Stability in Barter Exchange Markets

Sushmita Gupta
Department of Informatics, University of Bergen
Bergen, Norway
sushmita.gupta@ii.uib.no

Fahad Panolan
Department of Informatics, University of Bergen
Bergen, Norway
fahad.panolan@ii.uib.no

Saket Saurabh
The Institute of Mathematical Sciences, HBNI
Chennai, India
Department of Informatics, University of Bergen
Bergen, Norway
saket@imsc.res.in

Meirav Zehavi
Ben-Gurion University of the Negev
Beersheba, Israel
meiravze@bgu.ac.il

ABSTRACT

The notion of stability is the foundation of several classic problems in economics and computer science that arise in a wide-variety of real-world situations, including Stable Marriage, Stable Roommate, Hospital Resident and Group Activity Selection. We study this notion in the context of barter exchange markets. The input of our problem of interest consists of a set of people offering goods/services, with each person subjectively assigning values to a subset of goods/services offered by other people. The goal is to find a *stable transaction*, a set of cycles that is *stable* in the following sense: there does not exist a cycle such that every person participating in that cycle prefers to his current "status". For example, consider a market where families are seeking vacation rentals and offering their own homes for the same. Each family wishes to acquire a vacation home in exchange of its own home without any monetary exchange. We study such a market by analyzing a stable transaction of houses involving cycles of fixed length. The underlying rationale is that an entire trade/exchange fails if any of the participating agents cancels the agreement; as a result, shorter (trading) cycles are desirable.

We show that given a transaction, it can be verified whether or not it is stable in polynomial time, and that the problem of finding a stable transaction is NP-hard even if each person desires only a small number of other goods/services. Having established these results, we study the problem of finding a stable transaction in the framework of parameterized algorithms.

KEYWORDS

Algorithm Design; Stability; Barter Exchange; FPT

ACM Reference Format:
Sushmita Gupta, Fahad Panolan, Saket Saurabh, and Meirav Zehavi. 2018. Stability in Barter Exchange Markets. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 9 pages.

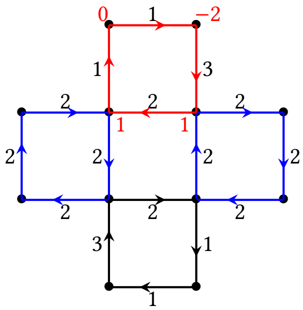
Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.

1 INTRODUCTION

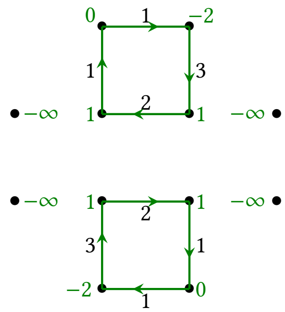
A stable assignment as a solution concept was formally studied by Gale and Shapley [8] in terms of Stable Marriage, a famous problem in economics, computer science and mathematics. The input of Stable Marriage consists of a set of people, each person strictly ranking a subset of people of the opposite sex. The goal is to find a matching (a set of men-women pairs, no two of which share a common member) that is *stable* in the following sense: there does not exist man-woman pair whose members prefer being matched to each other over their current "status". Here, every person prefers being matched to being unmatched. Algorithms relying on this notion of stability are routinely employed to handle a wide-variety of real-world situations, ranging from the assignment of graduating medical students to their first hospital appointments to the allocation of servers in a large distributed Internet service. Consequently, several books are dedicated to the study of Stable Marriage and its varied applications [9, 17, 18].

Our Model. We extend the idea of stability from pairs of agents to groups of agents. Informally, the input of our problem of interest, called Stable Barter, consists of a set of agents offering goods/services, each agent subjectively assigning values to a subset of goods/services offered by other agents. The goal is to find a set of cycles that is *stable* in the following sense: there does not exist a cycle which every agent prefers to its current "status". Here, every agent prefers to participate in a trade, rather than not. Let us also formally define Stable Barter in graph-theoretic terms, and then discuss the motivation behind the present formulation. We denote an instance of Stable Barter by (G, w, ℓ) where G is a digraph, $w : A(G) \to \mathbb{N}_0$ is a cost function, and $\ell \in \mathbb{N}$. A cycle in G on at most ℓ vertices is called an *exchange*. For an exchange X and a vertex v in X, we use $\text{pay}_X(v)$ and $\text{gain}_X(v)$ to denote the costs of the unique outgoing and incoming arcs of v in X, respectively. [1] The balance of v in X is given by $\text{balance}_X(v) = \text{gain}_X(v) - \text{pay}_X(v)$. A *transaction* $\mathscr{C}$ is a set of pairwise vertex-disjoint exchanges, where for any vertex v that is part of some exchange X in $\mathscr{C}$, we define $\text{balance}_{\mathscr{C}}(v) = \text{balance}_X(v)$, and otherwise $\text{balance}_{\mathscr{C}}(v) = -\infty$. A transaction $\mathscr{C}$ is said to be *stable* if there is no *blocking exchange* Y (a cycle of length at most ℓ

[1] Note that for a fixed exchange X, there is exactly one outgoing arc and exactly one incoming arc incident on every vertex in X.



(a) The set $\mathscr{C}$ of blue colored cycles is an example of transaction which is not stable due to red colored cycle X. $\text{balance}_{\mathscr{C}}(v) = 0$ for all v participating in the transaction $\mathscr{C}$. The balance of each vertex in the exchange X is marked with red colored digits in the figure.



(b) The set $\mathscr{C}$ of green colored cycles is an example of a stable transaction. The balance of each vertex in $\mathscr{C}$ is marked with green colored digits in the figure.

Figure 1: The graph in the figure (on the left side) with the weight function mentioned on the arcs and $\ell = 4$ is an instance of STABLE BARTER.

in G) such that for every vertex v in Y, $\text{balance}_Y(v) > \text{balance}_{\mathscr{C}}(v)$, that is $\text{gain}_Y(v) - \text{pay}_Y(v) > \text{gain}_X(v) - \text{pay}_X(v)$ where X is the exchange in $\mathscr{C}$ that contains v. The objective of the STABLE BARTER problem is to find a stable transaction. See Figure 1 for an illustration. Our definition of stability is similar to the notion of super stability mentioned in [15] and it is a stronger notion of stability compared with the strict stability mentioned in [21], where in a blocking exchange at least one of the agent *strictly* prefers the blocking exchange over the existing one.

Use of negative values. The balance values are "relative": positive and negative values do not represent absolute gain and loss. Negative values do not discourage agents from participating in a transaction. In particular, agents would only be incident to arcs they find acceptable. At first glance, it may look counter-intuitive that both contributor and receiver agree on the valuation of a tradable object, i.e. the contributor's "pay-value" and receiver's "gain-value" are the same. However, using a simple trick in the construction of the graph, we can incorporate situations where the receiver and donor do not agree. For ease of exposition, we will refer to the above as Model A and the following as Model B. Formally, in Model B we model situations where the contributor and receiver differ on their valuation of the trade with each other as follows. We are given a digraph D, and two cost functions $w_p, w_g : A(G) \to \mathbb{N}_0$ such that $w_p(uv)$ represents u's evaluation of the trade with v, and $w_g(uv)$ represents v's evaluation of the trade with u. For an exchange X, let uv, vw be arcs in X. Then, we define $\text{balance}_X(v) = w_g(uv) - w_p(vw)$.

Next, we show how to reduce STABLE BARTER in Model B to STABLE BARTER in Model A, and thereby exhibit that Model A is just as general as Model B. Given $\mathcal{I} = (D, \{w_p, w_g\}, \ell)$, an instance of STABLE BARTER in Model B, we create an instance $\mathcal{J} = (D', \{w\}, 2\ell)$ of STABLE BARTER in Model A as follows. For any arc $uv \in A(G)$ (with contributor u and receiver v), we add a dummy vertex x to split the arc u, v, such that ux is now an arc outgoing from u and xv is an arc incoming into v; then, we set $w(ux) = w_p(uv)$ and $w(xv) = w_g(uv)$. Note that there is a solution for STABLE BARTER in $\mathcal{I}$ if and only if there is a solution for STABLE BARTER in $\mathcal{J}$. Later in this section, we describe scenarios which are explicitly captured by STABLE BARTER in Model B.

