Cross-chain Swaps with Preferences

Eric Chan
University of California at Riverside

Marek Chrobak
University of California at Riverside

Mohsen Lesani
University of California at Riverside

***Abstract*—Extreme valuation and volatility of cryptocurrencies require investors to diversify often which demands secure exchange protocols. A cross-chain swap protocol allows distrusting parties to securely exchange their assets. However, the current models and protocols assume predefined user preferences for acceptable outcomes. This paper presents a generalized model of swaps that allows each party to specify its preferences on the subsets of its incoming and outgoing assets. It shows that the existing swap protocols are not necessarily a strong Nash equilibrium in this model. It characterizes the class of swap graphs that have protocols that are safe, live and a strong Nash equilibrium, and presents such a protocol for this class. Further, it shows that deciding whether a swap is in this class is NP-hard through a reduction from 3SAT, and further is Σ_2^{P}-complete through a reduction from $\exists\forall$DNF.**

I. Introduction

The multitude and volatility of cryptocurrencies force investors to diversify and frequenty trade their holdings. However, these currencies are hosted by distinct distributed blockchains and trading across blockchains is not atomic by default. This has led to the development of *cross-chain swap protocols* [19], [8], [10], [18], [13], [34] that allow distrusting parties to securely exchange their assets. Application of such swap protocols is not limited to trading digital currencies — they can be used for trading any type of digital assets (NFTs, for example), or even for trading physical objects by safely trasferring ownership documentation.

In a pioneering work, Herlihy [19] formalizes a cross-chain swap as a directed graph where vertices represent parties, and arcs represent assets to be exchanged. An execution of a *swap graph* is represented as the subset of arcs that are triggered in that execution. The *outcome* for each party is captured in five *predefined classes*: Deal, NoDeal, Discount, FreeRide, and Underwater. The classes Deal and NoDeal represent outcomes for a party where respectively, all and none of the arcs of that party are triggered. The class Discount represents outcomes where some of the outgoing arcs are not triggered, and FreeRide represents outcomes where at least one incoming but no outgoing arc is triggered. Outcomes in all these classes are considered acceptable by each party. The class Underwater captures all unacceptable outcomes, namely outcomes where at least one outgoing arc is triggered but not all incoming arcs are. Given this model of outcomes, Herlihy presented a *protocol based on hashed time-locks* and proved it to be *atomic*, meaning that it satisfies the conditions of *liveness, safety and strong Nash equilibrium*.

This project has been supported by NSF Career 1942711, DARPA YFA D22AP00146, and NSF grant CCF-2153723.

In practice, as noted in the original proposal [19], some parties may find it advantageous to exchange only *some* of their outgoing assets for only *some* of their incoming assets. As an example, suppose that Alina has a white shirt and white pants and she joins the swap hoping to trade for a black shirt and black pants. Coincidentally, Bohdan has exactly these items and joins the swap looking for the reverse trade. However, both of them would actually prefer to retain one white article of clothing and one black article of clothing, if possible. Thus, it would be preferable for both parties to, say, only swap the shirts or only swap the pants, although it is also acceptable to swap both. Such scenarios are not captured by the model in [19], because the outcomes with just one item swapped are in the class Underwater.

This leads to the natural question, left open in [19]: is there a more general swap model that allows each party to specify its personal preferences over all possible swap outcomes, and, at the same time, admits an atomic protocol.

Addressing this question, this paper introduces a general model of cross-chain swaps that we call *swap systems*. In a swap system, as in [19], the set of prearranged asset transfers is represented by a directed graph. Unlike in [19], however, in our model each party can specify its own preferences between all its possible outcomes (that is, between sets consisting of its incoming and outgoing arcs). These preferences can be arbitrary, as long as they form a poset and satisfy natural monotonicity conditions. This generality allows us to capture not only subjectivity of preferences, but also dependencies between assets. The example above (about trading clothing items) illustrates such a dependency: for the purpose of trading, Alina values her pair of items higher than the sum of their individual values. Such dependencies often arise in practice when a party intends to trade multiple assets — in fact, common investment strategies are guided by objectives (diversification, for example) that inherently involve asset dependencies.

As it turns out, Herlihy's protocol is not necessarily atomic in all swap systems, although it still satisfies the conditions of liveness, safety, and weak Nash equilibrium. We then present a characterization of swap systems that admit atomic protocols. The correctness proof of this characterization embodies such a protocol. We then focus on the problem of verifying whether a given swap systems has an atomic protocol. To this end, we provide a full characterization of the time complexity of this problem and show that it's computationally infeasible, by a novel proof of completeness in the complexity class Σ_2^{P}. As a stepping stone to this full characterization, we also include an easier proof of NP-hardness.

The paper is organized as follows.

- In section II we introduce our model of swap systems, including the definitions of atomic protocols in this model.
- In section III we show that our model is indeed a generalization of Herlihy's model.
- The full characterization of swap systems that admit atomic protocols is given in section IV.
- The decision problem of testing whether a swap systems admits an atomic protocol is studied in section V and section VI, first proving NP-hardness and then refining the proof to show Σ_2^{P}-completeness.

For readers interested in the practical impact of our work, the overall take-out message from this paper is this: (i) Even if some parties wish to specify outcome preferences not captured by the model in [19], it still may be possible to realize the swap with a protocol that is atomic and efficient. (ii) The challenge is that in order to determine whether it is possible, and to actually specify this protocol, one needs to solve a computationally infeasible decision problem. Naturally, for small number of parties this can still be done in practice – say by exhaustive search.

II. Swap Systems

As discussed in the introduction, Herlihy's model [19] for cross-chain swaps assumed that the rational behavior of participating parties is determined by preferences between five types of outcomes: DEAL, NODEAL, DISCOUNT, FREERIDE, and UNDERWATER. These preferences were assumed to be shared by all parties, and can be interpreted as a simple partial order on all possible outcomes. Some of these preferences are natural; for example, in DISCOUNT a party receives all incoming assets without trading all outgoing assets, making it preferable to DEAL. But, as explained in the introduction, in practice a party may consider some outcomes designated as UNDERWATER in [19] to be acceptable, or even preferable to DEAL. As another example, suppose that Alina possesses items `A` and `B` that she values at \$10 and \$12, and Bohdan possesses items `X` and `Y` that Alina values at \$11 and \$14. Providing that Alina's preferences are based only on the monetary value, she would accept to join the swap that allows her to swap both `A` and `B` for Bohdan's `X` and `Y`, but she would be even happier if she ends up swapping only `A` for `Y` instead. Similarly, there is no justification for the outcomes in FREERIDE to be incomparable to DEAL or DISCOUNT.

To represent such individual preferences, we now refine Herlihy's model by allowing each party to specify a partial order on all her possible outcomes of a protocol. Our model is very general in that (unlike in the example above) a party's preferences are not determined by numerical values of individual assets, but rather involve comparing directly whole sets of traded and acquired assets. The advantage of this approach is that it captures dependencies between assets, when a party values a set of assets higher or lower than the sum of their individual values. As an example, say that Alina owns a power drill and a shovel, while Bohdan is in possession of a pair of skis. Alina would not swap any of her items for any single ski, but she may be happy to swap both of her items for the pair. On the other hand, if, instead of skis, Bohdan needs to get rid of two skateboards, Alina may prefer to swap any of her items for one skateboard rather than swapping both for two skateboards.

Swap Systems. A *swap system* is specified by a pair $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ consisting of a digraph $\mathcal{D}$ that represents the pre-arranged asset transfers and a collection $\mathcal{P}$ of posets that specifies the preferences of each involved party among all of its potential outcomes. Next, we give a formal definition of these two components of $\mathcal{S}$.

Digraph $\mathcal{D} = (V, A)$ is called a *swap digraph*. Each vertex $v \in V$ represents a party that participates in the swap, and each arc $(u, v) \in A$ represents an asset that is to be transferred from party u to party v. By A_v^{in} and A_v^{out} we will denote the sets of vertex v's incoming and outgoing arcs, respectively. If $(x, v) \in A_v^{in}$ then x is called an *in-neighbor* of v, and if $(v, x) \in A_v^{out}$ then x is called an *out-neighbor* of v. Throughout the paper we assume that $\mathcal{D}$ does not have multiple arcs[1]. We also assume that $\mathcal{D}$ is weakly connected (otherwise a swap can be arranged for each connected component separately). To exclude some degenerate scenarios, we also assume that $|V| \geq 2$ and that $A_v^{in} \neq \emptyset$ and $A_v^{out} \neq \emptyset$ for each $v \in V$.

An *outcome* of a party $v \in V$ is a pair $\omega = \langle \omega^{in} \,|\, \omega^{out} \rangle$, where $\omega^{in} \subseteq A_v^{in}$ and $\omega^{out} \subseteq A_v^{out}$. An outcome represents the sets of acquired and traded assets, ω^{in} and ω^{out} respectively. The set of all possible outcomes of v will be denoted Ω_v. To reduce clutter, instead of arcs, in $\langle \omega^{in} \,|\, \omega^{out} \rangle$ we will often list only the corresponding in-neighbors and out-neighbors of v; for example, instead of $\langle \{(x, v), (y, v)\} \,|\, \{v, z\} \rangle$ we will write $\langle x, y \,|\, z \rangle$.

The collection $\mathcal{P} = \{\mathcal{P}_v\}_{v \in V}$ consists of *preference posets*. The preference poset of a party $v \in V$ is $\mathcal{P}_v = (\Omega_v, \preceq_v)$, where $\preceq_v$ is a partial order on Ω_v. We will write $\omega \prec_v \omega'$ if $\omega \preceq_v \omega'$ and $\omega \neq \omega'$. This poset naturally represents v's evaluation of its potential outcomes; that is, relation $\omega \preceq_v \omega'$ holds if v views outcome ω' to be better than outcome ω. The outcome where v does not participate in any transfer is $\text{NODEAL}_v = \langle \varnothing \,|\, \varnothing \rangle$ and the outcome where all of v's transfers are realized is $\text{DEAL}_v = \langle A_v^{in} \,|\, A_v^{out} \rangle$. Each preference poset $\mathcal{P}_v$ is assumed to have the following properties:

(p.1) DEAL is better than NODEAL: $\text{NODEAL}_v \prec_v \text{DEAL}_v$. Naturally, each party prefers swapping all assets over being completely excluded, as otherwise it would not even join the swap system.

(p.2) Inclusive Monotonicity: $(\omega_1^{in} \subseteq \omega_2^{in} \wedge \omega_2^{out} \subseteq \omega_1^{out}) \Rightarrow \omega_1 \preceq_v \omega_2$, for every two outcomes $\omega_1, \omega_2 \in \Omega_v$. That is, it's better to receive more assets and to trade fewer assets[2].

The preference pairs $\omega_1 \prec_v \omega_2$ that are determined by rules (p.1) and (p.2) above will be called *generic*. The size of the preference poset may be exponentially large with respect to

[1]This assumption is only for convenience – our model and results trivially extend to multi-digraphs, although this requires more cumbersome notation and terminology.

[2]Duuh.

the size of the swap digraph $\mathcal{D}$, but it is not necessary for a party to specify generic preferences as they are implied from the above rules. Therefore, throughout the paper, we assume that $\mathcal{P}_v$ is specified by its *generator set*, which is a subset of its non-generic preference pairs that, together with the generic pairs and transitivity, generate the whole poset. A generator set of a poset may not be unique. We use this convention in our examples and running time bounds. (This does not affect our hardness results — they hold even if the preference poset of each party is specified by listing *all* preference pairs.)

An outcome $\omega \in \Omega_v$ is called *acceptable* if $\omega \succeq \text{NODEAL}_v$. The set of acceptable outcomes of a node v will be denoted $\mathcal{A}_v$[3].

Throughout the paper, we will often omit subscript v in these notations (and others as well) if v is implicit in the context or irrelevant. On the other hand, if any ambiguity may arise, we will sometimes add a superscript to some notations specifying the digraph under consideration; for example we will write $\text{DEAL}_v^{\mathcal{D}}$ to specify that outcome $\text{DEAL}_v^{\mathcal{D}}$ is with respect to digraph $\mathcal{D}$.

Protocols. Given a swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$, a *swap protocol* $\mathbb{P}$ for $\mathcal{S}$ specifies actions of each party over time, in particular it determines how assets change hands. Initially, an asset represented by an arc $(u, v) \in A$ is in the possession of u, and, when $\mathbb{P}$ completes, this asset must be in possession of either u or v. If (u, v) ends up in the possession of v, we will say that the arc (u, v) has been *triggered*. The outcome of v after executing $\mathbb{P}$ is $\langle \omega^{in} \,|\, \omega^{out} \rangle$, where ω^{in} and ω^{out} are the sets of incoming and outgoing arcs of v that are triggered in this execution. In particular, we write $\mathbb{P}(v)$ for the outcome of v in an execution of protocol $\mathbb{P}$ in which all parties follow $\mathbb{P}$. If some party (possibly v itself) deviates from $\mathbb{P}$, we assume that v's outcome is also finalized when $\mathbb{P}$ completes, but it may be different from $\mathbb{P}(v)$.

A protocol may use appropriate cryptographic primitives. In particular, following [19], we assume the availability of *smart contracts*. A smart contract for an arc $a = (u, v)$ allows u to put asset a in an escrow secured with a suitable collection of hashed time-locks: each such time-lock is specified by a pair (h, τ), where $h = H(s)$ is a hashed value of a secret s and τ is a time-out value. In order to unlock this time-lock, v (and only v) must provide the value of s before time τ. If all time-locks of (u, v) are unlocked, v can claim a. This automatically triggers arc (u, v). If any time-lock times out, a is automatically returned to u. We describe a more elaborate hashed time-lock in the next section.

