International Journal of Game Theory (2023) 52:629–648
https://doi.org/10.1007/s00182-022-00833-0

ORIGINAL PAPER

Complexity of stability in trading networks

Tamás Fleiner[1,2] · **Zsuzsanna Jankó**[2,3] · **Ildikó Schlotter**[1,2] · **Alexander Teytelboym**[4]

Accepted: 19 September 2022 / Published online: 8 April 2023

Abstract

Efficient computability is an important property of solution concepts. We consider the computational complexity of finding and verifying various solution concepts in trading networks—multi-sided matching markets with bilateral contracts and without transferable utility—under the assumption of full substitutability of agents' preferences. It is known that outcomes that satisfy *trail stability* always exist and can be found in linear time. However, we show that the existence of *stable* outcomes—immune to deviations by arbitrary sets of agents—is an NP-hard problem in trading networks. We also show that even verifying whether a given outcome is stable is NP-hard in trading networks.

Keywords Matching markets · Market design · Computational complexity · Trading networks · Flow networks · Supply chains · Trail stability · Chain stability · Stability · Contracts

JEL Classification C78 · L14

We would like to thank Péter Biró, John Hatfield, Ravi Jagadeesan, Scott Kominers, Mike Ostrovsky, and Alex Westkamp for fruitful conversations about this project. Fleiner and Jankó were supported by the Hungarian Scientific Research Fund (OTKA grants K128611 and K143858) as well as the MTA-ELTE Egerváry Research Group. Jankó also acknowledges the support of the MTA-BCE Strategic Interaction Research Group. Part of Fleiner's research was carried out during two working visits at Keio University. Schlotter acknowledges the support of the Hungarian Academy of Sciences under its Momentum Programme (LP2021-2), and the Hungarian Scientific Research Fund (OTKA grants K128611 and K124171). The research by Fleiner and Schlotter reported in this paper was also supported by the BME-Artificial Intelligence FIKP grant of EMMI (BME FIKP-MI/SC) and by the BME NC TKP2020 grant of NKFIH Hungary. This work was supported by the Economic and Social Research Council grant number ES/R007470/1. Teytelboym is grateful for the hospitality of the Budapest University of Technology and Economics in August 2016.

✉ Alexander Teytelboym
alexander.teytelboym@economics.ox.ac.uk

Extended author information available on the last page of the article

1 Introduction

One of the most important features of the marriage market or the college admissions problems is that (pairwise) stable outcomes can always be found efficiently using the celebrated Deferred Acceptance algorithm (Gale and Shapley 1962). Efficient computability is key for practical applications of the Deferred Acceptance algorithm and its variants (Roth 1984; Abdulkadiroğlu and Sönmez 2003; Sönmez and Switzer 2013; Hatfield and Kominers 2014). Moreover, since in standard (one-to-one) marriage markets or (many-to-one) college admissions problems (pairwise) stable outcomes coincide with the core, there is no obvious need to select among various solution concepts.

However, in more complex, many-to-many matching markets there are many different, yet economically natural and interpretable solution concepts (Blair 1988; Sotomayor 1999; Echenique and Oviedo 2006; Klaus and Walzl 2009). For example, pairwise stable outcomes do not coincide with the core (Blair 1988). What makes a good solution concept? At the very least, it should make falsifiable predictions. But in economics good solution concepts often derive their appeal for normative reasons: they have sensible properties and make intuitive sense in particular settings. Efficient computability can serve as one such desirable property. As Papadimitriou (2007, p. 29–30) argues:

> The reason is simple: If an equilibrium concept is not efficiently computable, much of its credibility as a prediction of the behavior of rational agents is lost – after all, there is no clear reason why a group of agents cannot be simulated by a machine. Efficient computability is an important modelling prerequisite for solution concepts.

If agents cannot efficiently find certain deviations from an outcome in a matching market, then a stability concept that is based on these deviations may be too strong. On the other hand, if it is computationally hard to verify whether an outcome satisfying a particular solution concept exists, then this limits the applicability of this solution concept for matching market design. Finding a solution concept with attractive computational as well as economic properties would allow trading networks to be deployed in a variety of empirical (Fox 2017) and practical applications (Morstyn et al. 2018).

Our model and its connections to previous work. In this paper, we consider the computational complexity of various solution concepts in *trading networks*. A trading network is a matching market in which heterogeneous agents (firms) can sign many contracts that specify the terms of the trade, quality of the product, price etc. with their suppliers (upstream) and with their buyers (downstream). Following a seminal contribution by Ostrovsky (2008), we assume that agents' preferences are *fully substitutable*, that is, upstream and downstream contracts are complements, but contracts on the same side are substitutes. Following Ostrovsky (2008), Westkamp

(2010) and Hatfield and Kominers (2012), contracts in our model are discrete and utility is not transferable.[1] However, the model is general enough to capture not only wealth effects, but also distortionary frictions, such as sales taxes or bargaining costs (see, for example, Fleiner et al. 2019).

We allow firms to enter into upstream or downstream contracts (or both) with any other firm. Hence, we impose no restrictions on the structure of the trading network in contrast to models of supply chains (Ostrovsky 2008; Westkamp 2010; Hatfield and Kominers 2012) where no firm can be simultaneously upstream and downstream from another (that is, where the trading network has an acyclic structure).

However, the generality of such a setting comes at a price: here, *stable* outcomes, i.e., those immune to arbitrary blocks by sets of firms, do not necessarily exist (Hatfield and Kominers 2012). The non-existence of stable outcomes in a trading network model starkly contrasts models of supply chains where stable outcomes always exist (Hatfield and Kominers 2012), as well as the models with continuous prices without frictions where stable outcomes not only exist but also coincide with competitive equilibrium outcomes (Hatfield et al. 2013; Candogan et al. 2016; Fleiner et al. 2019).

Our contribution. Following Fleiner et al. (2016), we first point out that outcomes satisfying *trail stability*, an extension of pairwise or chain stability (Ostrovsky 2008), always exist and can be found in linear time (in the number of contracts) by an extension of the Deferred Acceptance algorithm (Theorem 1). Trail-stable outcomes are immune to locally blocking trails (sequences of distinct contracts in which a buyer of one contract is the seller in the next). In a locally blocking trail, agents simultaneously accept pairs of upstream and downstream contracts along the trail, but there must also be agents that "kick off" (and "complete") the blocking trail by unilaterally offering (or "accepting") a single downstream (or a single upstream) contract. One can think of an agent in a locally blocking trail as a manager in a firm receiving a phone call with offer to buy inputs. The manager keeps the offer on hold until she is able to find a new sale contract. If she is successful, the locally blocking trail continues and if the trail comes back to the firm, another manager picks up the phone. Trail stability is an attractive solution concept in settings where agents are myopic or have only partial commitment power.[2]

Considering stable outcomes, we prove that deciding whether a stable outcome exists in a trading network is NP-hard. In fact, we prove a stronger result (Corollary 1) by showing that deciding if a stable outcome exists is NP-complete even in a *flow network* (Fleiner 2014), which is a special case of a trading network.

To obtain these hardness results, we use a relaxed stability concept that we call *path-or-cycle stability* and that can be derived from trail stability by relaxing

[1] Several papers have studied trading networks in a perfectly transferable (Hatfield et al. 2013; Candogan et al. 2016) and imperfectly transferable utility settings (Fleiner et al. 2019). Hatfield et al. (2021) offer the most general model that generalizes settings with and without transfers.