Applications. The STABLE BARTER problem can model scenarios where transactions eschew financial exchanges for goods/services, the most obvious ones being where there is a humanitarian or altruistic consideration at stake. Here, KIDNEY EXCHANGE is perhaps a prototype problem. The context is that there are many patients with kidney disorders who would benefit from a transplantation of a healthy kidney, and each has a willing donor (e.g. a friend/family member) who is not medically compatible. The goal is to find a set of patients (with incompatible donors) who can exchange their donors among each other so that every patient is matched with a compatible donor. The common rule in a cycle of transplantations is that either all of them are carried out, or none. This necessitates that the length of the cycle be small, since there is always a chance that one of the members (be it a patient or a donor) backs out from an agreed upon exchange. In addition, the last argument also clearly motivates the demand that a transaction would be stable. Academic research on KIDNEY EXCHANGE has a long and varied history; we refer the reader to [1] for an extensive list as well as for a study of KIDNEY EXCHANGE in barter exchange markets.

In addition to kidney exchange, and organ donation in general, there are more traditional objects that are transacted in a similar non-monetary fashion. For example, we can point to vacation rentals [13], books [20], shoes [19] as tradable objects that have their own dedicated exchange markets. In particular, Intervac [13], the first home exchange network founded in 1953, provides a platform for home exchange for staying in holidays.

Our work, both the theoretical model and the results obtained for the STABLE BARTER problem, are applicable to these as well as to the KIDNEY EXCHANGE problem as follows. We set $w_p(uv) = 0$ representing the fact that contributor u (i.e "donor" in this case) is indifferent to who receives his/her organ, whereas $w_g(uv)$ encodes the compatibility of v (the recipient) receiving u's organ. Consequently, KIDNEY EXCHANGE can be modeled by STABLE BARTER in Model B. Moreover, KIDNEY EXCHANGE generalizes the STABLE ROOMMATE problem, and thus the latter can also be modeled by STABLE BARTER in Model B.

Notice that in KIDNEY EXCHANGE, the cost function $w_p(\cdot)$ maps all the edges to 0, but there are scenarios for which both w_p and w_g are non-constant functions. For instance, consider a market in which a large number of families are looking for vacation rentals and offering their own homes for the same. Each family wishes to acquire a vacation home in exchange of its own home without any monetary exchange. We study such a market by analyzing a stable transaction of houses involving cycles of fixed length. In this scenario, an arc uv represents an agent u who has a house in which agent v is interested for his/her vacation. Then, $w_p(uv)$ represents u's evaluation of v's suitability (for example: property

Problem	Complexity		Implication on Stable Barter	
	Existence	Max	Existence	Max
SMTI	P [8, 14]	NPC [14]		NPC for $\ell = 4$
SE(3,3)	NPH [2]	NPH [2]	NPH for $\ell = 6$	NPH for $\ell = 6$

Table 1: Here, P is the class of polynomial time solvable problems. NPH and NPC refer to NP-hard and NP-complete, respectively.

value, non-smoker, no-pets, and so on) and $w_g(uv)$ represents v's evaluation of u's property.[2]

Related Work. The Stable Marriage problem is among the most well-known matching problems with varied practical applications. A hard variant of this problem, called Stable Marriage with Ties and Incomplete Lists (SMTI), allows agents to submit incomplete preference lists with ties. Finding a stable matching is known to be polynomial-time solvable, but the finding a maximum sized stable matching is known to be NP-complete [14]. In recent years, the Stable Roommate problem, where there is only one type of agent (as opposed to two types in Stable Marriage), has been used to model the Kidney Exchange problem. Here also, each agent ranks a subset of other agents according to an order of preference. The same underlying idea applies to the Pairwise Kidney Exchange problem, where compatible donors are ranked by patients, and vice versa, in terms of the degree of compatibility, and a stable matching is desired. The Cycle Stable Roommate (CSR) problem, introduced in [16], extends the model of Pairwise Kidney Exchange as follows. Here, the goal is to find a matching M that admits no *blocking cycle*, defined to be a set (call it a *coalition*) of agents $(a_0, a_1, \ldots, a_{k-1})$ where $k \geq 2$, such that for each i, either (i) a_i is unassigned in M and finds a_{i+1} acceptable, or (ii) a_i prefers a_{i+1} to its partner assigned in M; all relations are computed modulo k. A matching is said to be a *cycle stable* matching, if it admits no blocking cycle. Irving [16] proved that deciding if a given instance of the Stable Roommate problem admits a cycle stable matching allocation is NP-complete, even when the cycle length is 3. A generalization of CSR was studied by Biró [2] as the *b-way stable ℓ- way exchange* (SE(b, ℓ)) problem where $b, l \in \mathbb{N}$; SE(3, 3) is proved to be NP-hard, even for deciding whether a solution exists. See Table 1 for the implications of known results for Stable Barter.

The notion of stability in the context of the Stable Marriage problem is inherently selfish when a outcome is viewed from the perspective of one side, men or women, but there are many real life scenarios in which varying degrees of cooperation is necessary to accomplish certain tasks. These types of applications are studied under the broad category of coalition games. Hedonic games and group activity selection are two settings under which cooperative games are studied. The former is a well-studied paradigm, while the latter is relatively new. In hedonic games, there are two basic notions of stability, one based on the individual and the other on the group. The latter notion corresponds to what is commonly known as *core stability*. Our definition of stable transaction is closely aligned with the standard notion of core stability in hedonic games. We remark that given the nature of applications modeled by the Stable Barter problem (altruistic and barter exchange of goods/services), individual-centric stability is taken for granted. This is because the nature of the exchange market ensures that no individual can unilaterally increase his/her payoff without doing the same for a larger group, which in our case is a cycle of length ℓ. For further references to core stability in Hedonic games, refer to [11] and [3, Ch. 15]. A recent paper [12] explores the computational complexity of the existence of core stable and Nash stable outcomes in the context of the graph-based Group Activity Selection problem (gGASP). These results are largely similar to the ones found in [11] for hedonic games, with subtle differences in the assignment of activities. This paper is of interest to us because it studies the parameterized complexity of this problem by showing the existence of FPT algorithms for computing a Nash stable outcome in trees, paths, stars, and small component graphs, as well as computing core stable solutions for small component graphs. A followup work [10] studies the complexity of computing a core stable solution in cliques, acyclic graphs, paths, stars, and small component graphs; the parameters are the number of activities and number of players.

[2]Note that u's evaluation of v's property is scalable based on u's evaluation of the various attributes of v's property.

Our Contribution. Our study encompasses various computational complexity issues concerning stability in barter exchange markets. Part of our contribution is conceptual: we generalize various stability models, while integrating explicit numerical valuations. Specifically, we present the following array of results for Stable Barter. Here, we denote an input instance for Stable Barter by (G, w, ℓ), where G is a directed graph, w is a weight function on the arc set of G and ℓ is the maximum length allowed in an exchange (cycle) in a transaction.

(i) A highly efficient and implementable dynamic programming based polynomial time verification protocol that takes as input a transaction and verifies if it is indeed stable. Our algorithm runs in time $O(\ell \cdot |A(G)|^2)$, where $A(G)$ denotes the arc set in G. Note that the exponent in the running time does not depend on ℓ.
(ii) An NP-completeness proof of the decision version of the Stable Barter problem, even when $\ell = 3$, the maximum degree (in-degree + out-degree) is at most 10 and the cost function assigns values from $\{1, 2, 3\}$, as well as an NP-completeness proof for testing whether there exists a transaction involving all the agents, even when $\ell = 3$, the maximum degree is at most 6 and the cost function assigns values from $\{1, 2\}$.
(iii) A W[1]-hardness proof of Stable Barter where the parameter is the number of exchanges in a transaction, i.e., it is unlikely that there is an algorithm for Stable Barter with time complexity $f(k) \cdot n^{O(1)}$ for any arbitrary function f, where k denotes the number of exchanges in a transaction and n is the total number of agents/vertices in the graph.
(iv) An algorithm of running time $\Delta^{O(k)} \cdot |A(G)|^{O(1)}$ to test whether there is a stable transaction involving k agents, where Δ is the maximum degree of the graph and $A(G)$ is the arc set of G. In other words, Stable Barter is fixed parameter tractable when parameterized by the number of

agents participating in the transaction plus the maximum degree of the graph.

