A THEORY OF AUTOMATED MARKET MAKERS IN DEFI

MASSIMO BARTOLETTI [a], JAMES HSIN-YU CHIANG [b], AND ALBERTO LLUCH-LAFUENTE [b]

[a] University of Cagliari, Cagliari, Italy
  e-mail address: bart@unica.it

[b] Technical University of Denmark, DTU Compute, Copenhagen, Denmark
  e-mail address: jchi@dtu.dk, albl@dtu.dk

ABSTRACT. Automated market makers (AMMs) are one of the most prominent decentralized finance (DeFi) applications. AMMs allow users to trade different types of crypto-tokens, without the need to find a counter-party. There are several implementations and models for AMMs, featuring a variety of sophisticated economic mechanisms. We present a theory of AMMs. The core of our theory is an abstract operational model of the interactions between users and AMMs, which can be concretised by instantiating the economic mechanisms. We exploit our theory to formally prove a set of fundamental properties of AMMs, characterizing both structural and economic aspects. We do this by abstracting from the actual economic mechanisms used in implementations, and identifying sufficient conditions which ensure the relevant properties. Notably, we devise a general solution to the *arbitrage problem*, the main game-theoretic foundation behind the economic mechanisms of AMMs.

1. INTRODUCTION

Decentralized finance (DeFi) is a software infrastructure, based on blockchains and smart contracts, which allows users to create and trade crypto-tokens without the intermediation of central authorities, unlike traditional finance [WPG+21, QZA+21]. *Automated Market Makers (AMMs)* are one of the main DeFi archetypes: roughly, AMMs are decentralized markets of crypto-tokens, providing users with three core operations: depositing crypto-tokens to obtain shares in an AMM; the dual operation of redeeming shares in the AMM for the underlying tokens; and swapping tokens of a given type for tokens of another type. The amount of tokens received by a user upon a swap is algorithmically determined by the AMM: roughly, this is the amount of tokens sent from the user to the AMM, times the *swap rate*, which is computed by the AMM based on its internal state and the input amount.

Despite the apparent simplicity of these operations, AMMs manifest an emerging behaviour, where users are incentivized to swap tokens to keep their swap rates aligned with the *exchange rate*, i.e. the ratio between the prices of the exchanged tokens given by external price oracles. Namely, if an AMM offers a better swap rate than the oracles' exchange rate, rational users will perform swaps to narrow the gap. Formally, the optimal strategy can be seen as the solution of a game, called the *arbitrage game*. Executing the optimal strategy closes the gap between AMM's and oracles' exchange rates, and in this sense AMMs offer users exhange rates that align towards the external, global exchange rates.

As of December 2022, the two AMM platforms leading by user activity, Uniswap and Curve Finance, alone hold \$3B and \$4B worth of tokens, and process \$1B and \$250M worth of transactions daily [uni22, cur22]. Although this massive adoption could suggest that AMMs are a consolidated, well-understood technology, in practice their economic mechanisms are inherently hard to design and implement. For instance, interactions with AMMs are sensitive to *transaction ordering attacks*, where actors with the power to influence the order of transactions in the blockchain can profit from an opportunistic behaviour, causing detriment to other users. The relevance of attacks to AMMs is witnessed by the proliferation of scientific literature on the topic [BCL22, ZQT$^+$21, DGK$^+$20, EMC20, QZG21]. Still, attacks to DeFi applications are not purely theoretical: indeed, there is a growing history of DeFi incidents, which have caused losses exceeding \$2.4B [def22] so far. These issues witness a need for foundational work to devise formal theories of AMMs which allow the understanding of their structural properties and of their economic incentive mechanisms.

Current descriptions of AMMs are either economic models [AKC$^+$21, AC20, EAC21, AEC20], which focus on the incentive mechanism alone, or concrete AMM implementations. While economic models are useful to understand the macroscopic financial aspects of AMMs, they do not precisely describe the interactions between AMMs and their users. Still, a precise formalisation of these interactions is fundamental to understand the structural and economic properties of AMMs, and to determine possible deviations from safe behaviour. Implementations, instead, reflect the exact behaviour of AMMs, but at a level of detail that hampers high-level understanding and reasoning. Moreover, the rich variety of implementations, proposals and models for AMMs, each featuring different economic mechanisms, makes it difficult to compare AMM designs or to provide a clear contour for the space of possible "well behaving" designs.

1.1. **Contributions.** In this paper we exploit techniques from concurrency theory to provide a formal backbone for AMMs and to study their fundamental properties. More specifically, our main contributions can be summarised as follows:

(1) We introduce a formal model of AMMs (section 2), which distils the common features of leading AMM implementations like Uniswap [uni21], Curve [cur21b], and Balancer [bal19]. The core of our model is a transition system that describes the evolution of AMM states resulting from the interaction between users and AMMs. A peculiar aspect of our model is that it abstracts from the *swap rate function*, a key economic mechanism of AMMs, which is used to determine the exchange rates between tokens.

(2) Building upon our model, in section 3 we define basic economic notions like token prices, exchange rates, slippage, net worth, and gain. We compute the gain resulting from swap actions (Lemma 3.2), and we establish a key relation between the gain of swap actions, the swap rate and the exchange rate: a swap action has a strictly positive gain *if and only if* the swap rate is strictly greater than the exchange rate between the swapped tokens (Lemma 3.3). Both lemmata are instrumental to prove many subsequent results.

(3) In section 4 we establish a set of structural properties of AMMs. In particular, we establish preservation results for the supply of tokens (Lemma 4.3) and for the global net worth (Lemma 4.5). We show that assets cannot be frozen within AMMs, i.e. users can always extract any amount of the token reserves deposited in AMMs (Lemma 4.8). In Lemma 4.9 we investigate when transactions can be reordered without affecting the resulting state. In Theorems 4.10 and 4.11 we study compositionality of deposit

and redeem transactions: in particular, we establish that two deposit actions on the same AMM can be merged in a single action (and similarly for two redeems), and that the effect of deposits and redeem actions can be reverted by suitable transactions. Remarkably, all the structural properties in section 4 do not depend on the choice of the swap rate function.

(4) In section 5 we devise sufficient conditions on swap rate functions that induce good behavioural properties of AMMs. These conditions allow us to extend to swap actions the additivity and reversibility properties enjoyed by deposit and redeem actions (Theorems 5.6 and 5.9), as well as to compute the gain of composed and reversed swaps (Lemmata 5.7 and 5.10). We study the effect of deposits and redeems on the swap rate and on the internal exchange rate (Lemma 5.13). We then study the properties of three notable swap rate functions: the constant sum, the constant product, and the constant mean.

(5) In section 6 we investigate the incentive mechanism of AMMs. We start by considering the *arbitrage problem*, which requires to find the action which maximizes the gain of a user. Performing such optimal action has the side effect of aligning the internal exchange rate of the AMM to the external exchange rate given by token price oracles. This gives one of the landmark economic properties of AMMs: assuming rational users, AMMs can be seen as price oracles themselves [AC20]. Notably, while solutions to the arbitrage problem are already known for specific swap rate functions, in Theorem 6.3 we generalize the result to any swap rate function respecting the conditions given in section 5. We then show that depositing tokens into AMMs incentivizes subsequent swaps (Theorem 6.6), while redeeming tokens disincentivizes them (Theorem 6.9). Finally, in Theorems 6.8 and 6.10 we relate the solution of the arbitrage problem in the states before and after a deposit or redeem action, and we compare their gains.

(6) In section 7 we discuss Maximal Extractable Value (MEV), a class of attacks where miners exploit their power of dropping and reordering user transactions (and inserting their own) to increase their gain to the detriment of users. These attacks are one of the most carefully studied AMM phenomena, occuring widely in practice and frequently making up the bulk of interactions with AMMs [QZG21]. The fact that our AMM model can accurately express these attacks supports the coherence of our modelling choices with respect to behaviour exhibited by actual AMM implementations.

(7) In section 8 we discuss some extensions to our basic AMM model to make it closer to the implementation of Uniswap [uni21], and their impact on the results in the paper.

(8) As a byproduct, we provide an open-source Ocaml implementation of our executable semantics as a companion of this paper.[1]

(9) We provide full proofs of all our statements in the Appendices.

1.2. **Related Work.** The work [AKC+21] proposed one of the first analyses of the incentive mechanism of Uniswap. This analysis was then generalised in [AC20] to *constant function AMMs (CFMMs)*, where, for a pair of token types, the reserves r_0, r_1 before a swap and the reserves r'_0, r'_1 after the swap must preserve the invariant $f(r_0, r_1) = f(r'_0, r'_1)$, for a given trading function f. Constant product AMMs, like Uniswap, are an instance of CFMMs, where $f(x, y) = xy$. Both works study the arbitrage problem, for constant product AMMs and CFMMs, respectively. The two works show that the solution can be efficiently computed,

---

[1]`https://github.com/blockchain-unica/defi-workbench`

and suggest that constant product AMMs accurately report exchange rates. Our work and [AC20] share a common goal, i.e. a theory of AMMs generalizing that of constant product AMMs. However, the two approaches are quite different. The work [AC20] considers a class of AMMs, i.e. CFMMs with a convex trading set, and studies the properties enjoyed by AMMs under these assumptions. Instead, in this paper we devise a minimal set of properties of the swap rate function which induce good behavioural properties of AMMs. Notably, we find conditions on the swap rate function which ensure that a given swap action maximizes the gain of the player (Theorem 6.3). Another difference is that the AMM model in [AC20] describes the evolution of a single AMM, abstracting away the other components of the state (i.e. the users and the other AMMs); instead, we model AMMs as *reactive systems*, borrowing techniques from concurrency theory. While the approach followed by [AC20] is still adequate to study problems that concern AMMs in isolation (e.g., arbitrage), viewing AMMs as reactive systems allows us to study what happens when many agents (users and AMMs) can interact. E.g., we are able to reason about Maximal Extractable Value (section 7).

The work [DKP21] generalises the arbitrage problem to the setting where a swap between two token types τ_0 and τ_n can be obtained through a sequence of n intermediate swaps between τ_i and τ_{i+1}, for $0 \leq i < n$. In practice, this represents the situation where users can interact with different AMM platforms, each one providing its own set of token pairs. To model this scenario, [DKP21] introduces *exchange networks*, i.e. multi-graphs where nodes are tokens, and edges are AMMs which allow users to swap the two endpoint tokens. To encompass different AMM platforms, each edge has its own price function, which determines how many output tokens are paid for a given amount of input tokens. The authors show that, under some conditions on the price functions (i.e., monotonicity, continuity, boundedness and concavity), the arbitrage problem always admits a non-trivial solution. In the special case of constant product AMMs, a closed formula for the solution is provided. Besides arbitrage, [DKP21] also considers the *optimal routing problem*, i.e. finding a strategy to maximize the amount of tokens τ_1 received for at most a given amount of tokens τ_0. Under the same assumptions on the price function used for the arbitrage problem, the optimal routing problem admits a solution. There are several differences between our approach and that of [DKP21], besides the fact that we assume the same swap rate function for all AMMs, and a graph instead of a multi-graph (i.e., we admit at most one AMM for each token pair). A technical difference is that we assume that the amount y of output tokens received for an amount x of input tokens is given by $y = SX(x, r_0, r_1) \cdot x$, whereas [DKP21] defines this amount as $y = f_{r_0, r_1}(x)$. This results in different structural properties for $SX(x, r_0, r_1)$ and $f(x)$ in order to achieve the desired behavioural properties of AMMs. Having the AMM reserves r_0, r_1 as parameters of our swap rate functions SX has a benefit, in that we can express conditions which relate states before and after a transaction: this is what happens, e.g., in the additivity, reversibility and homogeneity properties (Definitions 5.5, 5.8 and 5.11). As a consequence of this choice, compared to [DKP21] our theory encompasses also deposit and redeem actions, providing results that clarify how these actions interfere with swaps (e.g., Theorems 6.6, 6.9, 6.8, and 6.10).

A few alternatives to constant product AMMs have been studied. Balancer [bal19] generalizes the constant product function used by Uniswap to a constant (weighted geometric) mean $f(r_1, \cdots, r_n) = \prod_{i=1}^{n} r_i^{w_i}$, where the weight w_i reflects the relevance of a token τ_i in a tuple of tokens $(\tau_1, \cdots, \tau_n)$. This still fits within the CFMM setting of [AC20], thus inheriting its results about solvability of the arbitrage problem [EAC21]. Curve [Ego19] features a hybrid

of a constant sum and constant product function, optimized for large swap volumes between *stable coins*, where the swap rate can support large amounts with small sensitivity. To efficiently compute swap rates, implementations perform numerical approximations [cur21a]. Should these approximations fail to converge, these implementations still guarantee that the AMM remains liquid. The work [KFG21] proposes a constant product invariant that is adjusted dynamically based on the oracle price feed, thus reducing the need for arbitrage transactions, but at the cost of lower fee accrual. AMMs with *virtual* balances have been proposed [vir18] and implemented [moo20b,moo20a]. In these AMMs, the swap rate depends on past actions, besides the current funds balances in the AMM. This, similarly to [KFG21], aims to minimize the need for arbitrage transactions to ensure the local AMM swap rate tends towards the exchange rates. Establishing whether these sophisticated swap rate functions enjoy the properties in section 5 is an interesting open problem.

AMMs are well-known to suffer from transaction-ordering attacks, through which an adversary with the power of influencing the order of transactions (e.g., a miner) can extract value from user transactions. For instance, if the transaction pool contains a swap transaction sent by user A, then a miner M can extract value from A's swap through a transaction "sandwich" constructed as follows. First, M front-runs A's swap with its own swap, crafted so that A's swap decreases A's net worth as much as possible. Then, M closes the sandwich by appending another swap transaction which maximizes M's gain, and finalises the whole sandwich on the blockchain. In this way, A will always have a negative gain, which is counterbalanced by a positive gain of M. This and other kinds of attacks have fostered the research on adversarial and defensive strategies, and on empirical analyses of the impact of attacks [BCL22,CAE22,ZQT+21,QZG21,DGK+20,EMC20]. For instance, the work [BCL22] devises an optimal strategy through which an adversary can extract the maximal value from users' transactions (not only swaps, but also deposits and redeems), in the setting of Uniswap-like AMMs. The swap-rate-agnostic approach pursued by this paper could be exploited to generalise the attack of [BCL22] to AMMs beyond Uniswap.

A high-level survey on various AMM protocols is in [XVPC22].

Comparison with previous work. A preliminary version of this work was presented at COORDINATION 2021 [BCL21b]. The current version substantially extends it, streamlining the theory and providing additional results. A crucial difference between the two papers is that, while in [BCL21b] the semantics of swap actions was parameterized by an invariant between the old and the new token reserves, here we make the semantics parametric w.r.t. the swap rate function SX. This leads to a substantial simplification of the conditions that are put to obtain nice behavioural properties of swaps, and consequently of the corresponding proofs. Among the new results w.r.t. [BCL21b], we mention in particular the additivity and reversibility properties (Theorems 4.10, 4.11, 5.6, and 5.9), and the results that relate the gain of swaps before and after deposit/redeem actions (Theorems 6.6, 6.9, 6.8, and 6.10). Besides these extensions, the current paper includes a discussion of the constant sum and of the constant mean swap rate functions, a new section on MEV attacks (see section 7), and it provides detailed proofs for all its statements.

2. A formal model of Automated Market Makers

We introduce a formal, operational model of AMMs, which focusses on the common operations implemented by AMM platforms. In order to simplify the resulting theory, our model abstracts from a few features that are often found in AMM implementations, like e.g. fees, price updates, and guarded transactions. We discuss in §8 how to extend our model to make it closer to the Uniswap protocol [uni21].

