. Our model is a generalization of these problems. For example, when the serving size of each option is equal to the number of people who find it acceptable, then a minimum size optimistically valid menu is the same as a minimum size set cover. In addition to covering every agent by an option, we also restrict the ability of an option to simultaneously cover multiple agents based on its serving size. Consequently, multiple copies of an option may be needed to feed all the agents. Our goal of finding smallest menus that feed all agents can be also recast as an optimization problem where the number of agents that are fed are captured by a subadditive

function. We wish to find the smallest menu for which our objective function has value equal to the number of agents. Subadditive functions and their subtypes have been widely studied in the context of welfare maximization for item allocation [10, 11, 23, 27] and combinatorial procurement auctions [7, 8, 18]. Typically, there is no minimum size requirement on the set selected. Rather, in work on item allocation, there may be an upper bound on the cardinality of the set allocated to an agent [8, 13]. Here, the optimization objective is the value of the function, not the sets. In contrast, we have a given value of our function and we wish to pick the smallest set to achieve this value.

The term *menu selection* has been used in the past to refer to other research problems, especially in human-computer interaction (HCI). In the field of HCI, menu selection is concerned with what options to show to a user and how should they be presented in a drop-down menu or list [4, 38, 50, 52]. Outside of computer science, food researchers have also heavily studied the effect of menu selection on customer experience and restaurant sales [36, 37, 45, 48]. Here, menu selection refers to not only the dishes on offer but the fonts, designs, options, and other interface details.

B Omitted Material from Section 3

Optimistic Consumption

Theorem 3.1. [Characterization of optimistically valid menus] Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and a menu M , the following conditions are equivalent:

1. M is optimistically valid;
2. The consumption graph G_M admits matching of size n ;
3. There exists a picking sequence π and a possible outcome ω of π on M s.t. for each $k \in [n]$, $\omega(k) \neq \emptyset$.

Proof. Fix an instance I and a menu M . Recall that the optimistic consumption is precisely the size of the maximum cardinality matching in G_M . Thus the equivalence of statements (1) and (2) is straightforward. We now show the equivalence of these statements with there existing a consumption sequence ω .

Let M is optimistically valid. Choose x to be a maximum cardinality matching in the consumption graph G_M . Consequently, we have that for every agent $i \in N$, there exists $o \in A_i$ s.t. $x_{i,o} = 1$. For each $k \in [n]$, let π be an arbitrary ordering over N and set $\omega(k) = o$ s.t. $x_{\pi(k),o} = 1$. As x is a matching, we have that for each option $o \in O$, $\sum_{i \in N} x_{i,o} \leq q_o s_o$. Thus, for every agent $k \in [n]$, it must hold that the remaining menu M^k , after previous agents have picked, must satisfy $M^k|_{A_{\pi(k)}}$. Specifically, it must hold that $o \in M^k|_{A_{\pi(k)}}$ where o is s.t. $x_{\pi(k),o} = 1$. Thus, ω is indeed a consumption sequence of M .

For the converse, let there exist a consumption sequence ω s.t. for each $k \in [n]$, $\omega(k) \neq \emptyset$. Let π be a corresponding ordering over agents. Consider the matching x where for each $k \in [n]$, $x_{\pi(k),\omega(k)} = 1$ and $x_{\pi(k),o} = 0$ for all $o \neq \omega(k)$. As ω is a consumption sequence $\omega(k) \in A_{\pi(k)}$ and $q_{\omega(k)}(M) > 0$. Further, the number of agents matched to option o : $\sum_{i \in N} x_{i,o} = |\{k \in [n] | \omega(k) = o\}| \leq q_o s_o$. Therefore, x is indeed a matching of the consumption graph G_M and has size n . Therefore, M must be optimistically valid. \square

Pessimistic Consumption

Theorem 3.2. [Characterization of pessimistically valid menus] Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and a menu M , the following conditions are equivalent:

1. M is pessimistically valid;
2. For every agent $i \in N$, we have that $\text{size}(M|_{A_i}) = \sum_{o \in A_i} q_o(M) s_o \geq 1 + \rho_i(M)$;

3. For every picking sequence π and every $\omega \in \Omega(\pi, M)$, we have that for each $k \in [n]$, $\omega(k) \neq \emptyset$.

Proof. We shall first prove equivalence of pessimistic validity to the condition on pessimistic risk for each agent.

Equivalence of statements (1) and (2). Assume M is pessimistically valid. Thus every maximal matching in the consumption graph G_M must have size n . Fix agent $i \in N$. Consider a maximal matching where as many agents eat from A_i as possible. In particular, we consider a maximum cardinality matching from $N \setminus \{i\}$ to the multiset M^i where $q_o(M^i) = q_o(M)$ for all $o \in A_i$ and $q_o(M^i) = 0$ for all $o \notin A_i$. Note that the size of this matching is $\rho_i(M)$. As M is pessimistically valid, this matching must not be maximal. In particular, it must hold that $\rho_i(M) < \sum_{o \in A_i} q_o(M)s_o$. Consequently, we have that for all agents $i \in N$, $\sum_{o \in A_i} q_o(M)s_o \geq 1 + \rho_i(M)$.

Now it remains to prove the converse. Assume that M is s.t. for all agents $i \in N$, it holds that $\rho_i(M) < \sum_{o \in A_i} q_o(M)s_o$. Consequently, we have that for all agents $i \in N$, Fix agent $i \in N$ and an arbitrary maximal matching x in consumption graph G_M . It suffices to prove that for i and x we have that $\sum_{o \in A_i} x_{i,o} = 1$.

By assumption, we have that $\sum_{o \in A_i} q_o(M)s_o \geq 1$. Consequently, for at least one option in $o \in A_i$, it must be that $q_o(M) \geq 1$. By definition of pessimistic risk, at most $\rho_i(M)$ agents in $N \setminus \{i\}$ can be matched to an option in A_i . As we have that $\sum_{o \in A_i} q_o(M)s_o \geq 1 + \rho_i(M)$, any maximal matching should not leave agent i unmatched. Thus, as x is a maximal matching, it must hold that $\sum_{o \in A_i} x_{i,o} = 1$. Hence, M is pessimistically valid.

Equivalence of statement (3) to pessimistic validity. Let M be pessimistically valid. Consider a consumption sequence ω of M with a corresponding ordering π . Recall that for any $k \in [n]$, each preceding agent, under π , can consume at most one unit of an option acceptable to agent i . Consequently, we have that the difference between the number of units of acceptable options for agent $\pi(k)$ under M and that under M^k is $size(M|_{A_{\pi(k)}}) - size(M^k|_{A_{\pi(k)}}) \leq \rho_i$. As M is pessimistically valid, we have that $size(M|_{A_i}) = \sum_{o \in A_i} q_o(M)s_o \geq 1 + \rho_i(M)$. Consequently, $M^k|_{A_{\pi(k)}}$ must be non-empty. As a result, $\omega(k) \neq \emptyset$ for each $k \in [n]$.

Finally, for the converse, let M be s.t. for every consumption sequence ω of M , we have that for each $k \in [n]$, $\omega(k) \neq \emptyset$. Fix agent $i \in N$. Let x be maximum cardinality matching between $N \setminus \{i\}$ and $M|_{A_i}$. Consider an arbitrary ordering π under which $\pi(n) = i$. That is, i is the last agent to pick from M . Let $N^i = \{j \in N \setminus \{i\} \mid \sum_{o \in A_j} x_{j,o} = 1\}$. Consider a choice of ω s.t. for $\pi(k) \in N^i$ and the option o for which $x_{\pi(k),o} = 1$, we set $\omega(k) = o$. For all $k \in [n]$ s.t. $k \notin N^i$, choose $\omega(k)$ arbitrarily from M^k , whenever M^k is non-empty. It is straightforward to see that ω is indeed a consumption sequence for M . Consequently, it must hold that $\omega(n) \neq \emptyset$, by assumption. As a result, we must have that $M^n|_{A_i}$ must be non-empty. This gives us that the size of x if $\rho_i = \sum_{j \neq i} \sum_{o \in A_j} x_{j,o} < \sum_{o \in A_i} q_o(M)s_o$.

Hence, we have that for each $i \in N$, $size(M|_{A_i}) > \rho_i$ and thus, M is pessimistically valid. \square

C Omitted Material from Section 4

Computational Complexity

Proposition 4.1. Given a MENU SELECTION instance I and a menu M

1. the amount of M that is optimistically consumed $\lambda(M)$ can be computed in polynomial time,
2. it is NP-hard to compute the amount of M that is pessimistically consumed $\gamma(M)$, and
3. it can be checked in polynomial time if M is optimistically and/or pessimistically valid.

Proof. Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and menu M , in order to check if M is optimistically valid we need to check if there is a matching $x : N \times O \rightarrow [0, 1]$ where for each agent $i \in N$, $\sum_{o \in A_i} x_{i,o} = 1$ and for each option $o \in O$, $\sum_{i \in N} x_{i,o} \leq q_o(M) s_o$. This can be completed in polynomial time using a bipartite maximum matching algorithm (See West et al. [51] for an overview).