A detailed description of our results and outlines of our approaches are as follows.

Verification Protocol: We give a dynamic programming algorithm which tests whether a given transaction is stable in time $O(\ell \cdot |A(G)|^2)$. Our algorithm is fast, simple and highly implementable. Towards designing the algorithm, we prove that instead of testing the existence of a blocking exchange, it is enough to check for a blocking closed walk (a relaxed notion of blocking exchange). This key lemma allows us to design an efficient polynomial-time algorithm for verification, while exhaustive search would not even result in a fixed-parameter tractable algorithm with respect to ℓ, as its time complexity would be $O(\ell \cdot |A(G)|^{\ell})$.

Hardness of Stable Barter: We prove that Stable Barter is NP-complete even when $\ell = 3$, and the in-degree and out-degree of the graph are each upper bounded by 5. To derive NP-hardness, we give a polynomial time many-to-one reduction from an NP-complete problem, 3-Dimensional Matching. Due to paucity of space, the proof of this result is deferred to the full version of the paper.

Next, we consider the parameterized complexity of the problem with respect to various natural parameters. The first one is the number of agents that are *not* part of the output transaction. However, we show that it is already NP-hard to test whether there is a stable transaction involving all the agents. Then, we show that it is W[1]-hard to decide if a given instance of Stable Barter has a stable transaction with k exchanges, when parameterized by k. Towards that, we give a parameter preserving polynomial time many-to-one reduction from a W[1]-hard problem, Exact Cover.

Algorithm for Stable Barter: The Stable Barter problem is fixed parameter tractable when parameterized by the sum of the number of agents participating in the transaction and the maximum degree of the graph. Specifically, let (G, w, ℓ) be an instance of Stable Barter. Then, it is possible to determine if there is a stable transaction $\mathscr{C}$ involving k agents in time $\Delta^{O(k)}|A(G)|^{O(1)}$, where Δ denotes the maximum degree of G. Our algorithm uses the notion of *n-p-q*-separating collections, a derandomization tool used in the theory of algorithms.

2 PRELIMINARIES

We use $\mathbb{N}$ and $\mathbb{N}_0$ to denote the sets $\{1, 2, \ldots\}$ and $\{0, 1, \ldots\}$, respectively. For $n \in \mathbb{N}$, we use $[n]$ to denote the set $\{1, \ldots, n\}$. For convenience, $[0]$ represents $\emptyset$. The codomain of a function $f : D \to R$ is the set $\{f(x) \mid x \in D\}$.

Graphs. For a (di)graph G, we use $V(G)$ and $A(G)$ to denote the sets of vertices and arcs in G, respectively. An arc from u to v in G is denoted by uv. The in-neighbors and out-neighbors of a vertex v in G are denoted by $N^-_G(v)$ and $N^+_G(v)$, respectively. For $v \in V(G)$, the indegree and outdegree of v, denoted by $d^-_G(v)$ and $d^+_G(v)$, are the cardinalities of $N^-_G(v)$ and $N^+_G(v)$, respectively. The maximum degree of G, denoted by $\Delta(G)$, is equal to $\max_{v \in V(G)} d^-_G(v) + d^+_G(v)$. A walk in G is a sequence of vertices $u_1u_2\ldots u_\ell$, where $u_iu_{i+1} \in A(G)$ for all $i \in [\ell - 1]$ and its length, denoted by $|W|$, is equal to $\ell - 1$. A walk $u_1u_2\ldots u_\ell$ is called a closed walk when $u_1 = u_\ell$. If W is a walk $u_1u_2\ldots u_\ell$ and W' is a walk $v_1v_2\ldots v_{\ell'}$ with $u_\ell = v_1$, then WW' denotes the walk $u_1u_2\ldots u_{\ell-1}v_1v_2\ldots v_{\ell'}$. A walk is called a path when no vertex in the sequence repeats. A cycle is a closed walk when only the first and and last vertex is same and all other vertices do not repeat. A cycle of length 3 is called a triangle. For a collection of cycles $\mathscr{C}$ in a graph G, we use $V(\mathscr{C})$ and $A(\mathscr{C})$ to denote the sets of vertices and arcs present in $\mathscr{C}$, respectively. The distance from a vertex u to a vertex v, denoted by $d(u, v)$, is the minimum length of a path from u to v. For $X \subseteq V(G)$, $G[X]$ and $G \setminus X$ denote the graphs induced on X and $V(G) \setminus X$, respectively. For any graph related definition and notation, which is not explicitly defined here, we refer to [5].

Parameterized Complexity. A *parameterization* of a problem is the association of an integer k with each input instance, which results in a *parameterized problem*. The central notion of Parameterized Complexity is *fixed-parameter tractability (*FPT*)*. A parameterized problem Π is said to be FPT if there is an algorithm that solves it in time $f(k) \cdot |I|^{O(1)}$, where $|I|$ is the size of the input and f is a function that depends only on k. Finally, we recall that Parameterized Complexity also provides tools to refute the existence of parameterized algorithms for certain problems (under plausible complexity-theoretic assumption), in which context the notion of W[1]-hard is a central one. It is widely believed that a problem that is W[1]-hard is unlikely to be FPT, and we refer the reader to the book [4] for more information on this notion in particular, and on Parameterized Complexity in general.

3 VERIFICATION

In this section we prove the following theorem.

Theorem 3.1. *Let (G, w, ℓ) be an instance of Stable Barter. Given a transaction $\mathscr{C}$, it is possible to determine whether or not $\mathscr{C}$ is stable in time $O(\ell \cdot |A(G)|^2)$*

Given an instance (G, w, ℓ) of Stable Barter, a closed walk $W = v_1v_2\ldots v_rv_1$ in G, and an integer $i \in [r]$, we define $\text{balance}_W(i) = w(v_{i-1}v_i) - w(v_iv_{i+1})$ (here, $v_0 = v_r$ and $v_{r+1} = v_1$). For a transaction $\mathscr{C}$ we call a closed walk, $W = v_1v_2\ldots v_rv_1$, a *witness walk* for $\mathscr{C}$, if the length of W is at most ℓ and for any $i \in [r]$, $\text{balance}_W(i) > \text{balance}_{\mathscr{C}}(v_i)$.

Lemma 3.2. *Let (G, w, ℓ) be an instance of Stable Barter and $\mathscr{C}$ denote a transaction. Then, $\mathscr{C}$ is not stable if and only if there exists a witness walk for $\mathscr{C}$.*

Proof. The forward direction of the lemma is trivial. Now we prove the reverse direction. Let $W = v_1v_2\ldots v_rv_1$ denote a witness walk for $\mathscr{C}$. Using induction on r we prove that $\mathscr{C}$ is not stable. The base case is given by $r = 2$. Then W is a cycle $v_1v_2v_1$. By our assumption we know that $\text{balance}_W(v) > \text{balance}_{\mathscr{C}}(v)$ for any $v \in \{v_1, v_2\}$. This implies that W is an exchange that prevents transaction $\mathscr{C}$ from being stable.

Now consider the induction step, that is, $2 < r \leq \ell$. If W is a cycle, then clearly $\mathscr{C}$ is not stable. Suppose that W is a closed walk that is not a cycle, that is, a vertex appears more than once in W, say $v_i = v_j$. It is well known that given a closed walk, one can extract a closed subwalk (in fact, a cycle) of length $< |W|$. Since $v_i = v_j$, we note that $W' = v_iv_{i+1}\ldots v_j$ is a closed subwalk (of length strictly less than r) of W. Thus, $W'' = v_1\ldots v_iv_{j+1}\ldots v_rv_1$ is also a closed subwalk of W of length $< r$. Note that for any $k \in \{i+1, \ldots, j-1\}$, $\text{balance}_{W'}(k-(i-1)) = \text{balance}_W(k) > \text{balance}_{\mathscr{C}}(v_k)$. Moreover, as $v_i = v_j$, the balance of every vertex $v \in V(W'')$, excluding v_i, is the same in W'' and W.