Properties. For a swap protocol to be useful, it must guarantee that if all parties follow it then every party ends in an outcome at least as favorable as trading all their outgoing for all their incoming assets. Further, every conforming party should end up with an acceptable outcome, no matter whether other parties follow the protocol or not. Lastly, rational parties should have no incentive to deviate from the protocol. Herlihy [19] captured these properties using the concepts of uniformity and strong Nash equilibrium. Our definitions, below, are their natural extensions to the more general model of swap systems.

Uniformity. A swap protocol $\mathbb{P}$ is called *uniform* if it satisfies the following two conditions:

Liveness: If all parties follow $\mathbb{P}$, they all end in outcome DEAL *or better*, that is $\mathbb{P}(v) \succeq \text{DEAL}_v$ for all $v \in V$.

Safety: If a party conforms to $\mathbb{P}$, then its outcome will be acceptable, independently of the behavior of other parties.

A less restrictive concept of uniformity may also be of interest: We say that a protocol $\mathbb{P}$ is *weakly uniform* if it satisfies the safety condition above, but the liveness condition is replaced by the following *weak liveness* requirement: if all parties follow $\mathbb{P}$, then at least one party ends in an outcome strictly better than NODEAL. The assumptions on preference posets imply directly that a protocol that is uniform is also weakly uniform.

Nash equilibria and atomicity. We extend the concept of outcomes to sets of parties, where an outcome of a set is just a vector of individual outcomes. On this set we can then define a preference relation in a standard way, via a coordinate-wise ordering of outcomes. Formally, for any set of parties $C \subseteq V$, an *outcome vector of* C is $\bar{\omega} = (\omega_v)_{v \in C}$, where $\omega_v \in \Omega_v$ for all $v \in C$. Denote by $\bar{\Omega}_C$ the set of all outcome vectors of C. Given two outcome vectors $\bar{\omega}, \bar{\omega}' \in \bar{\Omega}_C$, we write $\bar{\omega} \preceq_C \bar{\omega}'$ if $\omega_v \preceq_v \omega'_v$ for all $v \in C$. If also $\bar{\omega} \neq \bar{\omega}'$ then we write $\bar{\omega} \prec_C \bar{\omega}'$. (In other words, $\bar{\omega} \prec_C \bar{\omega}'$ means that at least one party in C does strictly better in $\bar{\omega}'$ than in $\bar{\omega}$, and every party in C does at least as good.) In this notation, if all parties follow a protocol $\mathbb{P}$, then the outcome vector $\mathbb{P}(C)$ of a protocol $\mathbb{P}$ for a set of parties C is $(\mathbb{P}(v))_{v \in C}$.

We will say that a protocol $\mathbb{P}$ is a *strong Nash equilibrium* if no coalition of participating parties can improve its vector outcome by deviating from $\mathbb{P}$; more precisely, for every set C of parties, if $\bar{\omega}$ denotes the outcome vector of C in some execution of $\mathbb{P}$ where all parties in $V \setminus C$ follow $\mathbb{P}$, then we cannot have $\bar{\omega} \succ_C \mathbb{P}(C)$. We will call $\mathbb{P}$ *atomic* if it is both uniform and a strong Nash equilibrium.

Example 1. Consider a swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ whose digraph $\mathcal{D}$ is shown in Figure 1. The preference poset $\mathcal{P}_u$ is generated by two preference pairs $\text{DEAL}_u \prec \langle v \,|\, v \rangle \prec \langle v \,|\, w \rangle$, the preference poset $\mathcal{P}_v$ is generated by two preference pairs $\text{DEAL}_v \prec \langle u \,|\, u \rangle \prec \langle w \,|\, u \rangle$, and the preference poset $\mathcal{P}_w$ is generated by one preference pair $\text{DEAL}_w \prec \langle u \,|\, v \rangle$.

Consider also a swap protocol $\mathbb{P}$ for $\mathcal{S}$ such that if all parties follow $\mathbb{P}$ then all end up with outcome DEAL. Then $\mathbb{P}$ is not a strong Nash equilibrium, because for $C = \{u, v\}$, the parties in C can ignore $\mathbb{P}$ altogether and simply swap their assets between themselves, improving their outcomes. Nevertheless, as we show later in Section IV, $\mathcal{S}$ does have an atomic protocol. Roughly, instead of using the whole digraph $\mathcal{D}$, in this protocol only assets represented by arcs (u, w), (w, v) and (v, u) will be swapped. Then the outcome of each party will be better

[3]This definition can be relaxed to allow some outcomes incomparable to NODEAL be acceptable. In this extended model, the set $\mathcal{A}_v$ of acceptable outcomes would be part of a swap system specification, and would have to satisfy three conditions: (i) $\{\omega : \omega \succeq \text{NODEAL}_v\} \subseteq \mathcal{A}_v$, (ii) $\{\omega : \omega \prec \text{NODEAL}_v\} \cap \mathcal{A}_v = \emptyset$, and (iii) $\omega \in \mathcal{A}_v \wedge \omega \preceq \omega' \Rightarrow \omega' \in \mathcal{A}_v$. Our results can be extended naturally to this model. We adopted the simpler definition to streamline the presentation.

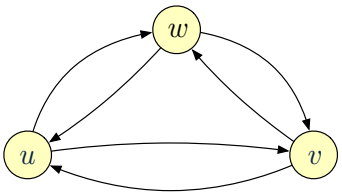

Fig. 1. The digraph $\mathcal{D}$ in the example.

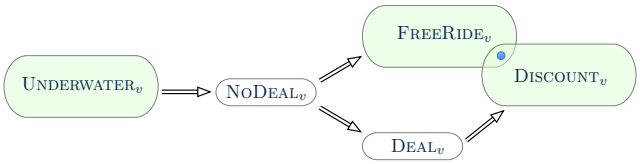

Fig. 2. The structure of a preference poset of a party v in an h-swap system. The arrows symbolize the preference relation. The one outcome in DISCOUNT$_v \cap$ FREERIDE$_v$ is $\langle A_v^{in} \mid \varnothing \rangle$.

than DEAL, and u and v will have no incentive to deviate from this protocol.

III. HERLIHY'S SWAP MODEL

In this section, we show that the concept of swap systems is a generalization of Herlihy's model [19]. To this end, we define a simple type of swap system called h-swap systems, and we show that it captures the model in [19]. In particular we prove that in h-swap systems, our definition of atomicity is equivalent to the definition in [19].

h-Swap Systems. Given a swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ and a party $v \in V$, define three sets of outcomes of v:

$$\begin{aligned} \text{DISCOUNT}_v &= \{\omega \mid \omega^{in} = A_v^{in} \wedge \omega^{out} \neq A_v^{out}\} \\ \text{FREERIDE}_v &= \{\omega \mid \omega^{in} \neq \varnothing \wedge \omega^{out} = \varnothing\} \\ \text{UNDERWATER}_v &= \{\omega \mid \omega^{in} \neq A_v^{in} \wedge \omega^{out} \neq \varnothing\} \end{aligned}$$

Since $A_v^{in} \neq \emptyset$ and $A_v^{out} \neq \emptyset$, all sets DISCOUNT$_v$, FREERIDE$_v$ and UNDERWATER$_v$ are well-defined, none of them contains NODEAL$_v$ nor DEAL$_v$, UNDERWATER$_v \cap$ (DISCOUNT$_v \cup$ FREERIDE$_v$) $= \emptyset$, DISCOUNT$_v \cap$ FREERIDE$_v = \{\langle A_v^{in} \mid \varnothing \rangle\}$, and

$$\begin{aligned} \Omega_v = \{&\text{NODEAL}_v\} \cup \{\text{DEAL}_v\} \cup \text{DISCOUNT}_v \\ &\cup \text{FREERIDE}_v \cup \text{UNDERWATER}_v. \end{aligned}$$

The inclusive monotonicity property (p.2) implies that all outcomes in FREERIDE$_v$ are better than NODEAL$_v$, and all outcomes in DISCOUNT$_v$ are better than DEAL$_v$.

We will call $\mathcal{S}$ an *h-swap system* if it satisfies the following conditions for all $v \in V$:

(h.1) If $\omega \in$ UNDERWATER$_v$ then $\omega \prec_v$ NODEAL$_v$,
(h.2) Party v has no other non-generic preferences besides these in (h.1).

In other words, in an h-swap system all preference posets are generated by relations $\omega \prec$ NODEAL for outcomes ω in UNDERWATER. Figure 2 illustrates the structure of a preference poset of an h-swap system[4]. Note that in an h-swap system, the set of acceptable outcomes of a node v is $\mathcal{A}_v = \Omega_v \setminus$ UNDERWATER$_v = \{$NODEAL$_v\} \cup \{$DEAL$_v\} \cup$ DISCOUNT$_v \cup$ FREERIDE$_v$. The preferences of an h-swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ are uniquely determined by its digraph $\mathcal{D}$, so it is not even necessary to specify $\mathcal{P}$.

The preference poset structure of h-swap systems, as defined above, captures the concept of a party's preferences assumed in the model from [19], except for the addition of preferences determined by inclusive monotonicity.

Comment. The model in [19] was not formulated in terms of posets, raising a question of how to formally capture a relation for pairs of outcomes between which preferences were not specified in [19]. In our model, such outcomes are considered incomparable in the poset (unless they are related by the inclusive monotonicity). One may try to consider another option: to allow arbitrary relations between such pairs, providing that the poset axioms are satisfied and the condition (h.1) holds. However, with this approach there is no meaningful way to extend such individual preferences to collective preferences of sets of parties (see the discussion later in this section).

h-Uniformity. To distinguish between our and Herlihy's definition of uniformity, we will refer to his concept as h-uniformity. A swap protocol $\mathbb{P}$ is called *h-uniform* if it satisfies the safety property and the following *h-liveness* condition: If all parties follow $\mathbb{P}$, they all end in outcome DEAL. This condition seems stricter than our definition of uniformity, but we show that in h-swap systems these two definitions are in fact equivalent. In fact, they are also equivalent to weak uniformity, as defined earlier in Section II.

Lemma 1. *Let $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ be an h-swap system in which some subset of arcs in $\mathcal{D}$ are triggered, and let Q be a path in $\mathcal{D}$ whose all internal nodes are in acceptable outcomes. Then, along Q, all triggered arcs of Q are before all non-triggered arcs of Q.*

Proof. If all arcs on Q are triggered, except possibly for the last one, we are done. Otherwise, let (x, y) be the first non-triggered arc on Q and let z be the successor of y. Since y's outcome is acceptable and (x, y) is not triggered, this outcome must be either NODEAL$_y$ or in FREERIDE$_y$. Therefore (y, z) is also not triggered. Repeating this argument, we obtain that all arcs on Q after (x, y) are not triggered. □

Theorem 1. *Let $\mathbb{P}$ be a swap protocol for an h-swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$, where $\mathcal{D}$ is strongly connected. Then the following three conditions are equivalent: (i) $\mathbb{P}$ is uniform, (ii) $\mathbb{P}$ is weakly uniform, (iii) $\mathbb{P}$ is h-uniform.*

Proof. Trivially, h-uniformity implies uniformity, which in turn implies weak uniformity. Thus it is sufficient to show that weak uniformity implies h-uniformity.

So assume that $\mathbb{P}$ is weakly uniform. As the safety condition is the same, it is sufficient to show that $\mathbb{P}$ satisfies the h-

[4]This figure differs slightly from Figure 3 in [19], which mistakenly showed the sets DISCOUNT$_v$ and FREERIDE$_v$ as disjoint.

liveness property. Assume that all parties follow $\mathbb{P}$. Then, from the assumptions about safety and weak liveness, all parties will end up in acceptable outcomes, with at least one party ending in an outcome strictly better than NoDeal.

Suppose, towards contradiction, that there is a party with outcome other than Deal. This gives us that some arc (x, y) is not triggered. Further, since some party has an outcome other than NoDeal, there must be a triggered arc (x', y'). By strong connectivity, there is a path P from x to y' whose first arc is (x, y) and the last arc is (x', y'). Then the existence of this path contradicts Lemma 1. □

h-Atomicity. The approach in [19] differs from ours in the way it formalizes the gain of a coalition (subset) of parties when they deviate from the protocol. Roughly, the definition in [19] captures a collective gain, while our definition views it as a vector of individual outcomes. In spite of this apparent difference, we show that in h-swap systems our concept of atomicity is in fact equivalent to the one in [19].

In the discussion below, let $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ be a fixed h-swap system. Following [19], we will define the h-outcome of a coalition $\mathcal{C}$ of parties by, in essence, contracting $\mathcal{C}$ into a single vertex. (The term "h-outcome" is ours, to better distinguish this concept from our concept of outcome vectors.) More formally, define $\mathcal{C}$'s incoming and outgoing arcs in a natural way: $A_{\mathcal{C}}^{in} = \bigcup_{v \in \mathcal{C}} A_v^{in} \setminus \bigcup_{v \in \mathcal{C}} A_v^{out}$ and similarly, $A_{\mathcal{C}}^{out} = \bigcup_{v \in \mathcal{C}} A_v^{out} \setminus \bigcup_{v \in \mathcal{C}} A_v^{in}$. The *h-outcomes* for $\mathcal{C}$ are pairs $\hat{\omega} = \langle \hat{\omega}^{in} \,|\, \hat{\omega}^{out} \rangle$ where $\hat{\omega}^{in} \subseteq A_{\mathcal{C}}^{in}$ and $\hat{\omega}^{out} \subseteq A_{\mathcal{C}}^{out}$. $\Omega_{\mathcal{C}}$ is the set of all h-outcomes of C. The preference poset and acceptable set of $\mathcal{C}$ are defined analogously to that of a single party in an h-swap system. That is, we define NoDeal$_C$, Deal$_C$, Discount$_C$, FreeRide$_C$, and Underwater$_C$ in the natural way, and we assume the analogues of conditions (p.1) and (p.2) for swap systems (in Section II) and conditions (h.1) and (h.2) for h-swap systems. The set of acceptable h-outcomes $\mathcal{A}_{\mathcal{C}}$ consists of all h-outcomes of $\mathcal{C}$ that are not in Underwater$_C$. (Note that if $\mathcal{C}$ consists of a single party then its h-outcome is identical to its outcome.)