[2] In a setting with continuous prices and distortionary frictions, trail-stable outcomes essentially coincide with competitive equilibrium outcomes (Fleiner et al. 2019).

the requirement of locally blocking trails that agents can only make an upstream (downstream) offer following the receipt of a downstream (upstream) offer. Thus we allow agents to offer an upstream and a downstream contract *simultaneously*, forming *blocking cycles*. In this setting, *path-or-cycle-stable* outcomes are immune to blocking sets in the form of paths (trails in which every agent is distinct) or cycles. The key motivation for this concept is that in flow networks path-or-cycle-stable outcomes coincide with stable outcomes (Theorem 3), because any blocking set in a flow network can be decomposed into a set of blocking cycles and blocking paths. This idea is analogous to a result by Hatfield et al. (2021) showing that in trading networks blocking sets can be decomposed into certain ordered blocking sets (that they call *chains*) under a monotonicity condition.

Path-or-cycle-stable outcomes do not always exist in trading networks. We show that the decision problem of whether a path-or-cycle-stable outcome (or, equivalently by Theorem 3, a stable outcome) exists in a flow network is NP-complete (Theorem 2). Therefore, the ability to minimally coordinate upstream and downstream contract offers renders the computational problem for existence intractable. The proof gives a reduction to this problem from the problem of partitioning a directed graph into two acyclic sets, which is known to be NP-complete (Bokal et al. 2004). Our result superficially resembles the problems of determining the existence of pairwise stable outcomes in two-sided hospital-resident markets with couples (McDermid and Manlove 2010), with sizes (Delacrétaz 2019), or with multidimensional knapsack constraints (Delacrétaz et al. 2019): in these models, as in ours, finding stable outcomes is hard due to the presence of various constraints. However, as far we know, none of our results has appeared elsewhere in the literature.

Finally, we show that even *verifying* that a particular outcome is not stable is NP-complete (Theorem 4). The proof provides a reduction from the set partition problem, which is known to be (weakly) NP-complete.

Our hardness results do not extend to more restricted models, such as the case of acyclic trading networks, or to alternative models with continuous prices without frictions. Determining computability properties of solution concepts in such settings is an open question.

Organization. Section 2 introduces the full model of trading networks and the special case of flow networks 2.3. In Sect. 3, we introduce trail stability 3.1, path-or-cycle stability 3.2, and stability 3.3, and state the main results about the computational complexity of finding outcomes that satisfy these solution concepts.

2 Preliminaries

Our notation follows Fleiner et al. (2016).

2.1 Our model

In a *trading network*, there is a finite set of agents (firms or consumers) F and a finite set of contracts X. A contract $x \in X$ is a bilateral agreement between a buyer $b(x) \in F$ and a seller $s(x) \in F$. Hence, $F(x) := \{s(x), b(x)\}$ is the set of firms associated with contract x and, more generally, $F(Y)$ is the set of firms associated with contract set $Y \subseteq X$. Given a set $Y \subseteq X$ of contracts, call $Y_B^f := \{x \in Y \mid b(x) = f\}$ and $Y_S^f := \{x \in Y \mid s(x) = f\}$ the sets of f's upstream and downstream contracts in Y – for which f is a buyer and a seller, respectively. Clearly, Y_B^f and Y_S^f form a partition over the set of contracts $Y^f := \{y \in Y \mid f \in F(y)\}$ which involve f, since an agent cannot be a buyer and a seller in the same contract. A firm f is a *terminal seller* if there are no upstream contracts in X for f in the network and f is a *terminal buyer* if the network does not contain any downstream contracts in X for f. An agent who is either a terminal buyer or a terminal seller is called a *terminal agent*. In an *acyclic* trading network, no agent can simultaneously buy and sell from another agent, even via intermediaries.

Every firm has a choice function C^f, such that $C^f(Y^f) \subseteq Y^f$ for any $Y^f \subseteq X^f$.[3] We say that a choice function of $f \in F$ satisfies the *irrelevance of rejected contracts (IRC)* condition if for any $Y \subseteq X$ and $C^f(Y) \subseteq Z \subseteq Y$ we have that $C^f(Z) = C^f(Y)$ (Blair 1988; Alkan 2002; Fleiner 2003; Echenique 2007; Aygün and Sönmez 2013).

For any $Y \subseteq X$ and $Z \subseteq X$, define the *chosen* set of upstream contracts

$$C_B^f(Y|Z) := C^f(Y_B^f \cup Z_S^f) \cap X_B^f$$

which is the set of contracts f chooses as a buyer when f has access to upstream contracts Y and downstream contracts Z. Analogously, define the chosen set of downstream contracts

$$C_S^f(Z|Y) := C^f(Z_S^f \cup Y_B^f) \cap X_S^f.$$

For brevity and by abuse of notation, we will also use $C^f(Y|Z) := (C_B^f(Y|Z)|C_S^f(Z|Y))$, so $C^f(Y|Z) = (Y'|Z')$ means that if f is offered upstream and downstream contracts Y and Z, respectively, then Y' and Z' are those among them that f chooses (with $Y' \subseteq Y$ and $Z' \subseteq Z$). We also define the *rejected* sets of contracts $R_B^f(Y|Z) := Y^f \setminus C_B^f(Y|Z)$ and $R_S^f(Z|Y) := Z^f \setminus C_S^f(Z|Y)$. An *outcome* $A \subseteq X$ is a set of contracts.

An outcome $A \subseteq X$ is *individually rational* for an agent $f \in F$ if $C^f(A^f) = A^f$. We call A *acceptable* if A is individually rational for all agents $f \in F$. For outcomes $W, A \subseteq X$, we say that A is (W, f)-*acceptable* if $A^f \subseteq C^f(W^f \cup A^f)$, i.e., if the agent f chooses all contracts from set A^f whenever she is offered A alongside W. An outcome A is W-*acceptable* if A is (W, f)-acceptable for all agents $f \in F$. Note that outcome A is individually rational for agent f if and only if it is $(\emptyset, f)$-acceptable.

[3] Since firms only care about their own contracts, we can write $C^f(Y)$ to mean $C^f(Y^f)$.

2.2 Assumptions on choice functions

We can now state our key assumption on choice functions introduced by Ostrovsky (2008).

Definition 1 The choice function of $f \in F$ is *fully substitutable* if for all $Y' \subseteq Y \subseteq X$ and $Z' \subseteq Z \subseteq X$ it is:

1 *Same-side substitutable* (SSS):
   (a) $R_B^f(Y'|Z) \subseteq R_B^f(Y|Z)$
   (b) $R_S^f(Z'|Y) \subseteq R_S^f(Z|Y)$
2 *Cross-side complementary* (CSC):
   (a) $R_B^f(Y|Z) \subseteq R_B^f(Y|Z')$
   (b) $R_S^f(Z|Y) \subseteq R_S^f(Z|Y')$

Contracts are substitutable if every firm regards any of its upstream or any of its downstream contracts as substitutes, but its upstream and downstream contracts as complements. Hence, rejected downstream (upstream) contracts continue to be rejected whenever the set of offered downstream (upstream) contracts expands or whenever the set of offered upstream (downstream) contracts shrinks.