We introduce here some general notation. We denote by fx the application of a function f to a value x (we use parentheses, e.g. $f(x)$, to resolve ambiguities). We denote with $\mathrm{dom}\, f$ the domain of f. We use the standard notation $f\{v/x\}$ to update a partial map f at point x: namely, $f\{v/x\}(x) = v$, while $f\{v/x\}(y) = fy$ for $y \neq x$.

2.1. AMM basics.

Tokens. We assume a set $\mathbb{T}_0$ of **atomic token types**, which represent native cryptocurrencies and application-specific tokens. For instance, $\mathbb{T}_0$ may include ETH, the native cryptocurrency of Ethereum, and WBTC, i.e. Bitcoins wrapped with the ERC20 interface for Ethereum tokens. A **minted token type** is an unordered pair of distinct atomic token types: if τ_0 and τ_1 are atomic token types and $\tau_0 \neq \tau_1$, then the minted token type $\{\tau_0, \tau_1\}$ represents shares in an AMM holding reserves of τ_0 and τ_1. We denote by $\mathbb{T}_1$ the set of minted token types. In our model, tokens are *fungible*, i.e. individual units of the same type are interchangeable. This means that amounts of tokens of the same type can be split into smaller parts, and two amounts of tokens of the same type can be joined. We use v, v', r, r', x, x' to range over nonnegative real numbers ($\mathbb{R}_{\geq 0}$). We write $\mathbb{T}$ for the universe of all token types, i.e. $\mathbb{T} = \mathbb{T}_0 \cup \mathbb{T}_1$, and we use $\tau, \tau', \ldots$ to range over $\mathbb{T}$. We write $r : \tau$ to denote r units of a token of type τ, either atomic or minted.

Wallets and AMMs. We assume a set of **users** $\mathbb{A}$, ranged over by $\mathsf{A}, \mathsf{A}', \ldots$ We model the **wallet** of a user A as a term $\mathsf{A}[\sigma]$, where the finite partial map $\sigma \in \mathbb{T} \rightharpoonup \mathbb{R}_{\geq 0}$ represents A's token balance. We model an **AMM** holding reserves of $r_0 : \tau_0$ and $r_1 : \tau_1$ (with $\tau_0 \neq \tau_1$) as an unordered pair $\{r_0 : \tau_0, r_1 : \tau_1\}$. Since the order of the token reserves in an AMM is immaterial, the terms $\{r_0 : \tau_0, r_1 : \tau_1\}$ and $\{r_1 : \tau_1, r_0 : \tau_0\}$ denote exactly the same AMM.

States. We model the interaction between users and AMMs as a labelled transition system (LTS). Its labels represent blockchain **transactions**, while the **states** $\Gamma, \Gamma', \Delta, \ldots$ are finite non-empty compositions of wallets and AMMs. Formally, states are terms of the form:

$$\mathsf{A}_1[\sigma_1] \mid \cdots \mid \mathsf{A}_n[\sigma_n] \mid \{r_1 : \tau_1, r_1' : \tau_1'\} \mid \cdots \mid \{r_k : \tau_k, r_k' : \tau_k'\}$$

and subject to the following conditions. For all $i \neq j$:

(1) $\mathsf{A}_i \neq \mathsf{A}_j$ (each user has a single wallet);
(2) $\{\tau_i, \tau_i'\} \neq \{\tau_j, \tau_j'\}$ (distinct AMMs cannot hold exactly the same token types).

Note that these conditions allow AMMs to have a common token type τ, e.g. as in $\{r_1 : \tau_1, r : \tau\}, \{r' : \tau, r_2 : \tau_2\}$, thus enabling indirect trades between token pairs not directly provided by any AMM. A state is *initial* when it has no AMMs, and its wallets hold only atomic tokens. We stipulate that the ordering of terms in a state is immaterial. Hence, we

consider two states Γ and Γ' to be equivalent when they contain the same terms (regardless of their order). For a term Q and a state Γ, we write $Q \in \Gamma$ when $\Gamma = Q \mid \Gamma'$, for some Γ'.

Transactions. State transitions are triggered by transactions $\mathsf{T}, \mathsf{T}', \ldots$, which can have the following forms (where τ_0 and τ_1 are atomic tokens):

- $\mathsf{A} : \mathsf{dep}(v_0 : \tau_0, v_1 : \tau_1)$. A deposits $v_0 : \tau_0$ and $v_1 : \tau_1$ to an AMM $\{r_0 : \tau_0, r_1 : \tau_1\}$, receiving in return some freshly-minted units of the token $\{\tau_0, \tau_1\}$;
- $\mathsf{A} : \mathsf{swap}(v, \tau_0, \tau_1)$. A tranfers $v : \tau_0$ to an AMM $\{r_0 : \tau_0, r_1 : \tau_1\}$, receiving in return some units of τ_1, which are removed from the AMM;
- $\mathsf{A} : \mathsf{rdm}(v : \{\tau_0, \tau_1\})$. A redeems v units of the minted token $\{\tau_0, \tau_1\}$: this means that some units of τ_0 and τ_1 are transferred from the AMM $\{r_0 : \tau_0, r_1 : \tau_1\}$ to A's wallet, and that v units of $\{\tau_0, \tau_1\}$ are burned.

We denote with $type(\mathsf{T})$ the type of T (i.e., dep, swap, or rdm), with $wal(\mathsf{T})$ the user whose wallet is affected by T, and with $tok(\mathsf{T})$ the set of token types affected by T. For example, if $\mathsf{T} = \mathsf{A} : \mathsf{swap}(v, \tau_0, \tau_1)$, then $type(\mathsf{T}) = \mathsf{swap}$, $wal(\mathsf{T}) = \mathsf{A}$, and $tok(\mathsf{T}) = \{\tau_0, \tau_1\}$.

Token supply. We use $S_\Gamma \tau$ to denote the **supply** of a token type τ in a state Γ, defined as the sum of the reserves of τ in all the wallets and the AMMs in Γ. Formally, we define $S_\Gamma \tau$ by induction on the structure of states as follows:

$$S_{\mathsf{A}[\sigma]}\tau = \begin{cases} \sigma\tau & \text{if } \tau \in \mathrm{dom}\,\sigma \\ 0 & \text{otherwise} \end{cases} \qquad S_{\{r_0:\tau_0, r_1:\tau_1\}}\tau = \begin{cases} r_i & \text{if } \tau = \tau_i \\ 0 & \text{otherwise} \end{cases} \qquad S_{\Gamma|\Gamma'}\tau = S_\Gamma\tau + S_{\Gamma'}\tau$$

For example, let $\Gamma = \mathsf{A}[1 : \tau_0, 2 : \{\tau_0, \tau_1\}] \mid \{3 : \tau_0, 4 : \tau_1\} \mid \{5 : \tau_0, 6 : \tau_2\}$. We have that $S_\Gamma\tau_0 = 9$, $S_\Gamma\tau_1 = 4$, $S_\Gamma\tau_2 = 6$, while $S_\Gamma\tau = 0$ for $\tau \notin \{\tau_0, \tau_1, \tau_2\}$. Note that $S_\Gamma\tau$ is always defined, since it is defined when Γ is an atomic term (wallet or AMM), and states Γ are *finite* compositions of atomic terms.

2.2. **AMM semantics.** We now formalise the transition rules between states. We write $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$ for a state transition from Γ to Γ', triggered by a transaction T. When $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$ for some Γ', we say that T is *enabled* in Γ. We denote with $\rightarrow^*$ the reflexive and transitive closure of $\rightarrow$. Given a finite sequence of transactions $\lambda = \mathsf{T}_1 \cdots \mathsf{T}_k$, we write $\Gamma \xrightarrow{\lambda} \Gamma'$ when $\Gamma \xrightarrow{\mathsf{T}_1} \cdots \xrightarrow{\mathsf{T}_k} \Gamma'$, and in this case we say that λ is enabled in Γ. We say that a state Γ is *reachable* if $\Gamma_0 \rightarrow^* \Gamma$ for some initial Γ_0. Hereafter, all the states mentioned in our results are implicitly assumed to be reachable. Given a partial map $\sigma \in \mathbb{T} \rightharpoonup \mathbb{R}_{\geq 0}$, a token type $\tau \in \mathbb{T}$ and a partial operation $\circ \in \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \rightharpoonup \mathbb{R}_{\geq 0}$ with $\circ \in \{+, -\}$, we define the partial map $\sigma \circ v : \tau$ as follows:

$$\sigma \circ v : \tau = \begin{cases} \sigma\{(\sigma\tau) \circ v / \tau\} & \text{if } \tau \in \mathrm{dom}\,\sigma \text{ and } (\sigma\tau) \circ v \in \mathbb{R}_{\geq 0} \\ \sigma\{v/\tau\} & \text{if } \tau \notin \mathrm{dom}\,\sigma \text{ and } \circ = + \end{cases}$$

These partial operations allow to increase/decrease the amount of tokens in a balance. For instance, if $\sigma = 5 : \tau_0$, then $\sigma + 1 : \tau_0 = 6 : \tau_0$, and $\sigma + 1 : \tau_1 = 5 : \tau_0, 1 : \tau_1$.

Deposit. Any user can create an AMM for two tokens τ_0 and τ_1, if such an AMM is not already present in the state. This is achieved by the transaction $A : \mathsf{dep}(v_0 : \tau_0, v_1 : \tau_1)$, through which A transfers $v_0 : \tau_0$ and $v_1 : \tau_1$ to the new AMM. In return for the deposit, A receives a certain positive amount of units of a new token type $\{\tau_0, \tau_1\}$, which is minted by the AMM.[2] We formalise this behaviour by the rule:

$$\frac{\sigma\tau_i \geq v_i > 0 \ \ (i \in \{0,1\}) \qquad S_\Gamma\{\tau_0, \tau_1\} = 0}{\begin{array}{l} A[\sigma] \mid \Gamma \xrightarrow{A:\mathsf{dep}(v_0:\tau_0, v_1:\tau_1)} \\ A[\sigma - v_0 : \tau_0 - v_1 : \tau_1 + v_0 : \{\tau_0, \tau_1\}] \mid \{v_0 : \tau_0, v_1 : \tau_1\} \mid \Gamma \end{array}} \ \ [\text{Dep0}]$$

Note that the premise $S_\Gamma\{\tau_0, \tau_1\} = 0$ implies that τ_0, τ_1 are distinct atomic tokens, since otherwise $\{\tau_0, \tau_1\}$ would not be a minted token. If Γ is reachable, then this premise also implies that Γ does *not* contain an AMM for the token pair τ_0, τ_1.

Once an AMM is created, any user can deposit tokens into it — *as long as* doing so preserves the ratio of the token reserves in the AMM. When a user deposits $v_0 : \tau_0$ and $v_1 : \tau_1$ to an existing AMM, it receives in return an amount of minted tokens of type $\{\tau_0, \tau_1\}$. This amount is the ratio between the deposited amount v_0 and the **redeem rate** of $\{\tau_0, \tau_1\}$ in the current state Γ, which is defined as follows for $i \in \{0, 1\}$:

$$RX_\Gamma^i(\tau_0, \tau_1) = \frac{r_i}{S_\Gamma\{\tau_0, \tau_1\}} \qquad \text{if } \{r_0 : \tau_0, r_1 : \tau_1\} \in \Gamma \qquad (2.1)$$

The effect of a deposit transaction on the state is then formalised by the following rule:

$$\frac{\sigma\tau_i \geq v_i > 0 \ \ (i \in \{0,1\}) \qquad v_i = v \cdot RX_\Gamma^i(\tau_0, \tau_1)}{\begin{array}{l} \Gamma = A[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta \xrightarrow{A:\mathsf{dep}(v_0:\tau_0, v_1:\tau_1)} \\ A[\sigma - v_0 : \tau_0 - v_1 : \tau_1 + v : \{\tau_0, \tau_1\}] \mid \{r_0 + v_0 : \tau_0, r_1 + v_1 : \tau_1\} \mid \Delta \end{array}} \ \ [\text{Dep}]$$

We anticipate that the premises of the [Dep] rule ensure that deposits preserve some key quantities across state transitions, namely:

- the ratio between the reserves of τ_0 and τ_1 in the AMM (see Lemma 4.4(a)). This ratio is always defined, since the reserves of a token in an AMM cannot be zeroed (see Lemma 4.2);
- the *net worth* of the user performing the action (see Lemma 4.5). In particular, the value of the minted tokens $\{\tau_0, \tau_1\}$ received by the user upon a deposit is equal to the value of the tokens τ_0, τ_1 transferred to the AMM;
- the *internal exchange rate* of the AMM (see Lemma 5.12). This preservation property holds for a relevant class of swap rate functions, called *homogeneous* (see Definition 5.11).

Redeem. Any user can redeem units of a minted token $\{\tau_0, \tau_1\}$, obtaining in return units of the underlying atomic tokens τ_0 and τ_1. Their actual amounts are determined by the redeem rate: the idea is that each unit of the minted token $\{\tau_0, \tau_1\}$ can be redeemed for equal fractions of τ_0 and τ_1 remaining in the AMM:

$$\frac{\sigma\{\tau_0, \tau_1\} \geq v > 0 \qquad v < S_\Gamma\{\tau_0, \tau_1\} \qquad v_i = v \cdot RX_\Gamma^i(\tau_0, \tau_1) \ \ (i \in \{0,1\})}{\begin{array}{l} \Gamma = A[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta \xrightarrow{A:\mathsf{rdm}(v:\{\tau_0, \tau_1\})} \\ A[\sigma + v_0 : \tau_0 + v_1 : \tau_1 - v : \{\tau_0, \tau_1\}] \mid \{r_0 - v_0 : \tau_0, r_1 - v_1 : \tau_1\} \mid \Delta \end{array}} \ \ [\text{Rdm}]$$

---

[2]The actual amount of received units is irrelevant. Here we choose v_0, but any other choice would be valid.

Note that the premise $v < S_\Gamma\{\tau_0, \tau_1\}$ ensures that the reserves are not depleted, i.e. $v_i < r_i$. Similarly to the [Dep] rule, the premises of [Rdm] ensure that:

- the net worth of the user performing the action is preserved (i.e., the net worth of burnt minted tokens is equal to that of the tokens received by A);
- the internal exchange rate of the AMM is unaffected by the transition, if the swap rate function is homogeneous.

Swap. Any user A can swap v units of τ_0 in her wallet for some units of τ_1 in an AMM $\{r_0 : \tau_0, r_1 : \tau_1\}$ through the transaction $A : \mathsf{swap}(v, \tau_0, \tau_1)$. Symmetrically, A can swap v of her units of τ_1 for units of τ_0 in the AMM through a transaction $A : \mathsf{swap}(v, \tau_1, \tau_0)$. The **swap rate** $SX(x, r_0, r_1)$ determines the amount of *output tokens* τ_1 that a user receives upon an amount of x *input tokens* τ_0 in an AMM $\{r_0 : \tau_0, r_1 : \tau_1\}$.

$$\frac{\sigma\tau_0 \geq x \qquad y = x \cdot SX(x, r_0, r_1) < r_1}{A[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Gamma \xrightarrow{A:\mathsf{swap}(x,\tau_0,\tau_1)} A[\sigma - x : \tau_0 + y : \tau_1] \mid \{r_0 + x : \tau_0, r_1 - y : \tau_1\} \mid \Gamma} \text{[Swap]}$$

The swap rate function is a parameter of our model: we will discuss in §5 some desiderata for this function, and the behavioural properties they induce on the AMM semantics. As an instance, we consider below the **constant product swap rate** [rva18], which is used in mainstream AMM implementations, like e.g. in Uniswap v2 [uni21], Mooniswap [moo20a] and SushiSwap [sus21]. We will use this swap rate function in all the examples in this paper.

