USING LATTICE GEOMETRY TO FIND ALL STABLE ALLOCATIONS

MAX CYTRYNBAUM

Abstract. In this paper, we give an algorithm to find all core allocations in a general model of multilateral many-to-many matching with contracts. We develop a notion of "information sharing" in lattices, showing how lattice geometry can be exploited to produce a relatively fast algorithm that returns the full set of core outcomes. We show how to apply our technique to more general economic problems and, as an application, construct the first algorithm to find *all* stable allocations in bilateral matching with contracts markets when agents have substitutable preferences.

I am very grateful to Scott Kominers for guidance and encouragement, as well as for providing the inspiration for this project.

1. Introduction

1.1. **Introduction and Motivation.** This paper extends an algorithm of Echenique and Yenmez (2013) to find *all* core allocations in a general model of multilateral many-to-many matching with contracts. Our model nests a large number of previous matching models. We show that, as in the case of matching with strict preferences over colleagues in Echenique and Yenmez (2013), core allocations can be characterized in terms of the fixed points of an antitone operator on a certain complete lattice. Our main contribution is to show how to fully exploit the geometric relationships implied by the problem's lattice structure. Specifically, we develop a notion of "information sharing" in lattices, showing how standard graph-theoretic techniques can be used to efficiently combine the geometric information created while searching for the full lattice of stable allocations. We construct and prove correctness for an algorithm that implements this idea, and show how this approach may be extended to a wider class of economic problems for which there is a complete lattice structure. To conclude, we restrict our attention to two-sided many-to-many matching with contracts with substitutable preferences, showing how information sharing may be used to find *all* stable allocations.

There are a variety of results showing the existence of a lattice of stable outcomes when preferences satisfy a substitutability condition. Usually, these results proceed by showing that substitutability implies the existence of an isotone operator on a complete lattice containing the stable matchings[1]. By iteration from a maximal element, one eventually arrives at an extremal stable matching. However, in the absence of substitutability, not only are stable matchings not guaranteed to exist, but the usual isotone operator is not guaranteed to find any that do exist. Moreover, even if substitutability does hold, it is often unclear how to find any but the extremal stable matchings.

Fortunately, our results show that, even without a substitutability guarantee, there still exists a relatively fast algorithm that either returns all *core* allocations or shows that none exist. The core is a natural replacement for stability when looking for reasonable economic outcomes in matching markets that either lack a natural bilateral structure or in which agents' preferences may not satisfy substitutability. Moreover, for bilateral matching with contracts markets where substitutability is guaranteed, the information sharing technique that we develop can be modified to produce an algorithm that finds *all* stable allocations, not just the extremal ones.

The information sharing idea developed in this paper is also of independent theoretical interest. Lattice structures arise frequently in economics, but it is often unclear how to make full use of a problem's lattice geometry. Section 4.2 shows that, for a certain class of economic problems, lattice geometry creates natural redundancies and shared geometric information that can be exploited to quickly compute objects of economic interest.

1.2. **Related Literature.** This paper is related to a broad literature on finding stable outcomes. Roth and Peranson (1999) is a well-known computational study examining the existence of stable outcomes in the National Residency Matching Program. For classical matching problems, Martínez et al. (2004) develops an algorithm for finding all stable matchings in a many-to-many matching market, while McVitie and Wilson (1971) and Irving and Leather (1986) show how to find all stable matchings in one-to-one matching markets. More recently, papers such as Sethuraman et al. (2006) and Schwarz and Yenmez (2011) have

[1]See, for instance, Hatfield and Milgrom (2005) or Hatfield and Kominers (2012).

shown how to find "median" stable matchings, which in some sense balance the interest of market participants. See Gusfield and Irving (1989) for a review of algorithms for finding all core matchings in the classical setting.

The algorithm developed in this paper is an extension of antitone operator approach developed in Echenique (2007) and Echenique and Yenmez (2013). Kominers (2010) shows that this approach finds all stable matchings in classical matching, while Kojima (2015) shows how to extend the approach to the case of matching with couples.

2. Characterizing Core Allocations

In this section, we show how core allocations can be characterized as the fixed points of an antitone operator on a certain complete lattice. First, we give details on our model and notation.

2.1. Model and Solution Concepts.

We consider a finite set of contracts X, each of which is associated with at least one agent $a \in A$. We call a subset $Y \subseteq X$ an *allocation*, and let 2^X denote the set of all allocations. Let $d(x)$ be the set agents associated with a contract $x \in X$, and extend this definition to allocations by writing $d(Y) \equiv \bigcup_{y \in Y} d(y)$. Note that, in general, we may have $|d(x)| > 2$ for multilateral contracts. For instance, we can view X as the set of potential multilateral relationships in a trading network[2]. In particular, we *do not* impose a two-sided market structure in this section.

For $a \in A$, we let $Y_a = \{y \in Y : a \in d(y)\}$ denote the set of contracts associated with that agent (note that we may have $Y_a = \emptyset$). Thus, 2^{X_a} denotes the set of all allocations naming an agent $a \in A$. We assume that each $a \in A$ has strict preferences over $Y \in 2^{X_a}$, where the utility of an allocation is given by the one-to-one function $U_a : 2^{X_a} \to \mathbb{R}$. Let $\succ_a$ denote the strict preference relation induced by these utility functions over bundles $Y \in 2^{X_a}$, with $\succeq_a$ denoting the weak relation. Thus, $Y \succeq_a Z \iff Y \succ_a Z$ or $Y = Z$. Throughout this paper, we will assume that $\emptyset_a \in X$ for all $a \in A$, where $\emptyset_a$ denotes a being unmatched. Note that $d(\emptyset_a) = \{a\}$.

An allocation $Y \subseteq X$ is said to be in the *core*[3] if there does not exist a non-empty blocking set $Z \subseteq X$ such that

$$U_b(Z_b) \geq U_b(Y_b) \qquad \forall b \in d(Z)$$

where *at least one* of the above inequalities holds strictly. We denote the core by $\mathrm{C}(X, U)$.

Similarly, an allocation Y is said to be in the *weak core* if there does not exist a non-empty blocking set $Z \subseteq X$ such that

$$U_b(Z_b) > U_b(Y_b) \qquad \forall b \in d(Z)$$

We denote the weak core by $\mathrm{WC}(X, U)$. Note that $\mathrm{WC}(X, U) \supseteq \mathrm{C}(X, U)$.

It is easy to see that core allocations need not exist in such a general setting. Our approach will be to construct an algorithm that either finds all core allocations or shows that none exist.

[2]This setting is thus a slight generalization of Hatfield and Kominers (2012) model of bilateral contracts in a trading network.

[3]The core concept defined here is sometimes also called the strong core or "core by weak domination"; see Roth and Sotomayor (1996), for instance. Here, we follow the core concept used in Echenique and Yenmez (2013).

2.2. **Fixed Preallocations and the Core.** We build an appropriate framework in which to generalize the fixed point construction discussed above. We start by generalizing the classical notion of a *prematching*.

Definition 2.1 (Preallocation)**.** We call a map $\phi : A \to 2^X$ a *preallocation* if $\phi(a) \in 2^{X_a}$ for all $a \in A$. Let $(2^X)^A$ denote the set of all preallocations.

Intuitively, a preallocation assigns each agent to a bundle of contracts naming him or her. We may think of $\phi(a)$ as the set of contracts "held" by agent a.

We can associate each allocation $Y \subseteq X$ with a unique preallocation ϕ_Y in a natural way by setting $\phi_Y(a) = Y_a$ for $a \in d(Y)$ and $\phi_Y(a) = \emptyset_a$ otherwise.

Remark 2.2. Note, however, that not all preallocations can be derived from allocations in this way. For example, consider the case where $\emptyset \neq \phi(a)_b \neq \phi(b)_a$. In the preallocation ϕ, a holds contracts naming b that are *not* in the bundle of contracts held by b naming a. In particular, there does not exist an allocation Y such that $\phi = \phi_Y$.

With this example in mind, we say that $\phi \in (2^X)^A$ is a *coordinated* preallocation if there exists an allocation Y such that $\phi = \phi_Y$. We denote the set of all cooordinated preallocations by $\mathrm{CP} \subseteq (2^X)^A$. Our method proceeds by identifying allocations $Y \in \mathrm{C}(X, U)$ with fixed points of an operator on preallocations, generalizing the construction in Echenique and Yenmez (2013).

For each agent $a \in A$, we define

$$V(\phi, a) = \{Z \in 2^{X_a} : \exists Y \in 2^X \, s.t. \, Y_a = Z, \, Y_b \succeq_b \phi(b) \; \forall b \in d(Y) \setminus \{a\}\}$$

Intuitively, $V(\phi, a)$ is the *possibility set* for an agent a at a preallocation ϕ. It contains all bundles of contracts naming a that are part of a larger economy Y in which every other agent $b \in d(Y)$ weakly prefers their contracts under Y to their contracts under the preallocation ϕ.

Next, we define an operator $T : (2^X)^A \to (2^X)^A$ by setting $T\phi(a) = \max V(\phi, a)$, where the maximum is taken under the preference relation $\succeq_a$ for each $a \in A$. Note that $\emptyset_a \in V(\phi, a)$ for any $\phi \in (2^X)^A$, so T is well-defined. Let $\mathcal{E}(T)$ denote the fixed points of T. Define $\mathrm{E}(T) = \{Y \in 2^X : \phi_Y \in \mathcal{E}(T)\}$, the collection of allocations Y whose corresponding preallocation ϕ_Y is fixed by T.

Before our first result, we note a simple fact: if $\phi \in \mathrm{CP}$, then $\phi(a) \in V(\phi, a)$. To see this, note that $\phi \in \mathrm{CP}$ means that there exists $Y \subseteq X$ with $\phi = \phi_Y$. Then Y is an allocation satisfying the conditions in $V(\phi_Y, a)$, so that $Y_a = \phi(a) \in V(\phi, a)$. With the definitions above, we have a simple result

Lemma 2.3. $\mathrm{E}(T) = \mathrm{C}(X, U)$

Proof. First, suppose that $Y \notin \mathrm{C}(X, U)$. Then by definition, there exists some blocking allocation $\emptyset \neq Z \subseteq X$ such $Z_b \succeq_b Y_b$ for all $b \in d(Z)$. Let $a \in d(Z)$ be such that the inequality above is strict, and consider $\phi = \phi_Y$. In particular, $Z_b \succeq_b \phi_Y(b)$ for all $b \in d(Z) \setminus \{a\}$, so that $Z_a \in V(\phi_Y, a)$. Then $T\phi_Y(a) = \max V(\phi_Y, a) \succeq_a Z_a \succ_a Y_a = \phi_Y(a)$, so $T\phi_Y(a) \neq \phi_Y(a)$, and $Y \notin \mathrm{E}(T)$.

Suppose, conversely, that $Y \notin \mathrm{E}(T)$ so that $\phi_Y \notin \mathcal{E}(T)$. Then there exists an agent $a \in A$ such that $T\phi_Y(a) = Z_a \neq \phi_Y(a)$ for some allocation Z. By the definition of $V(\phi_Y, a)$, we have $Z_b \succeq_b \phi_Y(b) = Y_b$ for $b \in d(Z) \setminus \{a\}$. We know ϕ_Y is coordinated, so by the simple fact

above $\phi_Y(a) \in V(\phi_Y, a)$. Then $Z_a = T\phi_Y(a) \succ_a \phi_Y(a) = Y_a$. Then Z is a blocking coalition for Y, so $Y \notin \mathrm{C}(X, U)$. □

Thus, we have identified $\mathrm{C}(X, U)$ with the set of *coordinated* preallocations $\phi \in \mathrm{CP}$ such that $T\phi = \phi$. This result shows that an algorithm that finds all $\phi \in \mathcal{E}(T)$ will also find all core matchings.

However, if there are *uncoordinated* preallocations that are also fixed by T, such an algorithm may return extraneous solutions not associated with any core matching. The following lemma, which is essential for the construction of our algorithm, shows that there are no such preallocations. Note that this result significantly generalizes the corresponding lemma in Echenique and Yenmez (2013) and also subsumes the main result of Kojima (2015).

Lemma 2.4. $\mathcal{E}(T) \subseteq \mathrm{CP}$

Proof. We begin with an important fact that will be used repeatedly. Suppose that $\phi \in \mathcal{E}(T)$. Then for any $a \in A$, we have $\phi(a) = T\phi(a) \in V(\phi, a)$. Thus, there exists an allocation Y such that $Y_a = \phi(a)$, and $Y_b \succeq_b \phi(b)$ for all $b \in d(Y) \setminus \{a\}$. Since $Y_a = \phi(a)$, then in fact $Y_b \succeq_b \phi(b)$ holds *for all* agents $b \in d(Y)$. For any $b \in d(Y)$, Y then satisfies the conditions in the definition of $V(\phi, b)$, so $Y_b \in V(\phi, b)$. Therefore, $\phi(b) = T\phi(b) \succeq Y_b \succeq \phi(b)$, so equality holds throughout. In particular, $Y_b = \phi(b)$ for all $b \in d(Y)$.

Fix $a_1 \in A$. Since $\phi \in \mathcal{E}(T)$, the argument above shows that there exists an allocation Y such that $Y_{a_1} = \phi(a_1)$, and, in particular, $Y_b = \phi(b)$ for all $b \in d(Y)$. Therefore, the collection of global allocations available to a_1 at ϕ

$$G(\phi, a_1) = \{Y \in 2^X : Y_{a_1} = \phi(a_1),\ Y_b \succeq \phi(b)\ \forall b \in d(Y)\}$$

is non-empty, so there exists an allocation

$$Y \in \operatorname*{argmax}_{Z \in G(\phi, a_1)} |d(Z)|$$

Let $A_1 = d(Y)$. If $A_1 = A$, we are done, since then by the construction above $Y_a = \phi(a)$ for all $a \in A$, so $\phi = \phi_Y$ and $\phi \in \mathrm{CP}$.

Then assume that $A_1 \neq A$, and pick $a_2 \in A \setminus A_1$. By the fact at the beginning of the proof, there exists an allocation Z such that $Z_{a_2} = \phi(a_2)$, and, in fact, $\phi(b) = Z_b$ for all $b \in d(Z)$. Define $A_2 = d(Z) \cap A_1^c$, which is non-empty by construction. Let $b \in A_2$. We will show that $d(\phi(b)) \cap A_1 = \emptyset$. That is, under the preallocation ϕ, agent $b \in A_2$ is *not* holding any contracts that name agents in A_1.

Suppose not, so there exists $c \in A_1 \cap d(\phi(b))$. Then, in particular, $c \in d(\phi(b)) = d(Z_b) \subseteq d(Z)$, so applying the fact proved at the beginning, $Z_c = \phi(c) = Y_c$. Since $c \in d(Z_b)$, there exists a contract $z \in Z_c$ naming both c and b. Then $b \in Z_c = Y_c$, so $b \in d(Y_c) \subseteq d(Y) = A_1$, so $b \in A_1 \cap A_2 = \emptyset$. This is a contradiction, so it must be the case that $d(\phi(b)) \cap A_1 = \emptyset$ for all $b \in A_2$.

Define $S = \bigcup_{b \in A_2} \phi(b)$. We have just shown that $d(S) \subseteq A_1^c$. We also have $d(S) = \bigcup_{b \in A_2} d(\phi(b)) = \bigcup_{b \in A_2} d(Z_b) \subseteq d(Z)$, so $d(S) \subseteq A_1^c \cap d(Z) = A_2$. Clearly $b \in d(\phi(b))$ for all $b \in A_2$, so $A_2 \subseteq d(S)$. Then $A_2 = d(S)$.

Set $W = Y \cup S$. We have now shown that $A_2 \neq \emptyset$ and $A_1 \cap A_2 = \emptyset$. Since $d(Y) = A_1$ and $d(W) = A_2$, it follows that $W \cap Y = \emptyset$, so we have

(1) $W_b = Y_b = \phi(b)$ for all $b \in A_1$.
(2) $W_b = S_b = \phi(b)$ for all $b \in A_2$.

Then apparently $W \in G(\phi, a_1)$ as defined above. However, by construction $|d(W)| > |d(Y)|$, which contradicts our original choice of Y. This finishes the proof. □

2.2.1. *Discussion.* Combining these lemmas, we see that searching for core allocations in a very general model of multilateral matching with contracts is equivalent to searching for the fixed points of T. Our algorithm depends heavily on this result, which shows, critically, that the fixed points $\mathcal{E}(T)$ are only as dense in $(2^X)^A$ as the core outcomes.

Our maximal domain results will show that this is not the case for the natural extension of this method to *weak* core outcomes $\mathrm{WC}(X, U)$. For weak core outcomes, where the lattice algorithm fails, T also fixes at a large number of extraneous, uncoordinated preallocations (see Lemma 4.1).

2.3. **The Lattice of Fixed Preallocations.** In this section, we generalize constructions from Echenique and Yenmez (2013) showing the the fixed points of the squared operator T^2 form a lattice. First, we define a natural partial order on the set of preallocations $(2^X)^A$.

Say that $\phi \succ \phi'$ if and only if $\phi(a) \succeq_a \phi'(a)$ for all $a \in A$, where at least one of these inequalities *holds strictly*. Thus, we write $\phi \succeq \phi'$ if and only if $\phi \succ \phi'$ or $\phi = \phi'$. This is a product order on a product space, which makes $(2^X)^A$ into a complete lattice. Next, we give a sequence of results concerning the operator T and its fixed points. These results are an almost direct extension of the results in Lemma 4 through Proposition 8 of Echenique and Yenmez (2013). For the purposes of illustration, we include the proof the first result for the more general framework of preallocations considered in this paper. The rest of the lemmas follow from straightforward extensions of the work in Echenique and Yenmez (2013).