Hatfield et al. (2012) showed that testing substitutability can be performed in polynomial time if there is a preference relation over the contracts, but may require an exponential number of queries if we have access only to the choice functions of the agents.

2.3 Special case: flow networks and flow-based choice functions

We first define flow-based choice functions. Flow-based choice functions work differently for terminal and non-terminal agents. A terminal agent has a flow-based choice function if it always accepts all offered contracts. However, a non-terminal agent has a flow-based choice function if it accepts as many upstream and downstream contracts as it can (using a linear preference ordering) subject to the constraint that the number of accepted upstream and downstream contracts is the same.

Definition 2 C^f is *flow-based* if

- whenever $f \in F$ is a terminal agent, $C^f(Y^f) = Y^f$ for any $Y \subseteq X$;
- whenever $f \in F$ is not a terminal agent
  - f has linear-order preferences $\succ_B^f$ over X_B^f and linear-order preferences $\succ_S^f$ over X_S^f, and
  - Given access to upstream contracts $Y \subseteq X_B^f$ and downstream contracts $Z \subseteq X_S^f$, a firm f with flow-based choice function will choose its k most preferred upstream contracts from Y according to $\succ_B^f$ and k most preferred downstream contracts from Z according to $\succ_S^f$ where $k := \min\{|Y|, |Z|\}$.

Since non-terminal agents with flow-based choice functions always pick the same number of upstream and downstream contracts, their preferences satisfy the so-called *Kirchhoff* equality.[4] It is immediate that flow-based choice functions defined in this way are fully substitutable and satisfy the IRC condition.

A *flow network* is a trading network in which there are exactly two terminal agents (and the remaining agents are non-terminal) and all choice functions are flow-based (Fleiner 2014).[5]

2.4 Computational complexity

We now provide a brief overview of the key concepts in computational complexity to ensure that our paper is self-contained.[6] Computational complexity theory deals with assessing the intrinsic hardness of computational problems. Most often we deal with a *decision problem* Q where for each *input instance* I, expressed as a string, there is a correct answer: *yes* or *no*; consequently we talk about *yes*– and *no*–instances of Q. To measure how hard or easy a certain computational problem is, we consider the resources, in terms of time and space, necessary for an algorithm that answers the problem correctly. The running time of a given algorithm is described by a function $T(n)$, denoting the maximum number of basic computational steps that the algorithm may perform on any input instance of length n.

We consider an algorithm *efficient*, if its running time can be upper-bounded by a polynomial of constant degree. Decision problems that can be solved by such an algorithm are said to be in P, the class of *polynomial-time solvable* problems. The class NP contains decision problem for which the answer *yes* can be efficiently *verified* given some short proof. More precisely, a decision problem Q belongs to the class NP, if for each *yes*–instance I of the problem, there exists a *proof* (or *witness*) w whose length is polynomial in $|I|$ and that can be used to verify in polynomial time that I is indeed a *yes*–instance. As an example, consider the problem of deciding if a Stable Roommates with Ties instance admits a stable matching (the SRT problem): in case of a *yes*–instance and a witness for this in the form of a stable matching, we can easily verify its correctness by checking all possible blocking pairs.

Although for numerous problems in NP, such as the SRT problem above, no polynomial-time algorithm is known to exist, we currently have no definitive proof showing that some problem $Q \in \mathsf{NP}$ is not in P. Nevertheless, we can infer the hardness of Q by using *polynomial-time reductions*: a problem Q' can be polynomially reduced to Q, if for each instance I' of Q' we can compute in polynomial time an instance I of Q such that I is a *yes*–instance of Q if and only if I' is a *yes*–instance of Q'. A problem Q is NP-*hard*, if *any* problem in NP can be reduced to Q; and we say that Q is NP-*complete*, if it is NP-hard and contained in NP. The first problem

[4] Flow-based choice functions are a special case of "separable" choice functions (Fleiner et al. 2016).

[5] In fact, what follows is a simplification of Fleiner's flow network model where all capacities have value 1 and flow values must be integral.

[6] For a more detailed introduction into the area of computational complexity, we refer the reader to the monograph by Garey and Johnson (1979), the book by Cormen et al. (2009) or Sects. 1 and 2 of the survey by Roughgarden (2010) that focuses specifically on economic applications.

ever shown to be NP-hard (by Cook 1971) was the problem of checking whether a formula in propositional logic is satisfiable (the SAT problem). Exploiting the transitivity of polynomial-time reductions, we can prove that a problem Q is NP-hard by giving a reduction from SAT or any other problem already known to be NP-hard.

3 Solution concepts and their computational complexity

3.1 Trail stability

In order to explain our first solution concept, we first define a *trail*.

Definition 3 A sequence $(x_1, \dots, x_M)$ of M distinct contracts for some $M \in \mathbb{N}^+$ is a *trail* if $b(x_m) = s(x_{m+1})$ holds for all $m \in \{1, \dots, M-1\}$.

Note that a firm may appear several times along the trail (as opposed to contracts, which may appear at most once). Further note that a trail T is a sequence of contracts; however, since all contracts of a trail are distinct, we may sometimes use T to refer to the *set* of contracts contained in T, whenever this does not cause any confusion. The following solution concept was introduced by Fleiner et al. (2016).

Definition 4 An outcome $A \subseteq X$ is *trail-stable* if

1. A is acceptable.
2. There is no trail $T = (x_1, x_2, \dots, x_M)$ such that $T \in (X \backslash A)^M$ and
   - a $\{x_1\}$ is (A, f_1)-acceptable for $f_1 = s(x_1)$, and
   - b $\{x_{m-1}, x_m\}$ is (A, f_m)-acceptable for $f_m = b(x_{m-1}) = s(x_m)$ whenever $1 < m \leq M$ and
   - c $\{x_M\}$ is (A, f_{M+1})-acceptable for $f_{M+1} = b(x_M)$.

Such a trail T is called a *locally blocking trail to* A.

Trail stability is a natural solution concept when firms interact mainly with their buyers and suppliers and deviations by arbitrary sets of firms are difficult to arrange. In a trail-stable outcome, no agent wants to drop his contracts and there exists no sequence of *consecutive* bilateral contracts comprising a trail such that any intermediate agent who is offered a downstream (upstream) contract along the trail wants to choose it alongside the subsequent upstream (downstream) contract in the trail. Importantly, we require that the first (final) agent wants to unilaterally offer (accept) the first (final) contract in the trail.[7] For trail stability, we do not require that intermediate agents simultaneously accept all the contracts along the locally blocking

[7] The trail and the order of conditional acceptances can, of course, be reversed with f_{M+1} offering the first upstream contract to seller f_M and so on.

trail. Instead they are simply required to myopically accept pairs of upstream and downstream contracts, one pair at a time, as they appear along the trail.[8]

Fleiner et al. (2016) proved that under full substitutability trail-stable outcomes always exist in trading networks and, under certain conditions, have a familiar lattice structure. Trail-stable outcomes may not be Pareto-efficient. In a variant of our model with continuous prices, Fleiner et al. (2019) showed that in the economy, trail-stable outcomes essentially coincide with competitive equilibrium outcomes even if distortionary frictions, such as sales taxes are present.[9] Here we focus on the computability of trail-stable outcomes.