Consider the following four arcs $v_{i-1}v_i, v_iv_{i+1}, v_{j-1}v_j$ and v_jv_{j+1} (here, if $i = 1$, then $v_0 = v_r$ and if $j = r$, then $v_{j+1} = v_1$). By assumption, we know that $\text{balance}_W(i) > \text{balance}_{\mathscr{C}}(v_i)$ and $\text{balance}_W(j) > \text{balance}_{\mathscr{C}}(v_j) = \text{balance}_{\mathscr{C}}(v_i)$. Thus, from the definition of $\text{balance}_W(\cdot)$, we have

$$w(v_{i-1}v_i) - w(v_iv_{i+1}) = \text{balance}_W(i) > \text{balance}_{\mathscr{C}}(v_i) \quad (1)$$

$$w(v_{j-1}v_j) - w(v_jv_{j+1}) = \text{balance}_W(j) > \text{balance}_{\mathscr{C}}(v_i) \quad (2)$$

By adding Equations 1 and 2, we get that $w(v_{i-1}v_i) - w(v_jv_{j+1}) + w(v_{j-1}v_j) - w(v_iv_{i+1}) > 2\,\text{balance}_{\mathscr{C}}(v_i)$. Then, either $w(v_{i-1}v_i) - w(v_jv_{j+1}) > \text{balance}_{\mathscr{C}}(v_i)$ or $w(v_{j-1}v_j) - w(v_iv_{i+1}) > \text{balance}_{\mathscr{C}}(v_i)$. Using v_i and v_j interchangeably (since $v_i = v_j$), we can deduce the following relations. If $w(v_{j-1}v_j) - w(v_iv_{i+1}) > \text{balance}_{\mathscr{C}}(v_i)$, then W' is a closed walk of length strictly less than r such that for all $s \in [|W'|]$, $\text{balance}_{W'}(s) > \text{balance}_{\mathscr{C}}(v_s)$. On the other hand, if $w(v_{i-1}v_i) - w(v_jv_{j+1}) > \text{balance}_{\mathscr{C}}(v_i)$, then the closed walk W'' is of length strictly less than r such that for all $s \in [|W''|]$, $\text{balance}_{W''}(s) > \text{balance}_{\mathscr{C}}(v_s)$. In either case, by the induction hypothesis, $\mathscr{C}$ is not stable. □

Proof of Theorem 3.1. Because of Lemma 3.2, to test whether $\mathscr{C}$ is stable or not, it is enough to check for the existence of a witness walk for $\mathscr{C}$. In what follows we design a dynamic programming (DP) algorithm $\mathcal{A}$ which outputs a witness walk for $\mathscr{C}$, if one exists. We use the term *solution* for a witness walk for $\mathscr{C}$. Algorithm $\mathcal{A}$ starts by guessing an arc v_1v_2 of a hypothetical solution W. The DP table entries of $\mathcal{A}$ are indexed with (i, v) where $v \in V(G)$ and $i \in \{2, 3, \ldots, \ell\}$. The DP table entry $T[i, v]$ stores a walk $W' = v_1v_2 \ldots v_{i+1}$ of length i with the following properties.

(i) $v = v_{i+1}$, and
(ii) for all $j \in [i] \setminus \{1\}$, $\text{balance}_{W'}(j) > \text{balance}_{\mathscr{C}}(v_j)$

Among all such walks, $T[i, v]$ stores a walk W' such that $w(v_iv_{i+1})$ is maximized. The entry for $T[i, v]$ can be computed from the set $\{T[i-1, u] \mid u \in N^-_G(v)\}$ as follows. We go through all the in-neighbors u of v with respect to non-increasing order on the weight of the arc uv and pick the first non-empty W' from $\{T[i-1, u] \mid u \in N^-_G(v)\}$ such that $w(xu) - w(uv) > \text{balance}_{\mathscr{C}}(u)$, where xu is the last arc in W'. In other words, we consider the set $\{w(uv) \mid uv \in A(G)\}$, and go through it in a non-increasing order and pick the first non-empty W' from $\{T[i-1, u] \mid u \in N^-_G(v)\}$ such that $w(xu) - w(uv) > \text{balance}_{\mathscr{C}}(u)$, where xu is the last arc in W'. Then we store $W'v$ in $T[i, v]$. Otherwise we set $T[i, v] = \emptyset$. Notice that for any $j \in [\ell]$, $T[j, v_1]$ either contains a closed walk or it is empty. If there exists $j \in [\ell]$ such that $T[j, v_1]$ contains a witness walk for $\mathscr{C}$, then algorithm $\mathcal{A}$ outputs $T[j, v_1]$. Otherwise $\mathcal{A}$ outputs No.

Each $T[i, v]$ can be computed in time $O(d^-_G(v))$. The number of choices for guessing an edge v_1v_2 is $|A(G)|$. Hence, the run time of the algorithm is $O(|A(G)| \sum_{i \in [\ell], v \in V(G)} d^-_G(v)) = O(\ell \cdot |A(G)|^2)$.

Now we prove the correctness of the algorithm. Clearly when algorithm $\mathcal{A}$ outputs a closed walk, it is a witness walk and hence $\mathscr{C}$ is not stable. Now assume that $\mathscr{C}$ is not stable. Then there is a witness walk W for $\mathscr{C}$. Let $|W| = r$ and $W = v_1v_2 \ldots v_rv_1$. Consider the execution of the algorithm when it first guesses the arc v_1v_2. Notice that $T[r, v_1]$ contains a walk $W' = v'_1v'_2v'_3 \ldots v'_rv_1$ (where $v'_1 = v_1$ and $v'_2 = v_2$) such that the end vertex is v_1 and for all $j \in [r] \setminus \{1\}$, $\text{balance}_W(j) > \text{balance}_{\mathscr{C}}(v'_j)$. Moreover $w(v'_rv_1) \geq w(v_rv_1)$. Also, since $w(v_rv_1) - w(v_1v_2) > \text{balance}_{\mathscr{C}}(v_1)$, $w(v'_rv_1) - w(v_1v_2) > \text{balance}_{\mathscr{C}}(v_1)$. This implies that W' is witness walk for $\mathscr{C}$ and $T[r, v_1] = W'$. Thus, if $\mathscr{C}$ is not stable, then the algorithm $\mathcal{A}$ always outputs a witness walk. □

4 HARDNESS

In this section we show that with respect to various natural parameters such as **(a)** the number of exchanges in a stable transactions (k) and **(b)** the number of agents *not participating* into a stable transaction (h), we do not expect to design an algorithm with running time either $f(k) \cdot n^{O(1)}$ or $g(h) \cdot n^{O(1)}$ for any computable functions f and g depending on k and h alone. In fact, parameterized by h, we do not even expect an algorithm with running time $n^{O(\tau(h))}$ for any function τ depending on h alone.

The second parameter is motivated by the following situation. Ideally, we would like to find a stable transaction which involves all the agents. So, it is natural to ask whether we can find a stable transaction which involves at least $n-h$ agents, where h, the parameter, is expected to be small. But, we show that it is indeed para-NP-hard.