Lemma 2.5. *T is antitone*

Proof. Let $\phi \preceq \phi'$ be preallocations. Fix $a \in A$, and let $Z \in V(\phi', a)$. Then there is an allocation $Y \subseteq X$ with $Y_a = Z$ such that $Y_b \succeq_b \phi'(b)$ for $b \in d(Y) \setminus \{a\}$. Then $Y_b \succeq_b \phi'(b) \succeq_b \phi_b$ also for all such agents, so we also have $Z \in V(\phi, a)$. Then $V(\phi, a) \supseteq V(\phi', a)$, so that $T\phi(a) \succeq T\phi'(a)$. The agent a was arbitrary, so $T\phi \succeq T\phi'$ under our partial order. □

The following lemmas follow exactly as in Echenique and Yenmez (2013), using the antitonicity of T.

Corollary 2.6. *T^2 is isotone, and $\mathcal{E}(T^2)$ is a non-empty complete lattice.*

Lemma 2.7. *No two preallocations ϕ and ϕ' can be compared under the partial order on $(2^X)^A$.*

Lemma 2.8. *There exist preallocations $\overline{\phi}$ and $\underline{\phi}$ such that for all $\phi \in \mathcal{E}(T)$, we have $\overline{\phi} \succeq \phi \succeq \underline{\phi}$. Moreover, if $\phi = \overline{\phi}$ or $\phi = \underline{\phi}$, then $\mathcal{E}(T) = \{\phi\}$.*

3. Sharing Lattice Information to Find All Core Allocations

3.1. **Introduction.** In this section, we give an algorithm that finds *all* core allocations in the model of multilateral matching with contracts considered above. Our algorithm builds upon the original approach in Echenique and Yenmez (2013). In particular, we show how to fully exploit the problem's complete lattice structure to more efficiently find the full set of core matchings.

The algorithm proceeds by successively initializing modified versions of the original matching problem, in which each agent has a truncated preference list. Our main contribution

is to realize that many of the subproblems created while searching the lattice $\mathcal{E}(T^2)$ share geometric information with other subproblems. We show how fast digraph algorithms may be used to combine this shared information and more quickly identify the full set of core matchings.

3.2. **Intuition and Notation.**

3.2.1. *Algorithm Intuition.* Let ϕ^* denote the largest preallocation under the partial order $\succ$. That is, $\phi^*(a) = \operatorname{argmax} U_a(\phi(a))$ for each $a \in A$. Suppose that the total number of agents $|A| = m > 0$. We will use $\langle\phi\rangle$ or $\langle\phi(1)\dots\phi(m)\rangle$ to denote a version of the original problem in which each agent a's preference list is truncated to allocations in 2^{X_a} ranked weakly below $\phi(a)$ by agent a. We will often identify a problem $\langle\phi\rangle$ with its maximal point ϕ. Let $\mathcal{E}(\phi) \equiv \{\phi' \in \mathcal{E}(T) : \phi' \preceq \phi\}$ denote the fixed points of T below ϕ.

By Tarski's Theorem[4], the sequence $(T^2)^k\phi^*$ converges to a preallocation $\overline{\phi}$, fixed by T^2, which is the maximal point of the lattice $\mathcal{E}(T^2)$. If $T\overline{\phi} = \overline{\phi}$, then by Lemma 2.8 showing that no elements of $\mathcal{E}(T^2)$ are ranked by "$\preceq$", $\mathcal{E}(T) = \{\overline{\phi}\}$. So suppose that $T\overline{\phi} \neq \overline{\phi}$. Clearly $\mathcal{E}(T) \subseteq \mathcal{E}(T^2)$ for any operator, and, by iteratively applying T^2, we have learned that $\mathcal{E}(T) \subseteq \mathcal{E}(T^2) \subseteq \{\phi' : \phi' \preceq \overline{\phi}\} \equiv \{\phi' \preceq \overline{\phi}\}$. We will sometimes denote relations of this type by $\mathcal{E}(T) \preceq \overline{\phi}$.

Since $T\overline{\phi} \neq \overline{\phi}$, apparently $\mathcal{E}(T) \subseteq \bigcup_{i=1}^m \{\phi' \preceq \overline{\phi} - e_i\}$, where e_i denotes the standard unit vector[5] in $\mathbb{R}^m$. Consider a subproblem of the form $\langle\overline{\phi} - e_i\rangle$, and let T_i be the operator corresponding to this subproblem, where agents' preferences are truncated above $\overline{\phi} - e_i$. The key insight of Echenique (2007), which generalizes to the current model, is that $\mathcal{E}(T) \cap \{\phi' \leq \overline{\phi} - e_i\} \subseteq \mathcal{E}(T_i)$. That is, the fixed points of T contained in the cone $\{\phi' \preceq \overline{\phi} - e_i\}$ are also fixed points of the new problem $\langle\overline{\phi} - e_i\rangle$ with operator T_i. Then apparently we have $\mathcal{E}(T) \cap \{\phi' \leq \overline{\phi} - e_i\} \subseteq \mathcal{E}(T_i) \subseteq \mathcal{E}(T_i^2)$, so information about the lattice of fixed points of T_i^2 can be used to bound $\mathcal{E}(T)$. The general result in our setting is as follows

Lemma 3.1. *Suppose $\phi \preceq \widehat{\phi}$, and let $\widehat{T}$ denote the core operator for the problem $\langle\widehat{\phi}\rangle$. If $\phi \in \mathcal{E}(T)$, then $\phi \in \mathcal{E}(\widehat{T})$. In particular, $\mathcal{E}(T) \cap \{\phi \leq \widehat{\phi}\} \subseteq \mathcal{E}(\widehat{T}^2)$.*

Proof. Let $\widehat{V}(\phi, a)$ be the defining set for $\widehat{T}$ in the subproblem. Then for any $a \in A$ and $\phi \in (2^X)^A$ we have $\widehat{V}(\phi, a) = \{Z \in 2^{X_a} : \exists Y \in 2^X \, s.t. \, Y_a = Z, \, \widehat{\phi}(b) \succeq_b Y_b \succeq_b \phi(b) \; \forall b \in d(Y) \setminus \{a\}\}$ then clearly $\widehat{V}(\phi, a) \subseteq V(\phi, a)$. We can compute $\widehat{T}\phi(a) = \max \widehat{V}(\phi, a) \leq \max V(\phi, a) = T\phi(a)$ for any preallocation ϕ.

Now, suppose that $\phi \in \mathcal{E}(T) \cap \{\phi' \preceq \widehat{\phi}\}$. Then we have have $\phi(a) = T\phi(a) \succeq_a \widehat{T}\phi(a)$ for all $a \in A$. By Lemma 2.4 above, $\mathcal{E}(T) \subseteq \text{CP}$, so $\phi = \phi_Y$ for some allocation $Y \in 2^X$. In particular, $Y_a = \phi(a) \preceq_a \widehat{\phi}(a)$, so $\phi(a) \in \widehat{V}(\phi, a)$. Then by the definition of $\widehat{T}$, we get $\widehat{T}\phi(a) \succeq_a \phi(a)$.

Combining this with the statement above, we have $\phi(a) = \widehat{T}\phi(a)$, so $\phi \in \mathcal{E}(\widehat{T})$. □

[4]This is easy to see. Note that $T^2\phi^* \preceq \phi^*$ by maximality, so $(T^2)^k\phi^* \preceq (T^2)^{k+1}\phi^*$ for any $k \geq 1$ by isotonicity. Then this sequence is monotonically decreasing and bounded above $\min \mathcal{E}(T^2)$ on a finite lattice, so it converges in finitely many iterations, say at $k = \ell$. Let $\phi' \in \mathcal{E}(T^2)$ another fixed point, then $\phi^* \succeq \phi'$, so $\overline{\phi} = (T^2)^\ell\phi^* \succeq (T^2)^\ell\phi' = \phi'$, so $\overline{\phi} = \max(\mathcal{E}(T^2))$.

[5]identifying $(2^X)^A$ with a grid in $\mathbb{R}^m$ as above.

Returning to the discussion above, we showed that $\mathcal{E}(T) \subseteq \bigcup_{i=1}^{m} \{\phi' \preceq \overline{\phi} - e_i\}$, a union of cones in the lattice of preallocations. That is, $\mathcal{E}(T) \subseteq \bigcup_{i=1}^{m} \mathcal{E}(\overline{\phi} - e_i)$. Fix i and consider $\mathcal{E}(\overline{\phi} - e_i)$, the fixed points of T below $\overline{\phi} - e_i$. For convenience, denote $S_i \equiv T_i^2$ and $\overline{\phi}_i \equiv \overline{\phi} - e_i$. Starting with $\overline{\phi}_i$, the unanimously most preferred preallocation in the subproblem $\langle \overline{\phi}_i \rangle$, Lemma 2.3 and the Tarski argument used above show that each iteration of S_i gives a "cone guarantee" on the fixed points $\mathcal{E}(T)$ of the form $\mathcal{E}(\overline{\phi}_i) \preceq S_i^k \overline{\phi}_i$.

Pursuing this strategy for $i = 1 \dots m$, we can bound the complete set of fixed points $\mathcal{E}(T)$ in a union of subproblem cones. We continue this strategy recursively by generating m new subproblems whenever a monotonic sequence of the form $S_i^k \overline{\phi}_i$ stops at a fixed point of S_i^k. In this way, we will eventually find the full set of core outcomes, directly extending the Echenique and Yenmez (2013) algorithm to the present setting.

However, depending on the structure of the lattice $\mathcal{E}(T^2)$, the strategy described above can actually be quite similar to greedy search. In particular, the number of subproblems generated at the kth recursive level scales exponentially in k as $|A|^k$.As we will show, many of the subproblems generated by this strategy are either redundant or do not need to be solved completely. We will show that we can capitalize on the structure of the preallocation lattice by allowing subproblems to share geometric information with each other. With this modification, evaluation of the full Tarski sequence $S_i^k \overline{\phi}_i$ is often unnecessary. In fact, it is often the case that subproblems can be stopped or removed entirely after a few iterations. We will illustrate this idea with a couple of simple examples.

3.2.2. *Graph-Theoretic Notation.* First, we briefly recall some graph-theoretic notation, which will be useful in the coming sections. A *graph* is a pair (V, E), where V is a collection of *vertices* and $E \subseteq V \times V$ is a collection of *edges*. In a *directed graph* (digraph), the order of vertices in an edge matters. A *directed path* is a sequence $e^1 \dots e^n$ of edges, where $e_2^k = e_1^{k+1}$ for $k \leq n - 1$. A subset $W \subseteq V$ is *strongly connected* if for any $v_i, v_j \in W$, there exist directed paths $v_i \to v_j$ and $v_j \to v_i$ using only vertices in W. Strong connectedness is an equivalence relation on V, and the (disjoint) maximal strongly connected subsets of V are called the *strong components* of G, which we denote by $\mathcal{G}$. Weak connectedness is defined similarly for an undirected graph, where an (undirected) path is defined as above, ignoring the order on vertices in an edge. The *reachable set* from a vertex v is the set of all $v' \in V$ such that there exists a directed path $v \to v'$.

By a *rooted tree*, we mean a graph that is connected with no cycles (paths starting and ending at the same vertex) and has one vertex designated as the *root*. Edges on a rooted tree are naturally oriented away from the root. A collection of such rooted trees is called a *forest* and, for such a collection, for a vertex w we let $r(w)$ denote the root of the tree containing w, and we let $c(w)$ denote the *children* of w, the vertices connected by an edge to w on a path away from the root, and $p(w)$ denote the parent vertex of w, defined similarly[6].

3.2.3. *Example 1 - Colliding Subproblem Cones.* In this section, we give a simple example of how subproblems can share information. For easy of visualization, consider the case $|A| = 2$. As noted previously, we may identify $(2^X)^A$ with a grid, in this case in $\mathbb{R}^2$. As above, we iterate T^2 to find $\overline{\phi} = \max \mathcal{E}(T^2)$, which we identify, for instance, with the integer lattice point $(10, 10) = \overline{\phi}$. Using the notation above, there are isotone operators $S_1 = T_1^2$ and $S_2 = T_2^2$ corresponding to the subproblems $\langle \overline{\phi} - e_1 \rangle$ and $\langle \overline{\phi} - e_2 \rangle$, respectively.

[6]See, for instance, Gross and Yellen (2005) for a more detailed introduction to graph-theoretic terminology.

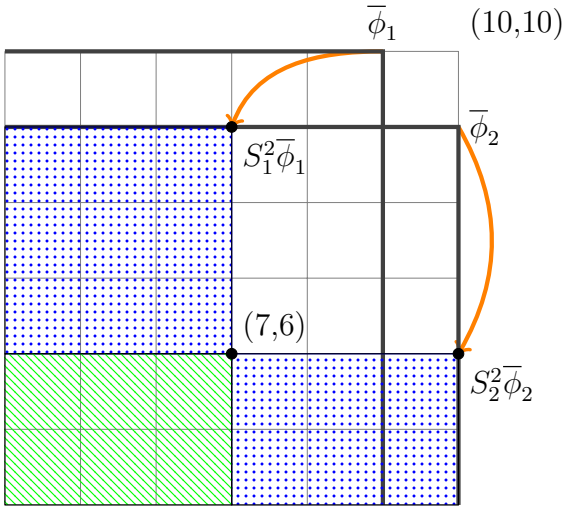

FIGURE 1. Combining Information from Subproblem Cones Guarantees

Beginning with subproblem 1, suppose that $S_1(9,10) = (8,10)$, $S_1^2(9,10) = (7,9)$, and $S_1^3(9,10) = (7,9)$. By our arguments above, we now have a cone guarantee on the fixed points in the first subproblem; specifically, $\mathcal{E}(T) \cap \{(x,y) \preceq (9,10)\} \preceq (7,9) \preceq (10,9) = \overline{\phi}_2$. Therefore, $\mathcal{E}(\overline{\phi}_1) = \mathcal{E}(T) \cap \{(x,y) \preceq (9,10)\} \subseteq \mathcal{E}(T) \cap \{(x,y) \preceq (10,9)\} = \mathcal{E}(\overline{\phi}_2)$. That is, the solutions to subproblem 1, the fixed points of T in $\{(x,y) \preceq (9,10)\}$, are *contained* in the set of solutions to subproblem 2.

Now suppose that we stop working on subproblem 1 and begin iterating with S_2. Suppose that $S_2(10,9) = (10,6) = S_2^2(10,9)$, so the sequence stops. Then we have learned that $\mathcal{E}(\overline{\phi}_1) \subseteq \mathcal{E}(\overline{\phi}_2) \preceq (10,6)$. However, by iterating S_1, we also learned that $\mathcal{E}(\overline{\phi}_1) \preceq (7,9)$. Then apparently $\mathcal{E}(\overline{\phi}_1)$ is contained in the cone intersection $\{(x,y) \preceq (7,9)\} \cap \{(x,y) \preceq (10,6)\} = \{(x,y) \preceq (7,6)\}$. That is, by *combining the information* from subproblem 2 with subproblem 1, we have learned that $\mathcal{E}(\overline{\phi}_1) \preceq (7,6)$.

Note that, even if we learn $\mathcal{E}(\overline{\phi}_1) \subseteq \mathcal{E}(\overline{\phi}_2)$, we may still wish to retain the information associated with subproblem 1. If, for instance, the outcome for subbproblem 2 was instead that $S_2(10,9) = (9,6) = S_2^2(10,9)$, then we would also know that $\mathcal{E}(\overline{\phi}_2) \subseteq \mathcal{E}(\overline{\phi}_1)$, so that $\mathcal{E}(\overline{\phi}_2) \preceq \min((9,6),(7,9)) = (7,6)$ and, in fact, $\mathcal{E}(\overline{\phi}_1) = \mathcal{E}(\overline{\phi}_2)$. The idea of equivalent subproblems is explored further in the next example.

3.2.4. *Example 2 - Chutes and Ladders.* In this section, we give intuition for how to efficiently combine information and track the relations between different subproblem cones. Consider the case $|A| = 3$, where we identify $(10,10,10) = \overline{\phi} = \max \mathcal{E}(T)$. After initializing subproblems at $(9,10,10)$, $(10,9,10)$ and $(10,10,9)$, suppose that we find $S_1(9,10,10) = (9,8,10)$, $S_2(10,9,10) = (10,9,7)$, and $S_3(10,10,9) = (9,10,9)$. As argued above, this shows that $\mathcal{E}(\overline{\phi}_1) \preceq (9,8,10) \preceq (10,9,10) = \overline{\phi}_2$, so that $\mathcal{E}(\overline{\phi}_1) \subseteq \mathcal{E}(\overline{\phi}_2)$. Similarly, our calculations with S_2 and S_3 show that $\mathcal{E}(\overline{\phi}_2) \subseteq \mathcal{E}(\overline{\phi}_3)$ and $\mathcal{E}(\overline{\phi}_3) \subseteq \mathcal{E}(\overline{\phi}_1)$. Combining all these relations, apparently $\mathcal{E}(\overline{\phi}_1) = \mathcal{E}(\overline{\phi}_2) = \mathcal{E}(\overline{\phi}_3) \preceq (9,8,7)$, so the subproblems are equivalent. Thus, we can collapse all of these subproblems to a new subproblem started at $\phi' = (9,8,7)$ *without losing any fixed points* of the original operator T.