Theorem 1 (Fleiner et al. 2016). *Suppose that in a trading network choice functions are fully substitutable and satisfy the IRC condition. Then a trail-stable outcome exists and can be found in time linear in the number of contracts.*

The proof of Theorem 1 follows immediately from the proof of existence of trail-stable outcomes provided by Fleiner et al. (2016) and Adachi (2017). Trail-stable outcomes are found by a generalized Gale-Shapley algorithm defined in Fleiner et al. (2016). In the generalized Gale-Shapley algorithm, at least one contract is rejected at each step and since the number of contracts is finite, the number of steps required to find a trail-stable outcome is bounded by the number of contracts.

We remark that the running time of the algorithm in Theorem 1 is optimal in the sense that the number of steps necessary for simply *reading* all contracts is linear in the number of contracts.[10] Whether or not our linear-time algorithm is practical depends on the total number of contracts. In some cases the number of contracts can be rather large because, for example, the same trade at two different discrete prices creates two different contracts.

3.2 Path-or-cycle stability

The definition of trail stability requires that there be initial and final agents that would unconditionally offer or accept an upstream or a downstream contract while all the intermediaries only make a downstream (upstream) offer after receiving an upstream (downstream) offer. Let us now relax this condition and allow agents to form blocking cycles: now every agent can offer an upstream and a downstream contract simultaneously without having to accept them individually. This is a mild strengthening of trail stability (whenever there is at most one contract between agents) that treats all blocking agents as intermediate agents in the blocking trail,

[8] If agents were required to accept to all the contracts along the locally blocking trail, this would lead to a strictly weaker solution concept (under full substitutability) called "weak trail stability" (Fleiner et al. 2016).

[9] The First Fundamental Welfare Theorem can, of course, fail in the presence of distortionary frictions.

[10] This optimality holds if the contracts are spelled out one by one in the problem description, and if the inputs have to be read completely. We leave open the question of whether it is possible to find a trail-stable outcome without spelling out all contracts (cf., Gonczarowski et al. 2019).

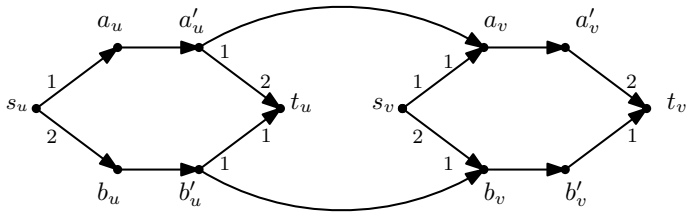

Fig. 1 Illustration of two node gadgets, D_u and D_v, and the two contracts in Z connecting them; the figure assumes $uv \in E$. The number on some contract x written next to some firm indicates the rank of the contract x in the preference ordering of the given firm

and thus allows for a small amount of additional coordination among blocking agents.

For the sake of further simplicity, let us only focus on *paths*, i.e., trails in which *all agents are distinct*.

Definition 5 A sequence $(x_1, \dots, x_M)$ of M contracts for some $M \in \mathbb{N}^+$ is a *path* if

- $b(x_m) = s(x_{m+1})$ holds for all $m \in \{1, \dots, M-1\}$, and
- all firms $s(x_1), s(x_2), \dots, s(x_M)$, and $b(x_M)$ are distinct.

Now, a trail that returns to its origin agent and goes through each of its agents only once is a *cycle*. More precisely:

Definition 6 A sequence $(x_1, \dots, x_M)$ of M contracts is a *cycle* if
- $b(x_m) = s(x_{m+1})$ holds for all $m \in \{1, \dots, M-1\}$,
- $b(x_M) = s(x_1)$, and
- all firms $s(x_1), s(x_2), \dots, s(x_M)$ are distinct.

Definition 7 An outcome $A \subseteq X$ is *path-or-cycle-stable* if

1. A is acceptable.
2. There is no path or cycle B such that $B \in (X \backslash A)^M$ and B is (A, f)-acceptable for each $f \in F(B)$. Such paths or cycles are called *blocking paths* and *blocking cycles*.

An interesting property of flow-based choice functions is that given an outcome $A \subseteq X$, any cycle C disjoint from A is a blocking cycle, as any firm which is offered a pair of additional upstream and downstream contracts will accept them. We will use this property to prove our second key result.

Theorem 2 *It is* NP*-complete to decide if a flow network admits a path-or-cycle-stable outcome.*

Proof The problem is in NP, since given an outcome A, we can check in linear time (with respect to the number of contracts) whether it admits a blocking path or a cycle: we only have to check that the contracts $X \backslash A$ do not contain a cycle or a path starting with an $(A, \cdot)$-acceptable contract and ending with an $(A, \cdot)$-acceptable

contract; both these tasks can be decided using, e.g., some variant of the depth-first search (DFS) algorithm on the directed graph representing $X \backslash A$ (see the book by Cormen et al. 2009 for more details on the DFS algorithm).

To prove NP-hardness, we present a polynomial reduction from the following problem: given a directed graph D, decide whether it is possible to partition the vertices of D into two acyclic sets V_1 and V_2. Here, a set U of vertices is *acyclic*, if there is no directed cycle in $D[U]$. This problem was proved to be NP-complete by Bokal et al. (2004).

Given our input $D = (V, E)$, we construct a set X of contracts and a set F of firms with flow-based choice functions (see Fig. 1 for an illustration). There will be at most one upstream contract and at most one downstream contract between any two firms, so we will denote a contract x with $s(x) = f_1$ and $b(x) = f_2$ as $f_1 f_2$. First, we introduce a *node gadget* D_v for each $v \in V$; the set of firms in D_v is $\{s_v, a_v, a'_v, b_v, b'_v, t_v\}$ and the set of contracts in D_V is $\{s_v a_v, s_v b_v, a_v a'_v, b_v b'_v, a'_v t_v, b'_v t_v\}$. Next, we add terminal firms s and t, together with the contracts $s s_v$ and $t_v t$ for each $v \in V$. Finally, we add the contract set $Z = \{a'_u a_v, b'_u b_v \mid uv \in E\}$ where recall that E is the set of edges of the input graph D.

Instead of describing the full preferences of each firm over its upstream and downstream contracts, we only define a partial ordering (and assume that the preferences of each firm respect this partial order). Namely, for each $v \in V$, we let s_v prefer $s_v a_v$ to $s_v b_v$, and we let t_v prefer $b'_v t_v$ to $a'_v t_v$. In Fig. 1, we indicate the rankings of the contracts with numbers 1 (highest rank), 2 (second highest) etc. Additionally, we let any firm in $\{a_v, a'_v, b_v, b'_v\}$ prefer all contracts not in Z to contracts in Z.

We claim that there exists an outcome in X admitting neither blocking paths nor blocking cycles if and only if the vertices of D can be partitioned into two acyclic sets.

"⇒": Let us suppose that there exists an outcome $A \subseteq X$ that does not admit any blocking paths or cycles.