Theorem 4.1. *Let (G, w, ℓ) be an instance of Stable Barter. It is* NP*-hard to test whether there is a stable transaction involving all the agents, even when $\ell = 3$ and $\Delta(G) = 6$.*

Proof Sketch. We give a polynomial time many-to-one reduction from an NP-complete problem, 3-Dimensional Matching. An input to this problem consists of a universe $A \uplus B \uplus C$ and a family $\mathcal{T} \subseteq A \times B \times C$, where $|A| = |B| = |C| = n$. The objective is to determine whether there exists a subfamily $\mathcal{M}$ of pairwise disjoint sets of cardinality n. Here we refer to an element $S = (a, b, c) \in \mathcal{T}$ as a 'set' and use the notation $S = \{a, b, c\}$. We construct an instance (G, w, ℓ) of Stable Barter as follows. For $u \in A \uplus B \uplus C$, let $f(u)$ be the number of sets in $\mathcal{T}$ which contain u. For every set $S = \{a, b, c\} \in \mathcal{T}$, we create a triangle $a_Sb_Sc_Sa_S$ with each arc being assigned the weight 1. For an element $u \in A \uplus B \uplus C$, let $S_1, \ldots, S_{f(u)}$ be the sets in $\mathcal{T}$ that contain u. Now we create a *selection gadget* using the (already created) vertex set $\{u_{S_1}, \ldots, u_{S_{f(u)}}\}$ and some new vertices as shown in Figure 2. The weight of each arc is also mentioned in Figure 2. Now we set $\ell = 3$. This completes the construction of the the reduced instance. Notice that $\Delta(G) = 6$.

Now we prove the forward direction of the correctness proof. Let $n = |A| = |B| = |C|$ and $\mathcal{M} \subseteq \mathcal{T}$ be a matching of cardinality n. Now we create a transaction $\mathscr{C}$ and then prove that it is stable. For each $S = \{a, b, c\} \in \mathcal{M}$, let C_S denotes the cycle $a_Sb_Sc_Sa_S$ and let $\mathscr{C}_{\mathcal{M}} = \{C_S \mid S \in \mathcal{M}\}$. Now for each $u \in A \uplus B \uplus C$, we create the following set of triangles. Let $\{S_1, \ldots, S_{f(u)}\}$ be the sets in $\mathcal{T}$ that contains u . Furthermore, since $\mathcal{M}$ is a matching of size n, there is a set $S_i \in \{S_1, \ldots, S_{f(u)}\}$ such that $S_i \in \mathcal{M}$. For each $j \in [i-1]$, let $C_j(u)$ be the cycle $u_{S_j}x_j(u)y_j(u)u_{S_j}$ and $C'_j(u)$ be the cycle $a_j(u)b_j(u)z_j(u)a_j(u)$. For each $j \in [f(u)-1] \setminus [i-1]$, let $C_j(u)$ be the cycle $u_{S_{j+1}}x_j(u)z_j(u)u_{S_{j+1}}$ and $C'_j(u)$ be the cycle $a_j(u)b_j(u)y_j(u)a_1(u)$. Let $\mathscr{C}_u = \{C_j(u), C'_j(u) \mid j \in [f(u)-1]\}$. We

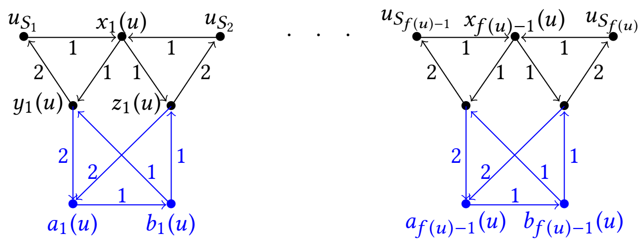


Figure 2: Selection gadget for $u \in A \uplus B \uplus C$

define $\mathscr{C}$ as follows.

$$\mathscr{C} = \mathscr{C}_{\mathcal{M}} \cup \left(\bigcup_{u \in A \uplus B \uplus C} \mathscr{C}_u \right).$$

$\mathscr{C}$ is a transaction involving all agents. No vertex of the form $x_i(u)$ and $a_i(u)$ is not part of any blocking cycle, because in both triangles they are part of, they have the same balance. Any vertex of the form u_{S_i} is part of at most three cycles and all except one of them are fully inside the section gadget for u. The balances of u_{S_i} in the triangles fully contained in the selection gadget are 1, while the balance of the other triangle in which u is part of $(u_{S_i} v_{S_i} w_{S_i} u_{S_i}$, where $(u, v, w) = S_i \in \mathcal{T})$ is 0. This implies that u_{S_i} prefers the triangles inside the section gadget, but its neighbors $x_{i-1}(u)$ and $x_i(u)$ is not part of any blocking cycle. As a result one can conclude that $\mathscr{C}$ is indeed a stable transaction.

Now towards the reverse direction of the correctness proof, notice that for the selection gadget for $u \in A \uplus B \uplus C$, the set of disjoint triangles which cover all the private vertices in the gadget (the vertices other than $u_{S_1}, \ldots, u_{S_{f(u)}}$) will cover all but one vertex from $\{u_{S_1}, \ldots, u_{S_{f(u)}}\}$. These unmatched vertices should be in a triangle (exchange) created for an element in $\mathcal{T}$, and these triangles will form a matching in $\mathcal{T}$. □

Now, we consider the parameterized complexity of Stable Barter when parameterized by the number of exchanges in a transaction.

Theorem 4.2. *Stable Barter is* W[1]*-hard parameterized by* k*, the number of exchanges in a transaction. Here, the length of the cycle is part of the input.*

Here, the number of cycles in a transaction is a parameter, and thus we use (G, w, ℓ, k) to denote an input instance of Stable Barter. We prove Theorem 4.2 by giving a polynomial time many to one reduction from a W[1]-hard problem, Exact Cover [6]. The problem is defined as follows.

Exact Cover	**Parameter:** k
Input: A universe U, a family $\mathcal{F}$ of subsets of U and $k \in \mathbb{N}$.	
Question: Is there a subfamily $\mathcal{F}'$ of $\mathcal{F}$, of cardinality at most k, such that each $u \in U$ is contained in exactly one set in $\mathcal{F}'$?	

We first give a construction that creates an instance of Stable Barter from an instance of Exact Cover.

Construction. Let $(U, \mathcal{F}, k)$ be an instance of Exact Cover. We construct an instance (G, w, ℓ, k') as follows. Let $U = \{u_1, \ldots, u_n\}$ and $\mathcal{F} = \{S_1, \ldots, S_m\}$. For each $u \in U$, we construct a vertex u in G. For each $S_i = \{u_1, \ldots, u_r\} \in \mathcal{F}$, we create $r = |S_i|$ vertices $\{S_i(j) \mid j \in [r]\}$ and create a cycle $u_1 S_i(1) u_2 S_i(2) \ldots u_r S_i(r) u_1$ with each arc having weight $4i$. Now for each $u_i \in U$, we create a selection gadget for u as follows. Add two new vertices, x_i and y_i, and the following five paths.

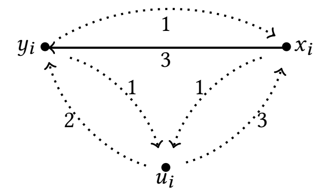


Figure 3: Selection gadget for u_i. Each dotted arrow represents a path with every arc weight in it is specified by the label. The length of the path from y_i to x_i is $2n-1$, while any other doted arrow represents a path of length n. The arrow from x_i to y_i represents an arc with weight 3.

(1) A path $P_{y_i u_i}$ of length n, from y_i to u_i with each arc having weight 1.
(2) A path $P_{x_i u_i}$ of length n, from x_i to u_i with each arc having weight 1.
(3) A path $P_{y_i x_i}$ of length $2n - 1$, from y_i to x_i with each arc having weight 1.
(4) A path $P_{u_i y_i}$ of length n, from u_i to y_i with each arc having weight 2.
(5) A path $P_{u_i x_i}$ of length n, from u_i to x_i with each arc having weight 3.

Now, we add an arc $x_i y_i$ with weight 3. This completes the construction of the selection gadget for u_i (see Figure 3). The selection gadgets for all u_i and u_j $(i \neq j)$ are disjoint. Finally, we add arcs $\{y_i x_{i+1} \mid i \in [n-1]\} \cup \{y_n x_1\}$ with each arc having weight 3. That is, $x_1 y_1 x_2 y_2 \ldots x_n y_n x_1$ is a cycle with each arc having weight 3. This completes the construction of G. We set $\ell = 2n$, $k' = k + 1$ and (G, w, ℓ, k') as the instance of Stable Barter. Clearly, the reduction can be done in polynomial time.