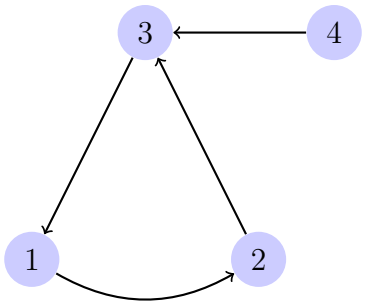

FIGURE 2. A Collision Digraph

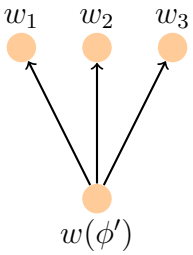

FIGURE 3. A Problem Forest

We have seen that collisions between subproblem cones give information relations of the form $\mathcal{E}(\overline{\phi}_i) \subseteq \mathcal{E}(\overline{\phi}_j)$ regarding the fixed points of T. Consider a digraph that tracks these collisions. Then this digraph has the form shown below, where subproblems correspond to vertices, and there is an edge $v_i \to v_j$ only if problem i "collides" with problem j (made precise below). Since, in particular, the digraph has a directed edge (v_i, v_j) only if $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi_j)$, the equivalence of subproblems is reflected in the connectedness of the graph.

Suppose now that $\overline{\phi}$ is found at some intermediate step in the algorithm, and consider a distinct subproblem near $\overline{\phi}$ started at $\phi_4 = (11, 11, 11)$ with associated isotone operator S. We begin iterating and find that $S(\phi_4) = (10, 10, 8)$, which shows $\mathcal{E}(\phi_4) \subseteq \mathcal{E}(\overline{\phi}_3)$ Then a *single iteration* of S has shown us that $\mathcal{E}(\phi_4) \subseteq \mathcal{E}(\overline{\phi}_3) \preceq (9, 8, 7) = \phi'$, using the relations above. Of course, the same conclusion would hold if we found that $S(\phi_4) \preceq \overline{\phi}_i$ for any i.

We know that $\mathcal{E}(\overline{\phi}_1) = \mathcal{E}(\overline{\phi}_2) = \mathcal{E}(\overline{\phi}_3) \preceq \phi'$. We can track these subproblem relations efficiently with a collection of rooted trees as follows: let vertices w_1, w_2, w_3 be children of $w(\phi')$, a vertex representing $\phi' = (9, 8, 7)$. We denote this relationship by $c(w(\phi')) = \{w_1, w_2, w_3\}$. Using the notation in the previous section, we have the root-vertex relation $r(w_i) = w(\phi')$ for $i = 1, 2, 3$. Note that, by our work above, we have $\mathcal{E}(\phi') = \mathcal{E}(\overline{\phi}_i)$ for all $w_i \in c(w(\phi'))$. Also, $\mathcal{E}(\overline{\phi}_i) = \mathcal{E}(\overline{\phi}_j)$ for all i, j children of the same node.

Suppose we maintain a collection of trees for which parent nodes and child nodes are all related in this way. Consider a solved subproblem $\langle \phi_i \rangle$ with $r(w_i) = w_k$. Then if at some step of the algorithm we obtain a guarantee that $\mathcal{E}(\phi_j) \subseteq \mathcal{E}(\phi_i)$ for an active subproblem $\langle \phi_j \rangle$, the relations built into our problem tree imply that, in fact, $\mathcal{E}(\phi_j) \subseteq \mathcal{E}(\phi_k)$, potentially *skipping a large region of lattice space* that we would otherwise have to search for elements of $\mathcal{E}(T)$. This dynamic gives our algorithm a "Chutes and Ladders" effect. In particular, we make efficient use of both the geometric information shared between active problems and the information gained during previous steps of the algorithm.

3.3. **Informal Algorithm Description.** In this section, we introduce notation and give a brief, intuitive explanation of the full algorithm, building on the examples above. Pseudocode and correctness proofs for the full algorithm follow. First, we discuss indexing.

Labels for Preallocations, Vertices, and Operators: During the algorithm, we build various graphs. For any such graph, each vertex will correspond to a problem $\langle \phi \rangle$, where $\phi \in (2^X)^A$. We thus require a way to denote correspondences between vertices and problems. We thus fix an index set $\mathcal{I}$, which we use to label problems and vertices. We will denote corresponding vertices and problems with the same subindex, for instance, $\phi_i \sim v_i \sim w_i$, where $i \in \mathcal{I}$. We allow the possibility that $\phi_i = \phi_j$ as preallocations for $i \neq j$. Thus, technically, a problem ϕ_i is a preallocation index pair $(\phi, i) \in (2^X)^A \times \mathcal{I}$, and we enforce that the index of every problem initialized during the algorithm is chosen to be unique. When clear, we let $v(\phi)$ and $w(\phi)$ denote the vertices corresponding to a certain problem $\langle \phi \rangle$, where $w(\phi) = w_k$ for some $k \in \mathcal{I}$. Thus, for instance, $v(\phi_i) = v_i$. For clarity, vertices of a digraph G are always denoted by v, while vertices of a forest F are always denoted by w. Similarly, the operator for a subproblem ϕ_i will be denoted by T_i. The algorithm consists of two alternating stages.

Stage 1 - Information Acquisiton: In the first stage, we consider a set of problems Q_0. As above, each problem $\langle \phi_i \rangle$ is associated with an isotone operator $S_i = T_i^2$. In this stage, we iteratively apply S_i to each $\phi_i \in Q_0$, creating new problems when necessary, as discussed below. Let Q be a set of problems $Q_0 \subseteq Q$, which we think of as the problems from previous steps of the algorithm that still contain useful information.

Stopping Conditions: During the algorithm, for a problem $\langle \phi_i \rangle$, we compute the Tarski sequence $(T_i^2)^k \phi_i$, $k \geq 0$. Define the following stopping conditions for an element $(T_i^2)^k \phi_i \equiv \phi'$ of this sequence.

(1) $\phi' \preceq \underline{\phi}$
(2) $T_i^2 \phi' = \phi'$
(3) There exists $\phi_j \in Q$ with $\phi' \preceq \phi_j$ and $\phi_i \not\preceq \phi_j$

Condition (1) corresponds to the problem exiting the lattice $\mathcal{E}(T^2)$. For Condition (2), the Tarski sequence fixes. If $T\phi' = \phi'$, by Lemma 2.7, $\{\phi'' \prec \phi'\} \cap \mathcal{E}(T) = \emptyset$. Otherwise, we need to initialize new problems below ϕ'. For Condition (3), the problem $\langle \phi_i \rangle$ "collides" with some problem in Q, so we stop the sequence and wait for the information sharing subroutine, discussed below.

Stage 2 - Information Combination: In this stage, we use efficient digraph algorithms to combine information from related problems. First, we define a graph structure that relates different problems solved during the algorithm

Definition 3.2. A collection of rooted trees F is said to be *problem forest* for a set of problems Q when the following hold:

(1) If $\phi_k \in Q$, then $w_k \in F$.
(2) If there exists a $w_k \in F$ such that $w_i, w_j \in c(w_k)$, then $\mathcal{E}(\phi_i) = \mathcal{E}(\phi_j)$.
(3) If $w_i = p(w_j)$, then $\phi_i \preceq \phi_j$ and $\mathcal{E}(\phi_j) = \mathcal{E}(\phi_i)$

Thus, siblings in a problem forest share fixed points of T, and the fixed points of a child are the same as the fixed points of its parent. For now, we assume the existence of such a forest for the problems considered below (construction in section 3.4). Next, we define a graph structure that allows us to combine the information from intersecting problem cones.

Definition 3.3. Let F be a collection of rooted trees. A digraph $G = (V, E)$ is said to be a *collision digraph* if $(v_i, v_k) \in E$ if and only if there exist $\phi_i \in Q_0$ and $\phi_j \in Q$ such that (i) $(T_i^2)^\ell \preceq \phi_j$ for some $\ell \geq 1$, (ii) $\phi_i \not\preceq \phi_j$, and (iii) $r(w_j) = w_k$.

That is, using the correspondence between vertices and problems, (v_i, v_k) is added to G only if during Stage 1 problem ϕ_i "collided" with some vertex w_j in the tree which has w_k as its root.

In stage 2 of the algorithm, the "information combination" stage, we start with a collection of problems Q_0 and a list of problem collisions $I(i)$ for each $\phi_i \in Q_0$. We form the collision digraph G and compute its strong components $\mathcal{G} = \{S_j\}_{j\in\mathcal{J}}$, where $\mathcal{J}$ is an index set. For $j \in \mathcal{J}$, pick $v \in S_j$ and compute its reachable set, by which we define H_j, the reachable set of component S_j. Since strong components are strongly connected, this definition is independent of the choice of v.

In Proposition 3.9, we show that our construction of the collision digraph and inductive assumption on the forest F guarantees that $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi_k)$ whenever $(v_i, v_k) \in E$. In particular, we can show that $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi_k)$ whenever $v_i \in S_j$ and $v_k \in H_j$.

Let $Q_{temp} = \emptyset$ and, for each strong component $S_j \in \mathcal{G}$, add $\phi = \min_{v_k \in H_j} \phi_k$ to Q_{temp}. By the guarantees above, we can combine all problems represented in strong component S_j of the digraph into a new active set of problems by letting $Q_0 \leftarrow Q_{temp}$ *without losing any points* of $\mathcal{E}(T)$. Note that, by changing Q_0 in this way, we may have actually caused new cone collisions. We can use these collisions to weakly improve the progress of the algorithm, as above. Thus, we run the information combination subroutine in stage 2 iteratively until there are no more cone collisions. This process terminates, as discussed in Lemma 3.11. After termination, we return to stage 1 with a new collection of active problems $Q_0 \leftarrow Q_{temp}$.

In general, we would like to spend as much time as possible in the information combination stage, since this stage improves the progress of the search for $\mathcal{E}(T)$ by using fast[7] digraph algorithms, as opposed to the often costly process of computing the operator T, which is equivalent to checking whether a given allocation is core.

3.4. **Full Algorithm.** In this section, we state the full algorithm to find all core allocations $\mathrm{C}(X, U)$. Throughout what follows, we maintain global variables F, a rooted forest, and Q, a collection of active and previously solved problems.

Initialization To initialize the algorithm, set $\mathcal{E}_C = \emptyset$ and create an empty forest $F = (\emptyset, \emptyset)$. Apply T^2 iteratively to each of ϕ^* and ϕ_*, the maximal and minimal allocations in $(2^X)^A$, to find $\overline{\phi} = \max \mathcal{E}(T^2)$ and $\underline{\phi} = \min \mathcal{E}(T^2)$, respectively. If $T\overline{\phi} = \overline{\phi}$ set $\mathcal{E}_C = \{\overline{\phi}\}$ and terminate. Similarly, if $T\underline{\phi} = \underline{\phi}$, then set $\mathcal{E}_C = \{\underline{\phi}\}$ and terminate. Otherwise, for $i = 1 \dots m$ add $\overline{\phi} - e_i$ to Q and Q_0, and add $w(\phi - e_i)$ to F as an isolated root. Then, run FindStrictCore. Subroutine details are given below.

Subroutine 3.4 (InformationAcquire)**.** *Initialize $Q_A = Q_0$, and set $\mathcal{E}_{temp} = Q_I = I = \emptyset$.*

If $\phi_i \in Q_0$ has either (i) $\phi_i \preceq \underline{\phi}$ or (ii) there exists $\phi_j \in Q_0$ s.t. $\phi_i \prec \phi_j$, then remove ϕ_i from Q_A, Q_0, and Q.

While $Q_A \neq \emptyset$, do the following:

For each $\phi_i \in Q_A$, compute the Tarski sequence $(T_i^2)^k \phi_i$ until a stopping condition (as above) is triggered for some $k = \ell$. Set $\phi' = (T_i^2)^\ell \phi_i$, and do the following:

(a) *Add $w(\phi')$ to F with $c(w(\phi')) = w(\phi_i)$*

[7]Tarjan's algorithm for finding strong components of a graph $G = (V, E)$ runs in $\mathcal{O}(|V| + |E|)$, for instance.

Algorithm 1 FindStrictCore

Input: Q_0

Initialize as above
Set $\mathcal{E}_C = \emptyset$
while $Q_0 \neq \emptyset$ **do**
  $(Q_0, I, \mathcal{E}_{temp}) \leftarrow InformationAcquire(Q_0)$
  $\mathcal{E}_C = \mathcal{E}_C \cup \mathcal{E}_{temp}$
  while $I \neq \emptyset$ **do**
    $(Q_0, I) \leftarrow InformationCombine(Q_0, I)$
  end while
end while
return $\mathcal{E}_C$

(b) *Remove* ϕ_i *from* Q_A.

For stopping conditions[8] *(2) and (3), do the following:*

(2) *(i) If* $T\phi' = \phi'$ *then add* ϕ' *to* $\mathcal{E}_{temp}$. *Else, (ii) for* $1 \leq i \leq m$, *add* $\phi' - e_i$ *to* Q_0, Q_A, *and* Q *if* $\phi' - e_i$ *is not already in* Q. *Finally, (iii) for each problem* ϕ'' *just added to* Q_0, *add* $w(\phi'')$ *to* F *as an isolated root.*

(3) *By assumption, condition (3) was triggered by a "collision" with at least one problem* $\phi_j \in Q$. *For all such* j, *(a) add* ϕ_i *to* Q_I *and (b) add* j *to the collision index set* $I(i)$.

Return $(Q_I, I, \mathcal{E}_{temp})$.

Subroutine 3.5 (InformationCombine)**.** *To initialize, form an empty digraph* $G = (V, E)$ *with* $V = E = \emptyset$, *set* $Q_{temp} = \emptyset$, *and add* i *to* $I(i)$ *for each* $\phi_i \in Q_0$. *Next, build the collision digraph as follows:*

For each $\phi_i \in Q_0$, *(i) add* v_i *to* V, *and for all collision indices* $j \in I(i)$, *(ii) compute* $w_k = r(w_j)$ *and (iii) add* v_k *to* V *and* (v_i, v_k) *to* E. *Next, do the following computations with* G*:*

(1) *Compute the strong components* $\mathcal{G}$ *of* G.
(2) *Remove* $\{v_k\} \in \mathcal{G}$ *if* $\phi_k \notin Q_0$, *leaving trimmed components* $\{S_j\}_{j \in \mathcal{J}}$
(3) *Compute the reachable set* H_j *for each* S_j *with* $j \in \mathcal{J}$.

For each $j \in \mathcal{J}$, *combine information to form new problems:*

(a) *Add* $\phi' = \min_{v_k \in H_j} \phi_k$ *to* Q_{temp}.

(b) *Add the vertex* $w(\phi')$ *to* F *with* $c(w(\phi')) = \{r(w_k) : v_k \in S_j\}$.

Reset $I(i) = \emptyset$ *for* $i \in \mathcal{I}$. *To check for new collisions, do the following:*

For each $\phi' \in Q_{temp}$, *if* $\exists \phi_k \in Q$ *and* $w_j \in c(w_i)$ *such that* $\phi' \preceq \phi_k$ *and* $\phi_j \not\preceq \phi_k$, *then add collision index* k *to* $I(i)$.

Set $Q_0 \leftarrow Q_{temp}$ *and* $Q \leftarrow Q \cup Q_{temp}$.

Return (Q_0, I).

3.5. **Algorithm Analysis.** In this section, we show termination and correctness of FindStrictCore. Along the way, we prove structure theorems for the collision digraph and problem forest defined above.

[8]If both stopping conditions (1) and (3) both occur, use condition (1): the problem has left the lattice $\mathcal{E}(T^2)$, so there is no reason to seek further information.

Our proofs of correctness and termination will make use of the following key lemma, which shows that we can track the relations between subsets of $\mathcal{E}(T)$ with the graph F constructed during the algorithm.

Lemma 3.6. *At any point during the execution of FindStrictCore*

(1) *F is a problem forest*
(2) *If $k \in I(i)$, where $r(w_i) = w_j$, then $\mathcal{E}(\phi_j) \subseteq \mathcal{E}(\phi_k)$*

Before proving Lemma 3.6, we give definitions, some needed auxiliary lemmas, and a structure result for the collision digraph G.

For a forest (a collection of rooted trees) define the *root set* $R(F) = \{r(w) : w \in F\}$. That is, $R(F)$ consists of all roots of trees in the forest. For clarity, if t is a rooted tree in F, we will let $r(t)$ denote the (unique) root node of this tree. The depth of a vertex w in a forest F is defined as its depth in the unique rooted tree containing it, denoted by $D(w, F)$. The reachable set of a vertex w in a forest F is denoted similarly by $H(w, F)$.

Next, we develop some graph-theoretic results for problem forests.

Lemma 3.7. *Let F be a rooted forest. Suppose that F' is obtained from F in one of the following ways*

(1) *Adding an isolated vertex to F*
(2) *Adding a vertex w and directed edges (w, w') with $w' \in R(F)$.*

Then the following are true

(i) *F' is also a rooted forest.*
(ii) *$H(w, F) = H(w, F')$, the reachable set of a vertex $w \in F$ is unchanged*
(iii) *$D(w, F') \geq D(w, F)$, the depth of a vertex $w \in F$ weakly increases.*

Proof. Note that a graph $G = (V, E)$ is a tree[9] if and only if G is connected and $|E| = |V| - 1$. Consider the forest F and let $\{t_i\}_{i \in I}$ be its constituent rooted trees. That is, $\{t_i\}_{i \in I}$ is the collection of maximal connected components of F. For case (1), no edges were added, so the new collection of maximal connected components is now $\{t_i\}_{i \in I} \cup \{w\}$. Each t_i is still a rooted tree by assumption, and $\{w\}$ is also trivially a rooted tree with root w.

For case (2) above, let w be the added vertex and J the subset of tree indices such that, for $i \in J$, edge $(w, r(t_i))$ is added to F. Let $i \in J$. Since each such t_i is connected and connectedness is transitive, then $t' = \cup_{i \in J} t_i$ is a (weak) connected component of F'. Then apparently the connected components of F' are $\{t_i\}_{i \notin J}$ and $\{t'\}$. It suffices to show that t' is a rooted tree. Let $V(t)$ and $E(t)$ denote the vertices and edges, respectively, of a tree t. By construction, $|V(t')| = 1 + \sum_{i \in J} |V(t_i)|$ and $|E(t')| = |J| + \sum_{i \in J} |E(t_i)| = |J| + \sum_{i \in J} (|V(t_i)| - 1) = \sum_{i \in J} |V(t_i)|$. Then $|V(t')| = |E(t')| + 1$, so, by the criterion above, t' is a tree.