We show that $Z \cap A = \emptyset$. To see this, first consider a contract $a'_u a_v \in Z$, and suppose for contradiction that $a'_u a_v \in A$. Since a_v has only one downstream contract $a_v a'_v$, this means that the contract $s_v a_v$ cannot be contained in A (because of the Kirchhoff equality for a_v). Note also that $s_v a_v$ is (A, a_v)-acceptable, because it is a contract preferred by a_v to $a'_u a_v$. Consider the path P from s through s_v to a_v. Clearly, if neither contract on P is in A, then it is a blocking path, otherwise the contract $s_v a_v$ is (A, s_v)-acceptable and hence a blocking path itself, a contradiction. So suppose now $b'_u b_v \in A$. Arguing analogously as before, we can prove that either the path from b'_u through t_u to t, or simply the contract $b'_u t_u$ is blocking. Thus we obtain that A cannot contain any contracts from Z, and only contains contracts within node gadgets and contracts where the seller or the buyer is the terminal agent s or the terminal agent t, respectively.

Therefore, we know that for each $v \in V$, at most one of the contracts $a_v a'_v$ and $b_v b'_v$ can be contained in A (since s_v can choose at most one of its downstream contracts by the Kirchhoff equality). Let $Q = \{v \in V \mid a_v a'_v \notin A\}$, and let $R = V \backslash Q$; clearly $b_v b'_v \notin A$ for any $v \in R$. It is not hard to see that both Q and R are acyclic. Indeed, any cycle within vertices of Q in D corresponds to a cycle using only contracts of

Z and the contracts $a_v a'_v$, $v \in Q$, and such a cycle cannot exists as it would block A. The same argument works to show the acyclicity of R, proving the first direction of our reduction.

"⇐": Assume now that Q and R are two acyclic subsets of V forming a partition. We define an outcome $A \subseteq X$ that contains the contracts ss_v, $s_v b_v$, $b_v b'_v$, $b'_v t_v$, and $t_v t$ for each $v \in Q$, and similarly, the contracts ss_v, $s_v a_v$, $a_v a'_v$, $a'_v t_v$, and $t_v t$ for each $v \in R$. We claim that there is no blocking path or cycle for A.

To see this, observe that by the Kirchhoff equality a contract that is $(A, \cdot)$-acceptable in itself for some firm (but is not contained in A) must be either $s_u a_u$ for some $u \in Q$ or $b'_v t_v$ for some $v \in R$. Since there is no path starting with a contract $s_u a_u$ and ending with a contract $b'_v t_v$, we know that no path can block A.

To show that A admits no blocking cycles, we simply use that blocking cycles must be disjoint from A. Note that any cycle has to use at least one contract of Z, as node gadgets are acyclic. We partition the contracts in Z into four sets as follows; recall that $A \cap Z = \emptyset$. Let $Z_{Q,R}$ denote those contracts in Z that leave a node gadget D_u with $u \in Q$ and arrive at a node gadget D_v with $v \in R$; we define $Z_{R,Q}$, $Z_{Q,Q}$, and $Z_{R,R}$ analogously. To see that no contract $xy \in Z_{Q,R} \cup Z_{R,Q}$ can be part of a blocking cycle for A, note that A contains either the unique upstream contract of x or the unique downstream contract of y, by the definition of A. In either case, any cycle that contains the contract xy must also contain a contract in A, and hence cannot be a blocking cycle.

By the same reasoning, no contract $b'_u b_v$ for some $u, v \in Q$ can be contained in a blocking cycle, since both the unique upstream contract of b'_u and the unique downstream contract of b_v are contained in A. Therefore, any contract of $Z_{Q,Q}$ used by a blocking cycle must be of the form $a'_u a_v$ for some $u, v \in Q$. Similarly, any contract of $Z_{R,R}$ used by a blocking cycle must be of the form $b'_u b_v$ for some $u, v \in R$. By the structure of the network, this implies that no cycle can use contracts both from $Z_{Q,Q}$ and from $Z_{R,R}$. However, any blocking cycle that uses only contracts in $Z_{Q,Q}$ and contracts of the form $a_u a'_u$ with $u \in Q$ directly corresponds to a cycle within $D[Q]$. Similarly, any blocking cycle using only contracts in $Z_{R,R}$ and contracts of the form $b_u b'_u$ with $u \in R$ yields a cycle within $D[R]$. Therefore, the acyclicity of Q and R ensures that A admits no blocking cycle, proving the correctness of our reduction. □

If we translate Theorem 2 into the language of *stable flows* introduced by Fleiner (2014), we obtain that it is NP-complete to decide whether a *completely stable flow* exists in a given network with preferences, where a flow is completely stable if it admits neither blocking paths nor blocking cycles. In fact, our statement holds not only in the discrete case (as implied directly by Theorem 2), but also in the continuous case where the flow can take real values as well; adjusting the proof of Theorem 2 to this case is straightforward. Hence we settle a conjecture posed by Fleiner (2014).

3.3 Stability

We now relax the assumption that blocking sets must be paths or cycles and consider general set blocks in trading networks (Hatfield et al. 2021) and flow networks (Fleiner 2014).

Definition 8 An outcome $A \subseteq X$ is *stable* if:
1. A is acceptable.
2. There exists no non-empty set of contracts $Z \subseteq X$, such that $Z \cap A = \emptyset$ and Z is (A, f)-acceptable for all $f \in F(Z)$; such sets are called *blocking*.

Stable outcomes are immune to deviations by arbitrary groups of firms, which can re-contract freely among themselves while keeping any of their existing contracts. Stable outcomes always exist in acyclic networks if choice functions are fully substitutable. However, Fleiner (2014) and Hatfield and Kominers (2012) showed that stable outcomes may not exist in general trading networks.

3.3.1 Stability in flow networks

We next prove that in flow networks path-or-cycle-stable outcomes coincide with stable outcomes.

Theorem 3 *In a flow network an outcome is path-or-cycle-stable if and only if it is stable.*

Proof Let X be a set of firms in a flow network. Using the definitions, it is immediate that a stable outcome A is also path-or-cycle-stable, as a blocking path or cycle is naturally a blocking set as well.

For the opposite direction, assume that A is a path-or-cycle-stable outcome. Towards contradiction, let us also assume that $B \subseteq X$ is a blocking set for A. Suppose first that B contains a cycle B_C. Then B_C is disjoint from A because $B_C \subseteq B \subseteq X \backslash A$. Moreover, B_C is (A, f)-acceptable for any firm f: if f is a terminal then it accepts all contracts offered, and if f is non-terminal then, by the Kirchhoff equality, it accepts B_C alongside with A since B_C^f is either empty or it contains an upstream and a downstream contract for f. Hence, B_C is a blocking cycle for A. This proves that B cannot contain any cycles. Let us now consider a path $P = (x_1, \ldots, x_p)$ in B that is maximal (in the sense that no contracts can be added to P to obtain a longer path). By the acyclicity of B, $s := s(x_1)$ must be a firm with no upstream contracts in B, and $t := b(x_p)$ must be a firm with no downstream contracts in B.

Since B is blocking for A, we know that B is (A, s)-acceptable. Recall that s has a flow-based choice function. Therefore either s is a terminal firm (always accepting every offered contract), or s must obey the Kirchhoff inequality, and therefore can accept the downstream contract $x_1 \in X_S^s \cap B$ only if there is a less preferred downstream contract in A. Note that this means that the contract x_1 is $(A, s(x_1))$-acceptable.