In order to compute the pessimistic consumption of M , we need to find a minimum sized maximal matching. The problem of finding a minimum size maximal matching is known to be NP-hard even on 3-regular bipartite graphs (GT10 in Appendix A1.1 [30]). Consequently the problem of computing the pessimistic consumption of a given menu is NP-hard in general.

Checking validity of M . To check if M is optimistically valid, it is sufficient to check if its consumption graph admits an agent saturating matching. This can be done in polynomial time using a bipartite maximum matching algorithm. For pessimistic validity, we know that we cannot check the pessimistic consumption in polynomial time. Recall from [Theorem 3.2](#) that M is pessimistically valid if and only if for every agent $i \in N$, $\text{size}(M|_{A_i}) \geq 1 + \rho_i$. Note that, ρ_i is the size of the maximum cardinality matching between $N \setminus \{i\}$ and $M|_{A_i}$. As a result, ρ_i can be computed by a bipartite maximum matching algorithm as well. Hence, it can be checked in polynomial time if M is pessimistically valid. \square

Proposition 4.2. *Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and size $B \geq n$ where $s_o \geq 3$, it is NP-complete to decide if there exists an optimistically valid menu of size at most B .*

Proof. Membership in NP. Given instance I and a size B , we can verify if I admits a valid menu of size at most B , by using the menu as a certificate. It can be checked in polynomial time if its size is at most B and whether it is optimistically valid, by [Proposition 4.1](#).

NP-hardness. We give a reduction from the 3DIMENSIONALMATCHING (3DM) problem. In this problem, we are given a tripartite hypergraph $G = (X, Y, Z, E)$ where $|X| = |Y| = |Z| = k$ and for each $e \in E$ we have that $e \in X \times Y \times Z$. We need to decide if there exists a (perfect) 3DM of size k . This problem is shown to be NP-hard, even when each vertex is contained in at most two hyperedges [30]. We reduce it to our setting as follows:

Given $G = (X, Y, Z, E)$ we create a vertex agent i_a for each vertex $a \in X \cup Y \cup Z$. Further, for each $e = (x, y, z) \in E$ we create an edge option o_e s.t. $\text{size}(o_e) = 3$ $o_e \in A_{i_a}$ if and only if $a \in \{x, y, z\}$. Thus we have $n = 3k$ agents and $m = |E|$ edges.

We now show that that an optimistically valid menu of size $n = 3k$ exists if and only if the given 3DM instance has a 3DM of size k .

Suppose an optimistically valid menu of size $n = 3k$ does indeed exist. Let this be M . Under M , for each $i \in N$, there exists $o \in A_i$ s.t. $q_M(o) \geq 1$. Recall that each option has serving size three. As M has size $3k$, at most k options must have been selected. Further, each option is contained in the acceptable set of exactly three agents, therefore, at least k distinct options are needed to feed all agents. As a result, we must have that $q_M(o) \leq 1$, for all $o \in O$. Consequently, we have that the set $\mu = \{e \in E | q_M(o) = 1\}$ must be a perfect matching.

For the converse, given a perfect matching $\mu \subset E$, we consider the menu $M = \{o_e | e \in \mu\}$. As μ is a perfect matching, for each $i \in N$, there exists an option $o \in M \cap A_i$. Further, as $s(o) = 3$ for each o , we have that one serving of o serves all three agents who find it acceptable. Consequently M is an optimistically valid menu. \square

Proposition 4.3. *Given a MENU SELECTION instance $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ and size $B \geq n$ where $s_o = 1$, it is NP-complete to decide if there exists a pessimistically valid menu of size at most B .*

Proof. Membership in NP. Given instance I and a size B , we can verify if I admits a valid menu of size at most B , by using the menu as a certificate. It can be checked in polynomial time if its size is at most B and whether it is pessimistically valid, by [Proposition 4.1](#).

NP-hardness. We give a reduction from the EXACT3COVER (X3C) problem, which is known to be NP-hard [30]. In this problem, given a universe of $3k$ elements \mathcal{U} and a family of ℓ subsets \mathcal{S} , where for each $S_t \in \mathcal{S}$, we have that $|S_t| = 3$. The objective is to determine if there exist k mutually disjoint subsets which cover \mathcal{U} . We shall show that an X3C exists for $\langle \mathcal{U}, \mathcal{S} \rangle$ if and only if a pessimistically valid menu of size n exists. Given an instance of X3C $\langle \mathcal{U}, \mathcal{S} \rangle$, we create a MENU SELECTION instance as follows:

- For each element $e \in \mathcal{U}$, create element agent i_e ,
- For each subset $S_t \in \mathcal{S}$, create three set agents j_1^t, j_2^t and j_3^t and options o_1^t, \dots, o_4^t .
- For each $o \in O$, we set $s_o = 1$
- For each $S_t \in \mathcal{S}$, let $S_t = \{e_1^t, e_2^t, e_3^t\}$. For each $t' \in [3]$, $A_{j_{t'}^t} = \{o_{t'}^t, o_4^t\}$. Further, add $o_{t'}^t$ to $A_{i_{e_{t'}}}$.

Consequently, we create an instance with $n = 3(k + \ell)$ agents and $m = 4\ell$ options. For each $e \in \mathcal{U}$, A_{i_e} contains exactly one option corresponding to each subset containing e . Note that the exact specification of which $o_{t'}^t$ is matched to which element agent is not strict. It is only needed that each element agent find a distinct option from the set acceptable. Further each $o_{t'}^t$ is acceptable to exactly two agents: one element agent and one set agent. As a result, for each element agent i_e and each $o \in A_{i_e}$, there is a distinct set agent who also finds that option acceptable. Similarly, for each set agent $j_{t'}^t$ only $o_{t'}^t$ and o_4^t are acceptable. Note that $j_{t'}^t$ is the only agent who finds both these options acceptable.

From [Theorem 3.2](#), we know that for a menu M to be pessimistically valid we need for each $i \in N$ that $\sum_{o \in A_i} q_o s_o \geq 1 + \rho_i$, where ρ_i is the size of the maximum cardinality matching from $N \setminus \{i\}$ to $M|_{A_i}$. For an element agent i_e , we have that $\rho_i = |\{o \in A_i | q_o(M) > 0\}|$. Consequently, for a menu M to be pessimistically valid, we need for every element agent i_e , it must hold that $q_o(M) \geq 2$ for some $o \in A_i$. Now consider a set agent $j_{t'}^t$. We can analogously see that if $q_{o_{t'}^t}(M) \leq 1$ AND $q_{o_4^t}(M) \leq 2$ then M is not pessimistically valid for $j_{t'}^t$. Thus, for M to be pessimistically valid, we need for each $t \in [\ell]$ and each $t' \in [3]$ that either $q_{o_{t'}^t} \geq 2$ or $q_{o_4^t} \geq 3$.

Constructing menu from X3C. We shall now show that an X3C exists for $\langle \mathcal{U}, \mathcal{S} \rangle$ if and only if a pessimistically valid menu of size n exists in the instance constructed. Let S_{t_1}, \dots, S_{t_k} be an exact 3-cover. Consider the following menu M where for each $t \notin \{t_1, \dots, t_k\}$, we set $q_{o_4^t} = 3$ and for each $t' \in \{1, 2, 3\}$ we set $q_{o_{t'}^t}(M) = 0$. For each $t \in \{t_1, \dots, t_k\}$, we set $q_{o_4^t} = 0$ and for each $t' \in \{1, 2, 3\}$ we set $q_{o_{t'}^t}(M) = 2$. The size of M is $6k + 3(\ell - k) = 3(\ell + k) = n$. Thus, it only remains to prove pessimistic validity.

We first show pessimistic validity for element agents i_e . Observe that as S_{t_1}, \dots, S_{t_k} is an X3C, for each $e \in \mathcal{U}$, there exists $t \in \{t_1, \dots, t_k\}$ s.t. $e \in S_t$. Note that, we have $o_{t'}^t \in A_{i_e}$ for some $t' \in [3]$ and $q_{o_{t'}^t}(M) = 2$. Further, as there is a unique set covering e , we have that $\rho_{i_e} = 1$. Therefore, M is pessimistically valid for each element agent i_e .

For set agent $j_{t'}^t$, there are two possible cases. If $t \in \{t_1, \dots, t_k\}$, we have that $q_{o_{t'}^t}(M) = 2$ and $q_{o_4^t}(M) = 0$. Consequently, $\rho_{j_{t'}^t} = 1$ and M is pessimistically valid for $j_{t'}^t$. The remaining case is when $t \notin \{t_1, \dots, t_k\}$. In this case, we have that $q_{o_4^t} = 3$ and $q_{o_{t'}^t} = 0$. Here, we have that $\rho_{j_{t'}^t} = 2$ which is strictly less than $q_{o_4^t}(M)$. Thus, M is pessimistically valid and has size n .