We show that t' is rooted with root w. Consider $w' \in V(t') \setminus \{w\}$. Then $w' \in V(t_i)$ for some $i \in J$. There exists a directed path $w \to R(t_i)$, and a directed path $R(t_i) \to w'$ because t_i is a rooted tree. Then there is a directed path $w \to w'$, which is unique because t' has no cycles. Then t' is a tree rooted at w, so F' is a rooted forest.

Next, we prove the statement about reachable sets. Case (1) above is clear, since no edges are added. For case (2), let $w \in t$ for some tree in F. If there is a new path from $w \to w'$ such that $w' \notin H(w, F)$, it uses one of the added edges (w_1, w_2). Then, apparently, there

[9]See Gross and Yellen (2005) for a proof.

exists a path $w \to w_1$ in F'. But, as shown above, w_1 is a root in the new forest F', so there exists no directed edge of the form (x, w_1). This is a contradiction, so it must be that $H(w, F) = H(w, F')$.

The final statement concerning vertex depth is trivial.

□

Next, we give a simple observation, showing how problem forests can be used to track fixed point relations.

Lemma 3.8. *Let F be a problem forest for Q, with $\phi_i, \phi_j \in Q$. Let t be a rooted tree in F, and suppose that $r(w_i) = w_j$. Then $\mathcal{E}(\phi_i) = \mathcal{E}(\phi_j)$.*

Proof. By assumption, t is a rooted tree, so there is a unique directed path of the form $w_j = w^0 \to w^1 \to \cdots \to w^n = w_i$ from the root to w_i. Then, for each vertex in the path is related by $p(w^k) = w^{k-1}$ for $1 \leq k \leq n$. Then, since F is a problem forest by assumption, we get a chain of inclusions $\mathcal{E}(\phi_{k-1}) = \mathcal{E}(\phi_k)$ for $1 \leq k \leq n$, showing that, as claimed, $\mathcal{E}(\phi_i) = \mathcal{E}(\phi_j)$. □

The following lemma shows that, conditional on the problem forest structure of F, the collision digraph can be used to combine the information contained in distinct problem cones.

Proposition 3.9. *Suppose that the following conditions hold when InformationCombine is called:*

(i) *F is a problem forest*
(ii) *For any $\phi_i \in Q_0$ with $r(w_i) = w_j$, if $k \in I(i)$ then $\mathcal{E}(\phi_j) \subseteq \mathcal{E}(\phi_k)$*

Let $G = (V, E)$ be the collision digraph and S one of its strong components, then

(1) *If $(v_i, v_k) \in E$ then $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi_k)$.*
(2) *If $v_i, v_j \in S$ then $\mathcal{E}(\phi_i) = \mathcal{E}(\phi_j)$.*
(3) *If $v_j \in S$ and $v_k \in H$, the reachable set of S, then $\mathcal{E}(\phi_i) \preceq \phi_k$.*

In particular, item (2) shows that the strong components of G identify groups of equivalent problems, which can then be consolidated. Statement (3) shows that fast digraph algorithms for computing reachable sets can be used to combine the information learned in different subproblems.

Proof. First, we characterize the edges of $G = (V, E)$. Consider $\phi_i \in Q_0$ at initialization of the subroutine and $j \in I(i)$ with $w_k = r(w_j)$ and $w_\ell = r(w_i)$. Since F is a problem tree, by Lemma 3.8 above $\mathcal{E}(\phi_i) = \mathcal{E}(\phi_\ell)$ and $\mathcal{E}(\phi_j) = \mathcal{E}(\phi_k)$. By our assumption about collision indices, we have $\mathcal{E}(\phi_\ell) \subseteq \mathcal{E}(\phi_j)$. Putting this together, we have $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi_k)$ for every $(v_i, v_k) \in E$. This shows that (1) above holds.

Now consider a strong component S (after removing singletons $\{v_i\}$ for $\phi_i \notin Q_0$). For $v_i, v_k \in S$, there are directed paths $v_i \to v_k$ and $v_k \to v_i$. Then, by the correspondence between directed edges and fixed point inclusion shown above, we have $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi_k)$ and $\mathcal{E}(\phi_k) \subseteq \mathcal{E}(\phi_i)$, so $\mathcal{E}(\phi_i) = \mathcal{E}(\phi_k)$ for all $v_i, v_k \in S$, completing the proof of (2) above.

Finally, if $\phi_i \in S$ and $\phi_k \in H$, then there exists a path $v_i \to v_k$ by definition, so by (3) above there is a chain of inclusions $\mathcal{E}(\phi_i) \subseteq \ldots \subseteq \mathcal{E}(\phi_k)$. In particular, $\mathcal{E}(\phi_i) \preceq \phi_k$. This completes the proof of the lemma. □

Before proving our key result, we show an auxiliary lemma about the relationship between F and the problem sets returned from each subroutine

Lemma 3.10. *Any distinct problem $\phi \in (2^X)^A$, is processed (the Tarski sequence for $\langle\phi\rangle$ is evaluated) at most once in Subroutine 3.4. Moreover, the following statements are true:*

(1) *If F is a problem forest when Subroutine 3.4 is initialized, then the set Q_0 returned from the subroutine satisfies $\phi_i \neq \phi_j \in Q_0 \implies r(w_i) \neq r(w_j)$.*
(2) *If Q_0 is returned from Subroutine 3.5, then $w(\phi) \in R(F)$ for each $\phi \in Q_0$.*

Proof. The first statement is a simple observation. If ϕ_i is processed more than once, then ϕ_i is added to Q_A more than once during the algorithm . Since $Q_A \subseteq Q_0$, and elements are *only added* (never removed) to Q_0 after redundancy checks (i) and (ii), then $\phi_i \in Q_0$ when it is added to Q_A the second time. But we check for this when we process stopping condition (2) at (ii), so this is impossible. Then any distinct preallocation is processed at most once.

For the other lemma statements, we work by induction on ℓ, the total number of evaluations of Subroutine 3.4 and 3.5. For $\ell = 0$ the statements are trivially true. Then assume that (1) and (2) above hold up to $\ell = n \geq 0$. There are several cases. Throughout, let (Q_0', F') and (Q_0'', F'') denote the states of (Q_0, F) at initialization and return, respectively, of the subroutine that finishes at $\ell = n + 1$.

Case 1: InformationAcquire (Subroutine 3.4) returns at $\ell = n + 1$. Note that Subroutine 3.4 is called either directly after initialization or after evaluation of Subroutine 3.5. If Subroutine 3.4 directly follows initialization of FindStrictCore, clearly $w(\phi) \in R(F')$ for $\phi \in Q_0'$ by construction. Otherwise, this statement holds by the inductive hypothesis and (2) above. Note that each ϕ added to Q_0 during this subroutine is initialized as an isolated root. Since problem indices are unique, this shows that, at any time before ϕ is processed (the Tarski sequence for $\phi \in Q_A$ is evaluated), $w(\phi) \in R(F)$. In particular, $D(w_i, F) = 0$ when ϕ_i is added to the Q_0 during the subroutine.

Note first that F'' is obtained from F' by adding vertices and edges satisfying the conditions of Lemma 3.7. Then, by the lemma, F'' is also a rooted forest. Suppose that for $\phi_i \neq \phi_j \in Q_0''$, $r(w_i) = r(w_j) = w$ in F''. Then there are directed paths $w \to w_i$ and $w \to w_j$. The directed path $w \to w_i$ can be represented as a sequence of vertices $w = w^1, w^2, \dots, w^k = w_i$ and similarly for w_j. Clearly, there is at least one vertex in the path $w \to w_j$ not in $w \to w_i$. Let k' denote the minimal index of such a vertex in either path. By minimality, the paths are identical up to $w_{k'-1}$.

Subcase 1: One of the paths, without loss $w \to w_i$, ends at $w_{k'-1}$. Then there is a directed path $w_i \to w_j$. But $D(w_j, F) = 0$ when ϕ_j is added to Q_0 as above, so all edges on this path must have been added during the subroutine, after ϕ_i was processed, since in-edges are only added to problems already in Q_0. Immediately after ϕ_j is processed, we have so that $p(w_j) = r(w_j) = w(\phi') \equiv w_k$ for some index k. But indices assigned during the algorithm are unique, so no problem processed after ϕ_i can have index k. Then no edge of the form (w', w_k) can ever be added, so this is a contradiction.

Subcase 2: Both paths $w \to w_i$ and $w \to w_j$ continue beyond $w_{k'-1}$. Then $w' \equiv w_{k'-1}$ has out-degree ≥ 2. Only vertices of out-degree 1 are ever added to F during the subroutine, so we must have $w' \in F'$, the problem forest at the beginning of the subroutine. By Lemma 3.7 the reachable sets for w' have $H(w', F') = H(w', F'')$. Then, in particular, $w_i \in F'$ and $D(w_i, F') \geq 1$ By Lemma 3.7 again, $D(w_i, F) \geq D(w_i, F') \geq 1$ for any state of the forest F during the subroutine. Then ϕ_i is never in Q_0 during the algorithm, since we showed that, for any $\phi_i \in Q_0$, $w_i \in R(F)$ at some point during the algorithm. This completes the proof for this case.

Case 2: Subroutine 3.5 returns at $\ell = n+1$ after Subroutine 3.4 returns at $\ell = n$. We will show that $w(\phi) \in R(F'')$ for all ϕ in the set of returned problems Q_{temp}. Let $G = (V, E)$ be the collision digraph formed during this subroutine. By definition, the strong components $\{S_j\}_{j\in\mathcal{J}}$ of G retained after removing irrelevant singletons at step (2) are disjoint. Then if $S_j \neq S_{j'}$ with $v_k \in S_j$ and $v_{k'} \in S_{j'}$ we must have $r(w_k) \neq r(w_{k'})$ by the inductive hypothesis and (1) above, so $\{r(w_k) : v_k \in S_j\} \cap \{r(w_k) : v_k \in S_{j'}\} = \emptyset$. Define $R(j, F) = \{r(w_k) : v_k \in S_j\}$, roots of F generated by vertices in S_i. At any point during the algorithm, call an index j *processed* if $\phi_j \in Q_{temp}$.

As we add problems ϕ to Q_{temp} and vertices $w(\phi)$ to F, the problem forest F changes, so root sets may change. However, working by induction the number of processed problems m, we can show that $R(F, j) = R(F', j)$ for any unprocessed index j and forest state F during this subroutine. The base case $m = 0$ is trivial. Assume this is true up to $m = k$. Suppose we process index j at $m = k+1$ and let $\mathcal{J}_u$ be the set of unprocessed indices, where F_j is the state of the forest before processing index j. For $i \in \mathcal{J}_u$, $R(F_j, i) = R(F', i)$ for $i \in \mathcal{J}_u$ by induction. In particular, $R(F_j, i)$ are pairwise disjoint for $i \in \mathcal{J}_u$. Then adding a new vertex $w(\phi_j)$ as a root with $c(w(\phi_j)) = R(F', j)$ cannot change any of the other component root sets $R(F', \ell)$ for $\ell \in \mathcal{J}_u \setminus \{j\}$. If k is processed next after j, this shows $R(F_k, \ell) = R(F', \ell)$ for $\ell \in \mathcal{J}_u \setminus \{j\}$, which completes the induction.

In particular, for any $\phi \in Q_{temp}$, $w(\phi) \notin F'$ at the beginning of the subroutine, so $w(\phi) \notin R(F_j, j) = R(F', j)$ for any j. Then no edge of the form $(w', w(\phi))$ is ever added during this procedure. This shows that each added vertex $w(\phi)$ is *still a root* in F'', when the subroutine returns, so we have shown (2) above for this case.

Case 3: Subroutine 3.5 returns at $\ell = n+1$ after Subroutine 3.5 returns at $\ell = n$. By the inductive hypothesis for problems returned from Subroutine 3.5, all inputs $\phi \in Q_0'$ to Subroutine 3.5 at $\ell = n+1$ have $w(\phi) \in R(F')$. Then $R(F', j) = \{r(w_k) : v_k \in S_j\} = \{w_k : v_k \in S_j\}$. Then the root sets are disjoint in this case as well, by disjointness of strong components. The inductive argument from case 2 then establishes (2) above for this case as well. □

We are now ready for the proof of our key lemma on the structure of the graph F during the algorithm.

Proof of Lemma 3.6. A straightforward induction on distinct states $(F, Q)_k$ of the pair (F, Q) shows that condition (1) of the definition of a problem forest is always satisfied. Specifically, any line in which we add ϕ to Q is paired with a line where $w(\phi)$ is added to F.

For the remaining statements, we work by induction on the sequence $(F, I)_k$ of distinct states of the pair (F, I). For $k = 0$, note that we initialize F as the collection of isolated roots $w(\overline{\phi} - e_i)$ for $1 \leq i \leq m$, so clearly F is a forest. The other defining conditions are trivial. Also, we initialize $I = \emptyset$, so item 2 above is trivially satisfied. Then assume by induction that the statements above hold for $(F, I)_k$ up to $k = n \geq 0$. By an abuse of notation, we will denote components of $(F, I)_k$ as F_k and I_k when the meaning is clear. We consider separately each case where the pair (F, I) can change.

Case 1: There is a transition $(F, I)_n \to (F, I)_{n+1}$ in Subroutine 3.4 when a stopping condition is triggered for problem ϕ_i at $(T_i^2)^\ell \phi_i = \phi'$, and $w(\phi')$ is added to F at processing step (a). Let F' and Q_0' denote the state of F and Q_0 at the beginning of InformationAcquire. From Lemma 3.10, we know that (i) for any distinct preallocation ϕ, the problem $\langle\phi\rangle$ is processed (the Tarski sequence evaluated) at most once. Moreover, the lemma also shows

that (ii) any $\phi \in Q_0'$ has $w(\phi) \in R(F')$ and, from the proof of this lemma, (iii) any ϕ_i added to Q_0 during the subroutine has $w(\phi_i) \in R(F)$ before ϕ_i is processed. We will use these facts below.

By the Tarski argument from the main text, we have $\mathcal{E}(T_i) \preceq \max \mathcal{E}(T_i^2) \preceq (T_i^2)^j \phi_i$ for any j. In particular, $\mathcal{E}(T_i) \preceq (T_i^2)^\ell \phi_i = \phi'$. Combining this with Lemma 3.1 above, $\mathcal{E}(\phi_i) = \mathcal{E}(T) \cap \{\phi'' \preceq \phi_i\} \subseteq \mathcal{E}(T_i) \preceq \phi'$. By monotonicity of the Tarski sequence, we have $\phi' \preceq \phi_i$, so $\mathcal{E}(\phi') \subseteq \mathcal{E}(\phi_i)$. Then $\mathcal{E}(\phi_i) = \mathcal{E}(\phi')$. Since $w(\phi') = p(w_i)$, we have shown that $\mathcal{E}(\phi_i) = \mathcal{E}(p(\phi_i))$ for the only *new* parent-child relation added to F_n. We add $w(\phi')$ with $c(w(\phi')) = \{\phi_i\}$, so there are no new sibling relations. Then by the inductive hypothesis, F_{n+1} also satisfies condition (2) and (3) defining a a problem forest. By (iii) above, F_n has $w(\phi_i) = r(w(\phi_i))$, so the modification $F_n \to F_{n+1}$ satisfies the conditions of Lemma 3.7. Then, by the lemma, F_{n+1} is also a rooted forest. This completes the proof that F_{n+1} is a problem forest.

By the single-processing condition of Lemma 3.10, we must have $I(i) = \emptyset$. Fix any $\ell \in \mathcal{I}$ and suppose that $k \in I_{n+1}(\ell) = I_n(\ell)$. By the inductive hypothesis on $(F, I)_n$, if $r(w_\ell) = w_j$ in problem forest F_n, then $\mathcal{E}(\phi_j) \subseteq \mathcal{E}(\phi_k)$. If we still have $r(w_\ell) = w_j$ in the forest F_{n+1}, we are done by induction. If not, then it must be that $r(w_\ell) = w(\phi')$. But $\ell \neq i$, so w_ℓ is a descendant of w_i in F_n. Case (1) of the proof of Lemma 3.10 shows that such a descendant relation is impossible. Then condition (2) holds as well, and we are done.

Case 2: There is a transition $(F, I)_n \to (F, I)_{n+1}$ in Subroutine 3.4 during processing of ϕ_i, where stopping condition (3) is triggered and the collection $I(i)$ is modified. Then $F_n = F_{n+1}$, so we need only check item (2) of the inductive hypothesis above. $I_n(i) = \emptyset$ because $I = \emptyset$ at initialization of the subroutine and by the single-processing condition, and each preallocation is processed at most once by Lemma 3.10. For stopping condition (3), we evaluated the Tarski sequence for $\langle \phi_i \rangle$ and found that $\phi' \preceq \phi_j \in Q$ for some collection of indices j, which were added to $I_n(i)$. By condition (iii) of Lemma 3.10, $w(\phi_i) \in R(F_n)$. We add $w(\phi') = p(w_i)$, so $r(w_i) = w(\phi')$ in F_{n+1}. Since $\phi' \preceq \phi_j$, $\mathcal{E}(\phi') \subseteq \mathcal{E}(\phi_j)$ for any new collision index j added during the change $(F, I)_n \to (F, I)_{n+1}$.

If $\ell \neq i$, then we have $I_n(\ell) = I_{n+1}(\ell)$. In this case, condition (2) holds by the argument at the end of case (1). Then we have shown that condition (2) holds for the pair $(F, I)_{n+1}$, and we are done.