Using the fact that B is (A, t)-acceptable, we can argue analogously to show that the contract x_p is $(A, b(x_p))$-acceptable. Finally, consider any intermediary firm f lying on path P; assume that $f = b(x_i) = s(x_{i+1})$ for some $i \in \{1, \ldots, p-1\}$. Again, either f is terminal (and thus accepts all offers) or it is non-terminal and obeys the Kirchhoff equality. In the latter case, the path P is (A, f)-acceptable because it contains exactly one upstream contract and one downstream contract for f, namely x_i and x_{i+1}. Hence we get that P is a blocking path for A, a contradiction. □

Theorems 2 and 3 imply that deciding whether a stable outcome exists in a flow network is NP-complete.

Corollary 1 *It is* NP*-complete to decide if a flow network admits a stable outcome.*

3.3.2 Stability in trading networks

We finally turn to the existence of stable outcomes in trading networks. Hatfield et al. (2021) showed that stability is closely related to a solution concept similar to trail stability. They define a *chain* as a set of contracts that can be ordered in a way that it forms a trail. Chain-stable outcomes are immune to blocks that form a chain in which every firm can simultaneously offer and accept all its contracts in the chain. Hatfield et al. (2021) show that, under additional conditions, stability is equivalent to *chain stability*. Chain stability contrasts to our definition of trail stability, which requires that firms only need to accept pairs of contracts along the trail, one pair at a time. Since a set of contracts that forms a path or a cycle is a chain, path-or-cycle stability is weaker than chain stability. Indeed, it is easy to see that in general trading networks path-or-cycle stability does not imply stability. But since flow networks are a special case of the trading networks (because flow-based choice functions are fully substitutable and satisfy the IRC condition), Corollary 1 implies that deciding whether a stable outcome exists is NP-hard in our trading networks model.

Corollary 2 *It is* NP*-hard to decide if a trading network admits a stable outcome, even if choice functions satisfy full substitutability and IRC conditions.*

In fact, dealing with stable outcomes is even trickier in trading networks. Our last result demonstrates that even *verifying* whether an outcome is stable is computationally intractable in general trading networks.[11]

Let INSTABILITY be the following decision problem. An instance of INSTABILITY is a trading network with a set X of contracts and a set F of agents with choice functions that satisfy the conditions of full substitutability and IRC, and an outcome $A \subseteq X$. The answer for an instance of INSTABILITY is YES if the

[11] The proof of Theorem 2 shows that in flow networks verifying whether an outcome is stable can be done in time linear in the number of contracts.

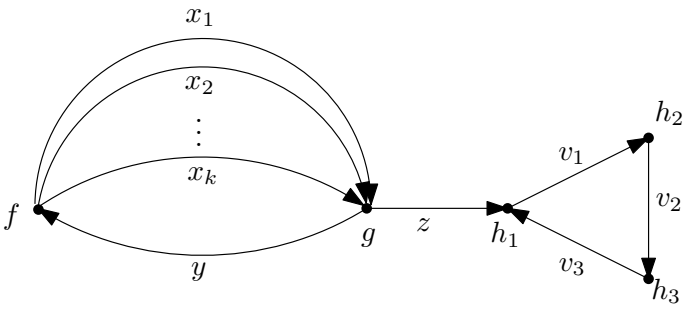

Fig. 2 Illustration of the trading network constructed in the proof of Theorem 4

particular outcome A is *not* stable (that is, if there is a set Z of contracts that blocks A), otherwise the answer is NO.

Theorem 4 *The* INSTABILITY *problem is* NP*-complete*[12] *even in the special case where the input consists of a trading network N and an outcome A in N such that either A is the only stable outcome in N, or there exists no stable outcome in N. Moreover, if choice functions are represented by oracles, then finding the right answer for an instance of* INSTABILITY *requires an exponential number of oracle calls in the worst case.*

Proof of Theorem 4 We present separate proofs for the theorem's two statements.

Proving the first statement. The INSTABILITY problem clearly belongs to the complexity class NP, as we can verify in polynomial time that a given set Z of contracts is a blocking set for an outcome A.

To show that INSTABILITY is NP-hard we reduce the NP-complete PARTITION problem to INSTABILITY. An instance of the PARTITION problem is given by a k-tuple $\Pi = (a_1, a_2, \ldots, a_k)$ of positive integers such that $a_1 \leq a_2 \leq \ldots \leq a_k$ holds. The answer to this problem is YES if and only if there is a subset I of $\{1, 2, \ldots, k\}$ such that $\sum_{i \in I} a_i = s$ where $2s = \sum_{i=1}^{k} a_i$. □

Intuition. In what follows, we will define a trading network with five agents, where certain contracts between two agents, f and g (see Fig. 2 for an illustration), will correspond to the integers in Π in a bijective manner. The main idea is that f and g can simultaneously engage in trade only if they are able to select, from all contracts corresponding to Π, a subset of contracts whose corresponding integers sum up to s, yielding a solution to Π. Hence, the ability of f and g to trade with each other depends on the solvability of our PARTITION instance Π. To obtain a reduction to INSTABILITY, we define an outcome in which only the remaining three vertices of the trading network

[12] Equivalently, the problem of verifying the stability of a given outcome is co-NP-complete.

participate and in which f and g do *not* trade; such an outcome is stable exactly if the Partition instance Π is *not* solvable, meaning that f and g are not able to trade.

Reduction. Given our instance Π of Partition, we define an instance of Instability as follows. First, we construct a trading network N which is based on an example by Fleiner (2014). The trading network N has agent set $\{f, g, h_1, h_2, h_3\}$ and contract set $\{x_1, x_2, \ldots, x_k, y, z, v_1, v_2, v_3\}$. The sellers and buyers of these contracts are as shown in Figure 2. Let h_1, h_2, and h_3 have flow-based choice functions as defined for non-terminal agents, with h_1 preferring z to v_3. Next, we define the choice function C^f_Π for f with the help of $s = \frac{1}{2}\sum_{i=1}^{k} a_i$ by

$$C^f_\Pi(Y|X) = \begin{cases} (\emptyset|\emptyset) & \text{if } Y = \emptyset, \\ (Y|X) & \text{if } Y = \{y\} \text{ and } \sum\{a_i : x_i \in X\} \leq s, \\ (Y|X \cap \{x_1, x_2, \ldots, x_t\}) & \text{if } Y = \{y\} \text{ and t is such that} \\ & \sum\{a_i : x_i \in X, i \leq t\} \leq s < \sum\{a_i : x_i \in X, i \leq t+1\} \end{cases}$$

for any $X \subseteq \{x_1, \ldots, x_k\}$ and $Y \subseteq \{y\}$.

One can readily check that C^f_Π satisfies the IRC condition. To see that it is also fully substitutable, first notice that f never rejects any upstream contracts, so it satisfies the conditions of both SSS and CSC with f as a buyer (that is, conditions 1(a) and 2(a) in Definition 1). To check the requirements for same-side substitutability with f taking the role of a seller (that is, condition 1(b) in Definition 1), let us fix a set Y of upstream contracts. If $Y = \emptyset$, then f rejects all downstream contracts. Otherwise (that is, if $Y = \{y\}$), suppose that f rejects some x_j from a set X of offered downstream contracts. This means that there exists an index $t < j$ such that $\sum\{a_i : x_i \in X, i \leq t\} \leq s < \sum\{a_i : x_i \in X, i < t+1\}$. But then, for any superset $X' \supseteq X$ of downstream contracts offered to f, the same condition will hold for some $t' \leq t < j$, and thus x_j will again be rejected. This proves same-side substitutability with f being a seller. To verify that C^f_Π also satisfies cross-side complementarity with f as a seller (that is, condition 2(b) in Definition 1), it suffices to observe that any downstream contract rejected while $Y = \{y\}$ is offered to f will get rejected again when $Y' = \emptyset$ is offered to f. Hence, we get that C^f_Π is fully substitutable.