Definition 2.1 (Constant product swap rate)**.** The constant product swap rate function is:

$$SX(x, r_0, r_1) = \frac{r_1}{r_0 + x}$$

The constant product swap rate ensures that, if an AMM $\{r_0 : \tau_0, r_1 : \tau_1\}$ evolves into $\{r_0 + x : \tau_0, r_1 - y : \tau_1\}$ upon a swap, then the product between the reserves is preserved:

$$(r_0 + x)(r_1 - y) = (r_0 + x)\left(r_1 - x \cdot \frac{r_1}{r_0 + x}\right) = r_0 r_1$$

Overall, the behaviour of the transition rules discussed above highlights some landmark properties of AMMs, namely:

- since neither deposits nor redeems affect the net worth of the users performing them, the only way for users to increase their net worth is to perform swaps. Since, as we will see in Lemma 4.5, the *global* net worth is constant, this means that increasing ones' net worth results in a decrease of someone else's net worth;
- the internal exchange rate of an AMM is affected only by swap actions (provided that the swap rate function is homogeneous). This is a natural behaviour, because swaps reflect the value of tokens perceived by users. We will show later in §5 that the constant sum/product/mean swap rate functions are homogeneous.

Example 2.2. Figure 1 shows a computation in our model. We discuss below the effect of the fired transactions, showing in Figure 2 the evolution of the token reserves in the AMM:

(1) $A : \mathsf{dep}(70 : \tau_0, 70 : \tau_1)$. Starting from an initial state, A creates a new AMM, depositing $70 : \tau_0$ and $70 : \tau_1$. In return, A receives 70 units of the minted token $\{\tau_0, \tau_1\}$.

$$\mathsf{A}[70:\tau_0, 70:\tau_1] \mid \mathsf{B}[30:\tau_0, 10:\tau_1]$$

$$\xrightarrow{\mathsf{A:dep}(70:\tau_0, 70:\tau_1)} \mathsf{A}[70:\{\tau_0, \tau_1\}] \mid \mathsf{B}[30:\tau_0, 10:\tau_1] \mid \{70:\tau_0, 70:\tau_1\}$$

$$\xrightarrow{\mathsf{B:swap}(30, \tau_0, \tau_1)} \mathsf{A}[70:\{\tau_0, \tau_1\}] \mid \mathsf{B}[0:\tau_0, 31:\tau_1] \mid \{100:\tau_0, 49:\tau_1\}$$

$$\xrightarrow{\mathsf{B:swap}(21, \tau_1, \tau_0)} \mathsf{A}[70:\{\tau_0, \tau_1\}] \mid \mathsf{B}[30:\tau_0, 10:\tau_1] \mid \{70:\tau_0, 70:\tau_1\}$$

$$\xrightarrow{\mathsf{A:rdm}(30:\{\tau_0, \tau_1\})} \mathsf{A}[30:\tau_0, 30:\tau_1, 40:\{\tau_0, \tau_1\}] \mid \mathsf{B}[30:\tau_0, 10:\tau_1] \mid \{40:\tau_0, 40:\tau_1\}$$

$$\xrightarrow{\mathsf{B:swap}(30, \tau_0, \tau_1)} \mathsf{A}[30:\tau_0, 30:\tau_1, 40:\{\tau_0, \tau_1\}] \mid \mathsf{B}[0:\tau_0, 27:\tau_1] \mid \{70:\tau_0, 23:\tau_1\}$$

$$\xrightarrow{\mathsf{A:rdm}(30:\{\tau_0, \tau_1\})} \mathsf{A}[82:\tau_0, 47:\tau_1, 10:\{\tau_0, \tau_1\}] \mid \mathsf{B}[0:\tau_0, 27:\tau_1] \mid \{18:\tau_0, 6:\tau_1\}$$

Figure 1: Interactions between two users and an AMM.

(2) $\mathsf{B}: \mathsf{swap}(30, \tau_0, \tau_1)$. B swaps 30 units of τ_0 for an amount y of units of τ_1 determined by the swap rate. Since we are assuming the constant product swap rate, we obtain $y = 30 \cdot {}^{70}/_{70+30} = 21$. This swap rate function ensures that swaps preserve the product between the token reserves in the AMM: in Figure 2, we show indeed that the swap results in a traversal along the curve $k = 70 \cdot 70$ from $\{70:\tau_0, 70:\tau_1\}$ to $\{100:\tau_0, 49:\tau_1\}$.

(3) $\mathsf{B}: \mathsf{swap}(21, \tau_1, \tau_0)$. B reverses the effect of his previous action by swapping 21 units of τ_1 for $y = 21 \cdot {}^{100}/_{49+21} = 30$ of τ_0. Figure 2 shows that the swap results in a traversal from $\{100:\tau_0, 49:\tau_1\}$ to $\{70:\tau_0, 70:\tau_1\}$ along the curve $k = 70 \cdot 70$.

(4) $\mathsf{B}: \mathsf{rdm}(30:\{\tau_0, \tau_1\})$. B redeems 30 units of the minted token $\{\tau_0, \tau_1\}$, accordingly reducing the token reserves in the AMM to $\{40:\tau_0, 40:\tau_1\}$. Note that the received tokens exhibit the same 1-to-1 ratio as after the initial deposit.

(5) $\mathsf{B}: \mathsf{swap}(30, \tau_0, \tau_1)$. B swaps 30 units of τ_0, receiving $y = 30 \cdot {}^{40}/_{40+30} \approx 17$ units of τ_1. Note that the swap rate, i.e. ${}^{40}/_{40+30} \approx 0.57$, has decreased w.r.t. the first swap, i.e. ${}^{70}/_{70+30} = 0.7$, even though the AMM has the same 1-to-1 reserves ratio. This is caused by the reduction in reserves occurred after A's redeem action: thus, the swap rate is sensitive not only to the ratio of reserves in the AMM, but also on their actual values.

(6) $\mathsf{A}: \mathsf{rdm}(30:\{\tau_0, \tau_1\})$. A redeems 30 units of the minted token $\{\tau_0, \tau_1\}$, thereby extracting 52 units of τ_0 and 17 units of τ_1 from the AMM. Note that the ratio of redeemed tokens is no longer 1-to-1 as in the previous redeem action, as the prior swap has changed the ratio between the funds of τ_0 and τ_1 in the AMM.

Finally, observe that the supply of both τ_0 and τ_1 remains constant. We will show in Lemma 4.3 that the supply of atomic token types is always preserved. ◇

3. Prices, exchange rates and net worth

In this section we introduce some economic notions which are pivotal for understanding the economic mechanisms of AMMs.

Token prices and exchange rates. We assume an external oracle that prices atomic tokens. Formally, we model this oracle as a function $P: \mathbb{T}_0 \to \mathbb{R}_{>0}$, assuming that the prices given by the oracle are constant along executions (see subsection 8.2 for a discussion about dynamic price updates). While the prices of atomic tokens are constant, that of minted
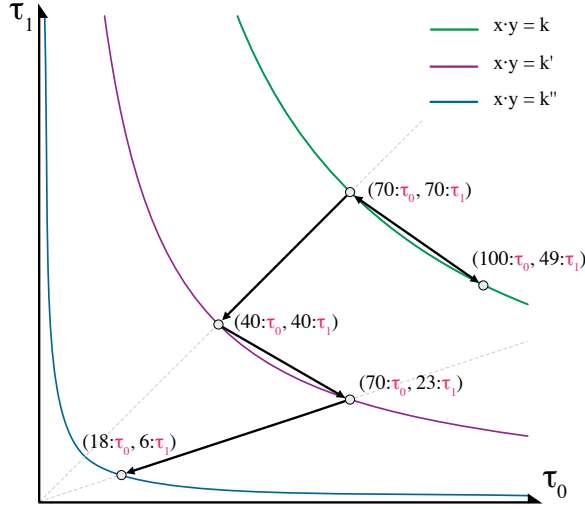
Figure 2: Evolution of reserves of AMM (τ_0, τ_1) along the trace in Figure 1.

tokens may vary at run-time as a function of the state. More precisely, the price $P_\Gamma\{\tau_0, \tau_1\}$ of a minted token $\{\tau_0, \tau_1\}$ depends both on the supply of the minted token in the users' wallets and on the reserves of τ_0 and τ_1 in the AMM:

$$P_\Gamma\{\tau_0, \tau_1\} \;=\; \frac{r_0 \cdot P\tau_0 + r_1 \cdot P\tau_1}{S_\Gamma\{\tau_0, \tau_1\}} \qquad \text{if } \{r_0 : \tau_0, r_1 : \tau_1\} \in \Gamma \tag{3.1}$$

For uniformity, we define $P_\Gamma \tau = P\tau$ when $\tau \in \mathbb{T}_0$. Lemma 4.2 will ensure that $r_0, r_1 > 0$ and $S_\Gamma\{\tau_0, \tau_1\} > 0$ in every reachable state Γ containing an AMM for the token pair τ_0, τ_1. Therefore, the price of the token $\{\tau_0, \tau_1\}$ is always defined and positive in reachable states.

The idea underlying Equation (3.1) is that the price of one unit of minted token must be equal to the value of the atomic tokens that can be obtained by redeeming the minted token. Indeed, by rule [RDM] and Equation (2.1) we have that:

$$P_\Gamma\{\tau_0, \tau_1\} \;=\; \frac{r_0}{S_\Gamma\{\tau_0, \tau_1\}} P\tau_0 + \frac{r_1}{S_\Gamma\{\tau_0, \tau_1\}} P\tau_1 = RX_\Gamma^0(\tau_0, \tau_1) \cdot P\tau_0 + RX_\Gamma^1(\tau_0, \tau_1) \cdot P\tau_1$$

which substantiates our desideratum. This intuition will be formalized later in Lemma 4.6.

The **exchange rate** $X(\tau_0, \tau_1)$ between atomic token types τ_0 and τ_1 is the number of units of τ_1 that one can buy with 1 unit of τ_0 at the price given by the external oracle:

$$X(\tau_0, \tau_1) \;=\; \frac{P\tau_0}{P\tau_1} \tag{3.2}$$

Hence, assuming an exchange at the prices of the external oracle, a user paying x units of τ_0 would receive $x \cdot X(\tau_0, \tau_1)$ units of τ_1.

Note that the exchange rate between two token types only depends on external oracles, neglecting the state of AMMs. However, AMMs themselves can act as (decentralised) price oracles [AC20], since they induce an exchange rate based on the effect of swaps in the current state. More precisely, the **internal exchange rate** $X_\Gamma(\tau_0, \tau_1)$ between two atomic token types τ_0 and τ_1 in a state Γ is the limit of the swap rate function as x approaches 0: [3]

$$X_\Gamma(\tau_0, \tau_1) \;=\; \lim_{x \to 0} SX(x, r_0, r_1) \qquad \text{if } \{r_0 : \tau_0, r_1 : \tau_1\} \tag{3.3}$$

_____

[3]This notion is also dubbed as marginal price [AC20] or spot exchange rate [XVPC22] in literature.

The intuition is similar to that in Equation (3.2): a user swapping x units of τ_0 for τ_1 through the AMM (for x *very small*) would expect to receive $x \cdot X_\Gamma(\tau_0, \tau_1)$ units of τ_1. We will see later in section 6 that rational users will perform actions that align the internal exchange rate to the one given by external oracles.

Slippage measures the discrepancy between the internal exchange rate and the actual ratio between the amounts of output and input tokens obtained upon the swap [XVPC22]:

$$\Delta X_\Gamma(x, \tau_0, \tau_1) \;=\; \frac{X_\Gamma(\tau_0, \tau_1)}{SX(x, r_0, r_1)} - 1 \qquad \text{if } \{r_0 : \tau_0, r_1 : \tau_1\} \tag{3.4}$$

Ideally, slippage should disadvantage large trades, i.e. trying to obtain a larger amount of tokens with a swap should make them more expensive, increasing the slippage. We will compute in sections 5.6-5.8 the internal exchange rate and the slippage of some common AMMs.

Example 3.1. Let $\Gamma = \mathsf{A}[82 : \tau_0, 47 : \tau_1, 10 : \{\tau_0, \tau_1\}] \mid \{18 : \tau_0, 6 : \tau_1\} \mid \mathsf{B}[\cdots]$ be the final state of the computation in Figure 1. We have that $S_\Gamma\{\tau_0, \tau_1\} = 10$, since only A's wallet contains units of the minted token. Assume that the prices of atomic tokens are $P\tau_0 = 5$ and $P\tau_1 = 9$. The price of the minted token $\{\tau_0, \tau_1\}$ is then:

$$P_\Gamma\{\tau_0, \tau_1\} = \frac{18 \cdot P\tau_0 + 6 \cdot P\tau_1}{10} = \frac{18 \cdot 5 + 6 \cdot 9}{10} = 14.4$$

The exchange rate between the two tokens is:

$$X(\tau_0, \tau_1) \;=\; \frac{P\tau_0}{P\tau_1} \;=\; \frac{5}{9} \;=\; 0.55$$

which means that to buy 1 unit of τ_0, one needs 0.55 units of τ_1. Note instead that the internal exchange rate is:

$$X_\Gamma(\tau_0, \tau_1) \;=\; \lim_{x \to 0} SX(x, 18, 6) = \frac{6}{18} \approx 0.33$$

We will see in Example 6.5 that the discrepancy between internal and oracle exchange rate can by exploited by users to increase their gain. The slippage of a $\mathsf{swap}(x, \tau_0, \tau_1)$ is:

$$\Delta X_\Gamma(x, \tau_0, \tau_1) \;=\; \frac{X_\Gamma(\tau_0, \tau_1)}{SX(x, 18, 6)} - 1 \;=\; \frac{x}{18}$$

from which we can see that the slippage grows with the input amount x. $\diamond$

Net worth and gain. The **net worth** of a user A is a measure of A's wealth in tokens (both atomic and minted). Formally, we define the net worth of A in a state Γ as:

$$W_\mathsf{A}(\Gamma) \;=\; \begin{cases} \sum_{\tau \in \mathrm{dom}\,\sigma} \sigma\tau \cdot P_\Gamma\tau & \text{if } \mathsf{A}[\sigma] \in \Gamma \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

Note that $W_\mathsf{A}(\Gamma) \in \mathbb{R}_{\geq 0}$, since balances σ are *finite* maps, and $P_\Gamma\tau$ is always defined.

The **global net worth** $W(\Gamma)$ in a state Γ is the sum of the net worth in users' wallets. Note that the token reserves in AMMs are not accounted for by $W(\Gamma)$, because their value is already recorded by minted tokens held in users' wallets. Indeed, the equality:

$$S_\Gamma\{\tau_0, \tau_1\} \cdot P_\Gamma\{\tau_0, \tau_1\} \;=\; r_0 \cdot P\tau_0 + r_1 \cdot P\tau_1$$

between the net worth of a minted token and the value of the AMM is a direct consequence of the definition of price in Equation (3.1).

We denote by $G_A(\Gamma, \lambda)$ the **_gain_** of user A upon performing a sequence of transactions λ enabled in state Γ (if λ is not enabled in Γ, we stipulate that the gain is zero):

$$G_A(\Gamma, \lambda) \;=\; W_A(\Gamma') - W_A(\Gamma) \qquad \text{if } \Gamma \xrightarrow{\lambda} \Gamma' \tag{3.6}$$

To maximize their gain, users can perform different interactions with the AMM, e.g., by investing tokens or trading units of differently priced token types.