Case 3: There is a transition $(F, I)_n \to (F, I)_{n+1}$ in Subroutine 3.4 during processing of ϕ_i, when stopping condition (2) is triggered and $w(\phi_i - e_j)$ is added to F_n as an isolated root at item (iii). We denote $\phi_i - e_j = \phi_k$. That F_{n+1} is still a rooted forest is immediate from Lemma 3.7, since the new vertex is isolated. No edges are added, so no new parent or child relations are added. Then, by the inductive hypothesis, F_{n+1} remains a problem forest. For condition (2) above, fix ℓ and let $j \in I(\ell)$. If $\ell \neq k$, then, because no edge is added to the graph, $r(w_\ell)$ is unchanged, so the condition holds by the inductive hypothesis. If $\ell = k$, then $I(i) = \emptyset$, so the condition is trivially true. We have shown (1) and (2) hold for $(F, I)_{n+1}$, so we are done.

Case 4 Before constructing the collision digraph in Subroutine 3.5, there is a transition $(F, I)_n \to (F, I)_{n+1}$ when we add i to $I(i)$. Then $F_n = F_{n+1}$, so it suffices to check that condition (2) holds. Let $r(w_i) = w_k$, so, by induction, $\mathcal{E}(\phi_i) = \mathcal{E}(\phi_k)$. In particular, $\mathcal{E}(\phi_k) \subseteq \mathcal{E}(\phi_i)$ for the index $i \in I(i)$, so condition (2) holds automatically. Since this was the only modification to I_n, condition (2) follows for other indices by the inductive hypothesis.

Case 5: There is a transition $(F, I)_n \to (F, I)_{n+1}$ by adding a vertex $w(\phi)$ to F_n while processing the collision digraph $G = (V, E)$ during Subroutine 3.5 at item (b). Note that $I_n = I_{n+1}$. By construction, we have $c(w(\phi)) \subseteq R(F_n)$, so this vertex addition is of the form required in Lemma 3.7, which shows that F_{n+1} is still a rooted forest.

Next, we show that defining conditions (2) and (3) still hold for F_{n+1}, making use of our characterization of the collision digraph in Lemma 3.9 above. Let $v_k, v_\ell \in S_j$, a strong component of $G = (V, E)$ (after removing irrelevant singletons at item (2), and denote $r(w_k) = w_{k'}$ and $r(w_\ell) = w_{\ell'}$. By (strong) induction, F' is a problem forest, so by Lemma 3.8 above, $\mathcal{E}(\phi_{\ell'}) = \mathcal{E}(\phi_\ell)$ and $\mathcal{E}(\phi_k) = \mathcal{E}(\phi_{k'})$. Moreover, the indices I' at initialization satisfy condition (2), by the inductive hypothesis again, so Lemma 3.9 shows that, in fact, $\mathcal{E}(\phi_{\ell'}) = \mathcal{E}(\phi_\ell) = \mathcal{E}(\phi_k) = \mathcal{E}(\phi_{k'})$. Now $w(\phi)$ is added to F_n with $c(w(\phi)) = \{r(w_k) : v_k \in S_j\}$. Then the previous argument shows that $w_i, w_k \in c(w(\phi))$ implies that $\mathcal{E}(\phi_i) = \mathcal{E}(\phi_k)$. Since these are the only *new* sibling relations added in F_{n+1}, by the inductive hypothesis we have shown condition (2) defining a problem forest for F_{n+1}.

For the final defining condition of a problem forest, consider adding ϕ as above and let $v_i \in S_j$. Clearly $i \in H_j$, so that $i \in K_j$. Then $\phi \preceq \phi_i$, so $\mathcal{E}(\phi) \subseteq \mathcal{E}(\phi_i)$, and the second statement of condition (3) is satisfied. If $v_k \in H_j$, then there is a directed path $v_i \to v_k$, so $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi_k)$ by statement (3) of Lemma 3.9 above. In particular, $\mathcal{E}(\phi_i) \preceq \phi_k$ for all $k \in K_j$. Then $\mathcal{E}(\phi_i) \preceq \phi = \min_{k \in K_j} \phi_k$, so that $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi)$, and we are done. This is the only new parent-child relation added to F_n, so, using the inductive hypothesis, F_{n+1} satisfies defining condition (3) of a problem forest.

For condition (2) regarding collision indices, it suffices to consider w_ℓ with $r(w_\ell) = w(\phi)$, because otherwise the statement follows directly from the inductive hypothesis, since $w(\phi)$ is the only root added to F_n. Let $k' \in I(\ell)$ and suppose $r(w_\ell) = w(\phi)$. Let w_j the root of w_ℓ in F_n Then it must be that $p(w_j) = w(\phi)$; for instance, by Lemma 3.7, since otherwise the reachable set of some $w' \in F_n$ would have changed as $F_n \to F_{n+1}$. Then by Lemma 3.8, we have $\mathcal{E}(\phi) = \mathcal{E}(\phi_j) \subseteq \mathcal{E}(\phi_{k'})$, where we use the inductive hypothesis in the subset relation, so condition (2) still holds for $(F, I)_{n+1}$. This finishes the proof for this case.

Case 6: There is a transition $(F, I)_n \to (F, I)_{n+1}$ in Subroutine 3.4 when we add collision index j to $I_n(i)$ for some newly created problem $\phi_i \in Q_{temp}$ when we "check for new collisions". Again, $F_n = F_{n+1}$, so it suffices to check that condition (2) holds. Now, by assumption, $\phi_i \preceq \phi_j$, so $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi_j)$. Let $r(w_i) = w_k$. By the inductive hypothesis on $F_n = F_{n+1}$, we have $\mathcal{E}(\phi_i) = \mathcal{E}(\phi_k)$, so condition (2) holds immediately. As above, this was the only modification to I_n, so condition (2) follows for other indices of I_{n+1} by the inductive hypothesis.

We have exhausted all cases, so the proof is complete. □

First, we handle termination analysis.

Lemma 3.11. *FindStrictCore terminates in finite time*

Proof. For any set $E \subseteq (2^X)^A$, define $d(E) = \max_{\phi \in E} \|\phi\|_1$, where the set of preallocations $(2^X)^A$ is identified with a grid as above[10]. Similarly, let $D(E) = \sum_{\phi \in E} \|\phi\|_1$. Note that by finiteness, both $d(\cdot)$ and $D(\cdot)$ take finitely many values on subsets of $(2^X)^A$. First, we show that, for any problem collection Q_0, InformationAcquire (Subroutine 3.4) terminates in finitely many iterations.

[10]For $x \in \mathbb{R}^m$, $\|x\|_1 = \sum_{i=1}^m |x_i|_1$ is the standard 1-norm .

Consider the sequence Q_A^k formed by distinct states of Q_A during Subroutine 3.4. Note that if for any $k \geq 0$ we have $Q_A^{k+1} = Q_A^k \setminus \{\phi\}$ for some problem ϕ, then clearly $d(Q_A^{k+1}) \leq d(Q_A^k)$. By inspection, all changes to Q_A are of this form, except when we add new problems after triggering stopping condition (2). For this state change, note that $\|\phi\|_1 > \max_{i=1,\dots,m} \|\phi - e_i\|_1$, so, in fact, if $Q_A^{k+1} = Q_A^k \setminus \{\phi\} \cup (\bigcup_{i=1}^m \{\phi - e_i\})$ then $d(Q_A^{k+1}) \leq d(Q_A^k)$. Then $d(Q_A^k)$ is monotonically decreasing and ≥ 0. In particular, $d(Q_A^k)$ converges in finite time, by finiteness of $(2^X)^A$.

Note that Q_I, the set returned by the algorithm, has $\phi \in Q_I \implies \phi \in Q_A^\ell$ for some ℓ. Then $d(Q_I) = \max_{\phi \in Q_I} \|\phi\|_1 = \|\phi'\|_1 \leq d(Q_A^\ell) \leq d(Q_0)$ for some $\phi' \in Q_I \cap Q_0^\ell$. So we have also shown $d(Q_0) \geq d(Q_I)$.

The inequality $d(Q_A^{k+1}) \leq d(Q_A^k)$ is *strict* whenever $\phi'' = \operatorname{argmax}_{\phi' \in Q_A^k} \|\phi'\|_1$. For any k, consider Q_A^k at the top of the algorithm's main for loop. Then $\phi'' \in Q_A^k$, so, for at least one iteration of the loop, $d(Q_A^k)$ decreases strictly. Since $d(Q_A^k)$ converges in finite iterations, the subroutine's while loop runs for only finitely many iterations, so the subroutine terminates.

Since all inputs are finite, Subroutine 3.5 clearly terminates in finite time.

Next, we show that the information combination stage of FindStrictCore terminates. Consider distinct states of Q_0 returned by InformationCombine, which form a sequence Q_0^k, during the inner while loop of FindStrictCore (line 6). We will show that the sequence $D(Q_0^k)$ is monotonically decreasing.

Consider a single execution of InformationCombine (Subroutine 3.5) with input Q_0^k. Let $S \in \mathcal{G}$ be a strong component of the collision digraph $G = (V, E)$ left in $\{S_j\}_{j \in \mathcal{J}}$ after removing irrelevant singletons at item (2). By construction, a vertex $v_i \in V$ has out-degree 0 if $\phi_i \notin Q_0^k$. Then each vertex v_i *not* associated with a preallocation $\phi_i \in Q_0^k$ must generate a singleton strong component $\{v_i\}$, which is removed at item (2), so $\{v_i : v_i \in S_j; j \in \mathcal{J}\} \subseteq \{v_i : \phi_i \subseteq Q_0^k\}$. Therefore, by disjointness of $\{S_j\}_{j \in \mathcal{J}}$, we can define a surjection $p : Q_0^k \to Q_{temp}$ by setting $p(\phi_i) = \min_{k \in K_j} \phi_k \in Q_{temp}$. By construction, for any $\phi_j \in Q_{temp}$, we have $p^{-1}(\{\phi_j\}) \supseteq \{\phi_k : w_k \in c(w_j)\}$. Note that, since i is added to $I(i)$ during initialization, if $\phi_i \in S_j$ and $r(w_i) = w_\ell$, then $v_i \to v_\ell$; in particular, $v_\ell \in H_j$, the reachable set of strong component j. Lemma 3.6 shows that F is a problem forest at any time during the algorithm, so if $\phi_k = p(\phi_i)$, then $\phi_k \preceq \phi_i$. By iterating, we have $\phi_\ell \preceq \phi_i$ for the root also. Then apparently $p(\phi_i) = \min_{k \in K_j} \phi_k \preceq \phi_i$ for any such i.

Using the map p above, we can express Q_0^k as a disjoint union of the form $Q_0^k = \bigcup_{\phi \in Q_{temp}} p^{-1}(\phi)$. Since p is a surjection, we may calculate $d(Q_{temp}) = \max_{\phi_i \in Q_{temp}} \|\phi_i\|_1 \leq \max_{\phi_i \in Q_{temp}} d(p^{-1}(\phi_i)) = d(Q_0^k)$. Similarly, since $D(\cdot)$ is an additive set function, we can compute $D(Q_{temp}) = \sum_{\phi \in Q_{temp}} D(\{\phi\}) \leq \sum_{\phi \in Q_{temp}} D(p^{-1}(\phi)) = D(Q_0^k)$ by surjection and the discussion above. Since Q_{temp} is returned from the subroutine as Q_0^{k+1}, we have shown $d(Q_0^{k+1}) \leq d(Q_0^k)$ and $D(Q_0^{k+1}) \leq D(Q_0^k)$ for any k.

In particular, the sequence $D(Q_0^k)$ is monotonically decreasing (and bounded below by 0), so it converges at a finite value of k, again using finiteness of $(2^X)^A$. Suppose that for $k \geq 0$ with Q_0^k the input to the subroutine InformationCombine, we have $D(Q_0^k) = D(Q_0^{k+1})$. By the previous discussion, $D(p^{-1}(\phi_j)) \geq D(\{\phi_j\})$ for any $\phi_j \in Q_{temp}$, so each of these weak inequalities must actually be equality. Since, in addition, $p(\phi_i) \preceq \phi_i$ for $\phi_i \in Q_0^k$, each of these inequalities under the partial order $\preceq$ must also be an equality, with $p^{-1}(\phi_i) = \{\phi_i\}$ for any $\phi_i \in Q_{temp}$. Then $Q_0^k = Q_{temp} = Q_0^{k+1}$. This argument shows that the *sequence of sets* Q_0^k actually converges in finite time as well.

Suppose, then, that $Q_0^k = Q_0^{k+1}$ for some k and consider a single execution of InformationCombine. Above, we showed that $p^{-1}(\phi_i) = \phi_i$ for any $\phi_i \in Q_{temp}$. Then, as noted above, we have $\{i\} = \{j : \phi_j \in p^{-1}(\phi_i)\} \supseteq \{j : w_j \in c(w_i)\}$, so the expression defining a collision in stopping condition (3) must be false for all $\phi_i \in Q_{temp}$. Then the subroutine must return $I = \emptyset$.

This shows that the information combination stage of the algorithm terminates. That is, every sequence of executions of InformationCombine (Subroutine 3.5) is finite.

We have shown that the sequence $d(Q_0^k)$ is monotonically decreasing during the information combination stage (the second while loop in FindStrictCore) as well as during a single evaluation of InformationAcquire (Subroutine 3.4). Then the sequence $d(Q_0^k)$ formed by distinct states of Q_0 at *any time* during FindStrictCore is monotonically decreasing. We complete the termination proof by showing that $d(Q_0)$ decreases strictly at *least once* during a single iteration of the outer while loop of FindStrictCore. Consider evaluating Subroutine 3.4 with input Q_0.

Case 1: The collection Q_I returned by the subroutine is empty. Note that, by inspection, $Q_0 \neq \emptyset$. Then apparently $0 = d(Q_0^{k+1}) = d(Q_I) < d(Q_0^k)$. *Case 2*: Suppose $Q_I \neq \emptyset$ and consider $\phi_i \in Q_I$. Since ϕ_i was added to Q_I, there must exist $k \geq 0$ such that $(T_i^2)^k \phi_i \neq (T_i^2)^{k+1}\phi_i$ by the definition of stopping condition (2). Suppose that the sequence stops at $\phi' = (T_i^2)^\ell \phi_i$. In particular, by monotonicity of the Tarski sequence for $\langle \phi_i \rangle$, we must have $\phi' \prec \phi_i$ *strictly*. Note that, for all such $\phi_i \in Q_I$, we add $w(\phi') = p(w_i)$ (at item (a)), so that $r(p(w_i)) = r(w(\phi'))$. Then $r(w_i) = r(w(\phi'))$, which we denote by $r(w_i) = w_k$.

The algorithm returns $Q_0' \equiv Q_I$ which is an input for Subroutine 3.5. In Subroutine 3.5, we form strong components of the collision digraph $G = (V, E)$. Suppose that W_j is the strong component containing v_i. We add i to $I(i)$ before forming the edges of G, so $(v_i, v_k) \in E$. Then $v_k \in H_j$, the reachable set of strong component S_j. Therefore, $\phi'' = \min_{\ell \in K_j} \phi_\ell \preceq \phi_k \preceq \phi' \prec \phi_i$, where the second inequality follows by Lemma 3.6, which shows that F is a problem forest, and because $w(\phi')$ is a descendant of $w(\phi_k)$. Using the map $p : Q_0' \to Q_{temp}$ above, we have $p(\phi_i) = \phi''$. We calculate as before $d(Q_{temp}) = \max_{\phi_j \in Q_{temp}} \|\phi_j\|_1 < \max_{\phi_j \in Q_{temp}} d(p^{-1}(\phi_j)) = d(Q_0')$, where the inequality is strict by the work above, noting that $\phi_i \in p^{-1}(\{\phi''\})$. Since $Q_0^{k+1} \leftarrow Q_{temp}$ when this subroutine returns, we have shown that, in both cases above, the sequence $d(Q_0^k)$ decreases *strictly* during a single execution of the outer while loop in FindStrictCore.

Above, we argued that $d(Q_0^k)$ converges in finite time. Therefore, the outer while loop of FindStrictCore executes only finitely many times, so the algorithm terminates. □

We are now ready to prove correctness of FindStrictCore. We will make heavy use of the collision digraph and problem forest structure results proved above.

Theorem 3.12. *FindStrictCore returns with $\mathcal{E}_C = \mathcal{E}(T)$. In particular, the algorithm finds all core outcomes $\mathrm{C}(X, U) = \{Y : \phi_Y \in \mathcal{E}_C\}$.*

Proof. We will show that at any point during the execution of FindStrictCore, a fixed point $\phi \in \mathcal{E}(T)$ satisfies either (i) $\phi \in \mathcal{E}_C$, the fixed points already found by the algorithm, or (ii) there exists a $\phi' \in Q_0$ such that $\phi \preceq \phi'$. We work by induction on k, the number of distinct states Q_0^k of the set of active problems Q_0 during the main loop of FindStrictCore[11]. For any set of problems W, we denote $\mathcal{E}(W) \equiv \bigcup_{\phi \in W} \mathcal{E}(\phi)$.

[11]Note that $\mathcal{E}_C$ changes only if Q_0 changes, so it suffices to consider distinct states of Q_0

For the base case, note that if $\overline{\phi} \in \mathcal{E}(T)$ or $\underline{\phi} \in \mathcal{E}(T)$, then, by Lemma 3.1 above, $\mathcal{E}(T) = \{\overline{\phi}\}$ or $\{\underline{\phi}\}$, so the algorithm is correct. Suppose the algorithm continues, and we set $Q_0 = \bigcup_{i=1}^m \{\overline{\phi} - e_i\}$ (identifying $(2^X)^A$ with a grid as above). Since $\mathcal{E}(T) \preceq \overline{\phi} = \max \mathcal{E}(T^2)$ and $\overline{\phi} \notin \mathcal{E}(T)$, the inductive hypothesis is clearly satisfied. Then suppose by induction that (i) and (ii) above are satisfied up to $k = n$. We show that these statements still hold for $k = n + 1$.