Observation 1. *For a cycle C in G,* $\sum_{v \in V(C)} \text{balance}_C(v) = 0$.

Next, we prove the correctness of the reduction.

Lemma 4.3. *If $(U, \mathcal{F}, k)$ is a* Yes *instance of Exact Cover, then (G, w, ℓ, k) is a* Yes *instance of Stable Barter.*

Proof. Let $\mathcal{F}'$ be a solution to $(U, \mathcal{F}, k)$. We give a stable transaction $\mathscr{C}$. For each $S = \{u_1, \ldots, u_r\} \in \mathcal{F}'$, let C_S denote the cycle $u_1 S(1) u_2 S(2) \ldots u_r S(r) u_1$ and let $\mathscr{C}_{\mathcal{F}'} = \{C_S \mid S \in \mathcal{F}'\}$. We define $\mathscr{C} = \mathscr{C}_{\mathcal{F}'} \cup \{x_1 y_1 x_2 y_2, \ldots, x_n y_n x_1\}$. Clearly, each cycle in $\mathscr{C}$ has length at most $2n$ and $|\mathscr{C}| \leq k + 1 = k'$.

Now we prove that $\mathscr{C}$ is a stable transaction. Towards that, we first claim that for any $i \in [n]$, there is no blocking exchange for $\mathscr{C}$ containing x_i. For a contradiction assume that there is a blocking exchange C for $\mathscr{C}$ containing x_i. Notice that $\text{balance}_{\mathscr{C}}(x_i) = 0$ and all the arcs incident with x_i have weights 1 or 3 (see Figure 3). Then, in the blocking exchange C, the weight of the incoming arc to x_i should be 3 and the weight of the outgoing arc from x_i should be 1. There is only one outgoing arc of weight 1 incident with x_i, which is the first arc on the path $P_{x_iu_i}$. This implies that C contains the path $P_{x_iu_i}$ (because all the internal vertices in $P_{x_iy_i}$ have in-degree 1 and out-degree 1). This implies that $u_i \in V(C)$ and the weight of the incoming arc of u_i in C is 1. Since all the out-going arcs from u_i have weights strictly larger than 1, $\text{balance}_C(u_i) < 0$. This is a contradiction to the assumption that C is a blocking exchange for $\mathscr{C}$, because $\text{balance}_{\mathscr{C}}(v) = 0$ for all $v \in V(\mathscr{C})$.

Now we show that for any $i \in [n]$, there is no blocking exchange for $\mathscr{C}$ containing y_i. For a contradiction assume that there is a blocking exchange C' for $\mathscr{C}$ containing y_i. Notice that $\text{balance}_{\mathscr{C}}(y_i) = 0$ and the arcs incident with y_i have weights 1, 2 or 3 (see Figure 3). Then, in the blocking exchange C', the weight of the outgoing arc from x_i should be 1 or 2. There is no out-going arc from y_i of weight 2. This implies that the weight of the outgoing arc from y_i should be 1. There are two outgoing arcs of y_i that have weight 1, the first arcs on paths $P_{y_ix_i}$ and $P_{y_iu_i}$. We have already shown that $x_i \notin V(C')$. This implies that $P_{y_iu_i}$ belongs to C'. Hence $u_i \in V(C)$ and the weight of the incoming arc of u_i in C' is 1. Since all the out-going arcs from u_i have weights strictly larger than 1, $\text{balance}_C(u_i) < 0$. This is a contradiction to the assumption that C' is a blocking exchange for $\mathscr{C}$.

We have shown that there is no blocking exchange containing a vertex from $\{x_i, y_i \mid i \in [n]\}$. Any cycle containing an internal vertex of a path $P \in \{P_{u_ix_i}, P_{u_iy_i}, P_{y_iu_i}, P_{x_iu_i}, P_{y_ix_i}\}$, also contains a vertex from $\{x_i, y_i \mid i \in [n]\}$. This implies that if there is a blocking exchange for $\mathscr{C}$, then it should be in the graph induced on $X = U \cup \{S_i(j) \mid i \in [n], j \in [|S_i|]\}$. Suppose there is a blocking exchange C for $\mathscr{C}$, fully contained in the graph $G[X]$. Since, for all $u \in U$, $\text{balance}_{\mathscr{C}}(u) = 0$, by the definition of blocking exchange, we have that $\text{balance}_C(v) > 0$ for all $v \in V(C) \cap U$. Since for all $x \in \{S(i) \mid S \in \mathcal{F}, i \in [|S|]\}$ the arcs incident with z have equal weights, we have that $\text{balance}_C(z) = 0$. This implies that for all $x \in V(C)$, $\text{balance}_C(x) > 0$, which is not possible, by Observation 1. ◻

Lemma 4.4. *If (G, w, ℓ, k') is a* Yes *instance of Stable Barter, then $(U, \mathcal{F}, k)$ is a* Yes *instance of Exact Cover.*

To prove Lemma 4.4, we first state some observations and prove some auxiliary lemmas.

Observation 2. *Let $i \in [n]$ and $z \in \{x_i, y_i\}$. There is only one cycle of length at most $2n$ containing z and u_i in G. The unique cycle of length at most $2n$ containing z and u_i is $P_{u_iz}P_{zu_i}$.*

Observation 3. *For any $i \in [n]$, $d(u_i, z) > n$ and $d(z, u_i) > n$, where $z \in \{x_j, y_j \mid j \in [n] \setminus \{i\}\}$.*

Lemma 4.5. *Let $\mathscr{C}$ be a stable transaction on (G, w, ℓ, k'). Then for all $i \in [n]$, $u_i \in V(\mathscr{C})$.*

Proof. For a contradiction assume that there exists $i \in [n]$ such that $u_i \notin V(\mathscr{C})$. Now we claim that $C = P_{u_ix_i}P_{x_iu_i}$ is a blocking exchange for $\mathscr{C}$. Notice that the length of C is $2n$. Since $u_i \notin V(\mathscr{C})$, no internal vertex of $P_{u_ix_i}$ or $P_{x_iu_i}$ is in $V(\mathscr{C})$. This implies that either $x_i \notin V(\mathscr{C})$, or the outgoing arc of x_i in $\mathscr{C}$ has weight 3, while the incoming arc of x_i in $\mathscr{C}$ has weight from $\{1, 3\}$. In either case C is a blocking exchange for $\mathscr{C}$ because $\text{balance}_C(x_i) = 2$. This is a contradiction to the assumption that $\mathscr{C}$ is stable. ◻

Lemma 4.6. *Let $\mathscr{C}$ be a stable transaction and $u_i \in V(\mathscr{C})$ for some $i \in [n]$. Then the cycle $C \in \mathscr{C}$, containing u_i will not contain any vertex from $\{x_j, y_j \mid j \in [n]\}$.*

Proof. By Observation 3, we have that $\{x_j, y_j \mid j \in [n], i \neq j\} \cap V(C) = \emptyset$. Suppose $x_i \in V(C)$. Then, by Observation 2, $C = P_{u_ix_i}P_{x_iu_i}$. Also, since all the incoming arcs to y_i are from the paths $P_{x_iy_i}$ and $P(u_iy_i)$, we have that $y_i \notin V(\mathscr{C})$. Since $y_i \notin V(\mathscr{C})$, all the internal vertices of $P_{u_iy_i}$ and $P_{y_iu_i}$ do not belong to $V(\mathscr{C})$. Now consider the cycle $C' = P_{u_iy_i}P_{y_iu_i}$. Notice that $V(\mathscr{C}) \cap V(C') = \{u_i\}$ and $\text{balance}_{C'}(u_i) = -1$, while $\text{balance}_C(u_i) = -2$. This implies that C' is a blocking exchange for $\mathscr{C}$, a contradiction.