Constructing X3C from menu. We now consider the case that there exists a pessimistically valid menu M of size n . We now show that we can use it to construct an exact 3-cover. Recall that we need for pessimistic validity that for each set S_t , either $q_{o_4^t} \geq 3$ or $q_{o_{t'}^t} \geq 2$ for each $t' \in \{1, 2, 3\}$. We first prove that for each element agent exactly one acceptable option must be in M . Fix $e \in \mathcal{U}$. We know that for pessimistic validity, it must hold that for at least one $o \in A_{i_e}$, $q_o \geq 2$. Suppose, for contradiction, that there are two distinct options $o \in A_{i_e}$ s.t. $q_o > 0$. As the size of the menu is exactly n , by pigeonhole principle this leaves at least one agent for whom no acceptable option in M , which contradicts the fact that M is pessimistically valid. Therefore, M must have exactly one option s.t. $q_o(M) > 0$. Specifically, for this option, $q_o(M) = 2$.

For agent i_e let the unique option be $o_{t'}^t$ for some $t' \in [3]$. We now show that this must mean that $q_{o_1^t}(M) = q_{o_2^t}(M) = q_{o_3^t}(M) = 2$. Suppose not, that let $q_{o_p^t} < 2$ for some $p \in [3]$. Thus, for pessimistic validity, we need that $q_{o_4^t} \geq 3$. This again means that there is more food in M from options corresponding to S_t than there are agents consuming it, which contradicts the pessimistic validity of M which has size n . Therefore, we have that under M for each $t \in [\ell]$ either i) $q_{o_4^t} = 3$ and $q_{o_1^t}(M) = q_{o_2^t}(M) = q_{o_3^t}(M) = 0$ or ii) $q_{o_4^t} = 0$ and $q_{o_1^t}(M) = q_{o_2^t}(M) = q_{o_3^t}(M) = 2$.

Recall that for each element agent, there is a unique option o that has $q_o(M) > 0$. Consequently, for any two sets S_t and S_v s.t. $q_{o_4^t}(M) = q_{o_4^v} = 0$, we have that $S_t \cap S_v = \emptyset$. Therefore, it must be that the sets with $q_{o_4^t}(M) = 0$ constitute an X3C of $\langle \mathcal{U}, \mathcal{S} \rangle$. □

ILP for Minimum Sized Pessimistically Valid Menus

We first recall the ILP for computing the minimum sized pessimistically valid menu.

$$\text{minimize } \sum_{o \in O} q_o s_o$$

subject to

$$\sum_{o \in A_i} w_o^i \geq 1 + \sum_{j \in N^i} y_j^i \quad i \in N \quad (4)$$

$$w_o^i \leq q_o s_o \quad i \in N, o \in A_i \quad (5)$$

$$w_o^i \leq x_o^i (n + s_o) \quad i \in N, o \in A_i \quad (6)$$

$$w_o^i \geq q_o s_o - (1 - x_o^i)(n + s_o) \quad i \in N, o \in A_i \quad (7)$$

$$y_j^i \geq x_o^i \quad i \in N, j \in N^i, o \in A_i \cap A_j \quad (8)$$

$$y_j^i \leq \sum_{o \in A_i \cap A_j} x_o^i \quad i \in N, j \in N^i \quad (9)$$

$$x_o^i \in \{0, 1\} \quad i \in N, o \in A_i$$

$$w_o^i \in \{0, 1, \dots, (n + s_o)\} \quad i \in N, o \in A_i$$

$$y_j^i \in \{0, 1\} \quad i \in N, j \in N^i$$

$$q_o \in \left\{ 0, 1, \dots, \left\lceil \frac{n}{s_o} \right\rceil \right\} \quad o \in O$$

Correctness of ILP We need to prove that this ILP is feasible only for choices of q_o for $o \in O$ that correspond to a pessimistically valid menu. To this end, we first prove that $w_o^i = x_o^i q_o s_o$ under every feasible solution $z = ((q_o)_{o \in O}, (x_o^i)_{i \in N, o \in A_i}, (w_o^i)_{i \in N, o \in A_i}, (y_j^i)_{i \in N, j \in N^i})$. Fix agent $i \in N$ and option $o \in A_i$.

Case 1: $x_o^i = 1$. In this case constraint 7 reduces to $w_o^i \geq q_o s_o$. Further, as a consequence of [Observation 4.1](#) constraint 6 becomes a trivial upper bound of constraint 5 and we get that $w_o^i = q_o s_o$.

Case 2: $x_o^i = 0$. In this case, constraint 6 insists that $w_o^i \leq 0$. Observe that this satisfies constraint 5. Further, from [Observation 4.1](#) we have that $n + s_o > q_o s_o$. Combining this with $x_o^i = 0$ makes constraint 7 trivially satisfied in this case. Thus, we must have that $w_o^i = 0$ whenever x_o^i .

Consequently, we have that $w_o^i = x_o^i q_o s_o$. As a result, our current constraints are simulating the constraints of the previous ILP for each agent $i \in N$. Thus, for a choice of q_o for each $o \in O$, the ILP has a feasible solution only if the corresponding menu is pessimistically valid for every agent.

It remains to show that the ILP is indeed always feasible for some choice of q_o for an arbitrary instance. Given MENU SELECTION instance I , observe that one trivial feasible solution is when we set $q_o = \lceil \frac{n}{s_o} \rceil$. Every option has enough quantity to feed *all* agents. As a result, this ILP will be feasible for all instances. Hence, the ILP finds a minimum sized pessimistically valid menu.

Size of ILP Observe that there are m variables of type q_o , at most nm variables of type x_o^i and w_o^i each and at most n^2 variables of type y_j^i . Further, there are n constraints corresponding to the Hall's theorem violation constraints, at most $3mn$ constraints corresponding to w_o^i simulating $x_o^i q_o s_o$ and at most $n^2(m + n)$ constraints corresponding to y_j^i being 1 if and only if $x_o = 1$ for some $o \in A_i \cap A_j$. Finally, there are at most $2nm + n^2 + m$ constraints specifying the ranges of values the variable can take. Consequently, the number of constraints is $O(n^2, mn^2)$.

Lenstra Jr [42] had shown that for every ILP, there is a corresponding parameterized algorithm that solves the ILP, when parameterized by the number of integer variables. As a result, our ILP leads to the following result.

Proposition C.1. *There is an $O^*(2^{n(n-1+m)})$ time algorithm for finding a minimum sized pessimistically valid menu.*

Parameterized Algorithm for Optimistic Menus

In this section we design a parameterized algorithm to find a minimum size menu that is optimistically valid. Towards this first we define maximum weighted partition. A tuple of subsets (S_1, S_2, \dots, S_k) of a set N is a partition of the set N if $\cup_{i \in [k]} S_i = N$ and $S_i \cap S_j = \emptyset$ for each pair $i, j \in [k]$, and $i \neq j$. The MAXIMUM WEIGHTED PARTITION problem is defined as follows [14]: Given a universe N , a set family \mathcal{F} over N , a non-negative integer k , and functions f_1, f_2, \dots, f_k , we ask if there is a partition (S_1, S_2, \dots, S_k) of N into k parts such that the sum $\sum_{i \in [k]} f_i(S_i)$ is maximum over all partitions (S_1, S_2, \dots, S_k) of N , where S_1, S_2, \dots, S_k are from \mathcal{F} .

Algorithm Overview. Let k denote the number of options that will be used in the menu. We iterate over the values of k starting from 1 to m . For a given value of k , we reduce our problem to a maximum weighted partition problem. We define the universe as the set of agents N . Let the set of options be $\{o_1, o_2, \dots, o_m\}$ with serving sizes s_1, s_2, \dots, s_m respectively. For each option $o_j \in O$, we define N_j to be the set of agents that accepts o_j . We define a family \mathcal{F} of subsets of N as follows: \mathcal{F} contains all the non-empty subsets of N_j for each $o_j \in O$. We define a function $f : \mathcal{F} \rightarrow \mathbb{Z}$ as follows. For a set $N' \subseteq N$ in the family \mathcal{F} , let O' denote the set of options that are accepted by all agents in N' and $o' \in \operatorname{argmin}_{j' \in O'} \{|N'| = r' \pmod{s_{j'}}\}$. Then, we define $f(N') = -r$ where $|N'| = r \pmod{s_{o'}}$. Now we find a partition (X_1, X_2, \dots, X_k) of N into k parts such that $X_i \in \mathcal{F}$ for each $i \in [k]$, and $\sum_{i \in [k]} f(X_i)$ is maximum. Let s_{max} denote the maximum serving size of any option, i.e., $s_{max} = \max_{j \in O} s_j$. We iterate over the values of k starting with 1 and going up to m and take the solution partition $(X_1, X_2, \dots, X_{k^*})$ of N that has maximum weight over all values of k .

For a part X_z where $z \in k^*$, let O_z denote the set of options that are accepted by all agents in X_z and $o_z \in \operatorname{argmin}_{j' \in O_z} \{|N'| = r' \bmod s_{j'}\}$. Then we construct M by setting $q_{o_z}(M) = \left\lceil \frac{X_z}{s_{o_z}} \right\rceil$ for each $z \in k^*$; for any other option $o \in O$, we set $q_o(M) = 0$.

The maximum weighted partition problem can be solved in time $O^*(2^n k^2 s_{\max})$, as shown by Björklund et al. [14]. We run the algorithm for maximum weighted partition at most m times. Therefore, our final running time is $O^*(2^n m^3 s_{\max})$.