The following lemma quantifies the gain of users upon firing a swap transaction. Note that this quantification does not depend on any of the properties of the swap rate function introduced later on in section 5: actually, it holds for any swap rate function.

Lemma 3.2 (Swap gain). *Let $\Gamma = \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$, and let $T = A : \text{swap}(x, \tau_0, \tau_1)$ be enabled in Γ. Then:*

$$G_A(\Gamma, T) = \quad x \cdot \big(SX(x, r_0, r_1)\, P\tau_1 - P\tau_0\big) \cdot \left(1 - \frac{\sigma_A\{\tau_0, \tau_1\}}{S_\Gamma\{\tau_0, \tau_1\}}\right) \qquad \text{if } A[\sigma_A] \in \Gamma$$

$$G_B(\Gamma, T) = -x \cdot \big(SX(x, r_0, r_1)P\tau_1 - P\tau_0\big) \cdot \frac{\sigma_B\{\tau_0, \tau_1\}}{S_\Gamma\{\tau_0, \tau_1\}} \qquad \text{if } B[\sigma_B] \in \Gamma,\ B \neq A$$

A direct consequence of Lemma 3.2 is that if A performs a swap between τ_0 and τ_1 and she holds all the units of the minted token $\{\tau_0, \tau_1\}$, then her gain will be zero. Further, A maximizes her gain when she has no minted tokens of type $\{\tau_0, \tau_1\}$. The lemma also implies that if the user performing the swap has a positive gain, then all the users who hold units of $\{\tau_0, \tau_1\}$ will have a negative gain.

The following lemma states that a swap transaction on an AMM $\{r_0 : \tau_0, r_1 : \tau_1\}$ has a strictly positive gain *if and only if* the swap rate is strictly greater than the oracle exchange rate between τ_0 and τ_1. This holds for *any* swap rate function, under the condition that the user who performs the swap has no minted tokens of type $\{\tau_0, \tau_1\}$.

Lemma 3.3 (Swap rate *vs.* exchange rate). *Let $\Gamma = A[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$ be such that $\sigma\{\tau_0, \tau_1\} = 0$, and let $T = A : \text{swap}(x, \tau_0, \tau_1)$ be enabled in Γ. Then:*

$$G_A(\Gamma, T) \circ 0 \iff SX(x, r_0, r_1) \circ X(\tau_0, \tau_1) \qquad \text{for } \circ \in \{<, =, >\}$$

Example 3.4. Let $\Gamma_0 = A[70 : \tau_0, 70 : \tau_1] \mid B[30 : \tau_0, 10 : \tau_1]$ be the initial state of the computation in Figure 1. Let $P\tau_0 = 5$ and $P\tau_1 = 9$. The users' net worth in Γ_0 and in the final state $\Gamma = A[82 : \tau_0, 47 : \tau_1, 10 : \{\tau_0, \tau_1\}] \mid B[0 : \tau_0, 27 : \tau_1] \mid \{18 : \tau_0, 6 : \tau_1\}$ is as follows:

$$W_A(\Gamma_0) = 70 \cdot P\tau_0 + 70 \cdot P\tau_1 = 980 \qquad\qquad W_B(\Gamma_0) = 30 \cdot P\tau_0 + 10 \cdot P\tau_1 = 240$$

$$W_A(\Gamma) = 82 \cdot P\tau_0 + 47 \cdot P\tau_1 + 10 \cdot P_\Gamma\{\tau_0, \tau_1\} = 977 \quad W_B(\Gamma) = 27 \cdot P\tau_1 = 243$$

Note that A's net worth of has decreased w.r.t. the initial state, while that of B has increased: indeed, the gain of A upon the sequence of transactions λ is $G_A(\Gamma, \lambda) = 977 - 980 = -3$, while that of B is $G_B(\Gamma, \lambda) = 243 - 240 = 3$. One may think that B has been more successful than A, but this depends on the users' goals. Note, e.g., that A holds 10 units of the minted token $\{\tau_0, \tau_1\}$, whose price may increase in the future. $\diamond$

4. Structural properties of AMMs

We now establish some structural properties of AMMs, which do not depend on the design of the economic mechanisms, i.e. on the choice of the swap rate function. These structural properties are the basis for AMM interactions that occur in the wild, and that cumulatively give rise to complex emerging behaviours like arbitrage and MEV. Hence, establishing these structural properties is a preliminary sanity check for our AMM model. We will provide further support for the coherence between our model and actual AMMs by showing that the above-mentioned complex behaviours are expressible in our model (see section 6 and section 7).

First, we establish that the AMMs' transition system is deterministic. This follows from the fact that, given a state Γ and a transaction T, there is at most one applicable rule. Note that determinism is a crucial property for blockchains, since it ensures that all the nodes in the blockchain network are able to reconstruct a common state from a sequence of transactions. Therefore, it makes sense that determinism holds also for our AMM model.

Lemma 4.1 (Determinism). *If $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$ and $\Gamma \xrightarrow{\mathsf{T}} \Gamma''$, then $\Gamma' = \Gamma''$.*

We can lift the statement to sequences of transactions by using a simple inductive argument. The same applies to other single-step results in this section.

Lemma 4.2 ensures that the reserves in an AMM cannot be zeroed, and that the same holds for the units of any minted token. Summing up, this ensures that the price of any minted token is always defined and positive.

Lemma 4.2 (Non depletion). *For all states Γ, if $\{r_0 : \tau_0, r_1 : \tau_1\} \in \Gamma$ then:*

(a) $r_i > 0$, *for $i \in \{0, 1\}$;*
(b) $S_\Gamma\{\tau_0, \tau_1\} > 0$.

4.1. **Preservation properties.** Lemma 4.3 ensures that transactions preserve the supply of *atomic* tokens. Minted tokens, instead, are preserved only by swap transactions, since deposit and redeem transactions, respectively, create and destroy minted tokens. This fact will be instrumental to prove the preservation of the net worth (see Lemma 4.5).

Lemma 4.3 (Preservation of token supply). *Let $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$. Then:*

(a) *for all $\tau \in \mathbb{T}_0$, $S_\Gamma \tau = S_{\Gamma'} \tau$*
(b) *if $type(\mathsf{T}) = \mathsf{swap}$, then for all $\tau \in \mathbb{T}_1$, $S_\Gamma \tau = S_{\Gamma'} \tau$*

Lemma 4.4 states that deposit and redeem transactions preserve the reserves ratio in AMMs, the redeem rate, and the price of minted tokens. These preservation properties will be exploited later on to determine the solution to the arbitrage game after deposits and redeems (see Theorems 6.8 and 6.10).

Lemma 4.4 (Preservation upon deposits/redeems). *Let $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$, with $\{r_0 : \tau_0, r_1 : \tau_1\} \in \Gamma$. If $type(\mathsf{T}) \in \{\mathsf{dep}, \mathsf{rdm}\}$, then:*

(a) *if $\{r'_0 : \tau_0, r'_1 : \tau_1\} \in \Gamma'$, then $r_1/r_0 = r'_1/r'_0$*
(b) $RX^i_\Gamma(\tau_0, \tau_1) = RX^i_{\Gamma'}(\tau_0, \tau_1)$, *for $i \in \{0, 1\}$*
(c) $P_\Gamma\{\tau_0, \tau_1\} = P_{\Gamma'}\{\tau_0, \tau_1\}$

Lemma 4.5 ensures that transactions (of any type) preserve the *global* net worth, whereas the net worth of individual users is preserved only by redeem and deposit transactions. A direct consequence of this preservation result is that users can increase their net worth only by performing swaps: Indeed, we will find in Theorem 6.3 that the solution of the arbitrage game only contains swap transactions. Furthermore, if a user has a positive gain, then some other user must have a loss.

Lemma 4.5 (Preservation of net worth). *Let $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$. Then:*
(a) *if $type(\mathsf{T}) \neq \mathsf{swap}$ then, for all A: $W_\mathsf{A}(\Gamma) = W_\mathsf{A}(\Gamma')$*
(b) $W(\Gamma) = W(\Gamma')$

The following lemma, which is a direct consequence of Lemma 4.5(a), supports the definition of the price of minted tokens in Equation (3.1): indeed, computing the net worth of a user A under that price definition corresponds to making A first redeem all her minted tokens, and then summing the price of the resulting atomic tokens.

Lemma 4.6. *Let $\Gamma \xrightarrow{\lambda} \Gamma'$, where λ contains only rdm actions of A. If $\mathsf{A}[\sigma] \in \Gamma'$ and $\dom \sigma \cap \mathbb{T}_1 = \emptyset$, then:*
$$W_\mathsf{A}(\Gamma) \;=\; \sum_{\tau \in \dom \sigma} \sigma\tau \cdot P\tau$$

Example 4.7. Let $\Gamma = \mathsf{A}[82 : \tau_0, 47 : \tau_1, 10 : \{\tau_0, \tau_1\}] \mid \{27 : \tau_0, 9 : \tau_1\} \mid \mathsf{B}[5 : \{\tau_0, \tau_1\}]$, and let $P\tau_0 = 5$ and $P\tau_1 = 9$. We have that $P_\Gamma\{\tau_0, \tau_1\} = 14.4$, and $W_\mathsf{A}(\Gamma) = 977$. Assume that A performs a transaction from Γ to redeem all 10 units of $\{\tau_0, \tau_1\}$ in her wallet. The resulting state is $\Gamma' = \mathsf{A}[100 : \tau_0, 53 : \tau_1] \mid \{9 : \tau_0, 3 : \tau_1\} \mid \cdots$. We compute A's net worth in Γ' using the oracle token prices:
$$W_\mathsf{A}(\Gamma') \;=\; 100 \cdot P\tau_0 + 53 \cdot P\tau_1 \;=\; 100 \cdot 5 + 53 \cdot 9 \;=\; 977$$
which is coherent with the net worth predicted by Lemma 4.6.   ◇

4.2. **Liquidity.** Lemma 4.8 ensures that funds cannot be *frozen* in an AMM, i.e. that users can always redeem arbitrary amounts of the tokens deposited in an AMM, as long as the reserves are not zeroed. Note that, since $\{r_0 : \tau_0, r_1 : \tau_1\} = \{r_1 : \tau_1, r_0 : \tau_0\}$, the statement also holds when swapping r_0 with r_1.

Lemma 4.8 (Liquidity). *Let Γ be such that $\{r_0 : \tau_0, r_1 : \tau_1\} \in \Gamma$. Then, for all $r_0' < r_0$, there exist $r_1' < r_1$, Γ' and λ only containing rdm transactions such that $\Gamma \xrightarrow{\lambda} \{r_0' : \tau_0, r_1' : \tau_1\} \mid \Gamma'$.*

4.3. **Reordering of transactions.** In general, given two transactions T_0 and T_1 and a state Γ, executing $\mathsf{T}_0\mathsf{T}_1$ or $\mathsf{T}_1\mathsf{T}_0$ from Γ yields different states. However, under some conditions it is possible to invert the order of the two transactions, preserving the resulting state. This is always the case, e.g., of two transactions which operate on disjoint sets of tokens. Lemma 4.9 establishes sufficient conditions for preserving the state upon reordering. Besides the case cited before, this is always possible if both transactions are deposits, or if they are bot redeems (case (a) of the statement). Note that, in these cases, the assumption that $\mathsf{T}_0\mathsf{T}_1$ is enabled in Γ implies that also $\mathsf{T}_1\mathsf{T}_0$ is such. This is no longer true when one of the two transactions is a deposit and the other one is a redeem. For instance, if T_1 redeems the minted tokens obtained upon a deposit T_0, then T_1 may not be enabled in Γ because

there are not enough minted tokens in the user's wallet. Therefore, case (b) of the statement uses the additional hypothesis that also $\mathsf{T}_1\mathsf{T}_0$ is enabled in Γ.

Lemma 4.9 (Reordering of transactions). *Let* $\Gamma \xrightarrow{\mathsf{T}_0\mathsf{T}_1} \Gamma_{01}$. *Then:*

(a) *if* $tok(\mathsf{T}_0) \cap tok(\mathsf{T}_1) = \emptyset$ *or* $type(\mathsf{T}_0) = type(\mathsf{T}_1) \in \{\mathsf{dep}, \mathsf{rdm}\}$, *then* $\Gamma \xrightarrow{\mathsf{T}_1\mathsf{T}_0} \Gamma_{01}$;

(b) *otherwise, if* $type(\mathsf{T}_0), type(\mathsf{T}_1) \neq \mathsf{swap}$ *and* $\Gamma \xrightarrow{\mathsf{T}_1\mathsf{T}_0} \Gamma_{10}$, *then* $\Gamma_{01} = \Gamma_{10}$.

As we shall see in section 6, it is actually desirable, and crucial for the economic mechanism of AMMs, that swaps interfere with other transactions that trade the same token type.

4.4. Additivity of deposit and redeem actions.
Deposit and redeem actions satisfy an additivity property: if a user performs two successive deposits (resp. redeems) on an AMM, then the same result can be obtained through a single deposit (resp. redeem). Instead, swap actions are not additive, in general: we will study sufficient conditions for the additivity of swap actions in section 5 (see Theorem 5.6).

Theorem 4.10 (Additivity). *Let* $\Gamma \xrightarrow{\mathsf{T}_0} \Gamma_0 \xrightarrow{\mathsf{T}_1} \Gamma_1$. *Then:*

(1) *if* $\mathsf{T}_0 = \mathsf{A} : \mathsf{dep}(v_0 : \tau_0, v_1 :\tau_1)$ *and* $\mathsf{T}_1 = \mathsf{A} : \mathsf{dep}(v_0' : \tau_0, v_1' :\tau_1)$, *then:*

$$\Gamma \xrightarrow{\mathsf{A}:\mathsf{dep}(v_0+v_0':\tau_0,v_1+v_1':\tau_1)} \Gamma_1$$

(2) *if* $\mathsf{T}_0 = \mathsf{A} : \mathsf{rdm}(v : \tau)$ *and* $\mathsf{T}_1 = \mathsf{A} : \mathsf{rdm}(v' : \tau)$, *then:*

$$\Gamma \xrightarrow{\mathsf{A}:\mathsf{rdm}(v+v':\tau)} \Gamma_1$$

4.5. Reversibility of deposit and redeem actions.
The following theorem establishes that deposit and redeem transactions are *reversible*: more precisely, the effect of a deposit action can be reverted by a redeem action, and *vice versa*, the effect of a redeem action can be reverted by a deposit action. The only exception is a deposit action that creates an AMM, through the rule [Dep0]. Swap actions are not reversible, in general: we will study sufficient conditions for their reversibility in section 5 (see Theorem 5.9).

Theorem 4.11 (Reversibility). *Let* $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$, *where* $type(\mathsf{T}) \in \{\mathsf{dep}, \mathsf{rdm}\}$ *and for all* $\tau \in \mathbb{T}_1$, *if* $S_\Gamma \tau = 0$ *then* $S_{\Gamma'} \tau = 0$. *Then there exists* T^{-1} *such that* $\Gamma' \xrightarrow{\mathsf{T}^{-1}} \Gamma$.