Define a protocol $\mathbb{P}$ to be a *strong Nash h-equilibrium* if it satisfies the following condition for every set $\mathcal{C}$ of parties: providing that the parties outside $\mathcal{C}$ follow $\mathbb{P}$, the parties in $\mathcal{C}$ cannot end up in an h-outcome better than their outcome resulting from following $\mathbb{P}$. $\mathbb{P}$ is called *h-atomic* if it is h-uniform and a strong Nash h-equilibrium.

Culminating the earlier discussion, the following theorem establishes that our model indeed captures the model introduced in [19].

Theorem 2. *Let $\mathbb{P}$ be a protocol for an h-swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$. $\mathbb{P}$ is atomic if and only if it is h-atomic.*

Proof. ($\Rightarrow$) Suppose that $\mathbb{P}$ is atomic. Theorem 1 implies that $\mathbb{P}$ is h-uniform. Thus, from the definition of h-uniformity, if all parties follow $\mathbb{P}$ then each party's outcome will be Deal.

It remains to show that $\mathbb{P}$ is a strong Nash h-equilibrium. Let $C \subseteq V$, and consider an execution of $\mathbb{P}$ in which all parties outside C follow $\mathbb{P}$. Since $\mathbb{P}$ is a strong Nash equilibrium, the outcome vector of C is not better than $(\text{Deal}_v)_{v \in C}$. Denote by $\hat{\omega}$ the h-outcome of C. We need to show that $\hat{\omega}$ is not better than Deal$_C$.

Towards contradiction, suppose that $\hat{\omega} \succ \text{Deal}_C$. The definition of preference posets for h-outcomes gives us that $\hat{\omega} \in \text{Discount}_C$. Now consider another execution of $\mathbb{P}$ where the parties in C behave just like before, but they also trigger all arcs connecting two members of C. This will not affect the execution of $\mathbb{P}$ for parties outside C. Then the outcome vector $\bar{\omega}$ of C consists of all arcs between C and $V \setminus C$ (in both directions) that are triggered in $\hat{\omega}$, as well as all arcs with both endpoints inside C. Since $\hat{\omega} \in \text{Discount}_C$, each $v \in C$ has all its incoming arcs in $\bar{\omega}$, and there is at least one $u \in C$ that has one arc to $V \setminus C$ that is not in $\bar{\omega}$. So the outcome of each $v \in C$ is either Deal$_v$ or Discount$_v$, and this u's outcome is Discount$_u$. But then $\bar{\omega}$ is better than $(\text{Deal}_v)_{v \in C}$, contradicting the assumption that $\mathbb{P}$ is a strong Nash equilibrium.

($\Leftarrow$) Now suppose that $\mathbb{P}$ is h-atomic; that is, $\mathbb{P}$ is h-uniform and is a strong Nash h-equilibrium. From Theorem 1 we obtain that $\mathbb{P}$ is uniform.

It remains to prove that $\mathbb{P}$ is a strong Nash equilibrium. Let $C \subseteq V$, and consider some execution of $\mathbb{P}$ in which all parties outside C follow $\mathbb{P}$. Since $\mathbb{P}$ is a strong Nash h-equilibrium, the h-outcome of C is not better than Deal$_C$. We need to show that C's outcome vector is not better than $(\text{Deal}_v)_{v \in C}$.

We again argue by contradiction. Suppose that C's outcome vector is $\bar{\omega} \succ (\text{Deal}_v)_{v \in C}$. Then each $v \in C$ has outcome in $\{\text{Deal}_v\} \cup \text{Discount}_v$ and there is some $u \in C$ with outcome in Discount$_u$. This implies that all parties in C have their incoming arcs in $\bar{\omega}$. Further, some outgoing arc of u is not in $\bar{\omega}$, and this arc must go to $V \setminus C$. We consider the h-outcome of C in the same run of $\mathbb{P}$, without changing the behavior of any members of C. (In the h-outcome of C the status of arcs internal to C is not relevant.) Denote this h-outcome by $\hat{\omega}$. Then $\hat{\omega}$ will include the same arcs between C and $V \setminus C$ (in both directions) as in $\bar{\omega}$. The properties of $\bar{\omega}$ established earlier imply that $\hat{\omega} \in \text{Discount}_C$, and thus $\hat{\omega} \succ \text{Deal}_C$, which contradicts our earlier assumption that $\mathbb{P}$ is a strong Nash h-equilibrium. □

Herlihy's Protocol. Herlihy presented a protocol for h-swap systems [19] that is h-atomic. We summarize this protocol that we will refer to as $\mathbb{H}$.

Since the generation and distribution of the swap system is not the focus of this paper, we assume a third-party service that reliably distributes information to the participating parties. The service begins by assembling a swap graph $\mathcal{D}$ and distributing it to every party. Each party p_i then generates and hashes a secret $h_i = \mathsf{hash}(s_i)$ and sends it back to the service.[5] The

[5]Herlihy describes an optimization where the service computes a feedback vertex set for $\mathcal{D}$ (*i.e.* the removal of this set would leave $\mathcal{D}$ acyclic). He refers to these parties as *leaders* and only uses the hashed secrets of these parties in subsequent steps of the protocol. As this is not a necessary step, we will ignore it for simplicity.

service distributes the hashed secrets as a vector $h_0...h_n$ to every party.

The protocol can be broken into two phases, which we call contract creation and secret propagation respectively. The contract creation phase, in essence, realizes $\mathcal{D}$. For every arc $(u, v) \in \mathcal{D}$, party u generates a smart contract with an escrowed asset to counterparty v. Each contract is *hash-locked* by a vector of hashlocks $h_0...h_n$ generated by the given vector of hashed secrets. A particular hashlock h_w on arc (u, v) unlocks when provided a hashkey (s_w, p, σ), where s_w is the preimage of h_w, p is any simple path from v to w (where w is the party that generated secret s_w), and σ is a sequence of signatures $sig(.., sig(s_w, w), .., v)$ backwards along path p. It should be noted that a single hashlock may have multiple hashkeys, as any simple path is acceptable. Hashlocks and hashkeys are also *time-locked*. Each hashkey only remains valid for a certain amount of time, scaling with the length of the path specified within it. Ignoring constant factors, a hashkey remains valid for $|p| \cdot \Delta$ time, where Δ is an upper bound of the time needed for a single step of a party. The longer the path in the hashkey, the longer the hashkey remains valid. A hashlock expires when all of its hashkeys expire, in which the escrowed asset is returned to the sender. That is, the hashkey containing the longest path from the recipient to the generator of the corresponding secret has expired. If all hashlocks in the vector are unlocked, the contract triggers and the escrowed asset is sent to the recipient.

When a party u observes that each of its incoming contracts has been created correctly, it enters the secret propagation phase. Party u first wants to propagate its own secret. This is done by unlocking hashlock h_u on each of their incoming arcs. Specifically, u generates hashkey $(s_u, u, sig(s_u, u))$ and uses this to unlock the corresponding hashlock on each arc in A_u^{in}. Party u also wants to propagate the secrets of others, which they learn by observing their own outgoing arcs. Let u observe on outgoing arc (u, v) that hashlock h_w was unlocked by hashkey (s_w, p, σ). Then u can generate hashkey $(s_w, u + p, sig(\sigma, u))$ and unlock the corresponding hashlock h_w on each arc in A_u^{in}.

Algorithm 1 Herlihy's Protocol For Vertex v

Input: Digraph $\mathcal{D}$, vector $\langle h_0, ..., h_n \rangle$, secret s_v

1: **for** every $(v, w) \in A_v^{out}$ **do** ▷ Phase 1
2: create contract to w hashlocked by $\langle h_0, ..., h_n \rangle$
3: **upon** contract for every $(u, v) \in A_v^{in}$ ▷ Phase 2
4: generate hashkey $k_1 = (s_v, v, sig(s_v, v))$
5: **for** every arc $(u, v) \in A_v^{in}$ **do**
6: unlock hashlock h_v with k_1
7: **while** no timed out hashlock **and** not all assets received **do**
8: **if** new hashkey $k_2 = (s, p, \sigma)$ on $(v, w) \in A_v^{out}$ **then**
9: generate hashkey $k_3 = (s, v + p, sig(\sigma, v))$
10: **for** every arc $(v, w) \in A_v^{in}$ **do**
11: unlock hashlock with k_3

Example 2. Consider the swap graph in Figure 3. Assume each party is given the same vector of hashed secrets. Parties start the protocol by creating their outgoing contracts using this vector. The party x creates the contract (x, v), the party u creates the contracts (u, x) and (u, v), the party v creates the contracts (v, u) and (v, y), and the party y creates the contract (y, u). Then when the party x observes that the contract (u, v) is created, it releases its secret on (u, x). Once the party u observes this secret on its outgoing contract (u, x), it applies it to its incoming contracts (v, u) and (y, u). Similarly, when the parties v and y observe the secret on their outgoing contracts, they apply it to their incoming contracts (x, v) and (v, y) respectively. Thus, the secret s_x is propagated through the whole graph. With similar steps, each other party releases its secret on its incoming contracts, and each secret is propagated by other parties to the rest of the graph. Thus, all secrets are eventually applied to all contracts, and all assets are transferred.

Lemma 2. *Consider the execution of $\mathbb{H}$ on a directed graph $\mathcal{D}$. Let u be a party that follows $\mathbb{H}$. After the execution is complete, for any $e \in A_u^{out}$ of u, e is triggered only if all arcs in A_u^{in} are triggered.*

If e is triggered, every hashlock on e was unlocked. Since u is following $\mathbb{H}$, it will observe whenever a hashlock on e is unlocked. Whenever a hashlock h_i is unlocked, u sees a hashkey $k_1 = (s_i, p, \sigma)$. Then, u can generate a hashkey $k_2 = (s_i, u + p, sig(\sigma, u))$ to post on the corresponding hashlocks h_i for their incoming arcs. Since k_1 was an acceptable hashkey, and increasing the length of a hashkey by 1 means it remains acceptable for Δ more time, u had sufficient time to post k_2 to all arcs in A_u^{in}. We repeat this argument for every hashlock on e, wherein when every hashlock on e is unlocked, every hashlock on every arc in A_u^{in} is also unlocked.

IV. A Characterization of Swap Systems with Atomic Protocols

As shown in [19], all swap systems considered in Herlihy's approach (that is all h-swap systems, in our terminology) have an atomic protocol, providing that the underlying digraph is strongly connected. In our more general model this is not always the case. Consider, for example, a swap system whose digraph is shown in Figure 3. The only non-generic preferences are: $\text{DEAL}_v \prec \langle u \,|\, u \rangle$ for v, and $\text{DEAL}_u \prec \langle v \,|\, v \rangle$ for u. Using the liveness condition for x and y, any atomic protocol needs to trigger arcs (u, x), (x, v), (v, y) and (y, u). But u and v can cooperatively deviate from the protocol by triggering only arcs (u, v) and (v, u), each obtaining a better outcome than if they followed the protocol. So this protocol cannot be a strong Nash equilibrium, and thus is not atomic.

This raises the question as to whether there exists a simple characterization of swap systems that admit atomic protocols. We provide such a characterization in this section. Interestingly enough, we show that if a swap system admits an atomic protocol, then it also admits an atomic protocol that is essentially equivalent to running Herlihy's protocol on a suitable subgraph.

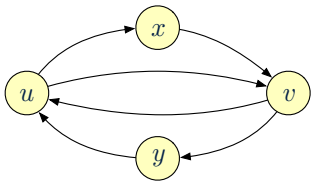

Fig. 3. The example of a swap system in which $\mathbb{H}$ is not a strong Nash equilibrium.

We saw Herlihy's protocol [19], denote by $\mathbb{H}$, in the previous section. Herlihy proved that $\mathbb{H}$ is h-atomic for h-swap systems (in our terminology).

Uniformity and Nash equilibrium of Herlihy's protocol. Let $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ be any swap system with strongly connected digraph $\mathcal{D}$. If all parties follow $\mathbb{H}$ then they all will end up in outcome DEAL. If v follows $\mathbb{H}$ then either it does not trigger any outgoing arcs, and thus its outcome is in $\{\text{NODEAL}_v\} \cup \text{FREERIDE}_v$, or it triggers some, but then also all its incoming arcs are triggered, so its outcome is in $\{\text{DEAL}_v\} \cup \text{DISCOUNT}_v$. In each case, regardless of the behavior of other parties, this outcome is at least as good as NODEAL_v, and thus acceptable. This means that $\mathbb{H}$ is uniform.