Proposition 4.4. *Given a MENU SELECTION instance I , let s_{\max} denote the maximum serving size of any option, i.e., $s_{\max} = \max_{o \in O} s_o$ and $m = |O|$. Algorithm 1 computes a minimum sized optimistically valid menu in $O^*(2^n m^3 s_{\max})$ time.*

Correctness of the Algorithm. We now prove the correctness of our algorithm and analyze its running time. Recall that given an instance $I = \langle N, O, (A_i)_{i \in N}, (s_j)_{j \in O} \rangle$, our goal is to select a optimistically valid menu M minimizing its *size*, where

$$\operatorname{size}(M) = \sum_{j \in O} q_j(M) \cdot s_j,$$

where $q_j(M)$ denotes the multiplicity of option o_j in M . A menu M is *optimistically valid* if $\lambda(M) = n$, i.e. every agent can be served fractionally with acceptable options.

To show the correctness of the algorithm, we show that the weight a partition of N (as defined in the algorithm) is the same as the waste of the menu that is constructed using the last step of our algorithm.

Recall that for each option o_j , let $N_j = \{i \in N : o_j \in A_i\}$. We define a set family $\mathcal{F} = \{X \subseteq N_j \mid X \neq \emptyset \text{ and } o_j \in O\}$. For each $X \in \mathcal{F}$, define its *waste* as

$$\operatorname{waste}(X) := \min_{j: X \subseteq N_j} ((s_j - (|X| \bmod s_j)) \bmod s_j).$$

That is, the minimum waste that created when all agents in X feeds from the same option. recall that we define the weight function $f(X) = -\operatorname{waste}(X)$.

For a partition $\mathcal{X} = (X_1, \dots, X_k)$ of N , the algorithm maximizes

$$F(\mathcal{X}) = \sum_{i=1}^k f(X_i) = - \sum_{i=1}^k \operatorname{waste}(X_i).$$

Therefore, the algorithm minimizes the waste when the agents in X_z feed from the same option for each $z \in [k]$. Note that there are m options, then there can be at most m sets of agents who feed from the same option. Since our algorithm take the partition that has minimum sum over all values of k , we find the minimum waste.

The following two lemmas show the equivalence between a valid menu and a maximum size partition of each value of k , then finally, use these lemmas we finally complete the proof of correctness.

Lemma C.2 (Partition \Rightarrow Menu). *Let $\mathcal{X} = (X_1, \dots, X_k)$ be a partition of N . Let menu M is constructed as follows: for each part X_z , pick an option o_z that attains the minimum in $\operatorname{waste}(X_i)$ and set $q_{o_z} = \lceil |X_i| / s_{o_z} \rceil$. Then*

$$\operatorname{size}(M) = n + \sum_{i=1}^k \operatorname{waste}(X_i) = n - F(\mathcal{X}).$$

Proof. Since $X_i \subseteq N_{j(i)}$, all agents in X_i accept option $j(i)$. The serving capacity for part X_i is $q_{j(i)}s_{j(i)}$, which equals $|X_i| + \text{waste}(X_i)$. Summing over the parts yields

$$\sum_j q_j(M)s_j = \sum_{i=1}^k (|X_i| + \text{waste}(X_i)) = n + \sum_{i=1}^k \text{waste}(X_i).$$

By definition of f , this equals $n - F(\mathcal{X})$. \square

Lemma C.3 (Menu \Rightarrow Partition). *Let M be any menu with at most k distinct options. There exists a partition $\mathcal{X} = (X_1, \dots, X_k)$ of N such that*

$$\text{size}(M) = n - F(\mathcal{X}).$$

Proof. Let the distinct options in M be $j_1, \dots, j_{k'}$ ($k' \leq k$). Consider an optimal fractional allocation witnessing $\omega(M) = n$. For each j_t , let $X^{(t)}$ be the set of agents served by option j_t . These $X^{(t)}$ form a partition of N (pad with empty parts if needed). The waste for part $X^{(t)}$ equals $q_{j_t}(M)s_{j_t} - |X^{(t)}|$. Choosing option j_t in the definition of $f(X^{(t)})$ gives $f(X^{(t)}) = -\text{waste}(X^{(t)})$. Summing over all parts yields $F(\mathcal{X}) = -\sum_t \text{waste}(X^{(t)})$, hence $\text{size}(M) = n - F(\mathcal{X})$. \square

Using [Lemmas C.2](#) and [C.3](#), we obtain the following proposition.

Proposition C.4 (Correctness). *For each k , let*

$$\text{opt}_k := \max_{\mathcal{X} \text{ partition into } k \text{ parts}} F(\mathcal{X}).$$

Then

$$\min_{M: M \text{ valid}} \text{size}(M) = \min_{k=1}^m (n - \text{opt}_k).$$

That is, the algorithm computes opt_k for all $k = 1, \dots, m$, and returns the menu induced by the partition achieving the maximum objective value. This menu is optimistically valid and has minimum possible size.

Proof. We show that the size of the menu M constructed using a maximum weight partition \mathcal{X} is minimum. Since \mathcal{X} is maximum weight partition, $F(\mathcal{X})$ is maximized over all partitions and $\sum_{i=1}^k \text{waste}(X_i) = F(\mathcal{X})$, we have that the total waste is minimized. As any feasible menu must have size n and M has size $n - F(\mathcal{X})$, the size of M is minimum. \square

Running Time Analysis For each fixed k , we must solve the maximum weighted partition problem with universe N , family \mathcal{F} , and weight function f . By the result of Björklund, Husfeldt, and Koivisto [14], this can be done in time $O^*(2^n k^2 s_{\max})$, where $s_{\max} = \max_{j \in [m]} s_j$ and the O^* notation suppresses polynomial factors in n . Since the algorithm must run this routine for each $k = 1, 2, \dots, m$, the overall running time is $O^*(2^n m^3 s_{\max})$.

D Omitted Material from [Section 5](#)

Limited Serving Sizes

Proposition 5.3. *Given a MENU SELECTION $I = \langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ where for each $o \in O$, $s(o) \in \{1, 2\}$, [Algorithm 2](#) finds a minimum sized optimistically valid menu in polynomial time.*

Proof. Construct the following auxiliary weighted graph $G = (V, E)$ where there is one vertex for each agent $i \in N$ and for each option o and agent $i \in N$ s.t. $o \in A_i$ and $s(o) = 1$, there is a vertex o^i . Here, add an edge between $i, j \in N$ only if there exists $o \in A_i \cap A_j$ s.t. $s(o) = 2$ and set $w(i, j) = 2$. Further, add an edge between $i \in N$ and a vertex o^i for $o \in O$ where $s(o) = 1$ and $o \in A_i$ and set $w(i, o^i) = 1$.

Consider an arbitrary matching μ in G . Observe that each edge $e \in \mu$ must contain at least one agent. For each edge $e = (i, j)$, we can add option o s.t. $o \in A_i \cap A_j$ and $s(o) = 2$. For each edge $(i, o) \in \mu$, it must be that $o \in A_i$ and $s(o) = 1$. Thus, based on a matching, we can prepare a menu that feeds all matched agents in μ .

We shall now show that a maximum weight matching in G has weight $n - t$ for $t \in [n]$ if and only if a minimum size optimistically valid menu has size $n + t$.

First consider a minimum size optimistically valid menu M . As size of each option is at most 2, $s(M) \leq 2n$. Further, if $s(M) = n + t$, exactly t distinct options in M of size 2 must have been ordered that are only consumed by one agent. For each such agent i , observe that A_i must only contain options of size 2. Further, all other agents who find an option A_i must already be sharing a different option of size 2 with another agent i' . It is easy to see that $A_{i'}$ must also only consist of options of size 2. All options in M that are fully consumed, are either of size 1 or each ordered quantity is consumed by two agents. Consequently, we can create a matching in G based on μ of weight $n - t$.

Now, consider a maximum weight matching. Let M_μ be the corresponding menu. M is not optimistically valid if and only if there are some unmatched agents under μ . As μ is a maximum weight matching, it must be that for each unmatched i , A_i only contains options of size 2. Further, all agents who also like an option in A_i must be matched to another agent. If they were matched to an option, being matched to i would increase the weight of the matching. Thus, it is unavoidable to order some $o \in A_i$ just for i . As a result, for each unmatched i , we can arbitrarily choose an option o from A_i and add it to M_μ . Thus, if $w(\mu) = n - t$, exactly t options of size 2 are ordered for the t unmatched agents, making M_μ have size $n - t + 2t = n + t$.

Running Time. The graph constructed consists of at most $n + nm$ vertices and at most $n^2 + nm$ edges. A maximum weight matching can be found using the blossom algorithm [25] in time $O(n^3(nm^2 + m^3))$. A maximum weight matching can have weight at most n and from it a menu can be constructed in time $O(n + m)$. As a result, the algorithm runs in polynomial time. \square

Structured Acceptability Relations

Lemma 5.4. *Given an MENU SELECTION instance I , for every $i, j \in N$ at most two of $A_i \cap A_j$, $A_i \setminus A_j$ and $A_j \setminus A_i$ are non-empty if and only if I has laminar acceptability relations.*

Proof. Let I be laminar. Fix $i, j \in N$ s.t. $|A_i| \leq |A_j|$. If $A_i \subseteq A_j$ then $A_i \setminus A_j$ must be non-empty. Else if A_i and A_j are disjoint, clearly, $A_i \setminus A_j = \emptyset$.