In general, the study of reversible computation models, which dates back to [Ben73], is an active area of research, which has led to a wide range of applications in software systems [MSG+20]. In particular, the reversibility of AMM actions has useful consequences on their behaviour. For instance, it guarantees that, starting from a "stable" state where no arbitrage is possible, after any transaction it is possible to return to the stable state. More in general, if the swap rate function satisfies the conditions of section 5 that ensure the additivity and reversibility also for swap actions, then for any sequence of transactions:

$$\Gamma_0 \xrightarrow{\mathsf{T}_1} \Gamma_1 \xrightarrow{\mathsf{T}_2} \cdots \Gamma_n$$

it is possible to fire another transaction and return to the state Γ_0. Indeed, by additivity we obtain that the effect of the sequence $\mathsf{T}_1 \cdots \mathsf{T}_n$ can be emulated by a single transaction T, and then reversibility ensures that T can be reversed, i.e.:

$$\Gamma_0 \xrightarrow{\mathsf{T}} \Gamma_n \xrightarrow{\mathsf{T}^{-1}} \Gamma_0$$

5. The swap rate function

In the previous section we have established some key structural properties of deposit and redeem actions, e.g. their additivity and reversibility. In general, these properties do not hold for swap actions: it is easy to find swap rate functions $SX \in \mathbb{R}_{\geq 0}^3 \to \mathbb{R}_{\geq 0}$ that make these properties false. Throughout this section we introduce some general properties of swap rate functions, and we discuss the properties they induce on the behaviour of AMMs. In sections 5.6-5.8 we then discuss the properties enjoyed by the swap rate functions used in some concrete AMM implementations. Coherently with these implementations, we assume that a swap rate function is defined and non-negative for all $x > 0$, and that the internal exchange rate (i.e., the limit of SX for x leading to 0, see (3.3)) is always defined.

5.1. **Output-boundedness.**
Output boundedness guarantees that an AMM has always enough output tokens τ_1 to send to the user who performs a $\mathsf{swap}(x, \tau_0, \tau_1)$.

Definition 5.1 (Output-boundedness). A swap rate function SX is *output-bounded* when, for all x, r_0, r_1 such that $x \geq 0$ and $r_0, r_1 > 0$:

$$x \cdot SX(x, r_0, r_1) < r_1$$

The following lemma establishes sufficient conditions for a swap action to be enabled.

Lemma 5.2. *Let $\mathsf{T} = \mathsf{A} : \mathsf{swap}(x, \tau_0, \tau_1)$, and let $\mathsf{A}[\sigma] \in \Gamma$. If $S_\Gamma\{\tau_0, \tau_1\} > 0$, $\sigma(\tau_0) \geq x$ and SX is output-bounded, then T is enabled in Γ.*

5.2. **Monotonicity.**
Consider a transaction $\mathsf{A} : \mathsf{swap}(x, \tau_0, \tau_1)$ on an AMM $\{r_0 : \tau_0, r_1 : \tau_1\}$. Without making any assumptions on the swap rate function, there is no relation between the effect of this transaction and that of a swap where the parameters have been varied. Monotonicity, instead, ensures that there exists a meaninful relation: the swap rate increases if we decrease the input amount x or the reserves of τ_0, and if we increase the reserves of τ_1. The intuition is that lower reserves of τ_0 in the AMM make the $x : \tau_0$ paid by A more "valuable" for the AMM, hence the AMM will output more units of τ_1 for the same input amount. Increasing the reserves of τ_1 in the AMM (keeping those of τ_0 unaltered) produces the same effect. Monotonicity on x also ensures that the internal exchange rate of the AMM is defined, for each token pair.

Definition 5.3 (Monotonicity). A swap rate function SX is *monotonic* when:

$$x' \leq x, \ r_0' \leq r_0, \ r_1 \leq r_1' \implies SX(x, r_0, r_1) \leq SX(x', r_0', r_1')$$

Further, SX is *strictly monotonic* when, for $i \in \{0, 1, 2\}$ and $\lhd_i \in \{<, \leq\}$:

$$x' \lhd_0 x, \ r_0' \lhd_1 r_0, \ r_1 \lhd_2 r_1' \implies SX(x, r_0, r_1) \lhd_3 SX(x', r_0', r_1')$$

where:

$$\lhd_3 = \begin{cases} \leq & \text{if } \lhd_i = \leq \text{ for } i \in \{0,1,2\} \\ < & \text{otherwise} \end{cases}$$

Note that strict monotonicity trivially implies monotonicity. The following lemma relates monotonicity of the swap rate function with the gain of swap transactions, concretising the intuition given before from the point of view of A's gain.

Lemma 5.4. *Let* $\Gamma = \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$ *and* $\Gamma' = \{r_0' : \tau_0, r_1' : \tau_1\} \mid \Delta$, *with* $r_0' \leq r_0$ *and* $r_1 \leq r_1'$, *and let* $\mathsf{T} = \mathsf{A} : \mathsf{swap}(x, \tau_0, \tau_1)$ *be enabled in* Γ *and in* Γ'. *If SX is monotonic, then* $G_{\mathsf{A}}(\Gamma, \mathsf{T}) \leq G_{\mathsf{A}}(\Gamma', \mathsf{T})$.

5.3. **Additivity.** To extend the additivity property of Theorem 4.10 to swap actions, we must require that the swap rate function is additive.

Definition 5.5 (Additivity). A swap rate function SX is *additive* when:

$$\alpha = SX(x, r_0, r_1), \ \beta = SX(y, r_0 + x, r_1 - \alpha x) \implies SX(x + y, r_0, r_1) = \frac{\alpha x + \beta y}{x + y}$$

The idea here is that a user fires a swap transaction (say, T_0) with input amount x in a state Γ, and then in the state reached after firing T_0, she fires another swap transaction (say, T_1) with input amount y on the same AMM. The definition of additivity requires that the swap rate of a swap transaction with input amount $x + y$ in Γ is in a given relation with the swap rates computed for T_0 and T_1 and with the input amounts x and y. Theorem 5.6 states that if this relation holds, then a single swap with input amount $x + y$ in Γ produces exactly the same effect of performing first T_0 and then T_1. Then, Lemma 5.7 allows us to compute the gain of this transaction as the sum of the gains of T_0 and T_1.

Theorem 5.6 (Additivity of swap). *Let* $\Gamma \xrightarrow{\mathsf{T}_0} \Gamma_0 \xrightarrow{\mathsf{T}_1} \Gamma_1$, *with* $\mathsf{T}_i = \mathsf{A} : \mathsf{swap}(x_i, \tau_0, \tau_1)$ *for* $i \in \{0, 1\}$. *If SX is additive, then:*

$$\Gamma \xrightarrow{\mathsf{A}:\mathsf{swap}(x_0+x_1,\tau_0,\tau_1)} \Gamma_1$$

Lemma 5.7 (Additivity of swap gain). *Let* $\mathsf{T}(x) = \mathsf{A} : \mathsf{swap}(x, \tau_0, \tau_1)$, *and let* $\Gamma \xrightarrow{\mathsf{T}(x_0)} \Gamma'$. *If SX is output-bounded and additive, then:*

$$G_{\mathsf{A}}(\Gamma, \mathsf{T}(x_0 + x_1)) = G_{\mathsf{A}}(\Gamma, \mathsf{T}(x_0)) + G_{\mathsf{A}}(\Gamma', \mathsf{T}(x_1))$$

5.4. **Reversibility.** The reversibility property in Theorem 4.11 states that the effect of deposit and redeem transactions can be reverted. We now devise a property of swap rate functions that give the same guarantee for swap transactions.

Definition 5.8 (Reversibility). A swap rate function SX is *reversible* when:

$$\alpha = SX(x, r_0, r_1) \implies SX(\alpha x, r_1 - \alpha x, r_0 + x) = \frac{1}{\alpha}$$

Consider now a state $\Gamma = \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$, and let $\alpha = \lim_{x \to 0} SX(x, r_0, r_1)$ be the internal exchange between τ_0 and τ_1 in Γ. If the swap rate function is reversible, then:

$$\lim_{x \to 0} SX(x, r_1, r_0) \;=\; \lim_{x \to 0} SX(\alpha x, r_1 - \alpha x, r_0 + x) \;=\; \lim_{x \to 0} \frac{1}{\alpha} \;=\; \frac{1}{\alpha}$$

from which we obtain:

$$X_\Gamma(\tau_1, \tau_0) \;=\; \frac{1}{X_\Gamma(\tau_0, \tau_1)} \tag{5.1}$$

The intuition of Definition 5.8 is that, to reverse the effect of a swap transaction T that pays $x : \tau_0$ to receive $y : \tau_1$, one must fire a swap transaction T^{-1} that pays $y : \tau_1$ to receive $x : \tau_0$. Of course, this results in the same AMM state that we had before performing T. Writing α for the swap rate $SX(x, r_0, r_1)$, the [SWAP] rule fixes $y = \alpha x$. Hence, assuming that in the initial state the AMM has reserves $r_0 : \tau_0$ and $r_1 : \tau_1$, after performing T its reserves will be $r_0 + x : \tau_0$ and $r_1 - \alpha x : \tau_1$. In this state, requiring that the swap rate for an input of $y : \tau_1$ is $\frac{1}{\alpha}$ (as done by Definition 5.8) implies that the AMM outputs $x : \tau_0$, reverting the reserves of the AMM to the initial values.

The following theorem formalises the intuition above, establishing that, when the swap rate function is reversible, swap transactions are reversible. Together with Theorem 4.11, all the AMM actions are reversible under this hypothesis.

Theorem 5.9 (Reversibility of swap). *Let $\mathsf{T} = \mathsf{A} : \mathsf{swap}(x, \tau_0, \tau_1)$, and let $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$. If SX is reversible, then there exists T^{-1} such that $\Gamma' \xrightarrow{\mathsf{T}^{-1}} \Gamma$.*

Lemma 5.10 allows us to compute the gain of the reverse transaction T^{-1} in the state reached after performing T as a function of the gain of T. As expected by preservation of the global net worth, the gain of T^{-1} is the opposite of that of T.

Lemma 5.10. *Let $\mathsf{T} = \mathsf{A} : \mathsf{swap}(x, \tau_0, \tau_1)$, and let $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$. If SX is reversible, then $G_\mathsf{A}(\Gamma, \mathsf{T}) = -G_\mathsf{A}(\Gamma', \mathsf{T}^{-1})$.*

5.5. **Homogeneity.** A swap rate function is homogeneous when the swap rate is not affected by a linear scaling of the three parameters. Homogeneity is useful to relate the swap rate before and after deposit or redeem transactions, since their effect is a linear scaling of the AMM reserves. Lemma 5.12 establishes one the the landmark properties of AMMs we have anticipated in section 2: when the swap rate function is homogeneous, deposits and redeems do not affect the internal swap rate.

Definition 5.11 (Homogeneity). A swap rate function SX is *homogeneous* when, for $a > 0$:

$$SX(ax, ar_0, ar_1) \;=\; SX(x, r_0, r_1)$$

Lemma 5.12 (Preservation of internal exchange rate upon deposits/redeems). *Let $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$, where $tok(\mathsf{T}) = \{\tau_0, \tau_1\}$ and $type(\mathsf{T}) \in \{\mathsf{dep}, \mathsf{rdm}\}$. If SX is homogeneous, then:*

$$X_\Gamma(\tau_0, \tau_1) = X_{\Gamma'}(\tau_0, \tau_1)$$

The following lemma shows that deposits increase swap rates, whilst redeems have the opposite effect. Dually, deposits decrease the slippage, while redeems increase it. In section 6 we will exploit this fact to show that deposits incentivize swaps, while redeems disincentivize them (see Theorems 6.6 and 6.9).

Lemma 5.13. *Let* $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$, *where* $\{r_0 : \tau_0, r_1 : \tau_1\} \in \Gamma$, $\{r_0' : \tau_0, r_1' : \tau_1\} \in \Gamma'$ *and* $tok(\mathsf{T}) = \{\tau_0, \tau_1\}$. *If* SX *is homogeneous and strictly monotonic, then for all* $x > 0$:

(a) $type(\mathsf{T}) = \mathsf{dep} \implies SX(x, r_0, r_1) < SX(x, r_0', r_1')$ *and* $\Delta X_\Gamma(x, \tau_0, \tau_1) > \Delta X_{\Gamma'}(x, \tau_0, \tau_1)$
(b) $type(\mathsf{T}) = \mathsf{rdm} \implies SX(x, r_0, r_1) > SX(x, r_0', r_1')$ *and* $\Delta X_\Gamma(x, \tau_0, \tau_1) < \Delta X_{\Gamma'}(x, \tau_0, \tau_1)$

It is easy to find swap rate functions that violate the properties discussed before: for instance $SX(x, r_0, r_1) = 1/x$ violates output-boundedness, additivity, reversibility and homogeneity. In the rest of the section we discuss some notable swap rate functions, used in actual AMM implementations, showing that they satisfy most of our properties.

5.6. **Constant sum swap rate.** The *constant sum* function mandates the sum of the token reserves in an AMM to remain constant, i.e. $r_0 + r_1 = k$, where the constant k is fixed upon the first deposit in the AMM.

Theorem 5.14 (Constant sum swap rate). *The* constant sum *swap rate function:*

$$SX(x, r_0, r_1) \;=\; 1$$

is monotonic, reversible, additive, and homogeneous. Furthermore, its internal swap rate and its slippage are given by:

$$X_\Gamma(\tau_0, \tau_1) \;=\; 1 \qquad\qquad \Delta X_\Gamma(x, \tau_0, \tau_1) \;=\; 0$$

Note that the constant sum function is *not* output-bounded, since the output amount may exceed the reserves of the output token. A positive aspect of constant sum AMMs is that they do not suffer from slippage. With constant sum AMMs, the internal exchange rate is always 1, and so there is zero slippage (see Equations (3.3) and (3.4)). A negative aspect is that constant sum AMMs do not allow the token reserves to grow unboundedly: indeed, the bound is fixed with the first deposit. This makes constant sum AMMs unsuitable for scenarios where one wants the liquidity of the AMM to increase over time, and to incentivise users to deposit through minted tokens. When the oracle and internal exchange rates are not aligned (i.e., when the prices of the two tokens are different), then rational users will drain the reserves of the most expensive token type held by the AMM. Despite these drawbacks, the constant sum swap rate is suitable situations where the two token types in the AMM are supposed to be equally prices, like for stablecoins. This is the case e.g. for mStable [mSt20].

5.7. **Constant product swap rate.** The constant product swap rate function (introduced before in Definition 2.1) enjoys all the properties discussed previously in this section.[4]

Theorem 5.15 (Constant product). *The constant product swap rate function is output-bounded, strictly monotonic, reversible, additive, and homogeneous. Furthermore, its internal swap rate and its slippage are given by:*

$$X_\Gamma(\tau_0, \tau_1) \;=\; \frac{r_1}{r_0} \qquad\qquad \Delta X_\Gamma(x, \tau_0, \tau_1) \;=\; \frac{x}{r_0}$$

Compared to the constant sum swap rate, a point in favour of the constant product is output-boundedness, which allows users to add unbounded liquidity to the AMM. A point against is slippage, which grows linearly with the amount of the input token. Therefore, when the internal exchange rate is aligned with the oracle's, users are disincentivised from

---

[4] The existence of other classes of swap rate functions enjoying all the six properties is an open question.

performing large swaps. The most prominent AMM platform adopting the constant product is Uniswap v2 [uni21]. Curve [cur20] uses a hybrid swap rate function, which approximates a constant sum for an interval of input values x, and behaves as a constant product outside the interval. In this way, it achieves a small slippage within the interval, at the same time allowing unbounded liquidity thanks to output-boundedness.

5.8. **Constant mean swap rate.** The constant mean swap rate function generalises the constant product by associating weights $w_0, w_1 \in \mathbb{R}_{>0}$ to the token types held by the AMM, so to preserve the following equality:

$$r_0^{w_0} r_1^{w_1} = (r_0 + x)^{w_0} (r_1 + y)^{w_1} \qquad \text{where } y = x \cdot SX(x, r_0, r_1)$$

The following theorem shows that the constant mean function enjoys most of the properties of the constant product, except reversibility.