Suppose $y_i \in V(C)$. Then, by Observation 2, $C = P_{u_iy_i}P_{y_iu_i}$. Also, since all the outgoing arcs from x_i are from the paths $P_{x_iy_i}$ and $P(x_iu_i)$, we have that $x_i \notin V(\mathscr{C})$. Since $x_i \notin V(\mathscr{C})$ all the internal vertices of $P_{x_iy_i}$ and $P_{y_ix_i}$ do not belong to $V(\mathscr{C})$. Now consider the cycle $C'' = P_{x_iy_i}P_{y_ix_i}$. Notice that $V(\mathscr{C}) \cap V(C'') = \{y_i\}$ and $\text{balance}_{C''}(y_i) = 2$, while $\text{balance}_C(y_i) = 1$. This implies that C'' is a blocking exchange for $\mathscr{C}$, a contradiction. ◻

Proof of Lemma 4.4. Let $\mathscr{C}$ be a stable transaction on (G, w, ℓ, k') such that $|\mathscr{C}| \leq k' = k + 1$. By Lemmas 4.5 and 4.6, we know that $U \subseteq V(\mathscr{C})$, and any cycle C which contains $u \in U$, will not contain any vertex from $\{x_i, y_i \mid i \in [n]\}$. This also implies that any cycle $C \in \mathscr{C}$, containing a vertex from U, is fully contained in the graph induced on $X = U \cup \{S_i(j) \mid i \in [n], j \in [|S_i|]\}$. Let $\mathscr{C}'$ be the set of cycles from $\mathscr{C}$ which contains at least one vertex from U. We know that $U \subseteq V(\mathscr{C}')$ and all cycles in $\mathscr{C}'$ are fully contained in $G[X]$. First we claim that $|\mathscr{C}'| \leq k$. Notice that there is at least one cycle in $G \setminus X$, $|\mathscr{C}| \leq k + 1$ and $\mathscr{C}$ is a stable transaction. This implies that $|\mathscr{C}'| \leq k$.

Now we claim that for every $C \in \mathscr{C}'$ and $v \in V(C)$, $\text{balance}_C(v) = 0$. Suppose not. Then by Observation 1, there is a vertex $v \in V(C)$ such that $\text{balance}_C(v) < 0$. Fix a vertex v such that $\text{balance}_C(v) < 0$. Since all the arcs incident with a vertex in $\{S_i(j) \mid i \in [n], j \in [|S_i|]\}$ have same weight, we have that $v \in U$. Let $v = u_i$. Since all the arcs incident with v in $G[X]$ have weights from $\{4j \mid j \in [n]\}$, $\text{balance}_C(u_i) \leq -4$. Let $C' = P_{u_ix_i}P_{x_iu_i}$. Notice that $\text{balance}_{C'}(x_i) = 2$, $\text{balance}_{C'}(u_i) = -2$ and all internal vertices of $P_{u_ix_i}$ and $P_{x_iu_i}$ are not in $V(\mathscr{C})$. This implies that C' is a blocking exchange for $\mathscr{C}$, a contradiction.

Thus, we have shown that $\mathscr{C}'$ is a collection of at most k cycles in $G[X]$, $U \subseteq V(\mathscr{C}')$ and for all $v \in V(\mathscr{C}')$, $\text{balance}_{\mathscr{C}'}(v) = 0$. This implies that for any $C \in \mathscr{C}'$ all the arcs in C have same weight. For any $C \in \mathscr{C}'$, if arcs in C has weight $4i$ then we define a set $S_C = S_i$.. Notice that $S_C = V(C) \cap U$, because the graph induced on arcs of weight $4i$ for any i, in $G[X]$ is the cycle $u_1S_i(1)u_2S_i(2)\ldots u_rS_i(r)u_1$, where $S_i = \{u_1, \ldots, u_r\}$. Hence $\{S_C \mid C \in \mathscr{C}'\}$ is an exact cover of $(U, \mathcal{F}, k)$. ◻

Lemmas 4.3 and 4.4 together prove Theorem 4.2.

5 STABLE BARTER: ALGORITHM

In this section we show that Stable Barter is FPT parameterized by the maximum degree Δ of the input graph and the number of agents participating in the transaction.

Theorem 5.1. *Let (G, w, ℓ) be an instance of* Stable Barter*. Then, it is possible to determine whether or not there is a stable transaction $\mathscr{C}$ involving k agents in time $O((e\Delta(G))^k \ell |A(G)|^2 \log |A(G)|)$.*

Towards the proof of Theorem 5.1 we first introduce the notion of *n-p-q-lopsided universal* family. Given a universe U and an integer ℓ, we denote all the ℓ-sized subsets of U by $\binom{U}{\ell}$. We say that a family $\mathcal{F}$ of sets over a universe U of size n is an *n-p-q-lopsided universal* family if for every $A \in \binom{U}{p}$ and $B \in \binom{U \setminus A}{q}$, there is an $F \in \mathcal{F}$ such that $A \subseteq F$ and $B \cap F = \emptyset$.

Lemma 5.2 ([7]). *There is an algorithm that given $n, p, q \in \mathbb{N}$ constructs an n-p-q-lopsided universal family $\mathcal{F}$ of cardinality $\binom{p+q}{p} \cdot 2^{o(p+q)} \log n$ in time $|\mathcal{F}| \cdot n$*

We would also need the following simple observation for our algorithm, whose correctness follows from the definition of stability.

Observation 4. *Let (G, w, ℓ) be an instance of* Stable Barter *and $\mathscr{C}$ be a stable transaction. Consider $V(\mathscr{C})$, the set of agents participating in the transaction $\mathscr{C}$. Then, $G \setminus V(\mathscr{C})$ is a graph that only contains cycles (if any) of length strictly greater than ℓ.*

Proof Theorem 5.1. Let (G, w, ℓ) be an instance of the Stable Barter, with the maximum degree (indegree+outdegree) of G being at most $\Delta = \Delta(G)$. Suppose that there exists a stable transaction $\mathscr{C}$ involving k agents, then we know that $\mathscr{C}$ is a collection of vertex disjoint directed cycles each of length at most ℓ such that the total number of vertices in the cycles is k. Thus, the number of vertices and arcs that participate in $\mathscr{C}$ is exactly k. Let $V(\mathscr{C})$ and $A(\mathscr{C})$ denote the set of vertices and arcs that participate in the cycles of $\mathscr{C}$, respectively.

Let $\mathscr{C}$ be a hypothetical stable transaction in G involving k agents. For our algorithm we will like to have a function $f : A(G) \to \{0, 1\}$ with the following property:

(P1) f assigns 1 to every arc in $A(\mathscr{C})$; and assigns 0 to every arc that is not present in $A(\mathscr{C})$ but is incident to a vertex in $V(\mathscr{C})$.

A function f satisfying the property (P1) with respect to a transaction $\mathscr{C}$ is called *nice with respect to* $\mathscr{C}$. Furthermore, a function $f : A(G) \to \{0, 1\}$ is said to be *nice* if f satisfies the property (P1) for *some* stable transaction in G involving k agents.

Given a function $f : A(G) \to \{0, 1\}$ our algorithm, $\mathbb{A}$, works as follows.

Step 1: Let A_1 denote the set of arcs in G that have been assigned 1 by f. Let G_1 denote the directed graph with the vertex set $V(G)$ and the arc set A_1.

Step 2: First we clean the graph G_1 by removing every connected component in G_1 that is not a directed cycle. After the cleaning process we know that G_1 is a collection of cycles. Let $\mathscr{C}^\star$ be the set of cycles of length at most ℓ in G_1. Next, we check whether or not $|V(\mathscr{C}^\star)| = k$, if no, then return that f is *bad*. Else, using Theorem 3.1 we check whether or not $\mathscr{C}^\star$ is stable in time $O(\ell \cdot |A(G_1)|^2)$. If Theorem 3.1 returns that $\mathscr{C}^\star$ is stable, then $\mathbb{A}$ returns $\mathscr{C}^\star$; else $\mathbb{A}$ returns that f is *bad*.

To show the correctness of our algorithm we need to run our algorithm with a nice function f. Towards this end, we first show that there exists a family of functions, $\mathcal{H} = \{f : A(G) \to \{0, 1\}\}$ such that if G contains a stable transaction, $\mathscr{C}$, of size k, then there exists a function $f \in \mathcal{H}$ such that f is nice with respect to $\mathscr{C}$. We will call the family of functions $\mathcal{H}$, a *nice family*.