We now claim that $\mathbb{H}$ is a *Nash equilibrium* in $\mathcal{S}$, in the sense that no single party can improve its outcome by deviating from $\mathbb{H}$, if all other parties follow $\mathbb{H}$. If all parties follow the protocol, all outcomes are DEAL. If any party v has outcome $\omega \succ \text{DEAL}_v$, then in ω there needs to be an incoming arc (u, v) but some outgoing arc (v, w) must be missing. (This holds for any preference poset, by properties (p.1) and (p.2).) In Herlihy's protocol a vertex triggers an outgoing arc only if all its incoming arcs are triggered. So if all parties other than v follow the protocol, then we get a contradiction by considering a path from w to u (following an argument similar to the proof of Theorem 1).

Note that this argument does not work for larger coalitions. In fact, in the swap system example discussed earlier in this section, $\mathbb{H}$, nor any other protocol, is a strong Nash equilibrium.

Characterization. Let $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ be a swap system for a set of parties V, and let $\mathcal{G}$ and $\mathcal{H}$ be two subgraphs of $\mathcal{D}$. $\mathcal{G}$ will be called *piece-wise strongly connected* if every connected component of $\mathcal{G}$ is strongly connected. $\mathcal{G}$ is called *spanning* if its vertex set is V. If $\mathcal{G}$ is spanning, we say that $\mathcal{H}$ *dominates* $\mathcal{G}$ if $\text{DEAL}_v^{\mathcal{H}} \succeq \text{DEAL}_v^{\mathcal{G}}$ for all vertices v in $\mathcal{H}$. In other words, if only the arcs in $\mathcal{H}$ are triggered, then all parties in $\mathcal{H}$ end in outcomes at least as good as if all their arcs in $\mathcal{G}$ were triggered. Also, $\mathcal{H}$ *strictly dominates* $\mathcal{G}$ if, in addition, there exists a party u of $\mathcal{H}$ such that $\text{DEAL}_u^{\mathcal{H}} \succ \text{DEAL}_u^{\mathcal{G}}$. That is, every party in $\mathcal{H}$ ends in an outcome at least as good and at least one party strictly improves their outcome when triggering the arcs of $\mathcal{H}$ instead of $\mathcal{G}$.

For example, consider the swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ in Example 1. The subgraph $\mathcal{G}_1 = \mathcal{D}$ is spanning, and is strictly dominated by the subgraph $\mathcal{H}$ consisting of vertices u, v and arcs (u, v) and (v, u). On the other hand, the subgraph $\mathcal{G}_2$ that has arcs (u, w), (w, v) and (v, u) is spanning, and there is no subgraph of $\mathcal{D}$ that strictly dominates it.

Theorem 3. *A swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ has an atomic swap protocol if and only if there exists a spanning subgraph $\mathcal{G}$ of $\mathcal{D}$ with the following properties: (c.1) $\mathcal{G}$ is piece-wise strongly connected and has no isolated vertices, (c.2) $\mathcal{G}$ dominates $\mathcal{D}$, and (c.3) no subgraph $\mathcal{H}$ of $\mathcal{D}$ strictly dominates $\mathcal{G}$.*

Proof. ($\Rightarrow$) Let $\mathbb{P}$ be an atomic swap protocol for $\mathcal{S}$. Define $\mathcal{G}$ to be the subgraph whose vertex set is V and whose arcs are the arcs triggered in an execution of $\mathbb{P}$ where all parties follow the protocol. By definition of $\mathbb{P}$'s atomicity, $\mathcal{G}$ is spanning.

We first show property (c.1). First, $\mathcal{G}$ cannot have any isolated vertices, since any isolated vertex v of $\mathcal{G}$ would have outcome $\text{NODEAL}_v^{\mathcal{D}}$ when all parties follow $\mathbb{P}$. This would contradict the uniformity (the liveness condition) of $\mathbb{P}$. Second, if $\mathcal{G}$ had a connected component B that is not strongly connected, then B would contain a strongly connected component C of $\mathcal{G}$ that has no arcs of $\mathcal{G}$ coming from $V \setminus C$ but has at least one arc of $\mathcal{G}$ going to $V \setminus C$. We could then consider another run of $\mathbb{P}$ in which the parties in C ignore $\mathbb{P}$ entirely and simply trigger the arcs of $\mathcal{G}$ that are within C. By the inclusive monotonicity property (p.2) of swap systems, this would strictly improve the outcome vector of C, contradicting $\mathbb{P}$ being a strong Nash equilibrium. We can thus conclude that such B cannot exist, completing the proof that $\mathcal{G}$ is piece-wise strongly connected.

Next, we consider property (c.2). By the uniformity (liveness) of $\mathbb{P}$, every party v must end in outcome $\text{DEAL}_v^{\mathcal{D}}$ or better when all parties follow $\mathbb{P}$. The arcs that are triggered at the conclusion of $\mathbb{P}$ are exactly the arcs in $\mathcal{G}$. Therefore $\text{DEAL}_v^{\mathcal{G}} \succeq \text{DEAL}_v^{\mathcal{D}}$, for all parties v.

Finally, we show property (c.3). Suppose there is a subgraph $\mathcal{H}$ that strictly dominates $\mathcal{G}$, towards contradiction. Let C be the set of vertices of $\mathcal{H}$. Modify the behavior of the parties in C to ignore $\mathbb{P}$ and instead trigger exactly the arcs of $\mathcal{H}$, giving C the outcome vector $(\text{DEAL}_v^{\mathcal{H}})_{v \in C}$. Then, $\mathbb{P}(C) = (\text{DEAL}_v^{\mathcal{G}})_{v \in C} \prec_C (\text{DEAL}_v^{\mathcal{H}})_{v \in C}$, as $\mathcal{H}$ strictly dominates $\mathcal{G}$. This contradicts the assumption that $\mathbb{P}$ is a strong Nash equilibrium, proving that $\mathcal{H}$ does not exists.

($\Leftarrow$) Suppose that $\mathcal{G}$ is a spanning subgraph that satisfies properties (c.1), (c.2) and (c.3). We show that then there is an atomic protocol for $\mathcal{S}$.

Let $\mathcal{S}^{\mathcal{G}}$ be the h-swap system with digraph $\mathcal{G}$. Our protocol, denoted $\mathbb{H}_{\mathcal{G}}$, simply executes Herlihy's protocol $\mathbb{H}$ on $\mathcal{S}^{\mathcal{G}}$. For simplicity, assume that $\mathcal{G}$ is strongly connected; otherwise we can apply our reasoning below to each strongly connected component of $\mathcal{G}$ separately. By the h-liveness condition of $\mathbb{H}_{\mathcal{G}}$, if all parties follow $\mathbb{H}_{\mathcal{G}}$ then each will end up in outcome $\text{DEAL}^{\mathcal{G}}$. Also, any party v that follows $\mathbb{H}_{\mathcal{G}}$ will not have any of its arcs outside $\mathcal{G}$ triggered and, by the safety property of $\mathbb{H}_{\mathcal{G}}$, will end up in an outcome that is acceptable in $\mathcal{S}^{\mathcal{G}}$, that is in $\{\text{NODEAL}_v^{\mathcal{G}}\} \cup \text{FREERIDE}_v^{\mathcal{G}} \cup \{\text{DEAL}_v^{\mathcal{G}}\} \cup \text{DISCOUNT}_v^{\mathcal{G}}$.

When comparing outcomes in the argument that follows, we will use notation "$\prec$" for the preference relation in the original swap system $\mathcal{S}$ (that is, *not* in the auxiliary system

$\mathcal{S}^{\mathcal{G}}$). Similarly, unless stated otherwise, the term "acceptable" also refers to the acceptability of an outcome in $\mathcal{S}$.

We first show that $\mathbb{H}_{\mathcal{G}}$ is uniform. Suppose that every party follows $\mathbb{H}_{\mathcal{G}}$. Then, by the h-uniformity of $\mathbb{H}_{\mathcal{G}}$, the outcome of each party v will be $\text{DEAL}_v^{\mathcal{G}}$. Using the assumptions that $\mathcal{G}$ is spanning and that it dominates $\mathcal{D}$, we obtain that $\text{DEAL}_v^{\mathcal{G}} \succeq \text{DEAL}_v^{\mathcal{D}}$ for all parties v, so $\mathbb{H}_{\mathcal{G}}$ indeed satisfies the liveness condition.

Next, we deal with the safety condition. Using the properties of $\mathbb{H}_{\mathcal{G}}$ established above, if a party v conforms to $\mathbb{H}_{\mathcal{G}}$ then we have two cases. Either the outcome ω of v satisfies $\omega \in \{\text{NODEAL}_v^{\mathcal{G}}\} \cup \text{FREERIDE}_v^{\mathcal{G}}$, in which case $\omega \in \{\text{NODEAL}_v^{\mathcal{D}}\} \cup \text{FREERIDE}_v^{\mathcal{D}}$ as well (because no edges of v outside $\mathcal{G}$ are triggered), so $\omega \succeq \text{NODEAL}_v^{\mathcal{D}}$, that is ω is acceptable. Or $\omega \in \{\text{DEAL}_v^{\mathcal{G}}\} \cup \text{DISCOUNT}_v^{\mathcal{G}}$, in which case, using the monotonicity property (p.2) for $\mathcal{S}$ and assumption (c.2), we obtain $\omega \succeq \text{DEAL}_v^{\mathcal{G}} \succeq \text{DEAL}_v^{\mathcal{D}} \succ \text{NODEAL}_v^{\mathcal{D}}$; that is ω is acceptable in this case as well. We conclude that $\mathbb{H}_{\mathcal{G}}$ satisfies the safety property, completing the proof that $\mathbb{H}_{\mathcal{G}}$ is uniform.

It remains to show that $\mathbb{H}_{\mathcal{G}}$ is a strong Nash equilibrium for $\mathcal{S}$. Assume that it is not, towards contradiction. Then there exists a coalition $C \subseteq V$ that, by deviating from $\mathbb{H}_{\mathcal{G}}$, can end in an outcome vector $\bar{\omega} \succ (\text{DEAL}_v^{\mathcal{G}})_{v \in C}$, even though all parties outside C follow $\mathbb{H}_{\mathcal{G}}$. We can assume C is maximal, in the sense that each party outside of C ends in an outcome that is not $\text{DEAL}^{\mathcal{G}}$ nor in $\text{DISCOUNT}^{\mathcal{G}}$. Otherwise, we can add those parties to C and the relation $\bar{\omega} \succ (\text{DEAL}_v^{\mathcal{G}})_{v \in C}$ will be preserved.

We first show that no arc $(u, v) \in A$ entering C from outside (that is $u \in V \setminus C$ and $v \in C$) is triggered. Assume such an arc is triggered, towards contradiction. Firstly, (u, v) must be in $\mathcal{G}$, otherwise u would not be following $\mathbb{H}_{\mathcal{G}}$ by creating/triggering this arc. By the h-safety property of $\mathbb{H}$ in $\mathcal{S}^{\mathcal{G}}$, $\mathbb{H}_{\mathcal{G}}$ guarantees that u must end up in an outcome acceptable in $\mathcal{S}^{\mathcal{G}}$. This means that u's outcome is in $\{\text{DEAL}_u^{\mathcal{G}}\} \cup \text{DISCOUNT}_u^{\mathcal{G}}$, contradicting the assumption that C is maximal. So, indeed, (u, v) cannot be triggered.

Further, without loss of generality we can assume that no arc from C to $V \setminus C$ is triggered. This is because, as we just showed, for each $v \in C$, v only receives arcs from other parties in C. Then, no member of C can have its outcome worsened if v changes its behavior and does not trigger any arc to $V \setminus C$.

Thus all arcs that appear in $\bar{\omega}$ are between members of C. Let $\mathcal{H}$ be the subgraph with vertex set C and the arcs that are in $\bar{\omega}$, that is $\bar{\omega} = (\text{DEAL}_v^{\mathcal{H}})_{v \in C}$. Since $\bar{\omega} \succ (\text{DEAL}_v^{\mathcal{G}})_{v \in C}$, then $\text{DEAL}_v^{\mathcal{H}} \succeq \text{DEAL}_v^{\mathcal{G}}$ for all $v \in C$ and $\text{DEAL}_w^{\mathcal{H}} \succ \text{DEAL}_w^{\mathcal{G}}$ for some $w \in C$. This means that $\mathcal{H}$ strictly dominates $\mathcal{G}$, contradicting (c.3). We conclude that no such C exists, and thus $\mathbb{H}_{\mathcal{G}}$ is a strong Nash equilibrium protocol. □

Comment: As some readers may have noticed, the proof of the ($\Rightarrow$) implication in Theorem 3 does not use the safety property of protocol $\mathbb{P}$. What this shows, in essence, is that in our setting of swap systems, a swap protocol that has the liveness and strong Nash equilibrium properties can be modified to also satisfy the safety property.

With Theorem 3 established, we can determine if a given swap system $\mathcal{S}$ permits an atomic protocol. Additionally, if it does, we can define such a protocol. Algorithm 2 describes how to check if a given swap protocol permits an atomic protocol. If it does not, it returns -1. Otherwise, it returns a set of strongly connected components. Then, the atomic swap protocol is running $\mathbb{H}$ with each component as the underlying graph.