Throughout, for a set E, we let E' denote the state of E at subroutine initialization, and E'' its state at algorithm return. There are two main cases, corresponding to each subroutine.

Case 1: There is a state change $Q_0^k \to Q_0^{k+1}$ and $\mathcal{E}_C^k \to \mathcal{E}_C^{k+1}$ when InformationAcquire (Subroutine 3.4) returns. InformationAcquire returns a set Q_I of "collided" problems, which will be combined during Subroutine 3.5. In the above notation, $Q_I'' = Q_0^{k+1}$ is the set of problems and $\mathcal{E}_{temp}''$ is the set of fixed points returned by the subroutine. By the inductive hypothesis, we have $\mathcal{E}(T) \subseteq \mathcal{E}_C' \cup \mathcal{E}(Q_0')$. Then, to show that no fixed points are "lost" during the subroutine, it suffices to show that $\mathcal{E}(Q_0') \subseteq \mathcal{E}_{temp}'' \cup \mathcal{E}(Q_I'')$.

Formally, we work by induction on $\ell \geq 0$ (*within* this subroutine), for distinct states of the tuple

$$(Q_A, Q_0, Q_I, \mathcal{E}_{temp})_\ell$$

Note that, by inspection of the algorithm, the last three sets in the tuple change *only if* Q_A changes. Therefore, it suffices to consider distinct states[12] of the working set of active problems Q_A. We will show by induction that for any $\ell \geq 0$ if $\phi'' \in \mathcal{E}(\phi)$ for some problem $\phi \in Q_0^\ell$ then either $\phi'' \in \mathcal{E}_{temp}^\ell$ or there exists $\phi' \in Q_A^\ell \cup Q_I^\ell$ with $\phi'' \preceq \phi' \preceq \phi$ (GAR). We call such a relationship a guarantee (GAR) on ϕ'' *afforded* by the problem $\phi' \in Q_A^\ell \cup Q_I^\ell$, which is either still active, or in the set of "collided" problems to be returned by the algorithm. In particular, this will show that $\mathcal{E}(Q_0^\ell) \subseteq \mathcal{E}(Q_A^\ell) \cup \mathcal{E}(Q_I^\ell) \cup \mathcal{E}_{temp}^\ell$ for all $\ell \geq 0$. By Lemma 3.11, this subroutine terminates with $Q_A'' = \emptyset$. Also, as we will show shortly, $\mathcal{E}(Q_0') \subseteq \mathcal{E}(Q_0'')$, where Q_0'' is the state of Q_0 when the algorithm returns. Then, assuming the inductive result above, we would have $\mathcal{E}(Q_0') \subseteq \mathcal{E}(Q_0'') \subseteq \mathcal{E}(Q_A'') \cup \mathcal{E}(Q_I'') \cup \mathcal{E}_{temp}'' = \mathcal{E}(Q_I'') \cup \mathcal{E}_{temp}''$, so that no fixed points are "lost" during this subroutine.

First, we show that $\mathcal{E}(Q_0') \subseteq \mathcal{E}(Q_0'')$. Consider the beginning of Subroutine 3.4, where we remove redundant subproblems (items (i) and (ii)). If $\phi \in Q_0'$ and $\phi \preceq \min \mathcal{E}(T^2)$, then $\mathcal{E}(\phi) = \emptyset$ since we checked $\underline{\phi} = \min \mathcal{E}(T^2) \notin \mathcal{E}(T)$. Otherwise, if we remove ϕ from Q_0 during this loop, then there is a $\phi \preceq \phi' \in Q_0$, so that $\mathcal{E}(\phi) \subseteq \mathcal{E}(\phi')$, and no fixed points are lost. Let Q_0^R denote the state of Q_0 after this removal step. Then $\mathcal{E}(Q_0^R) = \mathcal{E}(Q_0')$. After this step, we only add, never remove, points from Q_0, so we have $\mathcal{E}(Q_0') \subseteq \mathcal{E}(Q_0^R) \subseteq \mathcal{E}(Q_0'')$, as required.

We now begin the induction. The base case is clear, since we set $Q_A' \leftarrow Q_0'$ at initialization. Then suppose by induction that our hypothesis holds up to $\ell = t$. We will show that guarantees of the form (GAR) above still exist for all $\phi \in \mathcal{E}(Q_0^{t+1})$. By the argument above, it suffices to consider changes to Q_A after the redundant problem removal step, since no fixed points guarantees are lost during this step. There are several cases.

Subcase 1: There is a state change $Q_A^t \to Q_A^{t+1}$ when stopping condition (1) or (2) is triggered for a problem $\phi_i \in Q_A^t$, which is then removed $Q_A^{t+1} \leftarrow Q_A^t \setminus \{\phi_i\}$. Let $\phi_0 \in Q_0^t$ and $\phi \in \mathcal{E}(\phi_0)$ and suppose that $\phi \notin \mathcal{E}_{temp}^{t+1}$, the set of fixed points found by $\ell = t + 1$. We

[12]Note that by "distinct states" we mean distinct states that occur at outside of the algorithm's stopping condition processing code. Thus, we are allowed to remove a fixed point from $Q_A \cup Q_I$ and add it back later while processing a single problem ϕ_i without creating a "distinct" state.

show that the guarantee (GAR) on ϕ cannot be afforded by ϕ_i, so it is safe to remove ϕ_i from $Q_A^t \cup Q_I^t$. Suppose that $\phi \preceq \phi_i$. We have $\phi' = (T_i^2)^{m'}\phi_i$ for some $m' \geq 0$, and $\phi' \preceq \underline{\phi}$ or $T\phi' = \phi'$. By the Tarski argument discussed earlier in the text, $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi')$, so, in fact, $\phi \preceq \phi'$. But $\phi \neq \phi'$, since we check if $\phi' \in \mathcal{E}(T)$ when processing ϕ_i, and we assumed $\phi' \in \mathcal{E}_{temp}^{t+1}$. Then $\phi \prec \phi'$ strictly. For stopping condition (1), note that $\mathcal{E}(T) \subseteq \mathcal{E}(T^2) \succeq \underline{\phi}$. Combining these, $\phi \prec \phi' \preceq \underline{\phi}$, which is a contradiction. For stopping condition (2), note that by Lemma 2.8, which shows that no fixed points of T are ordered strictly by "$\prec$", if $T\phi' = \phi'$ then there can be no $\phi \prec \phi'$ fixed by T, so this also results in a contradiction. By the inductive hypothesis, there must be $\phi'' \neq \phi_i$ such that $\phi'' \in Q_A^t \cup Q_I^t$ and affords a guarantee $\phi \preceq \phi'' \preceq \phi_0$. Then $\phi'' \in Q_A^{t+1} \cup Q_I^{t+1}$ since it is not removed, so we are done.

Subcase 2: There is a state change $Q_A^t \to Q_A^{t+1}$ when stopping condition (3) is triggered for a problem $\phi_i \in Q_A^t$. Again, we have ϕ' the lowest element of the Tarski sequence, and, as above, $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi')$. We then remove ϕ_i and add $\phi' - e_i$ for $1 \leq i \leq m$, so $Q_A^{t+1} = Q_A^t \setminus \{\phi_i\} \cup \left(\bigcup_{i\in I}\{\phi' - e_i\}\right)$, where $I \subseteq \{1, \ldots, m\}$ are the indices such that $\phi' - e_i$ is non-redundant (as in item (ii)). Let $\phi \in \mathcal{E}(\phi_0)$ for $\phi_0 \in Q_0$, and suppose that $\phi \notin \mathcal{E}_{temp}^{t+1}$, so we need to show that there exists some guarantee for ϕ. Suppose $\phi \preceq \phi_i \preceq \phi_0$ as in (GAR) above, so that $\phi \preceq \phi'$ by the Tarski argument. In fact, $\phi \prec \phi'$, because we check if $\phi' \in \mathcal{E}(T)$, as in the previous case. Then we must have, $\phi \preceq \phi' - e_i$ for some i. If $i \in I$ then $\phi \preceq \phi' - e_i \prec \phi' \preceq \phi_0$, so $\phi' - e_i \in Q_A^{t+1}$ gives a new guarantee (GAR) on this fixed point for $\phi_0 \in Q_0$. No other points of $Q_A^t \cup Q_I^t$ are removed, so we are done. Suppose, conversely, that $i \notin I$; that is, the problem $\phi' - e_i$ is redundant. Then $\phi - e_i = \phi'' \in Q^t$. Note that $Q^t \subseteq Q_0^t \cup Q'$, and suppose that $\phi'' \in Q_0^t$. So we actually have $\phi \in \mathcal{E}(\phi'') \cap \mathcal{E}(\phi_0)$. By working with the guarantee (GAR) for $\phi'' \in Q_0^t$ (rather than the guarantee for $\phi_0 \in Q_0^t$), by the inductive hypothesis for $\ell = t$ there exists $\phi_g \in Q_A^t \cup Q_I^t$ such that $\phi \preceq \phi_g \preceq \phi'' = \phi' - e_i \prec \phi' \preceq \phi_i$. Then $\phi_g \neq \phi_i$, so, in particular, it is not removed at step $t+1$, and we have $\phi_g \in Q_A^{t+1} \cup Q_I^{t+1}$. Then, in either case, we still have a guarantee on all fixed points.

Subcase 3: There is a state change $Q_A^t \to Q_A^{t+1}$ when stopping condition (4) is triggered for a problem $\phi_i \in Q_A^t$. Note that in this case we just move ϕ_i between sets. Specifically, $Q_A^{t+1} = Q_A^t \setminus \{\phi_i\}$ and $Q_I^{t+1} = Q_I^t \cup \{\phi_i\}$. Then $Q_A^t \cup Q_I^t = Q_A^{t+1} \cup Q_I^{t+1}$, so none of the guarantees of the form (GAR) above can change, which finishes the subcases.

By the remarks at the beginning and since the algorithm terminates by Lemma 3.11, we have shown by induction that $\mathcal{E}(Q_0') \subseteq \mathcal{E}(Q_I'') \cup \mathcal{E}_{temp}''$, so we are done. Returning to the first induction, we have shown that in the case where $Q_0^k \to Q_0^{k+1}$ during InformationAcquire (Subroutine 3.4), we have, using the inductive hypothesis, that $\mathcal{E}(T) \subseteq \mathcal{E}(Q_0^k) \cup \mathcal{E}_C^k \subseteq \mathcal{E}(Q_0^{k+1}) \cup \mathcal{E}_C^k \cup \mathcal{E}_{temp}'' = \mathcal{E}(Q_0^{k+1}) \cup \mathcal{E}_C^{k+1}$.

Case 2: There is a state change $Q_0^k \to Q_0^{k+1}$ when InformationCombine (Subroutine 3.5) returns. In the notation above, Q_0' is the input to the subroutine, and $Q_{temp}'' = Q_0^{k+1}$ is the set returned by the subroutine. We will argue that, for any $\phi \in Q_0'$, there is a $\phi' \in Q_{temp}''$ such that $\mathcal{E}(\phi) \subseteq \mathcal{E}(\phi')$. It will follow immediately that $\mathcal{E}(Q_0^k) = \mathcal{E}(Q_0') \subseteq \mathcal{E}(Q_{temp}'') = \mathcal{E}(Q_0^{k+1})$. The proof relies crucially on the collision digraph and problem forest structure results developed above.

Fix $\phi_i \in Q_0'$. By construction, we have $v_i \in G$, where $G = (V, E)$ is the collision digraph built during the algorithm. Then $v_i \in S_j$, a strong component of G left over after trimming irrelevant singletons (at item (2)). By our key Lemma 3.7 on the structure of the rooted

forest F during the algorithm, F is a problem forest, as in Definition 3.2, at any time during the execution of FindStrictCore. Moreover, condition (2) of Lemma 3.7 also holds, showing that collision indices are compatible with F. Then Proposition 3.9, our result showing that fixed point relationships are reflected in the structure of the collisiion digraph, can be applied. Let H_j denote the reachable set of strong component S_j. By item (3) of the graph structure proposition, if $\phi_k \in H_j$, then $\mathcal{E}(\phi_i) \preceq \phi_k$. Now we have $\mathcal{E}(\phi_i) \preceq \min_{v_k \in S_j} \phi_k \equiv \phi'$, which is added to Q_{temp}. In particular, $\mathcal{E}(\phi_i) \subseteq \mathcal{E}(\phi')$, where $\phi' \in Q''_{temp} = Q_0^{k+1}$. This shows that $\mathcal{E}(Q_0^k) \subseteq \mathcal{E}(Q_0^{k+1})$, so, in particular, $\mathcal{E}(Q_0^k) \cup \mathcal{E}_C^k \subseteq \mathcal{E}(Q_0^{k+1}) \cup \mathcal{E}_C^{k+1}$ since $\mathcal{E}_C^k = \mathcal{E}_C^{k+1}$.

We have exhausted both cases, so this completes the induction. By our termination analysis in Lemma 3.11, $Q_0 = \emptyset$ when the algorithm returns, so we have shown that $\mathcal{E}(T) \subseteq \mathcal{E}(Q_0) \cup \mathcal{E}_C = \mathcal{E}_C$. Moreover, we only add ϕ to $\mathcal{E}_C$ after checking that $T\phi = \phi$, so clearly $\mathcal{E}_C \subseteq \mathcal{E}(T)$. Then $\mathcal{E}_C = \mathcal{E}(T)$. That is, by Lemma 2.3 and Lemma 2.4 above, for any $\phi \in \mathcal{E}_C$, we have $\phi = \phi_Y$ for some $Y \in \mathrm{C}(X, U)$, and this exhausts all core allocations. □

3.5.1. *Discussion.* Note that, given $\phi = \phi_Y \in \mathcal{E}_C$, we have $Y = \bigcup_{a \in A} Y_a = \bigcup_{a \in A} \phi(a)$, so the allocation corresponding to ϕ can be found by taking a union. The performance of the algorithm for a given problem instance depends on the exact structure of the lattice $\mathcal{E}(T^2)$. For instance, it could be that by stopping a Tarski sequence $(T^2)^k\phi$ before it converges and running InformationCombine, we miss some ϕ' in the sequence for which $T^2\phi' \prec \underline{\phi}$, showing that $\mathcal{E}(\phi) = \emptyset$. In particular, FindStrictCore and the original Echenique and Yenmez (2013) algorithm are incommensurable. Our algorithm uses the heuristic that evaluation of the T operator is very costly, and we prefer to make progress through the lattice by using fast graph algorithms to share information between problems whenever this is possible.

4. Maximal Domain Results and Generalizations

4.1. **Weak Core Allocations.** It is tempting to try to extend the fixed point construction $\mathrm{C}(X, U) = \mathrm{E}(T)$ and associated algorithm on the lattice $(2^X)^A$ to the full class of weak core allocations $\mathrm{WC}(X, U) \supseteq \mathrm{C}(X, U)$ in the general matching with contracts setting considered above. Indeed, for many classical matching models[13] $\mathrm{C}(X, U) = \mathrm{WC}(X, U)$, so the construction extends directly.

At first glance, it appears that such a generalization will be forthcoming. Similarly to the core case, for $a \in A$ we define a set

$$W(\phi, a) = \{Z \in 2^{X_a} : \exists Y \in 2^X \, s.t. \, Y_a = Z, \, Y_b \succ_b \phi(b) \, \forall b \in d(Y) \setminus \{a\}\} \cup \{\phi(a)\}$$

which can be thought of as the collection of allocations $Z \in 2^{X_a}$ with $Y \supseteq Z$ that agent a could successfully propose to the coalition $d(Y)$. Define a new operator $T_2 : (2^X)^A \to (2^X)^A$ by $T_2(\phi)(a) = \max_{\succeq_a} W(\phi, a)$. Note that in this case, the inequalities in the definition of $W(\phi, a)$ are *strict*, and we are also forced to include $\phi(a) \in W(\phi, a)$, which is not necessary for core. Define the fixed allocation set $\mathrm{E}(T_2) = \{Y \in 2^X : T_2(\phi_Y) = \phi_Y\}$ as above. With these modifications, we get a similar characterization of $\mathrm{WC}(X, U)$ as the (coordinated) fixed points of an operator defined on the lattice $(2^X)^A$.

Lemma 4.1. $\mathrm{WC}(X, U) = \mathrm{E}(T_2)$

[13]This is the case, for instance, in the marriage problem. See Blair (1988) and Echenique and Oviedo (2006) for more on solution concept equivalences in a classical setting.

Proof. Suppose $Y \notin \mathrm{E}(T_2)$. Denoting $\phi = \phi_Y$, we have $T_2(\phi) \neq \phi$. Then there exists $a \in A$ with $T_2(\phi)(a) \neq \phi(a)$. In particular, $T_2(\phi)(a) = Z_a$ for some Z satisfying $Z_b \succ_b \phi(b) = Y_b$ for $b \in d(Z) \setminus \{a\}$ Moreover, $\phi(a) \in W(\phi, a)$, so we have $Z_a \succ_a \phi(a)$. Then, in fact, $Z_b \succ_b Y_b$ for all $b \in d(Z)$, so Z blocks Y and $Y \notin \mathrm{WC}(X, U)$.