Next, define C^g_Π by

$$C^g_\Pi(X|W) = \begin{cases} (X|W) & \text{if } \sum\{a_i : x_i \in X\} \geq s, \\ (X|\emptyset) & \text{if } \sum\{a_i : x_i \in X\} < s \end{cases}$$

for any $X \subseteq \{x_1, \ldots, x_k\}$ and $W \subseteq \{y, z\}$.

It is straightforward to check that C^g_Π satisfies the condition of IRC, so let us check whether it is fully substitutable as well. First, since g always accepts all upstream contracts, both SSS and CSC clearly hold for g as a buyer. To check SSS as a seller for g, let us consider a fixed set X of upstream contracts offered for g. Then g either accepts all downstream contracts (in case X is such that $\sum\{a_i : x_i \in X\} \geq s$) or it rejects every downstream contract (otherwise); in either case, the set of contracts rejected by g as a seller satisfies the conditions of SSS. To check cross-side complementarity as a seller for g, note that if g rejects a set $W \subseteq \{y, z\}$ of contracts, then it must be the case that

$\sum\{a_i : x_i \in X\} < s$ for the offered set X of upstream contracts. But then g will reject W for any subset $X' \subseteq X$ too, so CSC holds as well. Hence, C^g_Π is fully substitutable.

Note that for any set of downstream contracts accepted by f, the corresponding set of integers in Π sum up to *at most* s; by contrast, agent g accepts a nonempty set of downstream contracts only if it receives a set of upstream contracts for which the corresponding set of integers in Π sum up to *at least* s. This will result in the key property of this trading network to "recognize" whether there is a set of elements in Π summing up to *exactly* s. Such recognition will be manifested here by the stability or instability of a certain outcome.

So far, based on our instance Π of Partition, we have determined a trading network N. To finish the construction of our Instability instance, we set an outcome $A = \{v_1, v_2, v_3\}$.

Correctness of the reduction. We have to show that the answer to our instance of the Partition problem is YES if and only if A is not stable, and furthermore, that N and A satisfy the conditions claimed in the theorem. To this end, we first prove the following claim.

Claim 1 *If $Z \subseteq \{x_1, x_2 \dots, x_k, y, z\}$ is a set of contracts acceptable for f and g, then either $Z = \emptyset$ or $\sum\{a_i : x_i \in Z\} = s$.*

Proof of Claim 1 Define $I = \{i : x_i \in Z\}$ and $X_I := \{x_i : x_i \in Z\}$. Since Z is acceptable for f and g, we have $C^f_\Pi(Z \cap \{y\}|X_I) = (Z \cap \{y\}|X_I)$ and $C^g_\Pi(X_I|Z \cap \{y, z\}) = (X_I|Z \cap \{y, z\})$. If $Z \cap \{y, z\} = \emptyset$, then $(Z \cap \{y\}|X_I) = C^f_\Pi(Z \cap \{y\}|X_I) = C^f_\Pi(\emptyset|X_I) = (\emptyset, \emptyset)$ implies $X_I = \emptyset$, which yields $Z = \emptyset$. If $Z \cap \{y, z\} \neq \emptyset$, then $(X_I|Z \cap \{y, z\}) = C^g_\Pi(X_I|Z \cap \{y, z\}) \neq (X_I|\emptyset)$ which implies $\sum_{i\in I} a_i \geq s$. As a consequence, $X_I \neq \emptyset$. Using this, we get $C^f_\Pi(Z \cap \{y\}|X_I) = (Z \cap \{y\}|X_I) \neq (\emptyset|\emptyset)$ which implies $\sum_{i\in I} a_i \leq s$. Consequently $\sum_{i\in I} a_i = s$. □

Assume now that the answer to our instance Π of Partition is YES, that is, $\sum_{i\in I} a_i = s$ for some $I \subseteq \{1, 2, \dots, k\}$. Define $X_I := \{x_i : i \in I\}$ and $W = \{y, z\}$. By definition, $C^f_\Pi(\{y\}|X_I) = (\{y\}|X_I)$, $C^g_\Pi(X_I|W) = (X_I|W)$, and since h_1 prefers z to v_3, h_1 chooses $\{z, v_1\}$ from $A \cup W$. Hence $X_I \cup W$ blocks A, so A is not stable.

For the other direction, assume now that A is not stable. This means that there is a blocking set Z for A. Clearly, Z must be acceptable for f and g (because $A^f = A^g = \emptyset$). Moreover, since $A \cap Z = \emptyset$, we know that $Z \subseteq \{x_1, x_2, \dots, x_k, y, z\}$. Hence, Claim 1 implies that $\sum\{a_i : x_i \in Z\} = s$, and the answer to the Partition problem is YES.

It remains to show that either there exists no stable outcome in N, or A is the unique stable outcome in N. Let S be any stable outcome in N. Now, if $S \cap \{x_1, x_2, \dots, x_k, y, z\} = \emptyset$, then since each agent in $\{h_1, h_2, h_3\}$ must choose the same number of upstream and downstream contracts, we know that either $S = \emptyset$, or $S = A$. It is easy to see that A blocks the empty outcome, so only $S = A$ is possible in this case. Suppose now $z_S := S \cap \{x_1, x_2, \dots, x_k, y, z\} \neq \emptyset$. Using again that agents h_1, h_2 and h_3 have flow-based choice functions as defined for non-terminal

vertices, we get that $z \notin S$. On the one hand, if $A \subseteq S$, then $\{z\}$ blocks S: h_1 prefers z to $v_3 \in A$, and by applying Claim 1 to Z_S we know $\sum\{a_i : x_i \in S\} = s$ which implies $z \in C^g_\Pi(S \cup \{z\})$. On the other hand, if $A \nsubseteq S$, then S must be disjoint from A by the flow-based choice functions of agents h_1, h_2 and h_3. But then A blocks S, proving that A is indeed the only outcome that can be stable.

Proving the second statement. To prove the second part of the theorem, define a trading network with agents f and g and with contracts y and x_i such that $f = b(y) = s(x_i)$ and $g = s(y) = b(x_i)$ for $1 \le i \le 2n$. Define the following choice function for f:

$$C^f(Y|X) = \begin{cases} (\emptyset|\emptyset) & \text{if } Y = \emptyset, \\ (Y|X) & \text{if } Y = \{y\} \text{ and } |X| \le n, \\ (Y|X \cap \{x_1, x_2, \dots, x_t\}) & \text{if } Y = \{y\} \text{ and } |\{x_i \in X : i \le t\}| = n. \end{cases} \tag{1}$$

As $C^f = C^f_\Pi$ for $\Pi = (1, 1, \dots, 1)$, C^f satisfies the conditions of full substitutability and IRC.