Theorem 5.16 (Constant mean swap rate). *The* constant mean *swap rate function:*

$$SX(x, r_0, r_1) \;=\; \frac{r_1}{x}\left(1 - \left(\frac{r_0}{r_0 + x}\right)^{\frac{w_0}{w_1}}\right)$$

is output-bounded, monotonic, additive, and homogeneous. Furthermore, its internal swap rate and its slippage are given by:

$$X_\Gamma(\tau_0, \tau_1) \;=\; \frac{r_1 w_0}{r_0 w_1} \qquad\qquad \Delta X_\Gamma(x, \tau_0, \tau_1) \;=\; \frac{x w_0}{r_0 w_1\left(1 - \left(\frac{r_0}{r_0 + x}\right)^{\frac{w_0}{w_1}}\right)} - 1$$

The most prominent AMM plaform using the constant mean swap rate is Balancer [bal19]. Users fix the weights w_0, w_1 of token types when an AMM is created; once fixed, these weights cannot be changed. The constant product swap rate can be seen as the special case of the constant mean where the two weights are equal.

6. The economic mechanism of AMMs

AMMs can be seen as games where users compete to increase their net worth. We now study the incentive mechanisms of AMMs from a game-theoretic perspective.

6.1. **Arbitrage.** The ***arbitrage game*** is a single-player, single-round game, where the player can perform a single move on a given AMM pair τ_0, τ_1 in order to maximize her gain. The initial game states have the form $\Gamma_0 = \mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$, where A is the player; the *moves* of A are all the possible transactions that can be fired by A (we also consider doing nothing as a possible move). More formally, a move is a sequence λ such that either $\lambda = \varepsilon$ (the empty sequence), or $\lambda = \mathsf{T}$ with $wal(\mathsf{T}) = \mathsf{A}$. The goal of A is to maximize her gain $G_\mathsf{A}(\Gamma_0, \lambda)$ on the AMM pair τ_0, τ_1. A *solution* to the game is a move λ that satisfies such goal. We study the arbitrage game under the assumption that A holds no minted tokens $\{\tau_0, \tau_1\}$. In this way, by Lemma 3.2, A's gain only depends on the input amount of A's swap, on the reserves of τ_0 and τ_1 in the AMM, and on their prices. In practice, AMM users are logically partitioned in two groups, e.g. liquidity providers (who perform deposits and redeems) and traders (who perform swaps), so basically here we are considering the arbitrage game from the traders' point of view. We further assume that A's balance is enough to allow A to perform the optimal swap. This is a common assumption in formulations of the arbitrage game: in practice, this can be achieved by borrowing the

needed amount of the input token from a lending pool via a flash-loan [QZLG21, WWL^{+}20]. Theorem 6.3 shows that a rational agent is incentivized to perform a swap to realign the internal and the oracle's exchange rate. The optimal solution to the arbitrage game can be approximated by multiple users who swap smaller amounts than the optimal one.

Before devising a solution to the arbitrage game, we examine the potential candidates for the solution. Observe that doing nothing (i.e., $\lambda = \varepsilon$) has clearly zero gain, as well as depositing or redeeming, as established by Lemma 4.5. Hence, if one of such moves is a solution, so are the other two: without loss of generality, we assume that A's move will be $\lambda = \varepsilon$ when there is no strategy which allows A to increase her gain.

We first show in Lemma 6.2 that, if a swap with input τ_0 and output τ_1 has a positive gain, then a swap with input τ_1 and output τ_0 will have a negative gain, whatever input amount is chosen. This holds whenever the swap rate function is monotonic and reversible. Lemma 6.1 is instrumental to prove Lemma 6.2, as it finds the needed relation between the swap rate function and the exchange rate. Passing from this relation to the gain of the swap transaction is obtained by means of Lemma 3.3.

Lemma 6.1. *If SX is strictly monotonic and reversible, then for all $x > 0$:*

$$SX(x, r_0, r_1) \geq X(\tau_0, \tau_1) \implies \forall y > 0.\ SX(y, r_1, r_0) < X(\tau_1, \tau_0)$$

Lemma 6.2 (Unique direction for swap gain). *Let $\Gamma = \mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$ be such that $\sigma\{\tau_0, \tau_1\} = 0$, and let $\mathsf{T}_d(x) = \mathsf{A} : \mathsf{swap}(x, \tau_d, \tau_{1-d})$, for $x > 0$ and $d \in \{0, 1\}$. If SX is output-bounded, strictly monotonic and reversible, then for all $y > 0$ such that $\sigma\tau_{1-d} \geq y$:*

$$G_\mathsf{A}(\Gamma, \mathsf{T}_d(x)) > 0 \implies G_\mathsf{A}(\Gamma, \mathsf{T}_{1-d}(y)) < 0$$

Theorem 6.3 devises a general solution to the arbitrage game, determining the swap transaction that maximizes A's gain. This is the transaction $\mathsf{A} : \mathsf{swap}(x_0, \tau_0, \tau_1)$ such that, in the state Γ' reached after performing it from the initial state, the internal exchange rate between τ_0 and τ_1 is aligned to the oracle's exchange rate. By Lemma 3.3, no move from Γ' can increase A's gain, i.e. the solution for the arbitrage game in Γ' is to do nothing. Lemma 6.2 guarantees that swaps in the other direction are not solutions, since they decrease A's gain. Note that if the internal exchange rate is already aligned to the oracle's, or if A has not enough balance to perform the optimal swap, then the solution to the arbitrage problem is to do nothing.

Theorem 6.3 (Arbitrage). *Let $\Gamma = \mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$ be such that $\sigma\{\tau_0, \tau_1\} = 0$. For all $x > 0$, let $\mathsf{T}(x) = \mathsf{A} : \mathsf{swap}(x, \tau_0, \tau_1)$. Let x_0 be such that:*

$$X_{\Gamma'}(\tau_0, \tau_1) = X(\tau_0, \tau_1) \qquad \text{where } \Gamma \xrightarrow{\mathsf{T}(x_0)} \Gamma' \tag{6.1}$$

If SX is output-bounded, strictly monotonic, additive and reversible, then:

$$\forall x \neq x_0\ :\ G_\mathsf{A}(\Gamma, \mathsf{T}(x_0)) > G_\mathsf{A}(\Gamma, \mathsf{T}(x))$$

Furthermore, if an x_0 satisfying Equation (6.1) exists, it is unique.

An implicit desideratum on these solutions is that, given a specific instance of the swap rate function, they are efficiently computable: this is the case, e.g., for the constant product, for which Lemma 6.4 finds a closed formula for the arbitrage solution.

Lemma 6.4 (Arbitrage and constant product). *Let $\Gamma = \mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\}$, and let:*

$$x_0\ =\ \sqrt{\frac{P\tau_1}{P\tau_0} r_0 r_1} - r_0 \tag{6.2}$$

If SX is the constant product swap rate and $x_0 > 0$, then $\mathsf{A} : \mathsf{swap}(x_0, \tau_0, \tau_1)$ is the solution to the arbitrage game in Γ.

Example 6.5. Consider an initial state $\Gamma = \mathsf{A}[\sigma] \mid \{18 : \tau_0, 6 : \tau_1\} \mid \Delta$. Assuming the constant product swap rate, and $P\tau_0 = 3$, $P\tau_1 = 4$, we have that:

$$X_\Gamma(\tau_0, \tau_1) \ = \ 6/18 \ < \ 3/4 \ = \ X(\tau_0, \tau_1)$$

$$X_\Gamma(\tau_1, \tau_0) \ = \ 18/6 \ > \ 4/3 \ = \ X(\tau_1, \tau_0)$$

By Theorem 6.3 it follows that the solution to the arbitrage game is $\mathsf{T}(x) = \mathsf{A} : \mathsf{swap}(x, \tau_1, \tau_0)$, for suitable x. By Lemma 6.4, we find that the optimal input value is:

$$x_1 \ = \ \sqrt{\frac{3}{4} \cdot 18 \cdot 6} - 6 = 3$$

and the corresponding output value is $x_1 \cdot SX(x_1, 6, 18) = 6$. We then obtain:

$$\Gamma \xrightarrow{\mathsf{T}(x_1)} \Gamma' \ = \ \mathsf{A}[\sigma - 3 : \tau_1 + 6 : \tau_0] \mid \{12 : \tau_0, 9 : \tau_1\}$$

This action maximizes A's gain $G_\mathsf{A}(\Gamma, \mathsf{T}(x_1)) = W_\mathsf{A}(\Gamma') - W_\mathsf{A}(\Gamma) = 6P\tau_0 - 3P\tau_1 = 6$. Any other action would result in a lower gain for A. Note that the internal exchange rate in Γ' is aligned to the oracle's: $X_{\Gamma'}(\tau_0, \tau_1) = 9/12 = 3/4 = X(\tau_0, \tau_1)$. $\diamond$

6.2. **Swaps after deposits.** We show in Theorem 6.6 that deposits incentivise swaps. Namely, if a user B performs a deposit on an AMM for the token pair τ_0, τ_1, and then a *different* user A performs a swap in the resulting state, then A's gain is increased w.r.t. the gain that she would have obtained by performing the same transaction *before* B's deposit. The intuition is that larger amounts of tokens in an AMM provide decrease the slippage, therefore attracting users interested in swaps.

Theorem 6.6 (Swap after deposit). *Let T_swap and T_dep be two transactions such that $wal(\mathsf{T}_\mathsf{swap}) = \mathsf{A} \neq wal(\mathsf{T}_\mathsf{dep})$ and, for $\ell \in \{\mathsf{swap}, \mathsf{dep}\}$, $type(\mathsf{T}_\ell) = \ell$ and $tok(\mathsf{T}_\ell) = \{\tau_0, \tau_1\}$. Let Γ be such that both T_swap and $\mathsf{T}_\mathsf{dep}\mathsf{T}_\mathsf{swap}$ are enabled in Γ. If the swap rate function is homogeneous and strictly monotonic, then:*

$$G_\mathsf{A}(\Gamma, \mathsf{T}_\mathsf{dep}\mathsf{T}_\mathsf{swap}) > G_\mathsf{A}(\Gamma, \mathsf{T}_\mathsf{swap})$$

Example 6.7. Let $\Gamma = \mathsf{A}[5 : \tau_0] \mid \{5 : \tau_0, 10 : \tau_1\} \mid \Delta$, let $\mathsf{T}_\mathsf{dep} = \mathsf{B} : \mathsf{dep}(40 : \tau_0, 80 :\tau_1)$, and let $\mathsf{T}_\mathsf{swap} = \mathsf{A} : \mathsf{swap}(5, \tau_0, \tau_1)$. Assuming the constant product swap rate, we have that:

$$\Gamma \xrightarrow{\mathsf{T}_\mathsf{swap}} \Gamma_s = \mathsf{A}[5 : \tau_1] \mid \{10 : \tau_0, 5 : \tau_1\} \mid \Delta$$

$$\Gamma \xrightarrow{\mathsf{T}_\mathsf{dep}} \Gamma_d = \mathsf{A}[5 : \tau_0] \mid \{45 : \tau_0, 90 : \tau_1\} \mid \Delta' \xrightarrow{\mathsf{T}_\mathsf{swap}} \Gamma_{ds} = \mathsf{A}[9 : \tau_1] \mid \{50 : \tau_0, 81 : \tau_1\} \mid \Delta'$$

Now, assuming $P\tau_0 = 1$ and $P\tau_1 = 1$, we have the following gains for A:

$$G_\mathsf{A}(\Gamma, \mathsf{T}_\mathsf{dep}\mathsf{T}_\mathsf{swap}) \ = \ 4 \ > \ 0 \ = \ G_\mathsf{A}(\Gamma, \mathsf{T}_\mathsf{swap})$$

as correctly predicted by Theorem 6.6. Note that in the state Γ before the deposit, A has zero gain from her swap, while the same transaction has a positive gain after the deposit. $\diamond$

Theorem 6.8 finds the solution of the arbitrage game after a deposit of another user. More precisely, let λ be the solution in Γ, and let λ' be the solution in the state Γ' reached after a deposit. If λ is empty, then also λ' is such. If λ is a swap with input τ_0 and output τ_1, then also λ' is such (but for the input amount).

Theorem 6.8 (Arbitrage after deposit). *Let* $\Gamma = A[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$, *and let:*

$$\Gamma \xrightarrow{\text{B:dep}(v_0:\tau_0, v_1:\tau_1)} \Gamma_d \qquad \text{where } \Gamma_d = A[\sigma'] \mid \{r_0' : \tau_0, r_1' : \tau_1\} \mid \Delta' \text{ and } B \neq A$$

Let λ *and* λ_d *be the solutions of the arbitrage game in* Γ *and in* Γ_d, *respectively. If SX is output-bounded, strictly monotonic, additive, reversible, and homogeneous, then:*

(1) *if* $\lambda = A : \text{swap}(x, \tau_0, \tau_1)$, *then*

$$\lambda_d = A : \text{swap}(ax, \tau_0, \tau_1) \qquad G_A(\Gamma_d, \lambda_d) = a\, G_A(\Gamma, \lambda) \qquad \text{where } a = \frac{r_1 + v_1}{r_1}$$

(2) *if* $\lambda = \varepsilon$, *then* $\lambda_d = \varepsilon$.

6.3. **Swaps after redeems.** We now study swaps and arbitrage after redeems. Conversely to what we have shown before in Theorem 6.6, we find that redeems disincentivise swaps (Theorem 6.9). Similarly to Theorem 6.8, if the solution to the arbitrage game in a state Γ is a swap, then after a redeem in Γ the solution is still a swap which only differs in the input amount (Theorem 6.10).

Theorem 6.9 (Swap after redeem). *Let* T_{swap} *and* T_{rdm} *be two transactions such that* $wal(T_{\text{swap}}) = A \neq wal(T_{\text{rdm}})$ *and, for* $\ell \in \{\text{swap}, \text{rdm}\}$, $type(T_\ell) = \ell$ *and* $tok(T_\ell) = \{\tau_0, \tau_1\}$. *Let* Γ *be such that both* T_{swap} *and* $T_{\text{rdm}} T_{\text{swap}}$ *are enabled in* Γ. *If the swap rate function is homogeneous and strictly monotonic, then:*

$$G_A(\Gamma, T_{\text{rdm}} T_{\text{swap}}) < G_A(\Gamma, T_{\text{swap}})$$

Theorem 6.10 (Arbitrage after redeem). *Let* $\Gamma = A[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$, *and let:*

$$\Gamma \xrightarrow{\text{B:rdm}(v:\{\tau_0, \tau_1\})} \Gamma_d \qquad \text{where } \Gamma_d = A[\sigma'] \mid \{r_0' : \tau_0, r_1' : \tau_1\} \mid \Delta' \text{ and } B \neq A$$

Let λ *and* λ_d *be the solutions of the arbitrage game in* Γ *and in* Γ_d, *respectively. If SX is output-bounded, strictly monotonic, additive, reversible, and homogeneous, then:*

(1) *if* $\lambda = A : \text{swap}(x, \tau_0, \tau_1)$, *then*

$$\lambda_d = A : \text{swap}(ax, \tau_0, \tau_1) \qquad G_A(\Gamma_d, \lambda_d) = a\, G_A(\Gamma, \lambda) \qquad \text{where } a = 1 - \frac{v}{S_\Gamma\{\tau_0, \tau_1\}}$$

(2) *if* $\lambda = \varepsilon$, *then* $\lambda_d = \varepsilon$.