For the converse, let I be such that for every $i, j \in N$ at most two of $A_i \cap A_j$, $A_i \setminus A_j$ and $A_j \setminus A_i$ are non-empty. As a result, at least one of these sets must be empty. Consider any two $i, j \in N$. If $A_i \cap A_j$ is empty, clearly, they satisfy laminarity. If $A_i \setminus A_j$ is empty, it must hold that $A_i \subseteq A_j$. Similarly, if $A_j \setminus A_i$ is empty, it must hold that $A_j \subseteq A_i$. Therefore, I satisfied laminarity. \square

Identical Acceptability Relations We first look at the case when all agents find the same set of options acceptable. As agents are identical, we can discard options that are not liked by any agent. Consequently, we can assume, without loss of generality, that $O = A_i$ for each $i \in N$. Consequently, every menu of size at least n will be both optimistically and pessimistically valid. Options with serving size n or more need only one quantity ordered, without ordering anything else, but options

Algorithm 3: Minimum Sized Valid Menus under Identical Acceptability Relations

Input: MENU SELECTION instance $\langle N, A, (s_o)_{o \in O} \rangle$ with identical acceptability relations

Output: Minimum size of valid menu c

- 1 Initialize small options set $A^s \leftarrow \{o \in A \mid s_o < n\}$
 - 2 Min sized large options set $A^\ell \leftarrow \arg \min \{s_o \mid o \in A, s_o \geq n\}$
 - 3 **if** $A^\ell \neq \emptyset$ **then** $c^\ell \leftarrow s_o$ where $o \in A^\ell$
 - 4 **else** $c^\ell \leftarrow \infty$
 - 5 $c^s \leftarrow \infty$
 - 6 **if** $A^s \neq \emptyset$ **then**
 - 7 $\mathcal{S} \leftarrow \{ks_o \mid o \in A^s, k \leq \lceil \frac{n}{s_o} \rceil\}$
 - 8 Let $s_1 \leq \dots \leq s_{|\mathcal{S}|}$ be the elements of \mathcal{S}
 - 9 $c^s \leftarrow \text{IDENT}(\mathcal{S}, n, 2mn)$
 - 10 **Return** $\min c^s, c^\ell$
-

with smaller serving sizes will have to be combined to feed all agents. So we choose either the smallest sized "large option" or the smallest possible combination of "small options". For small options, we effectively run a subset sum type DP to find the smallest menu that can feed everyone.

Proposition D.1. *Given identical MENU SELECTION instance $I = \langle N, A, (s_o)_{o \in A} \rangle$, Algorithm 3 finds the minimum size of an optimistically valid menu in polynomial time. This will also be the minimum size of a pessimistically valid menu.*

Proof. When agents have identical acceptability, we use $A \subseteq O$ to denote the universally accepted set of options, that is, for any $i \in N$, $A_i = A$.

We first show that every optimistically valid menu will also be pessimistically valid. Consider an optimistically valid menu M and the graph where there is a vertex for each agent and each option, with an edge from i to o if $o \in A_i$. Set vertex capacities where $c(i) = 1$ for each $i \in N$ and $c(o) = q_o(M)$ for each $o \in O$.

Now as M is optimistically valid, there must be a matching $\mu : N \times O \rightarrow \mathbb{R}^+$ s.t. $\sum_{o \in A_i} \mu(i, o) = 1$ and $\sum_{i \in N} \mu(i, o) \leq q_o(M)s_o$. In order to ensure that M is pessimistically valid, we need that for any other maximal matching μ' , we have that $\sum_{o \in A_i} \mu'(i, o) = 1$. Recall that μ' is a maximal matching if for every $i \in N$ s.t. $\sum_{o \in O} \mu'(i, o) < 1$, we have that for each $o \in A_i$, $\sum_{i \in N} \mu'(i, o) = q_o(M)s_o$. As agents have identical acceptability sets, any maximal matching μ' must in fact satisfy $\sum_{o \in O} \mu'(i, o) = 1$ for all $i \in N$. Consequently, M must be pessimistically valid.

It now remains to construct a minimum sized valid menu. Specifically, it suffices to find the smallest sized menu M s.t. $\sum_{o \in A} q_o(M) \geq n$. Observe that if a menu contains even one option $o \in A$ s.t. $s_o \geq n$, even if we discard all other options from this menu, it would continue to be valid. Consequently, a minimum sized valid menu would only involve multiple options if all of them have serving size less than n . We use this approach in Algorithm 3 and its subroutine Algorithm 4.

The algorithm proceeds by separating options into a small options set A^s consisting of the options of size less than n and picks the smallest size large options in the set A^ℓ containing the smallest options of size at least n , if any. Only one these options is needed to make a valid menu. For the small options set, Algorithm 3 creates a set \mathcal{S} , consisting of one element corresponding to k copies of $o \in A^s$ where $k \leq \lceil \frac{n}{s_o} \rceil$.

At this point the dynamic program IDENT (Algorithm 4) is invoked with \mathcal{S} and the goal of n . Starting from the largest sized element in \mathcal{S} , IDENT proceeds by considering two cases for each

Algorithm 4: IDENT

Input: $\langle \mathcal{S}, goal, t \rangle$
Output: Cost c

- 1 Let $s_1 \leq \dots \leq s_{|\mathcal{S}|}$ be the elements of \mathcal{S}
- 2 $t \leftarrow \min(|\mathcal{S}|, t)$
- 3 **if** $t > 1$ **then**
- 4 **if** $s_t \geq goal$ **then**
- 5 $t_{in} \leftarrow s_t$
- 6 **else**
- 7 $t_{in} \leftarrow s_t + \text{IDENT}(\mathcal{S}, goal - s_t, t - 1)$
- 8 $t_{out} \leftarrow \text{IDENT}(\mathcal{S}, goal, t - 1)$
- 9 **Return** $\min(t_{in}, t_{out})$
- 10 $t = 1$ */
- 11 **if** $s_1 \geq goal$ **then**
- 12 **Return** s_1
- 13 **Return** ∞

element s_t : in or out, captured by t_{in} and t_{out} . For t_{in} , the goal is reduced s_t and the remaining is filled by s_{t-1} or smaller elements. For t_{out} , s_t is not a part of the solution and the goal must be filled by s_{t-1} or smaller elements. The minimum of these two options is returned.

Running Time. Observe that the running time of [Algorithm 3](#) depends largely on the running time of [Algorithm 4](#) (IDENT), which is a dynamic program. The size of the dynamic program is based on the size of the set \mathcal{S} which contains $\lceil \frac{n}{s_o} \rceil$ elements for each $o \in O$. Consequently, the size of \mathcal{S} is at most nm , and consequently, IDENT has size at most $O(n^3m^2)$. \square

Optimistically Valid Menus for Laminar Acceptability Relations.

Under laminar acceptability relations, the acceptability sets of two agents are either disjoint, or one contains the other. Laminar sets can be represented by a containment tree where if $A^i \subsetneq A^j$, then A^j must be an ancestor of A^i in the containment tree (See ?? for some preliminaries on trees). Let A^1, \dots, A^k be the distinct acceptability sets labeled in a pre-order traversal of the containment tree. That is, if $A_i \subseteq A_j$ then $i \geq j$. The agents with acceptability set A^i can be fed from a combination of items across A^i or a combination of one child alone. Consequently, to find min sized menus, we need to determine the best combination of agents being fed from each acceptability set. To this end, for each internal node A^i , we need to decide how many of the corresponding agents eat from A^i and how many eat from each child of A^i .

To avoid going over all possible combinations here, we use the tree structure to build these combinations sequentially. Each node v other than the root inherits a “debt” from either its previous sibling (according to the label) or its parent (if no sibling comes earlier in the labeling), but not both. Now v needs to decide how much of this debt it will keep for the subtree rooted at v , and how much to pass off onto the next sibling, if any. Now the debt it retains plus n_v is the number of agents that need to be fed from the subtree rooted at v . Of this number, v now decides how many to feed from the set A^v and passes off the remaining to the first child (according to the labelling), if any.