Let the arc set $A(G)$ of the graph G be denoted by $\{1, 2, \ldots, m\}$. We will use Lemma 5.2 in our construction of a nice family. By applying Lemma 5.2 to the universe $U = \{1, 2, \ldots, m\}$, $p = k$ and $q = (\Delta - 1)k$, we obtain an m-p-q-lopsided-universal family $\mathcal{F}$ of size $\binom{p+q}{p} \cdot 2^{o(p+q)} \cdot \log m$ in time $O(\binom{p+q}{p} \cdot 2^{o(p+q)} \cdot m \log m)$. For every set $X \in \mathcal{F}$, we define f_X (the characteristic function of X) as follows: $f_X(x) = 1$ if $x \in X$ and $f_X(x) = 0$ otherwise. Finally, we define $\mathcal{H} := \{f_X \mid X \in \mathcal{F}\}$.

Next, we show that $\mathcal{H}$ is a nice family. Suppose G has a stable transaction $\mathscr{C}$ involving k agents. Let $A(\mathscr{C})$ denotes the set of arcs that are present in the cycles in $\mathscr{C}$ and let $B(\mathscr{C})$ denote the set of arcs that are not present in $A(\mathscr{C})$ but are incident to a vertex in $V(\mathscr{C})$. Clearly, $|A(\mathscr{C})| = k$ and $|B(\mathscr{C})| \leq (\Delta - 1)k$, since every vertex in $V(G)$ has its degree upper bounded by Δ. Since $\mathcal{F}$ is a m-k-$(\Delta-1)k$-lopsided-universal family, we know that there exists a set $X \in \mathcal{F}$ such that $A(\mathscr{C}) \subseteq X$ and $B(\mathscr{C}) \cap X = \emptyset$. By construction, f_X is nice with respect to $\mathscr{C}$. Hence, $\mathcal{H}$ is a nice family.

Finally, to test whether there exists a stable transaction $\mathscr{C}$ in (G, w, ℓ) involving k agents, we do as follows. For every $f_X \in \mathcal{H}$, we run the algorithm $\mathbb{A}$. If for any function $f_X \in \mathcal{H}$, $\mathbb{A}$ returns a stable transaction then we return the same. Else, we know that for every function $f_X \in \mathcal{H}$, $\mathbb{A}$ returns f_X is *bad*. In this case we return that G does not have a stable transaction of size k.

To argue the correctness observe that if G has a stable transaction $\mathscr{C}$ involving k agents, then there exists a nice function f with respect to $\mathscr{C}$. Let us consider the iteration of $\mathbb{A}$ when run with f. Observe that in G_1 every exchange (cycle) C of $\mathscr{C}$ occurs as a connected component such that the arc set of this connected component is precisely the arc set of C. Indeed, since f is nice with respect to $\mathscr{C}$, every arc that is incident to a vertex in C and does not belong to C has been assigned 0. Furthermore, by Observation 4 we know that if $W = V(\mathscr{C})$, then $G \setminus W$ has no cycle of length at most ℓ; so $G_1 \setminus W$ is a collection of directed cycles. In particular, the cleaning operation to obtain G_1 (Step 2) preserves all the directed cycles in $\mathscr{C}$. In other words, the only connected components of G_1 are the cycles in $\mathscr{C}$ itself and some cycles of length greater than ℓ. Hence, we can conclude that our algorithm $\mathbb{A}$ will correctly output $\mathscr{C}$. The correctness of last line follows from Theorem 3.1.

Finally, we prove the running time of the algorithm. Quite clearly, it is upper bounded by the size of $\mathcal{H}$ and the running time of $\mathbb{A}$. Thus, we have

$$
\begin{aligned}
|\mathcal{H}| \times O(\ell \cdot |A(G_1)|^2) &\leq O(\binom{(\Delta-1)k + k}{k} 2^{o(\Delta k)} \cdot \ell \cdot m^2 \log m) \\
&\leq O(\left(\frac{e\Delta k}{k}\right)^k 2^{o(\Delta k)} \ell \cdot m^2 \log m) \\
&= O((e\Delta)^k\, 2^{o(\Delta k)} \ell \cdot m^2 \log m)
\end{aligned}
$$

In the second inequality, we have used the well known inequality that $\binom{n}{k} \leq \left(\frac{ne}{k}\right)^k$, where e is the base of the natural logarithm. This concludes the proof. □

REFERENCES

[1] D. J. Abraham, A. Blum, and T. Sandholm. 2007. Clearing algorithms for barter exchange markets: Enabling nationwide kidney exchanges. In *Proceedings of the 8th ACM conference on Electronic Commerce (EC)*. 295–304.

[2] Péter Biró. 2007. Stable exchange of indivisible goods with restrictions. In *Proceedings of the 5th Japanese-Hungarian Symposium*. Citeseer, 97–105.

[3] F. Brandt, V. Conitzer, U. Endrisss, J. Lang, and A. D. Procaccia (Eds.). 2016. *Handbook of Computational Social Choice*. Cambridge Univ. Press.

[4] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. 2015. *Parameterized Algorithms*. Springer.

[5] R. Diestel. 2012. *Graph Theory, 4th Edition*. Graduate texts in mathematics, Vol. 173. Springer.

[6] Rodney G. Downey and Michael R. Fellows. 1995. Fixed-Parameter Tractability and Completeness II: On Completeness for W[1]. *Theor. Comput. Sci.* 141, 1&2 (1995), 109–131. https://doi.org/10.1016/0304-3975(94)00097-3

[7] Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. 2016. Efficient Computation of Representative Families with Applications in Parameterized and Exact Algorithms. *J. ACM* 63, 4 (2016), 29:1–29:60. https://doi.org/10.1145/2886094

[8] D. Gale and L. S. Shapley. 1962. College Admissions and the Stability of Marriage. *The American Mathematical Monthly* 69, 1 (1962), 9–15.

[9] Dan Gusfield and Robert W. Irving. 1989. *The Stable marriage problem - structure and algorithms*. MIT Press.

[10] A. Igarashi, R. Bredereck, and E. Elkind. 2017. On Parameterized Complexity of Group Activity Selection Problems on Social Networks. In *In Proceedings AAMAS'17*.

[11] A. Igarashi and E. Elkind. 2016. Hedonic games with graph-restricted communication. In *Proceedings of AAMAS'16*. 242–250.

[12] A. Igarashi, E. Elkind, and D. Peters. 2017. Group activity selection on social network. In *Proceedings of AAAI'17*.

[13] Intervac. [n. d.]. www.intervac-homeexchange.com.

[14] R.W. Irving, K. Iwama, D. F. Manlove, S. Miyazaki, and Y. Morita. 2002. Hard Variants of Stable Marriage. *Theoretical Computer Science* 276, 1-2 (2002), 261–279.

[15] Robert W. Irving. 1994. Stable marriage and indifference. *Discrete Applied Mathematics* 48, 3 (1994), 261 – 272. https://doi.org/10.1016/0166-218X(92)00179-P

[16] R. W. Irving. 2007. The cycle roommates problem: a hard case of kidney exchange. *Inform. Process. Lett.* 103, 1 (2007), 1–4.

[17] Donald Ervin Knuth. 1997. *Stable marriage and its relation to other combinatorial problems : an introduction to the mathematical analysis of algorithms*. Providence, R.I. American Mathematical Society.

[18] David F. Manlove. 2013. *Algorithmics of Matching Under Preferences*. Series on Theoretical Computer Science, Vol. 2. WorldScientific.

[19] NationalOddShoeExchange. [n. d.]. http://www.oddshoe.org/.

[20] ReadItSwapIt. [n. d.]. http://www.readitswapit.co.uk/TheLibrary.aspx. .

[21] Tayfun Sonmez. [n. d.]. Strategy-proofness and Essentially Single-valued Cores. *Econometrica* 67, 3 ([n. d.]), 677–689. https://doi.org/10.1111/1468-0262.00044