7. MAXIMAL EXTRACTABLE VALUE

Maximal Extractable Value (MEV) refers to a class of attacks to smart contracts where miners/validators exploit their power to reorder, drop or insert transactions in a block to "extract" value from the *mempool* (i.e., the set of transactions sent to the blockchain network, but not appearing yet in a block). Empirical research has shown that AMMs are routinely targeted by MEV attacks [DGK+20,QZG21,ZQC+21,ZQT+21], and indeed recent versions of the Ethereum protocol implementation include a MEV extraction mechanism [mev22]. This has negative effects on AMM users, as well as on transaction fees and network congestion.

We show that our AMM model makes it possible to faithfully express MEV attacks. Consider a constant product AMM for two token types τ_0, τ_1 with the same price, e.g. $P\tau_0 = P\tau_1 = 1$, and consider a state:

$$\Gamma = M[\cdots] \mid A[50 : \tau_0] \mid \{10 : \tau_0, 10 : \tau_1\} \mid \cdots$$

where we use A to impersonate a honest user, and M for a miner, acting as an adversary. By Lemma 3.3 we know that the AMM is in equilibrium in Γ, because, for each $x > 0$:

$$SX(x, 10, 10) = \frac{10}{10 + x} < 1 = X(\tau_0, \tau_1)$$

Therefore, neither a miner nor any other user can increase their net worth in Γ.

Assume now that A sends the transaction $T_A = A : \mathsf{swap}(50, \tau_0, \tau_1)$ to the blockchain network. Before being included in a block, T_A is added to the mempool, from where miners gather transactions to construct blocks. Any miner owning enough token units can increase their gain by firing A's transaction within a *sandwich* of M's swaps. For instance, assume that M's wallet is $M[40 : \tau_0, 1 : \tau_1]$. Then M can construct a block:

$$\lambda \;=\; M : \mathsf{swap}(40, \tau_0, \tau_1) \;\; T_A \;\; M : \mathsf{swap}(9, \tau_1, \tau_0)$$

We have that $\Gamma \xrightarrow{\lambda} \Gamma'$, where:

$$\Gamma \xrightarrow{M:\mathsf{swap}(40,\tau_0,\tau_1)} M[0 : \tau_0, 9 : \tau_1] \mid A[50 : \tau_0] \mid \{50 : \tau_0, 2 : \tau_1\} \mid \cdots$$

$$\xrightarrow{A:\mathsf{swap}(50,\tau_0,\tau_1)} M[0 : \tau_0, 9 : \tau_1] \mid A[0 : \tau_0, 1 : \tau_1] \mid \{100 : \tau_0, 1 : \tau_1\} \mid \cdots$$

$$\xrightarrow{M:\mathsf{swap}(9,\tau_1,\tau_0)} M[90 : \tau_0, 0 : \tau_1] \mid A[0 : \tau_0, 1 : \tau_1] \mid \{10 : \tau_0, 10 : \tau_1\} \mid \cdots \;= \Gamma'$$

This results in a positive gain for M, since:

$$G_M(\Gamma, \lambda) = W_M(\Gamma') - W_M(\Gamma) = 90 \cdot P\tau_0 - (40 \cdot P\tau_0 + 1 \cdot P\tau_1) = 49$$
$$G_A(\Gamma, \lambda) = W_A(\Gamma') - W_A(\Gamma) = 1 \cdot P\tau_1 - 50 \cdot P\tau_0 = -49$$

Summing up, M has managed to extract value from A's transaction in the mempool, improving her gain to the detriment of A's net worth.

The mechanism of *guarded transactions*, which allows users to specify a lower bound to the amount of tokens outputted upon a swap (see section 8), is a partial countermeasure against MEV attacks. For instance, in the scenario above A could have sent a guarded transaction $T'_A = A : \mathsf{swap}(50 : \tau_0, 8.3 : \tau_1)$, which would have ensured A to receive at least $8.3 : \tau_1$ upon the swap. This would have neutralised the sandwich attack described before, since after the first M's transaction, T'_A is no longer valid. Even though guarded transactions mitigate the issue of not knowing the state where one's transaction will be fired, they are not a complete defence against MEV attacks. Indeed, in [BCL22] it is shown that adversaries can craft sandwiches that extract value from *any* non-empty mempool of swap and dep (guarded) transactions. Further analyses the effect of MEV on constant-function AMMs are developed in [KDC22]. Several approaches to prevent MEV attacks are discussed in [HW22, ByCD$^+$21].

8. Variants of the basic model

Our AMM model abstracts from implementation-specific features, and from the features that are orthogonal to the core functionality of AMMs (e.g., governance). We discuss below some extensions and variants of our model to make it closer to actual implementations, and their impact on our theory.

8.1. **Fees.** In actual AMM implementations, the swap rate — and consequently, the semantics of [Swap] actions — also depends on a *trading fee* $1 - \phi$. For instance, incorporating this fee in the constant product swap rate function is usually done as follows:

$$SX_\phi(x, r_0, r_1) \;=\; \frac{\phi\, r_1}{r_0 + \phi\, x} \qquad \text{where } \phi \in [0, 1]$$

In this case, when the trading fee is zero (i.e., $\phi = 1$), the swap rate preserves the product between AMM reserves; a higher fee, instead, results in reduced amounts of output tokens received from swap actions. Intuitively, the AMM retains a portion of the swapped amounts, but the overall reserves are still distributed among all minted tokens, thereby increasing the redeem rate of minted tokens. The structural properties in section 4 are not affected by swap fees.

8.2. **Price updates.** An underlying assumption of our model is that the price of atomic tokens is constant, and consequently that exchange rates are stable. In the wild, prices and exchange rates can vary over time, possibly making the net worth of users holding minted tokens decrease — a phenomenon commonly referred to as *impermanent loss* [imp20].

Introducing price updates in our AMM model is straightforward: it suffices to extend states Γ with price oracles, parameterize with Γ the exchange rate X, and extend the AMM semantics with a rule to non-deterministically update token prices. Most of the structural properties in section 4 would not be affected by this extension: the exceptions are determinism (Lemma 4.1) and net worth preservation (Lemma 4.5(b), while part (a) would still be true for deposits and redeems). Technically, also the properties about swaps and incentives in section 5 and section 6 are preserved, although this happens because most of these properties assume sequences of deposits, redeems and swaps. If we allow these actions to be interleaved with price updates, some properties no longer hold: notably, the optimality of the solution λ to the arbitrage problem (Theorem 6.3) is lost if λ is front-run by a price update that alters the exchange rates, since this affects the condition provided by Theorem 6.3.

In practice, the assumption of constant exchange rates assumed by Theorem 6.3 may hold in the case of exchanges between stable coins [mak20]. Here, arbitrage ensures the alignment between swap rates and exchange rates, so users are hence incentivized to provide liquidity to AMMs, as the redeem rate is likely to increase over time.

8.3. **Guarded transactions.** The semantics of AMMs in section 2 defines how the state evolves upon transactions. In practice, when a user emits a transaction, she cannot predict the exact state in which it will be actually committed. This may lead to unexpected or unwanted behaviours. For instance, the gain of a swap transaction sent by A may be reduced if the transaction is front-run by a redeem transaction sent by B, as established by Theorem 6.9. The problem here is that redeems decrease the swap rate (by Lemma 5.13), and consequently the amount of output tokens received by A. As a partial countermeasure to this issue, Uniswap allows users to specify a lower bound y^{min} to the amount of received tokens. In our model, we could formalise this behaviour by amending the [Swap] rule as

follows:

$$\frac{\sigma\tau_0 \geq x > 0 \qquad y = x \cdot SX(x, r_0, r_1) \qquad y^{min} \leq y < r_1}{\mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Gamma \xrightarrow{\mathsf{A:swap}(x:\tau_0, y^{min}:\tau_1)} \\ \mathsf{A}[\sigma - x : \tau_0 + y : \tau_1] \mid \{r_0 + x : \tau_0, r_1 - y : \tau_1\} \mid \Gamma} \text{[Swap]}$$

Similar countermeasures apply to [Rdm] and [Dep] rules. For redeems, the user can enforce lower bounds v_0^{min}, v_1^{min} on the amount of received tokens τ_0, τ_1 as follows:

$$\frac{\sigma\{\tau_0, \tau_1\} \geq v > 0 \qquad v < S_\Gamma\{\tau_0, \tau_1\} \qquad v_i = v \cdot RX_\Gamma^i(\tau_0, \tau_1) \qquad v_i^{min} \leq v_i}{\begin{aligned} \Gamma \;=\;\; & \mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Gamma' \xrightarrow{\mathsf{A:rdm}(v:\{\tau_0,\tau_1\}, v_0^{min}:\tau_0, v_1^{min}:\tau_1)} \\ & \mathsf{A}[\sigma + v_0 : \tau_0 + v_1 : \tau_1 - v : \{\tau_0, \tau_1\}] \mid \{r_0 - v_0 : \tau_0, r_1 - v_1 : \tau_1\} \mid \Gamma' \end{aligned}} \text{[Rdm]}$$

Amending the [Dep] rule is more complex, since here we must define ranges for the deposited amounts v_0, v_1, and we must preserve the ratio between the AMM reserves. A possible way to achieve this behaviour is the following rule:

$$\frac{\sigma\tau_i \geq v_i > 0 \quad v = \frac{v_i}{RX_\Gamma^i(\tau_0,\tau_1)} \quad (v_0, v_1) = \begin{cases} (v_0^{max}, v_0^{max} \cdot \frac{r_1}{r_0}) & \text{if } v_1^{min} \leq v_0^{max} \cdot \frac{r_1}{r_0} \leq v_1^{max} \\ (v_1^{max} \cdot \frac{r_0}{r_1}, v_1^{max}) & \text{if } v_0^{min} \leq v_1^{max} \cdot \frac{r_0}{r_1} \leq v_0^{max} \end{cases}}{\begin{aligned} \Gamma \;=\;\; & \mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Gamma' \xrightarrow{\mathsf{A:dep}(v_0^{min}, v_0^{max}:\tau_0, v_1^{min}, v_1^{max}:\tau_1)} \\ & \mathsf{A}[\sigma - v_0 : \tau_0 - v_1 : \tau_1 + v : \{\tau_0, \tau_1\}] \mid \{r_0 + v_0 : \tau_0, r_1 + v_1 : \tau_1\} \mid \Gamma' \end{aligned}} \text{[Dep]}$$

These amendments, which are coherent with Uniswap implementation [uni21], preserve all the properties, both structural and economic, established in the previous sections, modulo a restatement of the properties which have transactions in their hypotheses. For instance, in Theorem 6.8, the scaling factor a will be computed on the actual deposited value, rather than on the parameter of the transaction. Note that, although the new rules can disable some transactions which were enabled with the rules in section 2, this does not affect the transactions reordering result (Lemma 4.9).

8.4. **Other variants.** There are further differences between our model and the existing AMM platforms, that could be accounted for in extensions of our model. Uniswap implements flash-loans as part of the swap actions: namely, the user can optionally borrow available pair funds [uni20a] whilst returning these within the same *atomic group* of actions. Further, Uniswap implements an exchange rate oracle, allowing smart contracts to interpret (averages of) recent swap rates as exchange rates [uni20b]. Balancer [bal19] extends token pairs to token *tuples*: a user can swap any two non-coinciding sets of supported tokens, such that the swap rate is maintained. In all AMM implementations, token balances are represented as integers: consequently, they are subject to rounding errors [rva18]. AMM platforms frequently implement a governance logic, which allow "governance token" holders to coordinate changes to AMM fee-rates or swap rate parameters.

9. Conclusions

We have proposed a theory of AMMs, which encompasses and generalizes the main functional and economic aspects of the mainstream AMM implementations, providing solid grounds for the design of future AMMs.

The core of our theory is a formal model of AMMs (section 2), based on a thorough inspection of leading AMM implementations like Uniswap [uni21], Curve [cur21b], and Balancer [bal19]. An original aspect of our model is that it is parametric with respect to the key economic mechanism — the *swap rate function* — that algorithmically determines exchange rates between tokens. Our model features an *executable semantics*, which can support future implementations and analysis tools; an open-source implementation of our semantics is available as a companion of this paper.

Building upon our model, we prove a set of properties characterizing both structural (section 4) and economic (section 3, section 6) aspects of AMMs. Structural properties include, e.g., that value cannot be created or destroyed (Lemma 4.5), that tokens cannot be frozen within an AMM (Lemma 4.8) and that some sequences of transactions can be reordered without affecting their semantics (Lemma 4.9). Concerning the economic properties, we address the *arbitrage problem*, the main game-theoretic foundation behind the economic incentives of AMMs. Theorem 6.3 provides sufficient conditions for the existence of solutions, and links the solutions to the expected relation between internal exchange rate and oracle's exchange rate. We show that deposits incentivize swaps, while redeems have the opposite effect. With respect to previous works, which focus on specific economic mechanisms, all our results are parametric with respect to the swap rate function. We identify indeed, for each property, a set of conditions on swap rate functions that are sufficient for the property to hold (section 5).

AMM platforms like Uniswap [uni21] and Curve [Ego19] have overtaken centralized cryptocurrency markets in size and usage. On the one hand, a better understanding of AMM design in cases where AMMs host the majority of the token's global swap volume is critical [AEC20]. On the other hand, the growth of AMMs is making them more attractive for malicious users, even if it is difficult to exactly quantify the effect of attacks.

This paper, together with our work on formalizing another DeFi archetype called *lending pool* [BCL21a], is the first step towards a general theory of DeFi [BCL21c]. We believe that a general theory encompassing interactions between different DeFi archetypes is crucial to be able to reason about their structural, economic and security aspects, as typical DeFi applications operate within a wider ecosystem, composed by a set of collaborating or competing agents, which interact through possibly separate execution environments.

References

[AC20]   Guillermo Angeris and Tarun Chitra. Improved price oracles: Constant function market makers. In *ACM Conference on Advances in Financial Technologies (AFT)*, pages 80–91. ACM, 2020. `https://arxiv.org/abs/2003.10001`. `doi:10.1145/3419614.3423251`.

[AEC20]   Guillermo Angeris, Alex Evans, and Tarun Chitra. When does the tail wag the dog? Curvature and market making. *arXiv preprint arXiv:2012.08040*, 2020. URL: `https://arxiv.org/abs/2012.08040`.

[AKC⁺21]   Guillermo Angeris, Hsien-Tang Kao, Rei Chiang, Charlie Noyes, and Tarun Chitra. An analysis of Uniswap markets. *Cryptoeconomic Systems*, 1(1), 2021. `doi:10.21428/58320208.c9738e64`.

[bal19]   Balancer whitepaper, 2019. `https://balancer.finance/whitepaper/`.

[BCL21a]   Massimo Bartoletti, James Hsin-yu Chiang, and Alberto Lluch-Lafuente. SoK: Lending pools in decentralized finance. In *Financial Cryptography Workshops*, volume 12676 of *LNCS*, pages 553–578. Springer, 2021. `doi:10.1007/978-3-662-63958-0_40`.

[BCL21b]   Massimo Bartoletti, James Hsin-yu Chiang, and Alberto Lluch-Lafuente. A Theory of Automated Market Makers in DeFi. In *Coordination Models and Languages*, volume 12717 of *LNCS*, pages 168–187. Springer, 2021. `doi:10.1007/978-3-030-78142-2_11`.

[BCL21c]   Massimo Bartoletti, James Hsin-yu Chiang, and Alberto Lluch-Lafuente. Towards a theory of decentralized finance. In *Financial Cryptography Workshops*, volume 12676 of *LNCS*, pages 227–232. Springer, 2021. `doi:10.1007/978-3-662-63958-0_20`.