Algorithm 5: Minimum Sized Optimistically Valid Menus under Laminar Acceptability Relations

Input: Menu Planning instance $\langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ with laminar acceptability relations

Output: Minimum size of optimistically valid menu c

- 1 Let A^1, \dots, A^k be the distinct acceptability sets s.t. sets are labeled in a pre-order traversal of the associated containment tree where $\{1, 2, \dots, k\}$ denote the acceptability sets ordered according to the pre-order traversal
 - 2 Let $n_v \leftarrow |\{i \in N \mid A_i = A^v\}|$
 - 3 Let $sib(v) \leftarrow \min\{v' \mid v' > v \text{ AND } v' \text{ is a sibling of } v\}$ if any, 0 otherwise
 - 4 Let $child(v) \leftarrow \min\{v' \mid v' \text{ is a child of } v\}$ if any, 0 otherwise
 - 5 Initialize $k \times n$ array *LaminarOpt*
 - 6 **for** $v = k$ **to** 1 **do**
 - 7 Let $t = \sum_{v': A^v \subsetneq A^{v'}} n_{v'}$
 - 8 **for** $debt = t$ **to** 0 **do**
 - 9 $u_1 \leftarrow \mathbb{1}(sib(v) > v)debt$
 - 10 $u_2 \leftarrow \mathbb{1}(child(v) > v)(n_v + debt)$
 - 11 $\mathcal{S}^v \leftarrow \{ks_o \mid o \in A^v, k \leq \lceil \frac{n}{s_o} \rceil\}$
 - 12 $LaminarOpt[v][debt] \leftarrow$
 $\min_{0 \leq t_1 \leq u_1} \min_{0 \leq t_2 \leq \max(u_2 - t_1, 0)} \left(\text{IDENT}(\mathcal{S}^v, n_v + debt - t_1 - t_2, nm) \right.$
 $\left. + \mathbb{1}(sib(v) > v)LaminarOpt[sib(v)][t_1] + \mathbb{1}(child(v) > v)LaminarOpt[child(v)][t_2] \right)$
 - 13
 - 14 **Return** $LaminarOpt[1][0]$
-

Algorithm Overview. Algorithm 5 starts at the root of the laminar containment tree, A^1 that is the largest acceptability set. Recall that we assume that the vertices of the tree are labeled in a pre-order traversal, so each internal vertex is labeled before its children. We say that n_t agents have acceptability set A^t . We use this ordering to run the dynamic program to sequentially decide, how many agents must eat from each vertex of the tree and use the algorithm for identical acceptability relations to find a minimum sized menu for these many agents.

We introduce the following notation: $sib(v)$ is used to denote the next sibling of v according to the labels, if any. If v has no siblings with a higher label, $sib(v)$ is set to 0. Similarly, $child(v)$ is used to denote the child of v with the smallest label. If v has no children, the value of $child(v)$ is set to 0. Further, we use $debt$ to denote the number of agents whose acceptability set is a superset, but must eat from the subtree rooted at v or its siblings.

Each vertex $v \in [k]$, other than the root, receives a $debt \geq 0$. If v has a sibling, some of this debt can be passed on to them. We set u_1 as equal to $debt$ if $sib(v) > 0$. That is, u_1 is set to the maximum debt that can be sent to $sib(v)$. We choose t_1 between 0 and u_1 as the debt passed onto $sib(v)$. The remaining $debt - t_1$ agents must be fed from the subtree rooted at v . We now can pass some off these agents on to the first child $child(v)$, if any. We analogously set u_2 as $n_v + debt - t_1$ if $child(v) > 0$ and choose t_2 as the debt passed on to $child(v)$. We then use Algorithm 3 to find the minimum sized menu to feed the remaining $n_v + debt - t_1 - t_2$ agents from A^v .

We make these decisions by using a dynamic program that we call $LaminarOpt[v][debt]$ where $v \in [k]$ and $debt$ can take any integral value between 0 and $\sum_{v': A^v \subsetneq A^{v'}} n_{v'} \leq n$. We fill this from $v = k, \dots, 1$ and for all values of $debt \in \{0, \dots, \sum_{v': A^v \subsetneq A^{v'}} n_{v'}\}$. Observe that v can only pass $t_1 > 0$

debt to $\text{sib}(v)$ if $\text{sib}(v) > v > 0$. Similarly, as the vertices are labeled in a pre-order traversal of the tree, each child's entries would be filled before we reach their parent. As a result, we can fill each entry of $\text{LaminarOpt}[v][\text{debt}]$ by considering at most n^2 other entries. We ultimately return $\text{LaminarOpt}[1][0]$.

Proposition 5.5. *Given a MENU SELECTION instance I with laminar acceptability relations, there exists a polynomial-time algorithm that finds the minimum size of a optimistically valid menu M .*

Proof. Fix a MENU SELECTION instance $I = \langle N, O, (s_o)_{o \in O}, (A_i)_{i \in N} \rangle$ with laminar acceptability relations. Let A^1, \dots, A^k be the distinct acceptability sets across all agents in N . Further, assume that these sets are labeled s.t. if $A^j \subseteq A^{j'}$, we have that $j \geq j'$. As these sets are laminar, we assume that they are labeled in a pre-order traversal of the associated containment tree.³

Consider an agent $i \in N$. Consider an optimistically valid menu M and a maximum sized integral matching x . As M is optimistically valid, there must exist $o \in A_i$ s.t. $x_{i,o} = 1$. Let A^j be the smallest acceptability set that contains o . That is, $A^j = \arg \max \{t \in [k] \mid o \in A_t\}$. Observe that A^j must be the same as A_i or a descendant of it in the laminar containment tree. In other words, i must eat from A_i or a descendant of it in the containment tree.

Consequently, in order to find min sized optimistic menus, we need to decide how many agents eat from each set. We can find min sized menus for each set using the identical case (Algorithm 3) as a subroutine. However, there may be exponentially many ways to divide agents among these sets. We present a clever way of making this decision sequentially and using dynamic programming to combine different independent subcases.

For each $t \in [k]$ let n_t be the number of agents who have acceptability set A^t . The agents eating from A_t can be agents with this exact acceptability set or with a superset (an ancestor of A_t). These agents can also eat from a descendant of A_t , and those with strictly larger acceptability sets can also eat from a sibling of A_t . Consequently, Algorithm 5 uses a dynamic program in a top down manner on the laminar containment tree as follows.

Correctness of Algorithm. We shall first show that every optimistically valid menu M can be broken into a distribution of agents eating from each set A^t for $t \in [k]$ and thus be considered by Algorithm 5.

Fix an arbitrary optimistically valid menu M . Choose a maximum cardinality optimistic consumption matching x s.t. $\sum_{i \in N, o \in O} x_{i,o} = n$. We shall now count the number of agents eating from each set A^v using the variables p_v . Initialize p_v to 0 for each variable $v \in [k]$.

For each option $o \in \tilde{M}$ s.t. $\sum_i x_{i,o} > 0$, let A^v be the smallest acceptability set containing o . That is, for each $v' \neq v$ s.t. $o \in A^{v'}$, $A^v \subsetneq A^{v'}$. We now increase p_v by $\sum_i x_{i,o}$ to account for the number of people eating o . Now, having done this for all distinct options in M , we now must have that $\sum_{v \in [k]} p_v = n$, as M is optimistically valid.

Further, for each $v \in [k]$, we only count the agents eating from it as those that are eating an option not contained in one of its descendants. Thus, the acceptability set of all these agents must be A^v or a superset of A^v . Consequently, it must hold that $p_v \leq n_v + \sum_{v': A^v \subsetneq A^{v'}} n_{v'}$. Additionally, for each leaf v , it must be that $p_v \geq n_v$ as agents with acceptability set A^v cannot eat from any other set. As a result, $p_v - n_v$ is precisely the number of agents with strictly larger acceptability sets. Thus, starting from the last leaf, we can calculate the *debt* inherited by each leaf in the tree. This then helps us calculate the *debt* inherited by each vertex whose children are all leaves. In this manner, we can calculate the *debt* inherited by each vertex and consequently the t_1 and t_2 values of

³A pre-order traversal of tree is one where each parent vertex is visited before its corresponding children. Given a collection of laminar sets, we can construct a corresponding containment tree s.t. if $S \subsetneq T$ then the vertex corresponding to T must be an ancestor of the vertex corresponding to S .

Algorithm 6: Minimum Sized Pessimistically Valid Menus under Laminar Acceptability Relations

Input: Menu Planning instance $\langle N, O, (A_i)_{i \in N}, (s_o)_{o \in O} \rangle$ with laminar acceptability relations
Output: Minimum size of valid menu c

- 1 Let A^1, \dots, A^k be the distinct acceptability sets
- 2 Let $n_v \leftarrow |\{i \in N \mid A_i = A^v\}|$
- 3 Consider the set of leaf vertices $\mathcal{L} \leftarrow \{v \in [k] \mid v \text{ is a leaf}\}$
- 4 Initialize $c \leftarrow 0$
- 5 **for** $v \in \mathcal{L}$ **do**
- 6 Let $t = n_v + \sum_{v': A^v \subsetneq A^{v'}} n_{v'}$
- 7 $\mathcal{S}^v \leftarrow \{ks_o \mid o \in A^v, k \leq \lceil \frac{n}{s_o} \rceil\}$
- 8 $c \leftarrow c + \text{IDENT}(\mathcal{S}^v, t, nm)$
- 9 **Return** c

each vertex. As we solve each subcase optimally using [Algorithm 3](#), for this sequence of t_1 and t_2 values for each vertex, the menu size stored would be less than or equal to that of M but still feed the same number of agents from each set A^v .

As a result, we have that each valid menu will be considered or something that feeds the same number of people with a smaller size. It is straightforward to see that the algorithm does not consider any invalid division of agents across the sets. Hence, as the algorithm returns the minimum size encountered, it must find a minimum sized optimistically valid menu for I .