Suppose $Y \notin \mathrm{WC}(X, U)$. Let Z be a blocking set and denote $\phi = \phi_Y$ as above. Then $Z_b \succ_b Y_b$ for $b \in d(Z)$ by definition. $d(Z) \neq \emptyset$ by definition, so fix $a \in d(Z)$. Then apparently $Z_a \in W(\phi, a)$ and, moreover, $Z_a \succ_a Y_a = \phi(a)$. In particular, $T_2(\phi)(a) \succ_a \phi(a)$, so that $\phi \notin \mathrm{E}(T_2)$. This completes the proof. □

Remark: Note that, by definition of the set $W(\phi, a)$, we must have $T_2\phi \succeq \phi$ for *any* $\phi \in (2^X)^A$. For the original operator T with $\mathrm{E}(T) = \mathrm{C}(X, U)$, this relationship only holds when $\phi \in \mathrm{CP}$; that is, $\phi = \phi_Y$ for some allocation $Y \subseteq X$. Because of this, T_2 may also fix *uncoordinated* preallocations, as we demonstrate shortly. Even more problematically, this fact shows that the monotonically decreasing Tarski sequence $(T^2)^k\phi$, which is the centerpiece of this approach, *can never actually decrease*.

The following example shows that the structure results for the operator T, which the fixed point approach crucially relies on, may fail for the operator T_2 even for the the setting of two-sided matching with bilateral contracts.

Proposition 4.2. *The operator T_2 may not be antitone. The squared operator T_2^2 may not be isotone. T_2 may fix uncoordinated preallocations; that is, $\mathcal{E}(T_2) \not\subseteq \mathrm{CP}$.*

Proof. Consider an embryonic trading network with two sellers and a buyer, so that $A = \{s_1, s_2, b\}$. We let $(i, j)^k$ denote the contract where i sells k units to j. Suppose that agents have preferences as follows:

$$\begin{aligned} s1 :& \{(s_1, b)\}, \emptyset_{s_1} \\ s2 :& \{(s_2, b)^2\}, \{(s_2, b)\}, \emptyset_{s_2} \\ b :& \{(s_1, b), (s_2, b)\}, \{(s_2, b)^2\}, \{(s_2, b)\}, \{(s_1, b)\}, \emptyset_b \end{aligned}$$

Thus, b prefers would like to execute two trades and prefers to buy from s_2, but, conditional on availability of a contract with both sellers, prefers to be diversified. The contract space is $X = \{(s_1, b), (s_2, b)^2, (s_2, b), \emptyset_{s_1}, \emptyset_{s_2}, \emptyset_b\}$, and we assume that all elements of 2^{X_a} not listed in the relations above are ranked below[14] the single-agent contract $\emptyset_a$ for $a \in A$.

Now define preallocations ϕ and ϕ' by setting $\phi'(a) = \emptyset_a$ for all $a \in A$ and $\phi(b) = \{(s_2, b)^2\}$, $\phi(s_1) = \emptyset_{s_1}$, $\phi(s_2) = \{(s_2, b)^2\}$. Note that $\phi = \phi_Y$, where $Y \in \mathrm{WC}(X, U)$ is the weak core allocation consisting of a single contract $\{(s_2, b)^2\}$. Then by Lemma 4.1 above $T_2(\phi) = \phi$. Since $\phi'(a) = \emptyset_a$ for all a, any trade listed above is a strict improvement for each agent involved, so every agent can propose its most preferred network structure. Thus, we get $\phi'' = T_2(\phi')$ defined by $\phi''(b) = \{(s_1, b), (s_2, b)\}$ and $\phi''(s_1) = \{(s_1, b)\}$ and $\phi''(s_2) = \{(s_2, b)^2\}$. ϕ'' is the unanimously most preferred preallocation, so, by the remarks above, $T_2(\phi'') = \phi''$.

Note that $\phi'' \succ \phi$ and $T_2(\phi'') \succ T_2(\phi)$, so T_2 is not antitone on the lattice $(2^X)^A$. We also have $\phi \succ \phi'$ but $T_2^2(\phi') \succ T_2^2(\phi)$, so T_2^2 is not isotone. Finally, $\{(s_2, b)\} \in \phi''(b)$, but $\{(s_2, b)\} \notin \phi''(s_2)$, so ϕ'' is not coordinated. However, we have shown that $T_2(\phi'') = \phi''$, so $\mathcal{E}(T_2) \not\subseteq \mathrm{CP}$. □

[14]Note that the specific order of allocations ranked below $\emptyset_a$ is irrelevant with respect to the evaluation of T_2, since trivially $\emptyset_a \in W(\phi, a)$ for any a.

4.1.1. *Discussion.* These results, and the previous remark, show that the fixed point approach to finding all weak core allocations is doomed to fail on the lattice $(2^X)^A$. Similar negative results can be shown, for instance, for an operator $T_3 : (2^X)^A \to (2^X)^A$ associated with the set of all stable matchings $\mathrm{S}(X, U)$. Our results do not prove, however, that there is no fixed point method for finding all core allocations $\mathrm{WC}(X, U)$ on *any lattice*. Indeed, our construction of an algorithm to find all stable matchings relies on adapting the approach for $\mathrm{C}(X, U)$ above to a lattice and partial order more appropriate for this solution concept.

4.2. **Generalized Information Sharing and Applications.** In this section, we briefly describe how the notion of information sharing and the collision digraph algorithm developed above can be extended to similar economic problems on more general complete lattices. This extension is the basis of our algorithm for finding all stable allocations. We also show how information sharing may be used to find Nash equilibria in games with strategic complementarities (GSC), extending Echenique (2007).

Let $(L, \preceq)$ be a complete lattice, and suppose $P \subseteq L$ is a set of points that we are would like to find. For instance, these could be stable allocations $\mathrm{S}(X, U)$ in the lattice $(2^X, \subseteq)$, where subsets are partially ordered by set inclusion. Let $F : L \to L$ be isotonic with $P \subseteq \mathcal{E}(F)$; that is, the fixed points of F *bound* P. Note that for any $x_i \in L$, where i is some index, the cone $\{x' \in L : x' \preceq x_i\} \equiv L_i \subseteq L$ is also a complete lattice. Suppose that an isotone operator $F_i : L_i \to L_i$ can be defined for each such sublattice.

4.2.1. *Fixed Point Guarantees.* To complete the construction, we need to assume some economic structure. Specifically, suppose that we can show $\mathcal{E}(F)$ satisfies $\mathcal{E}(F) \cap L_i \subseteq \mathcal{E}(F^i)$ (E). Guarantees of this form are surprisingly common in economics. For the problem of finding $\mathcal{E}(T) \subseteq (2^X)^A$, this guarantee is given by Lemma 3.1, which holds because truncating agents' preference profiles shrinks the set of available blocking allocations. The corresponding result in Echenique (2007) for Nash equilibria is similarly derived from the invariance of Nash equilibria under truncation of strategy profiles. For finding stable allocations, the guarantee is given by the invariance of substitutability under coarsening of the contract language X.

The following discussion shows how to extend our information sharing algorithm for finding $\mathcal{E}(T) \subseteq (2^X)^A$ to an arbitrary complete lattice, conditional on an economic guarantee of the form above.

4.2.2. *Example - Information Sharing and Collision Chains.* In this example, we show how to extend the notion of combining subproblem information to general complete lattices. Suppose that we currently have n active problems on lattices $L_i = \{x' \in L : x' \preceq x_i\}$. Denote $\mathcal{E}(F) \cap L_i \equiv \mathcal{E}(x_i)$ and $P_i = P \cap L_i$ Forming the Tarski sequence for F^i on the sublattice L_i, we find $y_i = \max \mathcal{E}(F^i)$, obtaining a guarantee of the form $P_i \subseteq \mathcal{E}(F) \cap L_i \subseteq \mathcal{E}(F^i) \preceq y_i$. Suppose then that there are collisions $y_i \preceq x_{i+1}$ for $1 \leq i \leq n-1$. Then, for instance, $P_1 \preceq y_1$, so $P_1 \subseteq P_2$. In fact, we can conclude that $P_1 \preceq y_2 \preceq x_3$, so $P_1 \preceq y_3$. Continuing in this way, we get a guarantee $P_1 \preceq y_i$ for $1 \leq i \leq n$. Since L is a complete lattice, we find that $P_1 \preceq \bigwedge_{i=1}^n y_i$, whereas without information sharing we only knew that $P_1 \preceq y_1$. Note that here $\bigwedge$ denotes the meet[15], or greatest lower bound, guaranteed to exist for any subset of a complete lattice. The full collision digraph construction for sharing information extends

[15]For the lattice $(2^X, \subseteq)$, for instance, and subsets $A_i \subseteq X$, we have $\bigwedge_{i=1}^n A_i = \bigcap_{i=1}^n A_i$, the standard intersection.

to an arbitrary complete lattice by simply replacing the pointwise minimum $\min_{k\in K}\phi_k$ over a reachable component K with the appropriate lattice meet.

To fully extend the algorithm, we discuss the intialization of new subproblems when, for instance, a Tarski sequence for the subproblem on L_i hits a fixed point of F^i. For $(2^X)^A$, we noted that $\{\phi' \prec \phi\} \subseteq \bigcup_{i=1}^m \{\phi' \preceq \phi - e_i\}$. In general, the lattice may not have such a regular structure. For instance, for the $(2^X, \subseteq)$, for $X_i \subseteq X$, we can form subproblems by noting that $\{Y \in 2^X : Y \subsetneq X_i\} \subseteq \bigcup_{x\in X_i}\{Y \in 2^X : Y \subseteq X \setminus \{x\}\}$. For more general problems, we work with whatever regularity is offered by the lattice.

4.2.3. *Games with Strategic Complementarities.* We briefly show how the above discussion can be used to incorporate information sharing in the Echenique (2007) algorithm for finding all pure strategy Nash equilibria in GSC. In GSC, we suppose a set of players $1 \leq i \leq n$ with strategy profiles S_i, each of which is a complete lattice. Then $S = \bigtimes_{i=1}^n S_i$ is also a complete lattice under the product order. Let $\beta_i : S \to 2^{S_i}$ denote the best response correspondence for player i. Then we can form a map $F : S \to S$ by setting $F_i = \sup \beta_i(s) \in S_i$. Lemma 2 of Echenique (2007) shows that this map is isotone on S. For a sublattice $S_r = \bigtimes_{i=1}^n \{s' \in S_i : s' \preceq s_i^r\}$, we can define a restricted best response correspondence β^r using only strategies in S_r, which extends to an operator $F_r : S_r \to S_r$ as above. Lemmas 3 and 4 of Echenique (2007) show that the guarantee $\mathcal{E}(F)\cap S_r \subseteq \mathcal{E}(F_r)$ of the form (E) above holds for this family of operators F_r on sublattices $S_r \subseteq S$, showing that the information sharing algorithm above may be applied.

5. Finding All Stable Matchings

In this section, we extend the information sharing technique developed above to an algorithm for finding all stable allocations in the setting of two-sided many-to-many matching with contracts under the assumption of substitutable preferences.

5.1. **Model and Notation.** In this section, we restrict our attention to two-sided many-to-many matching with contracts, and we may think of the contract language as $X = D\times H\times E$, where E is a finite set of contract terms. For instance, E could be a set of wages.

We define choice functions in the usual way for $Y \in 2^{X_a}$ as

$$C_a(Y) = \operatorname*{argmax}_{Z\subseteq Y} U_a(Z)$$

and, with abuse of notation, extend these functions to $Y \in 2^X$ by setting $C_a(Y) \equiv C_a(Y_a)$.

An allocation Y is said to be *stable* if it is

(1) *individually rational* - $C_a(Y) = Y_a$ for all $a \in A$
(2) *unblocked* - There is no allocation $Z \neq \emptyset$ such that $Z_b \subseteq C_b(Y \cup Z)$ for all $b \in d(Z)$.

We let $S(X, U)$ denote the set of all stable matchings.

We say that an agent $a \in A$ has *substitutable preferences* if for any $Y \subseteq Y' \subseteq X$ and $z \in X$ we have $z \in C_a(Y' \cup \{z\}) \implies z \in C_a(Y \cup \{z\})$. Similarly, a choice function C_a is said to satisfy *irrelevance of rejected contracts* if for any $Y \subseteq X$ and $z \notin Y$, if $z \in C_a(Y \cup \{z\})$ then $C_a(Y \cup \{z\}) = C_a(Y)$. In this setting, for a set $Y \subseteq X$, we let $C_D(Y) = \bigcup_{d\in D} C_d(Y)$ and similarly let $C_H(Y) = \bigcup_{h\in H} C_h(Y)$ denote the set of contracts chosen out of Y by the hospitals.

5.1.1. *Fixed Point Construction.* Consider $L = 2^X \times 2^X$, pairs of subsets of X, where we define a partial order by $(Z, W) \preceq (Z', W')$ if and only if $Z \subseteq Z'$ and $W' \subseteq W$. It is easy to check that this partial order defines a complete lattice. For instance, given a collection of $(Z_i, W_i) \in L$, the greatest lower bound is $\bigwedge_i (Z_i, W_i) = (\bigcap_i Z_i, \bigcup_i W_i) \in 2^X \times 2^X$. The least upper bound is attained similarly by swapping unions and intersections.

As in Hatfield and Kominers (2015), we may define an operator $\Phi : L \to L$ by $\Phi(Z, W) = (\Phi_D(W), \Phi_H(Z))$, where $\Phi^D(Z) = \{z \in X : z \in C_D(\{z\} \cup Z)\}$ and $\Phi^H(W) = \{z \in X : z \in C_H(\{z\} \cup W)\}$. As usual, we define the fixed point set $\mathcal{E}(\Phi) \equiv \{(Z, W) \in 2^X \times 2^X : \Phi(Z, W) = (Z, W)\}$. It is easy to see from the definition of substitutability that this operator is isotone on L. Therefore, by Tarski's Theorem, $\mathcal{E}(\Phi)$ is a non-empty lattice.

The following characterization of stable matchings is due to Hatfield and Kominers (2015)

Lemma 5.1 (Hatfield and Kominers (2015), Lemma 1)**.** *Assume that agents' preferences are substitutable and satisfy irrelevance of rejected contracts. Then for any $(Z, W) \in \mathcal{E}(\Phi)$, we have $Z \cap W \in \mathrm{S}(X, U)$. Conversely, for any $Y \in \mathrm{S}(X, U)$, there exists a unique $(Z, W) \in \mathcal{E}(\Phi)$ such that $Z \cap W = Y$.*

5.1.2. *Definition of Sublattice Operators.* - This lemma shows that, for the case of substitutable preferences, finding all stable matchings $\mathrm{S}(X, U)$ is equivalent to finding all fixed points of the operator $\mathcal{E}(\Phi)$. By isotonicity of Φ, the standard Tarski argument from previous sections shows that the sequence formed by $\Phi^k(X, \emptyset)$ is monotonically decreasing, and fixes at $(\overline{X}, \overline{Y}) = \max \mathcal{E}(\Phi)$. Similarly, iterating $\Phi^k(\emptyset, X)$ eventually fixes at a $(\underline{X}, \underline{Y}) = \min \mathcal{E}(\Phi)$. Note then that any $(Z, W) \in \mathcal{E}(\Phi)$ (corresponding to stable allocation $Z \cap W$) satisfies $\underline{X} \subseteq Z \subseteq \overline{X}$ and $\overline{Y} \subseteq W \subseteq \underline{Y}$.

This observation motivates our definition of operators on sublattices of L. Specifically, for a fixed element $(\overline{X}, \overline{Y}) \succeq (X_i, Y_i) \succeq (\underline{X}, \underline{Y})$, where i is an index, we let $L_i \equiv \{(Z, W) : (Z, W) \preceq (X_i, Y_i)\}$ and note that this is a complete sublattice. Define $\Phi_i : L \to L$ by setting $\Phi_i(Z, W) = (\Phi_i^D(W), \Phi_i^H(Z))$, where $\Phi_i^D(W) = \{z \in X_i \setminus \underline{X} : z \in C_H(\{z\} \cup W)\} \cup \underline{X}$ and $\Phi_i^H(Z) = \{z \in \underline{Y} \setminus Y_i : z \in C_D(\{z\} \cup Z)\} \cup Y_i$.

5.2. **Fixed Point Guarantees and Initializing Subproblems.** The following result shows that the family of operators introduced above satisfies the conditions required for an information sharing algorithm on the lattice $L = 2^X \times 2^X$.

Lemma 5.2. *Suppose that agents' preferences are substitutable over X. Let the sublattice L_i and $\Phi_i : L \to L$ be defined as above for some $(X_i, Y_i) \succeq (\underline{X}, \underline{Y})$. Then the following statements are true*

(1) *$\Phi_i : L_i \to L_i$ and is isotone.*
(2) *$\mathcal{E}(\Phi) \cap L_i \subseteq \mathcal{E}(\Phi_i)$.*

Proof. First we show (1). Since $\underline{X} \subseteq X_i$, and the hospitals are only allowed to choose from X_i, clearly $\Phi_i^D(W) \subseteq X_i$ for any $W \in 2^X$. By definition, $\Phi_i^H(Z) \supseteq Y_i$, so apparently $\Phi_i(Z, W) \preceq (X_i, Y_i)$ for any $(Z, W) \in L$. In particular, $\Phi_i : L_i \to L_i$. Suppose that $(Z, W) \preceq (Z', W')$ are elements of L_i, so that $Z \subseteq Z'$ and $W' \subseteq W$. By Proposition 4 of Hatfield and Kominers (2015), substitutability is preserved under a coarsening of the contract language to any $X' \subseteq X$. Thus, hospitals' preferences are substitutable over $X_i \setminus \underline{X} \subseteq X$, and, similarly, doctors' preferences are substitutable over $\underline{Y} \setminus Y_i \subseteq X$. In particular, by substitutability for doctors, $\{z \in \underline{Y} \setminus Y_i : z \in C_D(\{z\} \cup Z')\} \subseteq \{z \in \underline{Y} \setminus Y_i : z \in C_D(\{z\} \cup Z)\}$. Then $\Phi_i^H(Z') \subseteq \Phi_i^H(Z)$ by taking the union with Y_i on both sides. A similar argument shows

that $W' \subseteq W \implies \Phi_i^D(W) \subseteq \Phi_i^D(W')$. Then apparently $\Phi_i(Z,W) \preceq \Phi_i(Z',W')$, which finishes (1).