Algorithm 2 Generalized Atomic Swap Protocol

Input: Swap System $\mathcal{S} = (\mathcal{D}, \mathcal{P})$
Output: set of strongly connected components **or** -1

```
for every spanning subgraph G of D do
    if G is piece-wise strongly connected then          ▷ c.1
        if G dominates D then                           ▷ c.2
            for every subgraph H of D do                ▷ c.3
                if H strictly dominates G then
                    return -1
            return {C | C is an SCC of G}
return -1
```

Example 3. (Example 1 continued) To illustrate Theorem 3, consider again the swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ in Example 1. Let $\mathcal{G}_1 = \mathcal{D}$. Then $\mathcal{G}_1$ is spanning, satisfies conditions (c.1) and (c.2), but it does not satisfy condition (c.3) because it is strictly dominated by subgraph $\mathcal{H}$ consisting of vertices u, v and arcs (u, v) and (v, u). .

But we can take instead subgraph $\mathcal{G}_2$ consisting of arcs (u, w), (w, v) and (v, u). Then $\mathcal{G}_2$ is spanning, satisfies (c.1) and (c.2), and there is no subgraph of $\mathcal{D}$ that strictly dominates $\mathcal{G}_2$, so (c.3) holds as well. Therefore $\mathcal{S}$ has an atomic swap protocol. This protocol will completely ignore arcs outside $\mathcal{G}_2$. It will execute Herlihy's protocol, described in Section III, using $\mathcal{G}_2$ as the underlying graph. In fact, since $\mathcal{G}_2$ is a simple cycle, the full protocol of Herlihy is not needed — the simpler protocol that uses only one leader and does not need signatures can be used instead, see [19]. Roughly, this protocol chooses one leader, say u, who creates its secret, then the smart contracts based on this (hashed) secret are created by parties u, v and w on their outgoing arcs, with decreasing time-out values, and then the counter-parties claim the assets in these contracts, one by one, in the reverse order.

Example 4. We now give a larger example. Consider the swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ with digraph $\mathcal{D}$ in Figure 4 and with preference posets defined as follows. For $i = 1, 2$:

- The preference poset of u_i is generated by $\text{DEAL}_{u_i} \prec \langle v_i \,|\, v_i \rangle$ and $\text{DEAL}_{u_i} \prec \langle u_{3-i} \,|\, u_{3-i} \rangle$.
- The preference poset of v_i is generated by $\text{DEAL}_{v_i} \prec \langle u_i \,|\, u_i \rangle$ and $\text{DEAL}_{v_i} \prec \langle t_{3-i} \,|\, t_i \rangle \prec \langle v_{3-i} \,|\, v_{3-i} \rangle$.
- The preference poset of t_i is generated by $\text{DEAL}_{t_i} \prec \langle v_i \,|\, v_{3-i} \rangle$ and $\text{DEAL}_{t_i} \prec \langle t_{3-i} \,|\, t_{3-i} \rangle$.

We consider three candidates for the spanning sugraph $\mathcal{G}$. One candidate is $\mathcal{G}_1 = \mathcal{D}$. It's obviously spanning and it satisfies conditions (c.1) and (c.2) from Theorem 3. However,

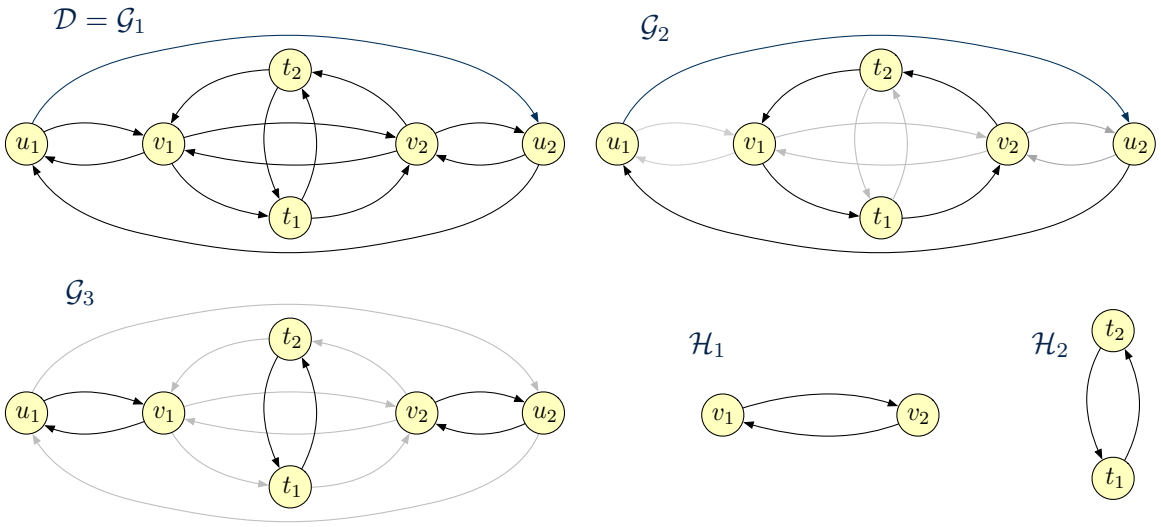

Fig. 4. Digraph $\mathcal{D}$ in the example illustrating Theorem 3.

it is strictly dominated by several subgraphs including the following two: subgraph $\mathcal{H}_1$ consisting of v_1 and v_2 with arcs (v_1, v_2) and (v_2, v_1), and subgraph $\mathcal{H}_2$ consisting of t_1 and t_2 with arcs (t_1, t_2) and (t_2, t_1).

Another candidate $\mathcal{G}_2$ consists of arcs (u_i, u_{3-i}), (v_i, t_i) and (t_i, v_{3-i}). for $i = 1, 2$. $\mathcal{G}_2$ is spanning and it satisfies condition (c.1). By inspecting the preferences of each vertex, it also satisfies (c.2). But it is strictly dominated by $\mathcal{H}_1$.

The third candidate $\mathcal{G}_3$ consists of arcs (u_i, v_i), (v_i, u_i), and (t_i, t_{3-i}), for $i = 1, 2$. It is also spanning and satisfies conditions (c.1) and (c.2). Also, the outcome of each vertex in $\mathcal{G}_3$ is maximal in its preference poset, so there is no subgraph of $\mathcal{D}$ that strictly dominates $\mathcal{G}_3$. Thus, by Theorem 3, $\mathcal{S}$ has an atomic swap protocol. This protocol is obtained by running $\mathbb{H}$ in $\mathcal{G}_3$.

V. NP-Hardness

Now that we have characterized the swap systems that permit an atomic protocol, a natural next question is the complexity of the corresponding decision problem: given a swap system, does it permit an atomic protocol? In the next two sections, we consider this. In this section, we define the corresponding decision problem and show it is in NP-hard. In the following section, we tighten this classification to Σ_2^{P}-complete. Although showing the problem is Σ_2^{P}-complete would imply it is NP-hard, we first present the NP-hardness proof since it is more digestible, and then present the more involved Σ_2^{P}-completeness proof.

Let SwapAtomic be the following decision problem: The input is a swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$, where $\mathcal{D}$ is a (weakly) connected digraph with no vertices of in-degree or out-degree 0. The objective is to decide whether $\mathcal{S}$ has an atomic swap protocol.

Theorem 4. SwapAtomic *is* NP*-hard, even for swap systems* $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ *in which digraph* $\mathcal{D}$ *is strongly connected.*

Proof. The proof is by showing a polynomial-time reduction from CNF. Recall that in CNF we are given a boolean expression α in conjunctive normal form, and the objective is to determine whether there is a truth assignment that satisfies α. In our reduction we convert α into a swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ such that α is satisfiable if and only if $\mathcal{S}$ has an atomic swap protocol.

Let $x_1, x_2, ..., x_n$ be the variables in α. The negation of x_i is denoted $\bar{x}_i$. We will use notation $\tilde{x}_i$ for an unspecified literal of variable x_i, that is $\tilde{x}_i \in \{x_i, \bar{x}_i\}$. Let $\alpha = c_1 \vee c_2 \vee ... \vee c_m$, where each c_j is a clause. Without loss of generality we assume that in each clause all literals are different.

We first describe the reduction. The fundamental approach is similar to other reductions from CNF: $\mathcal{D}$ will consist of so-called "gadgets" which are subgraphs used to simulate the role of variables and clauses. We then add some connections between these gadgets and specify appropriate preference pairs to assure that the resulting swap system $\mathcal{S}$ satisfies the required property, given above.

Specifically, in $\mathcal{D}$ there will be n gadgets corresponding to variables, m gadgets corresponding to clauses, and one more vertex called the *core vertex*. The x_i*-gadget* has vertices x_i and $\bar{x}_i$, that represent the literals of variable x_i, plus two additional vertices s_i and t_i. Its internal arcs are (s_i, x_i), $(s_i, \bar{x}_i)$, (s_i, t_i), and (t_i, s_i). The c_j*-gadget* has two vertices c_j and a_j, with internal arcs (c_j, a_j), (a_j, c_j). (See Figure 5.) We then add the following arcs: For each clause c_j and each literal $\tilde{x}_i$ in c_j, we add arc $(\tilde{x}_i, c_j)$. The core vertex, denoted b, is connected by arcs to and from each vertex in the above gadgets.

Next, we describe the preference posets $\mathcal{P}_v$, for each vertex v in $\mathcal{D}$. (See Figure 6.) As explained in Section II, an outcome $\langle \omega^{in} \,|\, \omega^{out} \rangle$ of a vertex v is specified by lists ω^{in} and ω^{out} of its in-neighbors and out-neighbors, respectively, and any preference poset can be uniquely defined by an appropriate set of generators.

The center vertex b's preference poset is generated by all

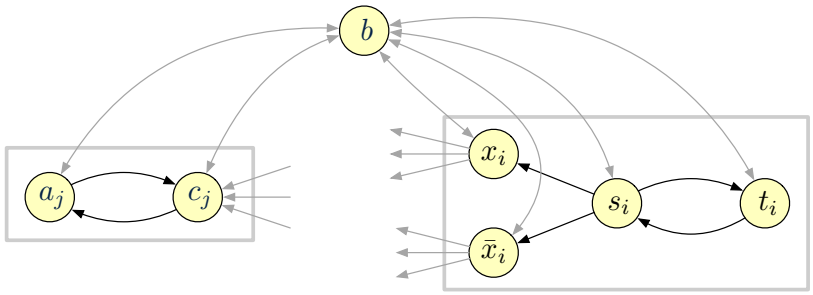

Fig. 5. The variable and clause gadgets in the proof of Theorem 4. The arcs to and from the core vertex b are shown as bi-directional arcs.

relations $\omega \prec \text{NODEAL}_b$, for $\omega \in \text{UNDERWATER}_b$. (This is the same poset as in h-swap systems.) For each vertex s_i, its preference poset is generated by relations $\langle b, t_i \,|\, t_i, \bar{x}_i\rangle \prec \langle t_i \,|\, t_i\rangle$, $\langle t_i \,|\, t_i\rangle \prec \langle b \,|\, b, x_i\rangle$, and $\langle t_i \,|\, t_i\rangle \prec \langle b \,|\, b, \bar{x}_i\rangle$. For each vertex t_i, its generators are $\text{DEAL}_{t_i} \prec \langle b \,|\, b\rangle$ and $\langle b, s_i \,|\, b\rangle \prec \langle s_i \,|\, s_i\rangle$. Each vertex $\tilde{x}_i \in \{x_i, \bar{x}_i\}$ has one generator $\text{DEAL}_{\tilde{x}_i} \prec \langle b \,|\, b\rangle$. Each vertex c_j has generators $\text{DEAL}_{c_j} \prec \langle b, \tilde{x}_i \,|\, b\rangle$, for each literal $\tilde{x}_i$ in c_j. The only generator of each vertex a_j is $\text{DEAL}_{a_j} \prec \langle b \,|\, b\rangle$.

With this, the description of $\mathcal{S}$ is complete. The construction of $\mathcal{S}$ clearly takes time that is polynomial in the size of α.

Applying Theorem 3, it remains to show that α is satisfiable if and only if $\mathcal{D}$ has a spanning subgraph $\mathcal{G}$ with the following properties: (c.1) $\mathcal{G}$ is piece-wise strongly connected and has no isolated vertices, (c.2) $\mathcal{G}$ dominates $\mathcal{D}$, and (c.3) no subgraph $\mathcal{H}$ of $\mathcal{D}$ strictly dominates $\mathcal{G}$.

($\Rightarrow$) Suppose that α is satisfiable, and consider some satisfying assignment for α. Using this assignment, we construct a spanning subgraph $\mathcal{G}$ of $\mathcal{D}$ that satisfies the three conditions (c.1)-(c.3) above.

Graph $\mathcal{G}$ will contain all vertices from the above construction and all arcs that connect b to all other vertices, in both directions. This makes $\mathcal{G}$ spanning and strongly connected, so (c.1) holds. Other arcs of $\mathcal{G}$ are defined as follows. For each true literal $\tilde{x}_i$, add to $\mathcal{G}$ arc $(s_i, \tilde{x}_i)$. For each clause c_j and each true literal $\tilde{x}_i$ in c_j, add to $\mathcal{G}$ arc $(\tilde{x}_i, c_j)$.