Running Time. Observe that the size of the *LaminarOpt* is at most n^2 . The laminar tree can contain at most n vertices, and the debt to each agent can be at most n . To fill each entry, [Algorithm 3](#) is called at most n^2 times and combined with other existing entries of *LaminarOpt*. Further [Algorithm 3](#) runs in time $O(n^3 m^2)$. Consequently, *LaminarOpt* runs in time $O(n^7 m^2)$. \square

Pessimistically Valid Menus for Laminar Acceptability Relations.

We now build pessimistically valid menus for the laminar setting. To this end, we first give a necessary and sufficient condition for pessimistic validity under laminarity.

Lemma D.2. *Given a MENU SELECTION instance I with laminar acceptability relations and menu M . Let v_1, \dots, v_t be the leaf nodes of the laminar containment tree. M is pessimistically valid if and only if for any $v \in \{v_1, \dots, v_t\}$, we have that $\sum_{o \in A^v} q_M(o) s_o \geq n_v + \sum_{v': A^v \subsetneq A^{v'}} n_{v'}$.*

Proof. Fix a MENU SELECTION instance $I = \langle N, O, (s_o)_{o \in O}, (A_i)_{i \in N} \rangle$ with laminar acceptability relations and a menu M . Let A^{v_1}, \dots, A^{v_t} be the t distinct minimal acceptability sets. That is, these sets are all pairwise disjoint and for all agents $i \in N$, there must exist at least one $j \in [t]$ s.t. $A^{v_j} \subseteq A_i$. It is straightforward to see that these must be the leaves of the laminar containment tree.

Let M be a pessimistically valid menu. Recall from [Theorem 3.2](#) that for a menu M to be pessimistically valid, for each agent $i \in N$ it must hold that $\text{size}(M|_{A_i}) = \sum_{o \in A_i} q_o(M) s_o \geq 1 + \rho_i(M)$, where M is the maximum cardinality matching from $N \setminus \{i\}$ to $M|_{A_i}$. Consider an agent $i \in N$ s.t. $A_i = A^v$ for some $v \in \{v_1, \dots, v_t\}$.

Consider the graph that determines ρ_i . One side of the bipartition will have $M|_{A_i}$ and the other has the set $N^i = \{j \in N \setminus \{i\} \mid A_i \cap A_j \neq \emptyset\}$. Observe that due to laminarity and the fact that $A_i = A^v$ is a minimal acceptability set, every agent $j \in N \setminus \{i\}$ s.t. $A_i \cap A_j \neq \emptyset$ satisfies $A_i \subseteq A_j$. As a result, every such agent j will have an edge to every option in A_i . Consequently,

$\rho_i(M) = \min(n_v - 1 + \sum_{v': A^v \subsetneq A^{v'}} n_{v'}, \text{size}(M|_{A_i}))$. As M is pessimistically valid, it must hold that $\sum_{o \in A^v} q_o(M) s_o \geq n_v + \sum_{v': A^{v_j} \subsetneq A^{v'}} n_{v'}$.

The remaining case is to show that when $\sum_{o \in A^v} q_M(o) s_o \geq n_v + \sum_{v': A^v \subsetneq A^{v'}} n_{v'}$ holds for each $v \in \{v_1, \dots, v_t\}$, M must be pessimistically valid. Let $\sum_{o \in A^v} q_M(o) s_o \geq n_v + \sum_{v': A^v \subsetneq A^{v'}} n_{v'}$ hold for each $v \in \{v_1, \dots, v_t\}$. Consider an arbitrary agent $i \in N$ and minimal acceptability sets v_{i_1}, \dots, v_{i_t} s.t. for each $v \in \{v_{i_1}, \dots, v_{i_t}\}$, we have that $A^v \subseteq A_i$. As I has laminar acceptability relations, we have that for every agent $j \in N \setminus \{i\}$ s.t. $A_i \cap A_j \neq \emptyset$, either $A_i \subseteq A_j$ or $A_j \subsetneq A_i$. Let $N_+^i = \{j \in N \setminus \{i\} | A_i \subseteq A_j\}$ and let $N_-^i = \{j \in N \setminus \{i\} | A_j \subsetneq A_i\}$.

Observe that for each $j \in N_+^i$, we have that $A^v \subseteq A_j$ for every $v \in \{v_{i_1}, \dots, v_{i_t}\}$. On the other hand, for each $j \in N_-^i$, we have that there must exist $v \in \{v_{i_1}, \dots, v_{i_t}\}$ s.t. $A^v \subseteq A_j$. As a result, we have that the number of agents who find an option in A_i acceptable is exactly $|N_+^i| + |N_-^i| = (\sum_{v': A^{v'} \cap A_i \neq \emptyset} n_{v'}) - 1$.

Recall that for every $v \in \{v_{i_1}, \dots, v_{i_t}\}$, we have that $\sum_{o \in A^v} q_M(o) s_o \geq n_v + \sum_{v': A^v \subsetneq A^{v'}} n_{v'}$. Consequently, for every $v \in \{v_{i_1}, \dots, v_{i_t}\}$, we have that $\sum_{o \in A^v} q_M(o) s_o \geq 1 + |N_+^i| + |\{j \in N_-^i | A^v \subseteq A_j\}|$. Thus, M is pessimistically valid. \square

Proposition 5.6. *Given a MENU SELECTION instance I with laminar acceptability relations, there exists a polynomial-time algorithm that finds the minimum size of a pessimistically valid menu M .*

Proof. Fix a MENU SELECTION instance $I = \langle N, O, (s_o)_{o \in O}, (A_i)_{i \in N} \rangle$ with laminar acceptability relations. We shall prove that [Algorithm 6](#) returns the minimum possible size of a pessimistically valid menu under I . We first begin with an algorithm overview.

Algorithm Overview. [Algorithm 6](#) proceeds by considering the laminar containment tree and the t leaves A^{v_1}, \dots, A^{v_t} . For each leaf v_j it adds the minimum size pessimistic menu for feeding $n_{v_j} + \sum_{v': A^{v_j} \subsetneq A^{v'}} n_{v'}$ from A^{v_j} using the algorithm for identical acceptability relations as a subroutine. The combined sizes for each leaf is returned.

Correctness. Firstly, it is straightforward to see that the menu constructed by [Algorithm 6](#) would satisfy [Lemma D.2](#), thus it is clearly pessimistically valid.

Let M' be a minimum sized pessimistically valid menu. From [Lemma D.2](#), we have that for each minimal acceptability set A^{v_j} , $\sum_{o \in A^{v_j}} q_{M'}(o) s_o \geq n_{v_j} + \sum_{v': A^{v_j} \subsetneq A^{v'}} n_{v'}$. Now as [Algorithm 3](#) always returns the minimum menu size to feed a given number of (identical) agents from a given set, it must be that $\sum_{o \in A^{v_j}} q_{M'}(o) s_o \geq \text{IDENT}(A^{v_j}, n_{v_j} + \sum_{v': A^{v_j} \subsetneq A^{v'}} n_{v'})$. If this is not so, either [Algorithm 3](#) would not be optimal or the condition $\sum_{o \in A^{v_j}} q_{M'}(o) s_o \geq n_{v_j} + \sum_{v': A^{v_j} \subsetneq A^{v'}} n_{v'}$ must not hold. This would be a contradiction.

Consequently, we have that the size of M' must be greater than or equal to the size returned by [Algorithm 6](#). Consequently, we have that [Algorithm 6](#) returns the minimum possible size of a pessimistically valid menu under I .

Running Time. [Algorithm 6](#) invokes [IDENT \(Algorithm 4\)](#) $|\mathcal{L}| \leq n - 1$ times and [IDENT](#) takes time $O(n^3 m^2)$. Consequently, [Algorithm 6](#) runs in time $O(n^4 m^2)$. \square

E Omitted Material from Section 6

Proof. Fix a MENU SELECTION instance $I = \langle N, O, (s_o)_{o \in O}, (A_i)_{i \in N} \rangle$.

Laminar. We first consider the laminar case. Given instance I , we shall create an alternate instance I' , also with laminar acceptability relations with the same number of agents and minimal acceptability sets, but which will always satisfy $WoP(I) \leq WoP(I')$ and then reason about its maximum possible waste of pessimism value. We will prove this bound as follows:

- i. Create an alternate instance I' with laminar acceptability relations, with the same number of agents and same number of minimal acceptability sets.
- ii. Prove that $WoP(I) \leq WoP(I')$.
- iii. Find a worst case choice of serving sizes and number of minimal acceptability sets for WoP
- iv. Demonstrate an instance where this bound is tight.

Creating I' . Given I , construct instance I' s.t. the laminar containment tree of I' is a star graph that has the same number of leaves as that of I . Let A^1, \dots, A^k be the distinct acceptability sets under I where A^1 is the root of the containment tree. Let $\mathcal{L} = \{A^{v_1}, \dots, A^{v_t}\}$ be the leaves of the containment tree in I .