For (2), let $(Z,W) \in \mathcal{E}(\Phi)$ such that $(Z,W) \preceq (X_i,Y_i)$; that is, $Z \subseteq X_i$ and $Y_i \subseteq W$. Note that, from the discussion above, $\underline{X} \subseteq Z$ and $W \subseteq \underline{Y}$. Let $z \in Z \setminus \underline{X} \subseteq X_i \setminus \underline{X}$. Then the fixed point assumption shows $z \in \Phi^D(W) = \{z \in X : z \in C_H(\{z\} \cup W)\}$. In particular, z is also chosen by the hospitals out of $X_i \setminus \overline{X}$, so $z \in \{z \in X_i \setminus \underline{X} : z \in C_H(\{z\} \cup W)\}$. Then $Z = (Z \setminus \underline{X}) \cup \underline{X} \subseteq \Phi_i^D(W)$. Clearly, $\{z \in X_i \setminus \underline{X} : z \in C_H(\{z\} \cup W)\} \subseteq \Phi^D(W)$, so taking the union with $\underline{X}$ on both sides gives $\Phi_i^D(W) \subseteq \Phi^D(W)$, noting that $\underline{X} \subseteq \Phi^D(W) = Z$. Then $\Phi_i^D(W) = Z$. Similarly, let $w \in W \setminus Y_i \subseteq \underline{Y} \setminus Y_i$. Then $w \in \Phi^H(Z) \implies w \in \{z \in \underline{Y} \setminus Y_i : z \in C_H(\{z\} \cup Z)\}$. This shows $W = (W \setminus Y_i) \cup Y_i \subseteq \Phi_i^D(Z)$. Clearly $\{z \in \underline{Y} \setminus Y_i : z \in C_H(\{z\} \cup Z)\} \subseteq \Phi^H(Z)$, so taking unions on both sides with Y_i, noting that $Y_i \subseteq W$ by assumption, gives $\Phi_i^H(Z) \subseteq W$, so, in fact, $\Phi_i^H(Z) = W$. Then we have shown $(Z,W) \in \mathcal{E}(\Phi_i)$, so $\mathcal{E}(\Phi) \cap L_i \subseteq \mathcal{E}(\Phi_i)$, completing the proof. □

5.2.1. *Initialization of Subproblems.* As in section 2, for $i \in \mathcal{I}$ some index set we let $\langle (X_i,Y_i) \rangle$ denote the problem of finding $\mathcal{E}(\Phi_i) \subseteq L_i = \{(X,Y) \preceq (X_i,Y_i)\}$. Recall that we form the Tarski sequence $\Phi_i^k(X_i,Y_i)$, which, unless another stopping condition is triggered, converges to $\max \mathcal{E}(\Phi_i) \succeq \mathcal{E}(\Phi) \cap L_i$. After the Tarski sequence fixes, we need to initialize new problems to find elements of $\mathcal{E}(\Phi)$ below $\max \mathcal{E}(\Phi_i)$. For the core algorithm, the Euclidean structure gave a natural way to do this, by forming problems $\overline{\phi} - e_i$ for $1 \leq i \leq m$, where we identified $(2^X)^A$ with a subset of $\mathbb{R}^m$.

Here, the lattice has a less regular structure; however, we can show that there is still a simple way to initialize subproblems without missing any points of $\mathcal{E}(\Phi)$. Suppose that a Tarski sequence converges to $(X',Y') = \max \mathcal{E}(\Phi_i)$. If $(Z,W) \in \mathcal{E}(\Phi)$ and $(Z,W) \prec (X',Y')$ then, by the lattice order, either there exists $x \in X' \setminus \underline{X}$ with $Z \subseteq X' \setminus \{x\}$ or there is $y \in \underline{Y} \setminus Y'$ with $Y' \cup \{y\} \subseteq W$. Thus, to ensure that we find the fixed point (Z,W), we initialize $|X' \setminus \underline{X}| + |\underline{Y} \setminus Y'|$ new problems of the form $\langle (X' \setminus \{x\}, Y') \rangle$ and $\langle (X', Y' \cup \{y\}) \rangle$, with x and y as above.

5.3. **Full Algorithm for Finding all Stable Allocations.** In this section, we describe the full information sharing algorithm for finding all stable allocations in two-sided many-to-many matching with contracts. In what follows, we assume that agents' preferences are substitutable, so from work in Hatfield and Milgrom (2005) the collection of stable allocations $\mathrm{S}(X,U)$ must be non-empty.

Stopping Conditions: As above, we let Q be a collection of problems, which we think of as previously solved problems that still contain information useful for finding $\mathcal{E}(\Phi)$. During the algorithm, for a problem $\langle (X_i,Y_i) \rangle$, we compute the Tarski sequence $\Phi_i^k(X_i,Y_i)$, $k \geq 0$. Define the following stopping conditions for an element $\Phi_i^k(X_i,Y_i) \equiv (X',Y')$ of this sequence.

(1) $(X',Y') \preceq (\underline{X},\underline{Y})$
(2) $\Phi_i(X',Y') = (X',Y')$
(3) There exists $(X_j,Y_j) \in Q$ with $(X',Y') \preceq (X_j,Y_j)$ and $(X_i,Y_i) \not\preceq (X_j,Y_j)$

Condition (1) corresponds to the problem exiting the lattice $\mathcal{E}(\Phi)$. For Condition (2), the Tarski sequence fixes, and we are forced to initialize new problems. For Condition (3), the problem $\langle (X_i,Y_i) \rangle$ "collides" with some problem in Q, so we stop the sequence and wait for the information sharing subroutine.

Initialization: To initialize the algorithm, set $Q = Q_0 = \mathcal{E}_C = \emptyset$, and create an empty forest $F = (\emptyset,\emptyset)$. Iterate $\Phi^k(\emptyset,X)$ and $\Phi^k(X,\emptyset)$, which converge in finitely many iterations

to $(\overline{X}, \overline{Y})$ and $(\underline{X}, \underline{Y})$, the largest and smallest elements of $\mathcal{E}(\Phi)$, respectively. Add each of these to the fixed point collection $\mathcal{E}_C$. For each $x \in \overline{X} \setminus \underline{X}$, add $(\overline{X} \setminus \{x\}, \overline{Y})$ to Q_0 and Q, and add the vertex $w(\overline{X} \setminus \{x\}, \overline{Y})$ to F as an isolated root Similarly, for each $y \in \underline{Y} \setminus \overline{Y}$, add $(\overline{X}, \overline{Y} \cup \{y\})$ to Q_0 and Q, and add the vertex $w(\overline{X}, \overline{Y} \cup \{y\})$ to F as an isolated root. Then run Algorithm 1, where we modify the subroutines InformationAcquire (Subroutine 3.4) and InformationCombine (Subroutine 3.5) as shown below.

Subroutine 5.3 (InformationAcquire)**.** *Initialize $Q_A = Q_0$, and set $\mathcal{E}_{temp} = Q_I = I = \emptyset$.*

If $(X_i, Y_i) \in Q_0$ has either (i) $(X_i, Y_i) \preceq (\underline{X}, \underline{Y})$ or (ii) there exists $(X_j, Y_j) \in Q_0$ s.t. $(X_i, Y_i) \prec (X_j, Y_j)$, then remove (X_i, Y_i) from Q_A, Q_0, and Q.

While $Q_A \neq \emptyset$, do the following:

For each $(X_i, Y_i) \in Q_A$, compute the Tarski sequence $\Phi_i^k(X_i, Y_i)$ until a stopping condition (as above) is triggered for some $k = \ell$. Set $(X', Y') = \Phi^\ell(X_i, Y_i)$, and do the following:

(a) *Add $w(X', Y')$ to F with $c(w(X', Y')) = w(X_i, Y_i)$*
(b) *Remove (X_i, Y_i) from Q_A.*

For stopping conditions[16] *(2) and (3), do the following:*

(2) *(i) If $\Phi(X', Y') = (X', Y')$, then add (X', Y') to $\mathcal{E}_{temp}$. Also, (ii) for each $x \in X' \setminus \underline{X}$ and $y \in \underline{Y} \setminus Y'$, add $(X' \setminus \{x\}, Y')$ and $(X', Y' \cup \{y\})$ to Q_0, Q_A, and Q if this problem is not already in Q. Finally, (iii) for each problem (X'', Y'') just added, add $w(X'', Y'')$ to F as an isolated root.*
(3) *By assumption, the stopping condition was triggered by a "collision" with some $(X_j, Y_j) \in Q$. Then (i) add (X_i, Y_i) to Q_I and (ii) add j to collision index set $I(i)$.*

Return $(Q_I, I, \mathcal{E}_{temp})$.

Subroutine 5.4 (InformationCombine)**.** *To initialize, form an empty digraph $G = (V, E)$ with $V = E = \emptyset$, set $Q_{temp} = \emptyset$, and add i to $I(i)$ for each $(X_i, Y_i) \in Q_0$. Next, build the collision digraph as follows:*

For each $(X_i, Y_i) \in Q_0$, (i) add v_i to V, and for all collision indices $j \in I(i)$, (ii) compute $w_k = r(w_j)$ and (iii) add v_k to V and (v_i, v_k) to E. Next, do the following computations with G:

(1) *Compute the strong components $\mathcal{G}$ of G.*
(2) *Remove $\{v_k\} \in \mathcal{G}$ if $(X_k, Y_k) \notin Q_0$, leaving trimmed components $\{S_j\}_{j \in \mathcal{J}}$*
(3) *Compute the reachable set H_j for each S_j with $j \in \mathcal{J}$.*

For each $j \in \mathcal{J}$, combine information to form new problems:

(a) *Add $(X', Y') = \left(\bigcap_{v_k \in H_j} X_k, \bigcup_{v_k \in H_j} Y_k \right)$ to Q_{temp}.*
(b) *Add the vertex $w(X', Y')$ to F with $c(w(X', Y')) = \{r(w_k) : v_k \in S_j\}$.*

Set $I(i) = \emptyset$ for $i \in \mathcal{I}$. To check for new collisions, do the following:

For each $(X_i, Y_i) \in Q_{temp}$, if $\exists (X_k, Y_k) \in Q$ and $w_j \in c(w_i)$ such that $(X', Y') \preceq (X_k, Y_k)$ and $(X_j, Y_j) \not\preceq (X_k, Y_k)$, then add collision index k to $I(i)$.

Set $Q_0 \leftarrow Q_{temp}$ and $Q \leftarrow Q \cup Q_{temp}$.

Return (Q_0, I).

[16] If both stopping conditions (1) and (3) both occur, use condition (1): the problem has left the lattice $\mathcal{E}(\Phi)$, so there is no reason to seek further information.

5.3.1. *Discussion.* We briefly note the substantive differences between this algorithm and Algorithm 1 for finding all core outcomes. Here, when forming new problems during InformationCombine, we replace the pointwise minima used in Algorithm 1 with the appropriate meet (greatest lower bound) for the lattice $2^X \times 2^X$. For the order "$\preceq$", the meet is given by intersections in the first component and unions in the second, as shown above. Note that, for this algorithm, the number of subproblems initialized when a Tarski sequence fixes is *not constant*. This is necessary because $2^X \times 2^X$ has a less regular structure than the Euclidean lattice $(2^X)^A$ used in section 2. Finally, for stable allocations, there is no equivalent of Lemma 2.7, since $\mathcal{E}(\Phi)$ are fixed points of an isotone (not antitone) operator. Thus, after finding $(Z, W) \in \mathcal{E}(\Phi)$, we still have to search the lattice $\{(X', Y') \prec (Z, W)\}$ strictly below (Z, W), whereas, for Algorithm 1, Lemma 2.7 shows that we can stop immediately. Because of this, Subroutine 5.3 has one fewer distinct stopping condition. Otherwise, the algorithm is structurally identical, with slightly different notation.

The following result is an easy extension of Lemma 3.11 and Theorem 3.12.

Theorem 5.5. *FindStable returns $\mathcal{E}_C = \mathcal{E}(\Phi)$. In particular, $\{Z \cap W : (Z, W) \in \mathcal{E}_C\} = \mathrm{S}(X, U)$, so the algorithm finds all stable allocations.*

Proof. This is a straightforward extension of the auxiliary lemmas and correctness proof provided above, using Lemma 5.1 from Hatfield and Kominers (2015) to get all of $\mathrm{S}(X, U)$ from the fixed point set $\mathcal{E}(\Phi)$. Here, we just note the substantive differences between the proofs.

In the analysis for Algorithm 1, we think of $(2^X)^A$ as a sublattice of $\mathbb{R}^m$. Thus, meet for this lattice is a pointwise minimum. For instance, we form new problems by setting $\phi = \min_{v_k \in H_j} \phi_k$. For finding all stable allocations, every such minimum must be replaced by the lattice meet for $L = 2^X \times 2^X$. Given a collection $\{(Z_i, W_i)\}_{i \in I} \subseteq 2^X \times 2^X$, the meet, or greatest lower bound, takes the form $\bigwedge_{i \in I} (Z_i, W_i) = \left(\bigcap_{i \in I} Z_i, \bigcup_{i \in I} W_i\right)$.

Our termination analysis in Lemma 3.11 needs to be slightly modified. For a set $E \subseteq 2^X \times 2^X$, define $d(E) = \max_{(Z,W) \in E} (|Z| + |X \setminus W|)$ and $D(E) = \sum_{(Z,W) \in E} (|Z| + |X \setminus W|)$. These functions are positive and take finitely many values on the lattice $2^X \times 2^X$. Moreover, $D(\cdot)$ is additive on disjoint subsets of $2^X \times 2^X$. The rest of the termination analysis follows as before.

Finally, as noted in the discussion above, there is an extra stopping condition in Algorithm 1 corresponding to the guarantee from Lemma 2.7 that no two points $\phi, \phi' \in \mathcal{E}(T^2)$ can be compared under $\preceq$. In the analysis, we use this lemma to argue that $\mathcal{E}(T) \cap \{\phi'' : \phi'' \prec \phi\} = \emptyset$ whenever $\phi \in \mathcal{E}(T)$. No such analysis is necessary for finding stable outcomes, because this stopping condition is not used. □

5.3.2. *Remark.* Consider the worst case scenario where $\Phi_i : L_i \to L_i$ fixes at $(X_i, Y_i) = \max L_i$ for every $(\underline{X}, \underline{Y}) \preceq (X_i, Y_i) \preceq (\overline{X}, \overline{Y})$. If we were to initialize *every* subproblem of the form $(X_i \setminus \{x\}, Y_i)$ and $(X_i, Y_i \cup \{y\})$, we would create $\left(|\overline{X}| - |\underline{X}| + |\underline{Y}| - |\overline{Y}|\right)!$ problems in total, whereas greedy search only needs to check $2^{|\overline{X}| - |\underline{X}| + |\underline{Y}| - |\overline{Y}|}$ points. However, in Subroutine 5.3, we only initialize non-redundant problems, so this does not happen, and the algorithm does weakly better than greedy search in all cases.

6. Conclusion

The approach developed above gives an alternative method for computing reasonable economic outcomes in general matching markets when substitutability is not guaranteed. We show how our technique has several applications to computing objects of interest in economic problems for which there is a lattice structure. The information sharing approach developed here also gives the first algorithm for finding all stable allocations in bilateral matching with contracts with substitutable preferences.

References

Blair, C. (1988). The lattice structure of the set of stable matchings with multiple partners. *Mathematics of Operations Research 13*, 619–628.

Echenique, F. (2007, July). Finding all equilibria in games with strategic complements. *Journal of Economic Theory 135*(1), 514–532.

Echenique, F. and J. Oviedo (2006). A theory of stability in many-to-many matching markets. *Theoretical Economics 1*, 233–273.

Echenique, F. and M. B. Yenmez (2013). A solution to matching with preferences over colleagues. *Games and Economic Behavior 59*, 46–71.

Gross, J. L. and J. Yellen (2005). Graph theory and its applications. *CRC Press*.

Gusfield, D. and R. Irving (1989). The stable marriage problem: Structure and algorithms. *MIT Press*.

Hatfield, J. W. and S. D. Kominers (2012). Matching in networks with bilateral contracts. *American Economic Journal: Microeconomics 4*, 176–208.

Hatfield, J. W. and S. D. Kominers (2015). Contract design and stability in many-to-many matching. Harvard Business School Working Paper.

Hatfield, J. W. and P. Milgrom (2005). Matching with contracts. *American Economic Review 95*, 913–935.

Irving, R. and P. Leather (1986). The complexity of counting stable marriages. *SIAM Journal of Computing 15*, 655–667.

Kojima, F. (2015). Finding all stable matchings with couples. *Journal of Dynamics and Games*.

Kominers, S. D. (2010). Matching with preferences over colleagues solves classical matching. *Games and Economic Behavior 68*(2), 773–780.

Martínez, R., J. Massó, A. Neme, and J. Oviedo (2004). An algorithm to compute the full set of many-to-many stable matchings. *Mathematical Social Sciences 47*, 187–210.

McVitie, D. G. and L. B. Wilson (1971). The stable marriage problem. *Communications of the ACM 14*, 486–490.

Roth, A. E. and E. Peranson (1999). The effects of the change in the NRMP matching algorithm. *American Economic Review 89*, 748–780.

Roth, A. E. and M. Sotomayor (1996, November). Stable outcomes in discrete and continuous models of two-sided matching: A unified treatment. *Review de Econometria*.

Schwarz, M. and M. B. Yenmez (2011, March). Median stable matching for markets with wages. *Journal of Economic Theory*.

Sethuraman, J., C. Teo, and L. Qian (2006). Many-to-one matching: Geometry and fairness. *Mathematics of Operations Research 31*, 581–596.