Condition (c.2) can be verified through routine inspection, by observing that $\text{DEAL}_v^{\mathcal{D}} \preceq \text{DEAL}_v^{\mathcal{G}}$ holds for each vertex v, directly from the above specification of the preference posets. For example, for a vertex s_i, if $\tilde{x}_i$ is the true literal of x_i then we have $\text{DEAL}_{s_i}^{\mathcal{G}} = \langle b \,|\, b, \tilde{x}_i\rangle \succeq \text{DEAL}_{s_i}^{\mathcal{D}}$. For any $\tilde{x}_i$, let $C(\tilde{x}_i)$ be the set of clauses that contain $\tilde{x}_i$. If $\tilde{x}_i$ is true then $\text{DEAL}_{\tilde{x}_i}^{\mathcal{G}} = \langle b, s_i \,|\, b, C(\tilde{x}_i)\rangle = \text{DEAL}_{\tilde{x}_i}^{\mathcal{D}}$, and if $\tilde{x}_i$ is false then $\text{DEAL}_{\tilde{x}_i}^{\mathcal{G}} = \langle b \,|\, b\rangle \succeq \text{DEAL}_{\tilde{x}_i}^{\mathcal{D}}$. For each clause c_j, denote by $T(c_j)$ the set of true literals in c_j. Since we use a satisfying assignment, each $T(c_j)$ is non-empty. For a clause c_j, for any true literal $\tilde{x}_i \in T(c_j)$, applying the inclusive monotonicity property (p.2) we have $\text{DEAL}_{c_j}^{\mathcal{G}} = \langle b, T(c_j) \,|\, b\rangle \succeq \langle b, \tilde{x}_i \,|\, b\rangle \succeq \text{DEAL}_{c_j}^{\mathcal{D}}$.

It remains to verify condition (c.3). Let $\mathcal{H}$ be a subgraph of $\mathcal{D}$, and suppose that $\mathcal{H}$ dominates $\mathcal{G}$, that is $\text{DEAL}_v^{\mathcal{H}} \succeq \text{DEAL}_v^{\mathcal{G}}$ for all vertices v in $\mathcal{H}$.

We claim first that $\mathcal{H}$ must contain b. Indeed, otherwise for any vertex v of $\mathcal{H}$ we would have $\text{DEAL}_v^{\mathcal{H}} = \omega = \langle \omega^{in} \,|\, \omega^{out}\rangle$ with $b \notin \omega^{in}$ and $\omega \succeq \text{DEAL}_v^{\mathcal{G}}$. The only vertices that have such outcomes ω are $t_i's$. For any t_i, the only outcome ω that has these properties is $\langle s_i \,|\, s_i\rangle$. But then $\mathcal{H}$ would also have to contain s_i contradicting the earlier statement. (We note that $\text{DEAL}_{s_i}^{\mathcal{G}} = \langle b \,|\, b, \tilde{x}_i\rangle$ where $\tilde{x}_i$ is the true literal of x_i, and there is no strictly better outcome for s_i.)

So we can assume from now on that $\mathcal{H}$ contains b. The idea of the remaining argument is to show that the assumption that $\mathcal{H}$ dominates $\mathcal{G}$ implies that in fact $\mathcal{H} = \mathcal{G}$ — so $\mathcal{H}$ cannot strictly dominate $\mathcal{G}$. To this end, we examine the arcs of $\mathcal{D}$ one by one. For each arc (u, v), we use the relation $\text{DEAL}_z^{\mathcal{H}} \succeq \text{DEAL}_z^{\mathcal{G}}$ for some $z \in \{u, v\}$, to show that (u, v) belongs to $\mathcal{H}$ if and only if it belongs to $\mathcal{G}$. We will divide this argument into a sequence of claims.

Claim 1: *$\mathcal{H}$ contains all arcs (v, b) and (b, v), for $v \neq b$.* From the definition of the preference poset of b, $\mathcal{H}$ must contain all incoming arcs of b. This gives us that $\mathcal{H}$ is spanning. For each $v \neq b$, any outcome $\omega \succeq \text{DEAL}_v^{\mathcal{G}}$ that has outgoing arc (v, b) must also have incoming arc (b, v). This proves the claim.

Claim 2: *$\mathcal{H}$ does not contain any arc (a_j, c_j) or (c_j, a_j).* Indeed, no outcome of a_j that has arc (a_j, c_j) is better than $\langle b \,|\, b\rangle = \text{DEAL}_{a_j}^{\mathcal{G}}$, so $\mathcal{H}$ cannot contain (a_j, c_j). Similarly, no outcome of c_j that has arc (c_j, a_j) is better than $\langle b, T(c_j) \,|\, b\rangle = \text{DEAL}_{c_j}^{\mathcal{G}}$, so $\mathcal{H}$ cannot contain (c_j, a_j).

Claim 3: *For each literal $\tilde{x}_i$ in clause c_j, $\mathcal{H}$ contains $(\tilde{x}_i, c_j)$ iff $\tilde{x}_i$ is true.* Suppose first that $\tilde{x}_i$ is a literal in c_j that is true. There is no outcome of c_j that does not contain $(\tilde{x}_i, c_j)$ and is better than $\langle b, T(c_j) \,|\, b\rangle = \text{DEAL}_{c_j}^{\mathcal{G}}$, so $\mathcal{H}$ must contain $(\tilde{x}_i, c_j)$. Next, suppose that $\tilde{x}_i$ is false. Then no outcome of $\tilde{x}_i$ that contains $(\tilde{x}_i, c_j)$ is better than $\langle b \,|\, b\rangle = \text{DEAL}_{\tilde{x}_i}^{\mathcal{G}}$. Thus $\mathcal{H}$ cannot contain $(\tilde{x}_i, c_j)$.

Claim 4: *For each literal $\tilde{x}_i$, $\mathcal{H}$ contains arc $(s_i, \tilde{x}_i)$ iff $\tilde{x}_i$ is true.* Suppose first that literal $\tilde{x}_i$ is true. From the previous claim, we have that the outgoing arcs to $C(\tilde{x}_i)$ are in $\mathcal{H}$. There is no outcome of $\tilde{x}_i$ that contains the arcs to $C(\tilde{x}_i)$, does not contain arc $(s_i, \tilde{x}_i)$, and is better than $\langle b, s_i \,|\, b, C(\tilde{x}_i)\rangle = \text{DEAL}_{\tilde{x}_i}^{\mathcal{G}}$. Thus $\mathcal{H}$ must contain $(s_i, \tilde{x}_i)$. Next, suppose that $\tilde{x}_i$ is false, and let $\bar{\tilde{x}}_i$ be the negation of $\tilde{x}_i$ (that is, the true literal of x_i). Then there is no outcome of s_i that contains arc $(s_i, \tilde{x}_i)$

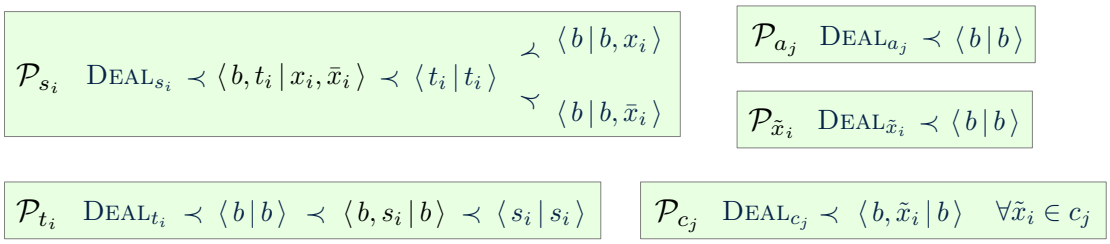

Fig. 6. The specifications of preference posets in the reduction. We include in the figure some generic preferences for s_i and t_i, to illustrate the relationship of some outcomes to the corresponding DEAL outcome.

and is better than $\langle b \,|\, b, \bar{\tilde{x}}_i \rangle = \text{DEAL}^{\mathcal{G}}_{s_i}$. Thus $\mathcal{H}$ cannot contain $(s_i, \tilde{x}_i)$.

Claim 5: *$\mathcal{H}$ does not contain any arc (s_i, t_i) or (t_i, s_i).* There is no outcome of s_i that has an arc (s_i, t_i) and is better than $\langle b \,|\, b, \tilde{x}_i \rangle = \text{DEAL}^{\mathcal{G}}_{s_i}$, where $\tilde{x}_i$ is the true literal of x_i. So $\mathcal{H}$ cannot contain (s_i, t_i). Also, there is no outcome of t_i that has arc (t_i, s_i), does not have arc (s_i, t_i), and is better than $\langle b \,|\, b \rangle = \text{DEAL}^{\mathcal{G}}_{t_i}$. So $\mathcal{H}$ cannot contain (t_i, s_i).

($\Leftarrow$) Assume now that $\mathcal{D}$ has a spanning subgraph $\mathcal{G}$ that satisfies properties (c.1)-(c.3). From $\mathcal{G}$ we will construct a satisfying assignment for α.

Since $\text{DEAL}^{\mathcal{G}}_b \succeq \text{DEAL}^{\mathcal{D}}_b$, $\mathcal{G}$ must contain incoming arcs of b from all other vertices. For each $v \neq b$, any outcome of v that is at least as good as $\text{DEAL}^{\mathcal{D}}_v$ and contains arc (v, b) must also contain arc (b, v). So $\mathcal{G}$ contains all outgoing arcs of b.

Next, we claim that, for each variable x_i, $\mathcal{G}$ contains at most one of arcs (s_i, x_i) and $(s_i, \bar{x}_i)$. Indeed, towards contradiction, suppose that $\mathcal{G}$ contains both arcs (s_i, x_i) and $(s_i, \bar{x}_i)$. The best possible outcome of s_i with both arcs (s_i, x_i) and $(s_i, \bar{x}_i)$ is $\langle b, t_i \,|\, x_i, \bar{x}_i \rangle$, and using the preferences of s_i, we obtain $\langle t_i \,|\, t_i \rangle \succ \langle b, t_i \,|\, x_i, \bar{x}_i \rangle \succeq \text{DEAL}^{\mathcal{G}}_{s_i}$. Regarding t_i, we have already established that t_i has arcs to and from b, and the best such outcome for t_i is $\langle b, s_i \,|\, b \rangle$. Thus, using the preferences of t_i we obtain $\langle s_i \,|\, s_i \rangle \succ \langle b, s_i \,|\, b \rangle \succeq \text{DEAL}^{\mathcal{G}}_{t_i}$. So we could take $\mathcal{H}$ to consist of s_i, t_i, and arcs (s_i, t_i) and (t_i, s_i), and this $\mathcal{H}$ would strictly dominate $\mathcal{G}$, contradicting our assumption that $\mathcal{G}$ satisfies condition (c.3).

Using the claim in the previous paragraph, we construct a satisfying assignment for α as follows: For each variable x_i, set it to true if $\mathcal{G}$ contains (s_i, x_i); otherwise set it to false. (Note that $\mathcal{G}$ may not contain any arc (s_i, x_i), $(s_i, \bar{x}_i)$, in which case we could set the value of x_i arbitrarily.) This truth assignment is well defined.

We now argue that this truth assignment satisfies α. Consider any clause c_j. Vertex c_j must have at least one incoming arc $(\tilde{x}_i, c_j)$ in $\mathcal{G}$, because otherwise we couldn't have $\text{DEAL}^{\mathcal{G}}_{c_j} \succeq \text{DEAL}^{\mathcal{D}}_{c_j}$. Similarly, if $\mathcal{G}$ contains this arc $(\tilde{x}_i, c_j)$ then it must also contain arc $(s_i, \tilde{x}_i)$, because otherwise we couldn't have $\text{DEAL}^{\mathcal{G}}_{\tilde{x}_i} \succeq \text{DEAL}^{\mathcal{D}}_{\tilde{x}_i}$. This implies that literal $\tilde{x}_i$ is true in our truth assignment, so clause c_j is true as well. As this holds for each clause, we can conclude that α is satisfied. □

VI. Σ_2-COMPLETENESS

In the previous section, we showed that SwapAtomic is NP-hard. In this section, we tighten the complexity classification of SwapAtomic, and show that it is in fact Σ_2^{P}-complete. Recall that $\Sigma_2^{\mathsf{P}} = \mathsf{NP}^{\mathsf{NP}}$ is the class of problems at the 2nd level of the polynomial hierarchy that consists of problems solvable non-deterministically in polynomial time with an NP oracle.

Our proof is based on a reduction from a restricted variant of the $\exists\forall$DNF problem. An instance of $\exists\forall$DNF is a boolean expression $\alpha = \exists \mathbf{x} \forall \mathbf{y} \beta(\mathbf{x}, \mathbf{y})$, where $\mathbf{x} = (x_1, ..., x_k)$ and $\mathbf{y} = (y_1, ..., y_l)$ are vectors of boolean variables and $\beta(\mathbf{x}, \mathbf{y})$ is a quantifier-free boolean expression in disjunctive normal form, that is $\beta(\mathbf{x}, \mathbf{y}) = \tau_1 \vee \tau_2 \vee ... \vee \tau_m$, and each term τ_g is a conjunction of literals involving different variables. The goal is to determine whether α is true. $\exists\forall$DNF is a canonical Σ_2^{P}-complete problem [29], [26]. The restriction of $\exists\forall$DNF that we use in our proof, denoted $\exists\forall\mathsf{DNF}_{1x}$, consists of instances $\alpha = \exists \mathbf{x} \forall \mathbf{y} \beta(\mathbf{x}, \mathbf{y})$ where each term of β includes exactly one $\mathbf{x}$-literal and one or more $\mathbf{y}$-literals that involve different variables.

Lemma 3. *$\exists\forall\mathsf{DNF}_{1x}$ is Σ_2^{P}-complete.*

The proof can be found in the supplemental material.

We remember that SwapAtomic is the decision problem of deciding whether a swap system has an atomic protocol.

Theorem 5. *SwapAtomic is Σ_2^{P}-complete.*

The complete proof can be found in the supplemental material. In the remainder of this section, we briefly present a high-level description of our reduction and the accompanying proof.