We set the distinct acceptability sets under I' as $A^1 \cup o^*, A^{v_1}, \dots, A^{v_t}$. Here, we add an additional option o^* with $s_{o^*} = 1$ to the root of the tree. Further, we set the number of agents with acceptability set A^v as $n_v = 1$ if $v \in L$ and $n_1 = n - t$ where n is the number of agents in I .

Observe that any menu that under I' each agent find the same set acceptable as under I or finds a superset of it acceptable. This means that any menu that is optimistically valid under I will continue to be optimistically valid under I' . Consequently, $\min_{M \in \Lambda(I)} \sum_{o \in O} q_M(o) s_o \geq \min_{M \in \Lambda(I')} \sum_{o \in O} q_M(o) s_o$.

Further, for pessimistically valid menus, from [Lemma D.2](#), the number of agents that need to be fed by any minimal set under I' is at least as much as that under I . As a result, any menu that is pessimistically valid under I' will be pessimistically valid under I . Consequently, $\min_{M \in \Gamma(I)} \sum_{o \in O} q_M(o) s_o \leq \min_{M \in \Gamma(I')} \sum_{o \in O} q_M(o) s_o$. Thus, we have that

$$WoP(I) = \frac{\min_{M \in \Gamma(I)} \sum_{o \in O} q_M s_o}{\min_{M \in \Lambda(I)} \sum_{o \in O} q_M s_o} \leq \frac{\min_{M \in \Gamma(I)} \sum_{o \in O} q_M s_o}{\min_{M \in \Lambda(I')} \sum_{o \in O} q_M s_o} \leq \frac{\min_{M \in \Gamma(I')} \sum_{o \in O} q_M s_o}{\min_{M \in \Lambda(I')} \sum_{o \in O} q_M s_o} = WoP(I').$$

Worst Case $WoP(I')$. We now compute the worst case value for the waste of pessimism for I' . Firstly, given an acceptability set A^v if $\alpha = \min_{o \in A^v} s_o$ then the minimum possible menu size to feed n' agents from A^v would be at most $\alpha \lceil \frac{n'}{\alpha} \rceil$. That is, in order to feed all n' agents who find A^v acceptable, we can simply order the item with the smallest serving size $\lceil \frac{n'}{\alpha} \rceil$ times in order to find a valid menu. Consequently, the minimum sized pessimistic menu size to feed n agents under I' would involve feeding $n - t + 1$ agents from each leaf vertex A^{v_j} for $j \in [t]$. Thus, the size of the smallest pessimistic menu will be at most $\sum_{j \in [t]} \alpha_j \lceil \frac{n-t+1}{\alpha_j} \rceil$ where $\alpha_j = \min_{o \in A^{v_j}} s_o$.

Now recall that under I' exactly one agent has $A_i = A^{v_j}$ for each $j \in [t]$. To feed this agent under an optimistically valid menu, exactly one option needs to be selected from A^{v_j} . Specifically, this should be the item with the smallest serving size, i.e. it must have size α_j . Thus, the minimum sized optimistically valid menu must have size at least $\sum_{j \in [t]} \alpha_j$. If for some $j \in [t]$, $\alpha_j > 1$, this can additionally feed some agents at the root. Further, we had added option o^* to A^1 with size $s_{o^*} = 1$. If $\sum_j \alpha_j < n$, we can order $n - \sum_j \alpha_j$ quantities of o^* . Consequently, a minimum sized optimistic menu will have size $\max(n, \sum_{j \in [t]} \alpha_j)$. We now give a bound on $WoP(I')$ based on which term comes in the denominator.

Case 1: $\sum_{j \in [t]} \alpha_j \geq n$. Recall that the size of the minimum sized pessimistic menu is

$$\begin{aligned} \min_{M \in \Gamma(I')} \sum_{o \in O} q_M(o) s_o &\leq \sum_{j \in [t]} \alpha_j \left\lceil \frac{n-t+1}{\alpha_j} \right\rceil \\ &\leq \left\lceil \frac{n-t+1}{\min_{j \in [t]} \alpha_j} \right\rceil \sum_{j \in [t]} \alpha_j. \end{aligned}$$

$$\begin{aligned} \text{This gives us that } WoP(I') &= \frac{\left\lceil \frac{n-t+1}{\min_{j \in [t]} \alpha_j} \right\rceil \sum_{j \in [t]} \alpha_j}{\sum_{j \in [t]} \alpha_j} \\ &\leq \left\lceil \frac{n-t+1}{\min_{j \in [t]} \alpha_j} \right\rceil. \end{aligned}$$

Observe that this bound would be tight only when $\min_j \alpha_j = \alpha_j$. However, we still need $\sum_{j \in [t]} \alpha_j \geq n$. Thus, in order to maximize the waste of pessimism, we need $\alpha_j = \frac{n}{t}$ for all $j \in [t]$. Thus, to maximize $WoP(I')$, we need that $\sum_{j \in [t]} \alpha_j = n$. This brings us to the next case.

Case 2: $\sum_{j \in [t]} \alpha_j \leq n$. In this case, we have that the denominator in the Waste of Pessimism bound will be n . For the numerator we have that

$$\begin{aligned} \min_{M \in \Gamma(I')} \sum_{o \in O} q_M(o) s_o &\leq \sum_{j \in [t]} \alpha_j \left\lceil \frac{n-t+1}{\alpha_j} \right\rceil \\ &\leq \sum_{j \in [t]} (n-t+\alpha_j) \\ &= t(n-t) + \sum_{j \in [t]} \alpha_j. \end{aligned}$$

Again, to maximize $WoP(I)$, we set $\sum_j \alpha_j = n$. Consequently, it remains to choose a value of t that maximizes the function $t(n-t)$. By some straightforward calculus, we get that we need $t = \frac{n}{2}$, in order to maximize $WoP(I')$. Recall that we assume $\sum_{j \in [t]} \alpha_j = n$. By definition, for each $j \in [t]$, it must be that $\alpha_j \geq 1$.

For this case, in order to ensure that the upper bound is tight, we require that $\sum_{j \in [t]} \alpha_j = n$ and $\sum_{j \in [t]} \alpha_j \left\lceil \frac{n-t+1}{\alpha_j} \right\rceil = \sum_{j \in [t]} (n-t+\alpha_j)$. To this end, we need that for every $j \in [t]$, α_j divides $n-t$. This includes the choice of $\alpha_j = \frac{n}{t}$, which gave the tight bound for Case 1, which in this setting gives us $\alpha_j = 2$ for all j . To simplify our analysis for the upper bound, we assume $\alpha_j = 2$.

To this end, we get that for any MENU SELECTION instance I with n agents,

$$WoP(I) \leq \frac{(n/2)(n/2+2)}{n} = \frac{n+4}{4}.$$

Tight example. Consider the following instance where this bound is tight: set $n = 4t'$, $O = \{o_1, \dots, o_{2t'+1}\}$ where $s_o = 2$ for all $o \in O$. Further, set $A_i = \{o_i\}$ for $t \leq 2t'$ and $A_i = O$ for $i > 2t'$. Here, the minimum sized optimistic menu would have to order at least one quantity of o_i to feed agent i for each $i \leq 2t'$. Each o_i can also simultaneously feed agent $2t' + i$ for $i \leq 2t'$. Consequently, we have that the minimum optimistically valid menu has size n .

Meanwhile, in order to ensure pessimistic validity, each of $n/2$ minimal sets need to feed $n/2 + 1$ people. As $n = 4t'$, we need enough of o_i to feed $2t' + 1$ agents. As $s_{o_i} = 2$, we need to order $t' + 1$ quantities of o_i , for each $i \in [2t']$. Thus, the minimum pessimistic menu would have size $(\frac{n}{2})(\frac{n}{2} + 2)$. This exactly matches the upper bound obtained.

Chained. We now consider the case where the laminar tree is simply a path. That is, for the distinct acceptability sets A^1, \dots, A^k , we have that they form a chain of subsets $A^k \subseteq \dots \subseteq A^1$.

Observe that under a pessimistically valid menu must be able to feed everyone from A^k , while an optimistically valid menu need not. As we argued in the case for laminar, the minimum sized menu to feed n agents from A^k would have size at most $\alpha \lceil \frac{n}{\alpha} \rceil \leq n + \alpha - 1$ where $\alpha = \min_{o \in A^k} s_o$. Consequently, in order to maximize the waste of pessimism under I , we need α to be suitably large.

If $\alpha \geq n$, clearly, there would be no waste of pessimism, as even in the minimum sized optimistic menu, to feed the agents with acceptability set A^k , we need a size of at least α which will allow us to feed all other agents as well. Thus, the maximum possible waste of pessimism would come when $\alpha = n - 1$. As a result, we have that

$$WoP(I) \leq \frac{n + n - 1}{n} = \frac{2n - 1}{n} \leq 2.$$

A tight example would come when we have n agents with two options $O = \{o_1, o_2\}$ where $s_{o_1} = n - 1$ and $s_{o_2} = 1$. We set $A_1 = A_2 = \dots = A_{n-1} = \{o_1\}$ and $A_n = \{o_1, o_2\}$.

Identical. Observe that under identical acceptability relations every optimistically valid menu is pessimistically valid. Hence there would be no waste of pessimism. \square