Next, define the following choice function

$$C^g_0(X|Y) = \begin{cases} (X|Y) & \text{if } |X| \ge n+1, \\ (X|\emptyset) & \text{if } |X| \le n. \end{cases}$$

For any $I \subseteq \{1, 2, \dots, 2n\}$ with $|I| = n$, define $X_I := \{x_i : i \in I\}$ and let

$$C^f_I(X|Y) = \begin{cases} (X|Y) & \text{if } |X| \ge n+1 \text{ or if } X = X_I, \\ (X|\emptyset) & \text{if } |X| \le n \text{ and } X \ne X_I. \end{cases}$$

It is straightforward to check that choice functions C^g_0 and C^g_I above are also fully substitutable and satisfy the condition of IRC.

Now assume that an instance of Instability is given by the above network and an outcome $A = \emptyset$. Assume that the choice functions are not given explicitly, but by oracles. Moreover, we know exactly that the choice function of f is the one defined in (1) and we know that the choice function of g is either C^g_0 or C^g_I for some I. It is easy to check that A is not stable if and only if $C^g = C^g_I$ for some I, and in this case the only blocking set is $Z = X_I \cup \{y\}$. So if one has to decide stability of A, then one must determine the $C^g(Z)$ values for all such possible Z, and this means $\binom{2n}{n}$ oracle calls. □

Let us stress that in the Instability problem we assume that every choice function is fully substitutable. This assumption allows us to concentrate on the computational intractability of the Instability problem itself since we do not have to take into account the time necessary to recognize whether agents have fully substitutable choice functions. Therefore, the second statement of Theorem 4 is unrelated to an important result by Hatfield et al. (2012) which shows that any algorithm that can decide if some agent f has a substitutable choice function C^f must perform an expected number of oracle calls accessing C^f that is exponential in the number of contracts. Analogously, the statement of Corollary 2 is also unrelated to testing substitutability of choice functions.

References

Abdulkadiroğlu A, Sönmez T (2003) School choice: a mechanism design approach. Am Econ Rev 93(3):729–747

Adachi H (2017) Stable matchings and fixed points in trading networks: a note. Econ Lett 156:65–67

Alkan A (2002) A class of multipartner matching markets with a strong lattice struture. Econ Theory 19(4):737–746

Aygün O, Sönmez T (2013) Matching with contracts: comment. Am Econ Rev 103(5):2050–2051

Blair C (1988) The lattice structure of the set of stable matchings with multiple partners. Math Oper Res 13(4):619–628

Bokal D, Fijavž G, Juvan M, Kayll PM, Mohar B, (2004) The circular chromatic number of a digraph. J Graph Theory 46(3):227–240

Candogan O, Epitropou M, Vohra, RV (2016) Competitive equilibrium and trading networks: A network flow approach. In: EC '16: Proceedings of the 2016 ACM Conference on Economics and Computation, pp. 701–702

Cook SA (1971) The complexity of theorem-proving procedures. In: Proceedings of the Third Annual ACM Symposium on Theory of Computing, STOC '71, New York, NY, USA, pp. 151–158. Association for Computing Machinery

Cormen TH, Leiserson CE, Rivest RL, Stein C (2009) Introduction to Algorithms, Third Edition (3rd ed). The MIT Press

Delacrétaz D (2019) January Stability in matching markets with sizes

Delacrétaz D, Kominers SD, Teytelboym A (2019) Matching mechanisms for refugee resettlement

Echenique F (2007) Counting combinatorial choice rules. Games Econ Behav 58(2):231–245

Echenique F, Oviedo J (2006) A theory of stability in many-to-many matching markets. Theor Econ 1(1):233–273

Fleiner T (2003) A fixed-point approach to stable matchings and some applications. Math Oper Res 28(1):103–126

Fleiner T (2014) On stable matchings and flows. Algorithms 7:1–14

Fleiner T, Jankó Z, Tamura A, ATeytelboym (2016) Trading networks with bilateral contracts. Working paper, Oxford University

Fleiner T, Jagadeesan R, Jankó Z, Teytelboym A (2019) Trading networks with frictions. Econometrica 87(5):1633–1661

Fox JT (2017) Specifying a structural matching game of trading networks with transferable utility. Am Econ Rev 107(5):256–60

Gale D, Shapley LS (1962) College admissions and the stability of marriage. Am Math Mon 69(1):9–15

Garey M, Johnson DS (1979) Computers and Intractability: a guide to the theory of NP-completeness. W. H Freeman and Company, New York

Gonczarowski YA, Nisan N, Ostrovsky R, Rosenbaum W (2019) A stable marriage requires communication. Games Econ Behav 118:626–647

Hatfield JW, Kominers SD (2012) Matching in networks with bilateral contracts. Am Econ J Microecon 4(1):176–208

Hatfield JW, Kominers SD (2014) Hidden substitutes. Technical report, Mimeo

Hatfield JW, Immorlica N, Kominers SD (2012) Testing substitutability. Games Econ Behav 75(2):639–645

Hatfield JW, Kominers SD, Nichifor A, Ostrovsky M, Westkamp A (2013) Stability and competitive equilibrium in trading networks. J Polit Econ 121(5):966–1005
Hatfield JW, Kominers SD, Nichifor A, Ostrovsky M, Westkamp A (2021) Chain stability in trading networks. Theor Econ 16(1):197–234
Klaus B, Walzl M (2009) Stable many-to-many matchings with contracts. J Math Econ 45(7–8):422–434
McDermid EJ, Manlove DF (2010) Keeping partners together: algorithmic results for the hospitals/residents problem with couples. J Combinat Optim 19(3):279–303
Morstyn T, Teytelboym A, McCulloch MD (2018) Bilateral contract networks for peer-to-peer energy trading. IEEE Trans Smart Grid 10(2):2026–2035
Ostrovsky M (2008) Stability in supply chain networks. Am Econ Rev 98(3):897–923
Papadimitriou CH (2007) The complexity of finding nash equilibria. In: Nisan N, Roughgarden T, Éva T, Vazirani VV (eds) Algorithmic Game Theory. Cambridge University Press
Roth A (1984) The evolution of the labor market for medical interns and residents: a case study in game theory. J Polit Econ 92(6):991–1016
Roughgarden T (2010) Computing equilibria: a computational complexity perspective. Econ Theory 42(1):193–236
Sönmez T, Switzer T (2013) Matching with (branch-of-choice) contracts at the United States Military Academy. Econometrica 81(2):451–488
Sotomayor M (1999) Three remarks on the many-to-many stable matching problem. Math Soc Sci 38:55–70
Westkamp A (2010) Market structure and matching with contracts. J Econ Theory 145(5):1724–1738

Authors and Affiliations

Tamás Fleiner[1,2] · Zsuzsanna Jankó[2,3] · Ildikó Schlotter[1,2] · Alexander Teytelboym[4]

Tamás Fleiner
fleiner@cs.bme.hu

Zsuzsanna Jankó
zsuzsanna.janko@uni-corvinus.hu

Ildikó Schlotter
schlotter.ildiko@krtk.hu

1 Budapest University of Technology and Economics, Budapest, Hungary

2 Institute of Economics, Centre for Economic and Regional Studies, Budapest, Hungary

3 Corvinus University of Budapest, Budapest, Hungary

4 University of Oxford, Oxford, UK