According to Theorem 3, a swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ has an atomic swap protocol if and only if $\mathcal{D}$ has a spanning subgraph $\mathcal{G}$ with the following properties: (c.1) $\mathcal{G}$ is piece-wise strongly connected and has no isolated vertices, (c.2) $\mathcal{G}$ dominates $\mathcal{D}$, and (c.3) no subgraph $\mathcal{H}$ of $\mathcal{D}$ strictly dominates $\mathcal{G}$. This characterization is of the form $\exists\mathcal{G}. \neg\exists\mathcal{H}. \pi(\mathcal{G}, \mathcal{H})$, where $\pi(\mathcal{G}, \mathcal{H})$ is a polynomial-time decidable predicate, so it immediately implies that SwapAtomic is in Σ_2^{P}. Thus it remains to show that SwapAtomic is Σ_2^{P}-hard.

To prove Σ_2^{P}-hardness, we present a polynomial-time reduction from the above-defined decision problem $\exists\forall\mathsf{DNF}_{1x}$. Let the given instance of $\exists\forall\mathsf{DNF}_{1x}$ be $\alpha = \exists \mathbf{x} \forall \mathbf{y} \beta(\mathbf{x}, \mathbf{y})$, where $\mathbf{x} = (x_1, ..., x_k)$ and $\mathbf{y} = (y_1, ..., y_l)$ are vectors of boolean

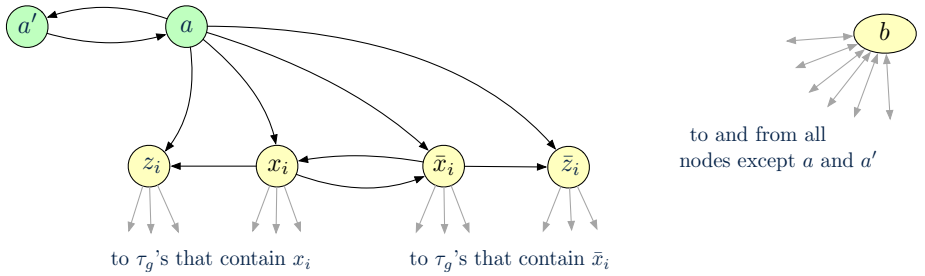

Fig. 7. The construction of digraph $\mathcal{D}$ in the proof of Σ_2^{P}-hardness. This figure shows vertices a, a', b, and an $\exists$-gadget for variable x_i. The arcs to and from b are shown as bi-directional arrows at b.

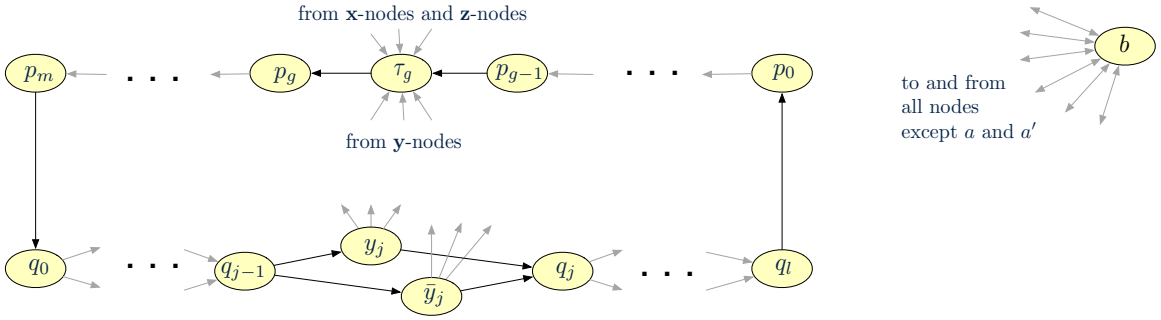

Fig. 8. The construction of digraph $\mathcal{D}$ in the proof of Σ_2^{P}-hardness. This figures shows the $\forall$-gadget, namely the part of $\mathcal{D}$ that contains the vertices that simulate setting the values of the y_j-variables and the terms τ_g. The arcs to and from b are shown as bi-directional arrows at b.

variables and $\beta(\mathbf{x}, \mathbf{y}) = \tau_1 \vee \tau_2 \vee ... \vee \tau_m$, where each τ_g is a conjunction of one $\mathbf{x}$-literal and one or more $\mathbf{y}$-literals. Our reduction converts α into a swap system $\mathcal{S} = (\mathcal{D}, \mathcal{P})$ such that α is true if and only if $\mathcal{D}$ has a spanning subgraph $\mathcal{G}$ that satisfies conditions (c.1)-(c.3) from Theorem 3.

The following informal interpretation of $\exists\forall\mathsf{DNF}_{1x}$ will be helpful in understanding our reduction. Say that a truth assignment to some variables "kills" a term τ_g if it sets one of its literals to false. A truth assignment $\boldsymbol{\phi}$ to the $\mathbf{x}$-variables will kill some terms, while others will survive. Thus α will be true for assignment $\boldsymbol{\phi}$ iff there is no assignment $\boldsymbol{\psi}$ for the $\mathbf{y}$-variables that kills all terms that survived $\boldsymbol{\phi}$. In our reduction, the existence of this assignment $\boldsymbol{\phi}$ will be represented by the existence of subgraph $\mathcal{G}$. The non-existence of $\boldsymbol{\psi}$ that kills all terms that survived $\boldsymbol{\phi}$ will be represented by the non-existence of a subgraph $\mathcal{H}$ that strictly dominates $\mathcal{G}$.

Throughout this section, the negation of a boolean variable x_i will be denoted $\bar{x}_i$. We will also use notation $\tilde{x}_i$ for an unspecified literal of x_i, that is $\tilde{x}_i \in \{x_i, \bar{x}_i\}$. The same conventions apply to the variables y_j.

We now give an overview of our reduction. The digraph $\mathcal{D}$ consists of several "gadgets". There will be $\exists$-gadgets, which correspond to the variables x_i and will be used to set their values, through the choice of subgraphs that $\mathcal{G}$ includes. Then there is the $\forall$-gadget, that contains "sub-gadgets" representing the literals $\tilde{y}_j$ and the terms τ_g. These gadgets will allow for the values of the variables y_j to admit all possible assignments. If any setting of these values kills all terms not yet killed by the variables x_i, this gadget will contain a subgraph $\mathcal{H}$ that strictly dominates $\mathcal{G}$. Figure 7 shows a single $\exists$-gadget and Figure 8 shows the $\forall$-gadget. As we explain the high-level intuition, we gradually visit vertices and explain their purpose.

The argument is based on several ideas. One, we design the preference posets of $\tilde{x}_i$'s so that $\mathcal{G}$ is forced to choose between two possible subsets of arcs within the $\exists$-gadget. The choice between these two subsets of arcs corresponds to choosing a truth assignment for variable x_i. We focus on the literals $\tilde{x}_i$ that are set to false, since these kill the terms where they appear. If $\tilde{x}_i$ is set to false, its arcs to the terms τ_g's in which the literal appears will be included in $\mathcal{G}$ (the first subset), otherwise its arc to $\tilde{z}_i$ will be included in $\mathcal{G}$ (the second subset).

Another idea is that vertices outside of the $\forall$-gadget have their preference posets defined in such a way that their arcs in $\mathcal{G}$ define an outcome that is already the best for them. Therefore, if a subgraph $\mathcal{H}$ that strictly dominates $\mathcal{G}$ does indeed exist, we know it must appear in the $\forall$-gadget. This leads into the key idea of the $\forall$-gadget. The vertices in this gadget can have outcomes that are better than their outcomes in $\mathcal{G}$. All the arcs in these better outcomes together form the cycle

$$\begin{aligned} \mathcal{C} \quad = \quad & q_0 \to \tilde{y}_1 \to ... \to \tilde{y}_l \to q_l \to \\ & p_0 \to \tau_1 \to ...\tau_m \to p_m \to \\ & q_0 \end{aligned} \tag{1}$$

for some choice of the literals $\tilde{y}_1, ..., \tilde{y}_l$. We design the preference posets of each τ_g so that its outcome in $\mathcal{G}$ can only be improved (specifically, towards $\mathcal{C}$) only if it receives an arc from one of its literals — in other words, if it is killed by that literal. This way, $\mathcal{G}$ will have a strictly dominating subgraph $\mathcal{H}$ (namely cycle $\mathcal{C}$) only if all terms are killed, i.e. when α is false.

Next, we provide brief insight to the important vertices and how they help capture the ideas above. Firstly, we want to simulate a truth assignment for variable x_i, which we represent by having $\mathcal{G}$ choose between two subsets of arcs in the corresponding $\exists$-gadget for x_i. Intuitively, one subset corresponds to assigning x_i to true while the other subset corresponds to assigning x_i to false. These two subsets of arcs are established by how we define the preference posets of x_i and $\bar{x}_i$. In order to force $\mathcal{G}$ to make a choice (instead of taking all the arcs), we introduce the auxiliary vertex a, which has arcs to every literal $\tilde{x}_i$. The graph $\mathcal{D}$ has two strongly connected components: (1) a and a', and (2) all the other vertices. We claim graph $\mathcal{G}$ cannot include any arcs from a to the literal vertices. Otherwise, since there is no edge from the second component to the first, dropping those arcs always results in a better outcome for the first component. However, that contradicts condition (c.3). Then, for $\mathcal{G}$ to satisfy condition (c.2), $\mathcal{G}$ is forced to make a choice between the two subsets of arcs. Specifically, $\mathcal{G}$ must choose to include all arcs from x_i to its terms (corresponds to setting x_i to be false) or all arcs from $\bar{x}_i$ to its terms (corresponds to setting x_i to be true).

Next, we want to simulate a term τ_g being killed. We achieve this by designing the preference poset of each τ_g so that if it receives its arc from its $\mathbf{x}$-literal, it would prefer its outcome in the cycle $\mathcal{C}$ over its outcome in $\mathcal{G}$. A term τ_g can also be killed by one of its $\mathbf{y}$-literals, which we describe later.

Now, we want to simulate checking whether there is a truth assignment for the $\mathbf{y}$-variables that make $\forall \mathbf{y} \beta(\boldsymbol{\phi}, \mathbf{y})$ false, where $\boldsymbol{\phi}$ is a truth assignment over the $\mathbf{x}$ variables. In other words, $\mathcal{G}$'s assignment of the $\mathbf{x}$ variables have killed some terms and now we want to see if $\mathcal{H}$ can give an assignment of the $\mathbf{y}$ variables that kill the surviving terms.

First, we need to simulate a truth assignment for each variable y_j. This is simple: we flank y_j and $\bar{y}_j$ by vertices q_{j-1} and q_j as seen in Figure 8. We define the preference posets of q_{j-1} and q_j in a way that only one of y_j or $\bar{y}_j$ can be included in the cycle $\mathcal{C}$, thus forcing $\mathcal{H}$ to choose between them. If $\mathcal{H}$ selects a vertex $\tilde{y}_j$, then $\tilde{y}_j$ will send an arc to every term it appears in. This corresponds to assigning $\tilde{y}_j$ to false.

We additionally define τ_g's poset so that if it receives an arc from any of its $\mathbf{y}$-literals, it wants to join $\mathcal{C}$. At this point, we have represented τ_g being killed if it receives either its $\mathbf{x}$-literal arc or any of its $\mathbf{y}$-literal arcs. (The $\tilde{z}_i$ vertices are actually used for this purpose. They help distinguish between when a term is killed by their $\mathbf{x}$-literal and when a term survived in which it needs to be killed by a $\mathbf{y}$-literal.)

The $\forall$-gadget is designed in the following manner: the preference posets of the q_j, $\tilde{y}_j$, and p_g vertices are such that if every τ_g prefers $\mathcal{C}$, so will they; otherwise, if any τ_g does not prefer $\mathcal{C}$, then none of the vertices can cooperatively deviate to improve their outcomes. In other words, each τ_g acts as a bottleneck for the cycle $\mathcal{C}$, thus we focus only on the τ_g's.

We give an analogy to better understand the remainder of the reduction. Each τ_g is given a vote to whether or not they want to participate in cycle $\mathcal{C}$. In order for $\mathcal{C}$ to pass, it must receive a *unanimous* vote from every τ_g. Vertex τ_g only casts its vote to join $\mathcal{C}$ if it receives an arc from either its $\mathbf{x}$-literal arc or any of its $\mathbf{y}$-literal arcs (corresponds to being killed). Then, $\mathcal{G}$'s selection in each $\exists$-gadget (truth assignment over $\mathbf{x}$ variables) caused *some* τ_g's to vote for $\mathcal{C}$. Now, $\mathcal{H}$ is tasked with selecting vertices $\tilde{y}_1, ..., \tilde{y}_l$ (truth assignment over $\mathbf{y}$ variables) so that the *remaining* τ_g's also vote for $\mathcal{C}$. At the end of this, if the τ_g's unanimously voted for $\mathcal{C}$, then there is an $\mathcal{H}$ that strictly dominates $\mathcal{G}$, namely the subgraph induced by $\mathcal{C}$. (This corresponds to giving an assignment $\mathbf{y} \mapsto \boldsymbol{\psi}$ such that $\beta(\boldsymbol{\phi}, \boldsymbol{\psi})$ is false.) Otherwise, if $\mathcal{H}$ cannot give such a selection over $\tilde{y}_1, ..., \tilde{y}_l$, then there is no $\mathcal{H}$ that strictly dominates $\mathcal{G}$. (This corresponds to $\forall \mathbf{y} \beta(\boldsymbol{\phi}, \mathbf{y})$ being true, i.e. α is true.)

The remaining vertices are primarily used for convenience and to influence the behavior/preference posets of their neighbors. In other words, they make the topology of $\mathcal{G}$ and $\mathcal{H}$ predictable, holding them to a particular form. For example, vertex b is used to guarantee condition (c.1), the piece-wise strong connectivity condition. Also, it is used where vertices would otherwise have no incoming or outgoing arcs.

To conclude the section, we provide some brief insight to both directions of the proof. In the ($\Rightarrow$) implication, we show that if α is true, then the swap graph $\mathcal{D}$ has a subgraph $\mathcal{G}$ that satisfies the properties of Theorem 3. We begin by fixing some truth assignment $\mathbf{x} \mapsto \boldsymbol{\phi}$ that makes α true. We convert $\boldsymbol{\phi}$ into a graph $\mathcal{G}$ that satisfies the properties of Theorem 3 using the ideas described above. Conditions (c.1) and (c.2) can be verified by routine inspection, leaving condition (c.3) that $\mathcal{G}$ does not have a strictly dominating subgraph $\mathcal{H}$. The idea is, towards contradiction, if such an $\mathcal{H}$ existed, we could convert it into an assignment $\boldsymbol{\psi}$ of the $\mathbf{y}$ variables so that $\beta(\boldsymbol{\phi}, \boldsymbol{\psi})$ is false. This contradicts the fact that α is true.

In the ($\Leftarrow$) implication, we show that if $\mathcal{D}$ has a subgraph $\mathcal{G}$ that satisfies the properties of theorem 3, then α is true. We begin by showing that the topology of $\mathcal{G}$ must have a certain form; specifically, it is representative of the graph $\mathcal{G}$ we constructed in the proof of the ($\Rightarrow$) implication. This allows us to reconstruct an assignment of the $\mathbf{x}$ variables. We then show, again by contradiction, that $\forall \mathbf{y} \beta(\boldsymbol{\phi}, \mathbf{y})$ must be true. If it were not, we can take a falsifying assignment $\mathbf{y} \mapsto \boldsymbol{\psi}$ and convert it into a subgraph $\mathcal{H}$ that strictly dominates $\mathcal{G}$. However, this contradicts condition (c.3) of $\mathcal{G}$.

VII. Related Works

The fair exchange problem [23], [16], [7], [4], [5] was of interest even before the blockchain technology. It arises when two parties want to exchange their assets, and the outcome must be either that the two parties end up trading their assets, or that they both keep their assets. However, in contrast to the swap problem, some trust in a third party is often assumed. The optimistic fair exchange protocol [23] relies on invisible trusted parties: parties that work as a background service and intervene only in case of a misbehaviour. Similarly, the secure group barter protocol [16] studies multi-party barter with semi-trusted agents.

To the best of our knowledge, it was back in 2013 when the notion of cross-chain swaps first emerged in an online forum [35]. Atomic cross-chain swap is since an active problem for the blockchain community [8], [35], [9], [10]. The two wiki pages [8] and [35] and later platforms such as deCRED [13] proposed protocols for bilateral swaps. However, these projects offer only two-party transactions. Later, protocols for cross-chain swaps and transactions [19], [20], [18], [34] emerged that can work for an arbitrary number of parties; however, they assumed the predefined preference relation that we saw earlier for all the parties.

These protocols motivated a host of follow-up research. The time and space complexity [21] and privacy guarantees [14] of the protocol were improved. The former [21] uses a model where each asset is assigned two numerical values, one by its current owner and one by the intended recipient. These values can then be used to determine preferences for each party, and can be extended to sets of parties by considering the difference between the total values of incoming and outgoing assets. To assure that their swap graphs have atomic protocols, restrictions (similar in spirit to our Theorem 3) are placed on allowed swap graphs. As we discussed in the introduction, such value-based preferences cannot express dependencies between assets. So the model in [21] would not capture some natural scenarios, for example trades involving assets from an investment portfolio with fixed proportions between different asset classes. Their way of extending individual preferences to coalitions (sets of parties) is different from our model, and it involves a tacit assumption that the coalition members agree on these values. Nevertheless, the approach in [21] is natural and worth studing, and in particular it would be of interest to investigate the time complexity to determine whether a swap graph has an atomic protocol in that model. We suspect that this problem may be computationally easier than for our swap systems.

Further, extensions to support off-chain steps [30] and reduce the asset lock-up time [37] appeared. Others presented hardness and impossibility results [39], [12] formal verification [25], and protocols with all-or-nothing guarantees [38] and success guarantees under synchrony assumptions [36]. Others proposed moving assets [32] and smart contracts [17] across blockchains, and executing code that spans multiple blockchains [28], and presented implementations for industrial blockchains [3], [2], [11], [33].

Payment channel networks process multi-hop payments in the same blockchain through a sequence of channels using Hash Timelock Contracts [27], [1] or adaptor signatures [22]. Recent protocols such as AMCU [15], Sprites [24] and Thora [6] support more general topologies for transactions.

In contrast to previous work, this paper presented a generalized model of swaps where each party can specify a personalized preference on their set of incoming and outgoing assets in a finer manner, e.g. dependencies between subsets of acquired and traded assets.

VIII. Conclusion

We presented a general swap model that allows each party to specify their preference on their possible outcomes. We saw that Herlihy's pioneering protocol is a uniform and Nash strategy in this model; however, it is not a strong Nash strategy. We presented a characterization of the class of swap graphs that have uniform and Strong Nash protocols. Interestingly, Herlihy's protocol is such a strategy when executed on a particular subgraph of the swap graphs in this class. We further presented reductions that shows the NP-harness and Σ_2^{P}-completeness of the decision problem for this class.

Acknowledgements. I would like to thank Annie Semb for her unconditional love and support over the years. I will remember you always. I miss you dearly. *Eric.*

References

[1] Raiden network. https://raiden.network/.

[2] Submarine swap in lightning network. https://wiki.ion.radar.tech/tech/research/submarine-swap.

[3] What is atomic swap and how to implement it. https://www.axiomadev.com/blog/what-is-atomic-swap-and-how-to-implement-it/.

[4] N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for fair exchange. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, CCS '97, pages 7–17, New York, NY, USA, 1997. ACM. URL: http://doi.acm.org/10.1145/266420.266426, doi:10.1145/266420.266426.

[5] N. Asokan, Victor Shoup, and Michael Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communications*, 18:593–610, 1997.

[6] Lukas Aumayr, Kasra Abbaszadeh, and Matteo Maffei. Thora: Atomic and privacy-preserving multi-channel updates. *Cryptology ePrint Archive*, 2022.

[7] Michael Ben-Or, Oded Goldreich, Silvio Micali, and Ronald L. Rivest. A fair protocol for signing contracts. In Wilfried Brauer, editor, *Automata, Languages and Programming*, pages 43–52, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg.

[8] bitcoinwiki. Atomic swap. https://en.bitcoin.it/wiki/Atomic_swap. [Online; accessed 23-January-2021].

[9] bitcoinwiki. Hashed timelock contracts. https://en.bitcoin.it/wiki/Hashed_Timelock_Contracts. [Online; accessed 23-January-2021].

[10] Sean Bowe and Daira Hopwood. Hashed time-locked contract transactions. https://github.com/bitcoin/bips/blob/master/bip-0199.mediawiki. [Online; accessed 23-January-2021].

[11] Gewu Bu, Riane Haouara, Thanh-Son-Lam Nguyen, and Maria Potop-Butucaru. Cross hyperledger fabric transactions. In *Proceedings of the 3rd Workshop on Cryptocurrencies and Blockchains for Distributed Systems*, pages 35–40, 2020.

[12] Eric Chan and Mohsen Lesani. Brief announcement: Brokering with hashed timelock contracts is np-hard. In *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, pages 199–202, 2021.

[13] deCRED. On-chain atomic swaps for decred and other cryptocurrencies. https://github.com/decred/atomicswap. [Online; accessed 27-January-2019].

[14] Apoorvaa Deshpande and Maurice Herlihy. Privacy-preserving cross-chain atomic swaps. In *International Conference on Financial Cryptography and Data Security*, pages 540–549. Springer, 2020.

[15] Christoph Egger, Pedro Moreno-Sanchez, and Matteo Maffei. Atomic multi-channel updates with constant collateral in bitcoin-compatible payment-channel networks. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 801–815, 2019.

[16] Matt Franklin and Gene Tsudik. Secure group barter: Multi-party fair exchange with semi-trusted neutral parties. In Rafael Hirchfeld, editor, *Financial Cryptography*, pages 90–102, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[17] Enrique Fynn, Alysson Bessani, and Fernando Pedone. Smart contracts on the move. In *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 233–244. IEEE, 2020.

[18] Ethan Heilman, Sebastien Lipmann, and Sharon Goldberg. The arwen trading protocols. In *International Conference on Financial Cryptography and Data Security*, pages 156–173. Springer, 2020.

[19] Maurice Herlihy. Atomic cross-chain swaps. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, PODC '18, pages 245–254, New York, NY, USA, 2018. ACM. URL: http://doi.acm.org/10.1145/3212734.3212736, doi:10.1145/3212734.3212736.

[20] Maurice Herlihy, Barbara Liskov, and Liuba Shrira. Cross-chain deals and adversarial commerce. *Proc. VLDB Endow.*, 13(2):100–113, October 2019.

[21] Soichiro Imoto, Yuichi Sudo, Hirotsugu Kakugawa, and Toshimitsu Masuzawa. Atomic cross-chain swaps with improved space and local time complexity. In *International Symposium on Stabilizing, Safety, and Security of Distributed Systems*, pages 194–208. Springer, 2019.

[22] Giulio Malavolta, Pedro Moreno-Sanchez, Clara Schneidewind, Aniket Kate, and Matteo Maffei. Anonymous multi-hop locks for blockchain scalability and interoperability. *Cryptology ePrint Archive*, 2018.

[23] Silvio Micali. Simple and fast optimistic protocols for fair electronic exchange. In *Proceedings of the Twenty-second Annual Symposium on Principles of Distributed Computing*, PODC '03, pages 12–19, New York, NY, USA, 2003. ACM. URL: http://doi.acm.org/10.1145/872035.872038, doi:10.1145/872035.872038.

[24] Andrew Miller, Iddo Bentov, Surya Bakshi, Ranjit Kumaresan, and Patrick McCorry. Sprites and state channels: Payment networks that go faster than lightning. In *International Conference on Financial Cryptography and Data Security*, pages 508–526. Springer, 2019.

[25] Zeinab Nehaï, François Bobot, Sara Tucci-Piergiovanni, Carole Delporte-Gallet, and Hugues Fauconnier. A tla+ formal proof of a cross-chain swap. In *23rd International Conference on Distributed Computing and Networking*, pages 148–159, 2022.

[26] Christos Papadimitriou. *Computational Complexity*. Addison-Wesley Publishing Company, 1994.

[27] Joseph Poon and Thaddeus Dryja. The bitcoin lightning network: Scalable off-chain instant payments, 2016.

[28] Peter Robinson and Raghavendra Ramesh. General purpose atomic crosschain transactions. In *IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2021, Sydney, Australia, May 3-6, 2021*, pages 1–3. IEEE, 2021. doi:10.1109/ICBC51069.2021.9461132.

[29] Marcus Schaefer and Christopher Umans. Completeness in the polynomial-time hierarchy: a compendium. *Sigact News*, 33(3):32–49, 2002.

[30] Narges Shadab, Farzin Houshmand, and Mohsen Lesani. Cross-chain transactions. In *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–9. IEEE, 2020.

[31] M. Sharir. A strong-connectivity algorithm and its applications in data flow analysis. *Computers Mathematics with Applications*, 7(1):67–72, 1981. doi:https://doi.org/10.1016/0898-1221(81)90008-0.

[32] Marten Sigwart, Philipp Frauenthaler, Christof Spanring, and Stefan Schulte. Decentralized cross-blockchain asset transfers. *arXiv preprint arXiv:2004.10488*, 2020.

[33] Sri AravindaKrishnan Thyagarajan, Giulio Malavolta, and Pedro Moreno-Sánchez. Universal atomic swaps: Secure exchange of coins across all blockchains. *Cryptology ePrint Archive*, 2021.

[34] Sri AravindaKrishnan Thyagarajan, Giulio Malavolta, and Pedro Moreno-Sanchez. Universal atomic swaps: Secure exchange of coins across all blockchains. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1299–1316. IEEE, 2022.

[35] Tier Nolan. Alt chains and atomic transfers. https://bitcointalk.org/index.php?topic=193281.msg2224949#msg2224949, 2013. [Online; accessed 23-January-2021].

[36] Rob van Glabbeek, Vincent Gramoli, and Pierre Tholoniat. Feasibility of cross-chain payment with success guarantees. In *Proceedings of the 32nd ACM Symposium on Parallelism in Algorithms and Architectures*, pages 579–581, 2020.

[37] Yingjie Xue and Maurice Herlihy. Hedging against sore loser attacks in cross-chain transactions. In *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, PODC'21, page 155–164, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3465084.3467904.

[38] Victor Zakhary, Divyakant Agrawal, and Amr El Abbadi. Atomic commitment across blockchains. *Proceedings of the VLDB Endowment*, 13(9).

[39] Alexei Zamyatin, Mustafa Al-Bassam, Dionysis Zindros, Eleftherios Kokoris-Kogias, Pedro Moreno-Sanchez, Aggelos Kiayias, and William J Knottenbelt. Sok: Communication across distributed ledgers. In *International Conference on Financial Cryptography and Data Security*, pages 3–36. Springer, 2021.