[BCL22]   Massimo Bartoletti, James Hsin-yu Chiang, and Alberto Lluch-Lafuente. Maximizing extractable value from Automated Market Makers. In *Financial Cryptography*, volume 13411 of *LNCS*, pages 3–19. Springer, 2022. `doi:10.1007/978-3-031-18283-9_1`.

[Ben73]   C. H. Bennett. Logical reversibility of computation. *IBM J. Res. Dev.*, 17:525–532, November 1973.

[ByCD⁺21] Carsten Baum, James Hsin yu Chiang, Bernardo David, Tore Kasper Frederiksen, and Lorenzo Gentile. SoK: Mitigation of front-running in decentralized finance. Cryptology ePrint Archive, Report 2021/1628, 2021. `https://ia.cr/2021/1628`.

[CAE22]   Tarun Chitra, Guillermo Angeris, and Alex Evans. Differential privacy in constant function market makers. 13411:149–178, 2022. `doi:10.1007/978-3-031-18283-9_8`.

[cur20]   Curve website, 2020. URL: `https://www.curve.fi`.

[cur21a]   Curve computation of invariant constant, 2021. `https://github.com/curvefi/curve-contract/blob/a1b5a797790d3f5ef12b0e358892a0ce47c12f85/contracts/pool-templates/base/SwapTemplateBase.vy#L206`.

[cur21b]   Curve token pair implementation, 2021. `https://github.com/curvefi/curve-contract/blob/a1b5a797790d3f5ef12b0e358892a0ce47c12f85/contracts/pool-templates/base/SwapTemplateBase.vy`.

[cur22]   Curve statistics, 2022. `https://www.curve.fi/dailystats`.

[def22]   Documented timeline of exchange hacks, 2022. `https://cryptosec.info/exchange-hacks/`.

[DGK⁺20]   P. Daian, S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, and A. Juels. Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In *IEEE Symposium on Security and Privacy*, pages 910–927. IEEE, 2020. `doi:10.1109/SP40000.2020.00040`.

[DKP21]   Vincent Danos, Hamza El Khalloufi, and Julien Prat. Global order routing on exchange networks. In *Financial Cryptography Workshops*, volume 12676 of *LNCS*, pages 207–226. Springer, 2021. `doi:10.1007/978-3-662-63958-0_19`.

[EAC21]   Alex Evans, Guillermo Angeris, and Tarun Chitra. Optimal fees for geometric mean market makers. In *Financial Cryptography Workshops*, volume 12676 of *LNCS*, pages 65–79. Springer, 2021. `doi:10.1007/978-3-662-63958-0_6`.

[Ego19]   Michael Egorov. Stableswap - efficient mechanism for stablecoin, 2019. `https://curve.fi/files/stableswap-paper.pdf`.

[EMC20]   Shayan Eskandari, Seyedehmahsa Moosavi, and Jeremy Clark. SoK: Transparent Dishonesty: Front-Running Attacks on Blockchain. In *Financial Cryptography*, pages 170–189, Cham, 2020. Springer International Publishing. `doi:10.1007/978-3-030-43725-1_13`.

[HW22]   Lioba Heimbach and Roger Wattenhofer. SoK: Preventing transaction reordering manipulations in decentralized finance. *CoRR*, abs/2203.11520, 2022. `arXiv:2203.11520`, `doi:10.48550/arXiv.2203.11520`.

[imp20]   Uniswap Documentation: Understanding Returns, 2020. `https://uniswap.org/docs/v2/advanced-topics/understanding-returns/`.

[KDC22]   Kshitij Kulkarni, Theo Diamandis, and Tarun Chitra. Towards a Theory of Maximal Extractable Value I: Constant Function Market Makers. *CoRR*, abs/2207.11835, 2022. `arXiv:2207.11835`, `doi:10.48550/arXiv.2207.11835`.

[KFG21]   Bhaskar Krishnamachari, Qi Feng, and Eugenio Grippo. Dynamic curves for decentralized autonomous cryptocurrency exchanges. In *International Symposium on Foundations and Applications of Blockchain (FAB)*, volume 92 of *OASIcs*, pages 5:1–5:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. `doi:10.4230/OASIcs.FAB.2021.5`.

[mak20]   Makerdao website, 2020. `https://https://makerdao.com`.

[mev22]   MEV-geth, 2022. `https://github.com/flashbots/mev-geth`.

[moo20a]   Mooniswap implementation, 2020. `https://github.com/1inch-exchange/mooniswap/blob/02dccfab2ddbb8a409400288cb13441763370350/contracts/Mooniswap.sol`.

[moo20b]   Mooniswap whitepaper, 2020. `https://mooniswap.exchange/docs/MooniswapWhitePaper-v1.0.pdf`.

[MSG+20]   Claudio Antares Mezzina, Rudolf Schlatte, Robert Glück, Tue Haulund, James Hoey, Martin Holm Cservenka, Ivan Lanese, Torben Æ. Mogensen, Harun Siljak, Ulrik Pagh Schultz, and Irek Ulidowski. Software and reversible systems: A survey of recent activities. In *Reversible Computation: Extending Horizons of Computing - Selected Results of the COST Action IC1405*, volume 12070 of *LNCS*, pages 41–59. Springer, 2020. `doi:10.1007/978-3-030-47361-7_2`.

[mSt20]   mStable — introducing constant sum bonding curves for tokenised assets, 2020. `https://medium.com/mstable/introducing-constant-sum-bonding-curves-for-tokenised-assets-6e18879cdc5b`.

[QZA+21]   Kaihua Qin, Liyi Zhou, Yaroslav Afonin, Ludovico Lazzaretti, and Arthur Gervais. CeFi vs. DeFi - comparing centralized to decentralized finance. *CoRR*, abs/2106.08157, 2021. URL: `https://arxiv.org/abs/2106.08157`, `arXiv:2106.08157`.

[QZG21]   Kaihua Qin, Liyi Zhou, and Arthur Gervais. Quantifying blockchain extractable value: How dark is the forest? 2021. URL: `https://arxiv.org/abs/2101.05511`, `arXiv:2101.05511`.

[QZLG21]   Kaihua Qin, Liyi Zhou, Benjamin Livshits, and Arthur Gervais. Attacking the DeFi ecosystem with flash loans for fun and profit. In *Financial Cryptography*, volume 12674 of *LNCS*, pages 3–32. Springer, 2021. `doi:10.1007/978-3-662-64322-8_1`.

[rva18]   Formal specification of constant product market maker model & implementation, 2018. `https://github.com/runtimeverification/verified-smart-contracts/blob/uniswap/uniswap/x-y-k.pdf`.

[sus21]   SushiSwap token pair implementation, 2021. `https://github.com/sushiswap/sushiswap/blob/94ea7712daaa13155dfab9786aacf69e24390147/contracts/uniswapv2/UniswapV2Pair.sol`.

[uni20a]   Uniswap flash loan implementation, 2020. `https://github.com/Uniswap/uniswap-v2-core/blob/4dd59067c76dea4a0e8e4bfdda41877a6b16dedc/contracts/UniswapV2Pair.sol#L172`.

[uni20b]   Uniswap oracle template, 2020. `https://github.com/Uniswap/uniswap-v2-periphery/blob/dda62473e2da448bc9cb8f4514dadda4aeede5f4/contracts/examples/ExampleOracleSimple.sol`.

[uni21]   Uniswap token pair implementation, 2021. `https://github.com/Uniswap/uniswap-v2-core/blob/4dd59067c76dea4a0e8e4bfdda41877a6b16dedc/contracts/UniswapV2Pair.sol`.

[uni22]   Uniswap statistics, 2022. `https://info.uniswap.org`.

[vir18]   Improving frontrunning resistance of x*y=k market makers, 2018. `https://ethresear.ch/t/improving-front-running-resistance-of-x-y-k-market-makers/1281`.

[WPG+21]   Sam M. Werner, Daniel Perez, Lewis Gudgeon, Ariah Klages-Mundt, Dominik Harz, and William J. Knottenbelt. Sok: Decentralized finance (defi), 2021. `arXiv:2101.08778`.

[WWL+20]   Dabao Wang, Siwei Wu, Ziling Lin, Lei Wu, Xingliang Yuan, Yajin Zhou, Haoyu Wang, and Kui Ren. Towards understanding flash loan and its applications in DeFi ecosystem. *arXiv preprint arXiv:2010.12252*, 2020. `https://arxiv.org/abs/2010.12252`.

[XVPC22]   Jiahua Xu, Nazariy Vavryk, Krzysztof Paruch, and Simon Cousaert. Sok: Decentralized exchanges (DEX) with automated market maker (AMM) protocols. *ACM Comput. Surv.*, nov 2022. `doi:10.1145/3570639`.

[ZQC+21]   Liyi Zhou, Kaihua Qin, Antoine Cully, Benjamin Livshits, and Arthur Gervais. On the just-in-time discovery of profit-generating transactions in DeFi protocols. In *IEEE Symp. on Security and Privacy*, pages 919–936. IEEE, 2021. `doi:10.1109/SP40001.2021.00113`.

[ZQT+21]   Liyi Zhou, Kaihua Qin, Christof Ferreira Torres, Duc V Le, and Arthur Gervais. High-Frequency Trading on Decentralized On-Chain Exchanges. In *IEEE Symp. on Security and Privacy*, pages 428–445. IEEE, 2021. `doi:10.1109/SP40001.2021.00027`.

APPENDIX A. PROOFS FOR SECTION 3

Proof of Lemma 3.2. Let Γ and T be as in the hypotheses, let $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$, and let $y = x \cdot SX(x, r_0, r_1)$. By definition of gain (Equation 3.6), we have that:

$$G_\mathsf{A}(\Gamma, \mathsf{T}) \;=\; W_\mathsf{A}(\Gamma') - W_\mathsf{A}(\Gamma)$$

By definition of net worth (Equation 3.5), we have that:

$$W_\mathsf{A}(\Gamma) = \sigma_\mathsf{A}(\tau_0) \cdot P\tau_0 \;+\; \sigma_\mathsf{A}(\tau_1) \cdot P\tau_1$$
$$+ \sigma_\mathsf{A}\{\tau_0, \tau_1\} \cdot \frac{r_0 \cdot P\tau_0 + r_1 \cdot P\tau_1}{S_\Gamma\{\tau_0, \tau_1\}}$$
$$+ \sum_{\tau \notin \{\tau_0, \tau_1, \{\tau_0, \tau_1\}\}} \sigma_\mathsf{A}(\tau) \cdot P_\Gamma \tau$$
$$W_\mathsf{A}(\Gamma') = (\sigma_\mathsf{A}(\tau_0) - x) \cdot P\tau_0 \;+\; (\sigma_\mathsf{A}(\tau_1) + y) \cdot P\tau_1$$
$$+ \sigma_\mathsf{A}\{\tau_0, \tau_1\} \cdot \frac{(r_0 + x) \cdot P\tau_0 + (r_1 - y) \cdot P\tau_1}{S_{\Gamma'}\{\tau_0, \tau_1\}}$$
$$+ \sum_{\tau \notin \{\tau_0, \tau_1, \{\tau_0, \tau_1\}\}} \sigma_\mathsf{A}(\tau) \cdot P_{\Gamma'} \tau$$

Since $S_\Gamma\{\tau_0, \tau_1\} = S_{\Gamma'}\{\tau_0, \tau_1\}$ and $P_\Gamma \tau = P_{\Gamma'} \tau$ for all $\tau \neq \{\tau_0, \tau_1\}$:

$$W_\mathsf{A}(\Gamma') - W_\mathsf{A}(\Gamma) = y \cdot P\tau_1 - x \cdot P\tau_0 + \sigma_\mathsf{A}\{\tau_0, \tau_1\} \frac{x \cdot P\tau_0 - y \cdot P\tau_1}{S_\Gamma\{\tau_0, \tau_1\}}$$
$$= \left(y \cdot P\tau_1 - x \cdot P\tau_0 \right)\left(1 - \frac{\sigma_\mathsf{A}\{\tau_0, \tau_1\}}{S_\Gamma\{\tau_0, \tau_1\}} \right)$$
$$= x \cdot \left(SX(x, r_0, r_1)\, P\tau_1 - P\tau_0 \right)\left(1 - \frac{\sigma_\mathsf{A}\{\tau_0, \tau_1\}}{S_\Gamma\{\tau_0, \tau_1\}} \right)$$

Using similar calculations, for $\mathsf{B} \neq \mathsf{A}$, we obtain:

$$G_\mathsf{B}(\Gamma, \mathsf{T}) = \sigma_\mathsf{B}\{\tau_0, \tau_1\} \frac{x \cdot P\tau_0 - y \cdot P\tau_1}{S_\Gamma\{\tau_0, \tau_1\}} \qquad \square$$

Proof of Lemma 3.3. Let $y = x \cdot SX(x, r_0, r_1)$. Since $\sigma\{\tau_0, \tau_1\} = 0$, by Lemma 3.2 we have that:

$$G_\mathsf{A}(\Gamma, \mathsf{T}) \circ 0 \iff y\, P\tau_1 - x\, P\tau_0 \circ 0$$
$$\iff \frac{y}{x} \circ \frac{P\tau_0}{P\tau_1}$$
$$\iff SX(x, r_0, r_1) \circ X(\tau_0, \tau_1) \qquad \square$$

APPENDIX B. PROOFS FOR SECTION 4

Proof of Lemma 4.1. Straightforward inspection of the rules [DEP0], [DEP], [RDM], [SWAP] in section 2. $\qquad \square$

Proof of Lemma 4.2. For item (a), we proceed by induction on the length of a computation $\Gamma_0 \to^* \Gamma$, where Γ_0 is initial. The base case (computation of zero steps) is trivial, since initial states does not contain AMMs. For the inductive case, note that rule [DEP0] requires that the initial reserves of an AMM are strictly greater than zero. The rules that decrease the token reserves in AMMs, i.e. [RDM] and [SWAP], have premises that ensure that the reserves cannot be zeroed.

For item (b), we proceed by induction on the length of a computation $\Gamma_0 \to^* \Gamma$, where Γ_0 is initial. The base case is trivial, since initial states do not contain AMMs. For the inductive case, we assume that Γ satisfies the property, and we prove that it is preserved by a transition $\Gamma \to \Gamma'$. Assume that Γ' contains an AMM $\{r'_0 : \tau_0, r'_1 : \tau_1\}$. By item ((a)), $r'_0 > 0$ and $r'_1 > 0$. There are the following cases, depending on the rule used to infer $\Gamma \to \Gamma'$:

- [DEP0], [DEP]. Trivial, because deposits can only increase the supply of minted tokens.
- [SWAP]. Trivial, because swap actions do not affect the supply of minted tokens.
- [RDM]. Assume that $\{r_0 : \tau_0, r_1 : \tau_1\} \in \Gamma$. By contradiction, suppose that the [RDM] action burns all the supply of the minted token, i.e. it burns $v = S_\Gamma\{\tau_0, \tau_1\}$ units. The rule premise requires $v > 0$, and it implies:

$$r'_0 = r_0 - v\frac{r_0}{S_\Gamma\{\tau_0, \tau_1\}} = 0 \qquad r'_1 = r_1 - v\frac{r_1}{S_\Gamma\{\tau_0, \tau_1\}} = 0$$

  Therefore, we would have $r'_0 = r'_1 = 0$ — contradiction. $\qquad\square$

Proof of Lemma 4.3. By cases on the rule used in the transition $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$. It is straightforward to check that, in all the rules, the changes applied to atomic tokens cancel out. Further, the [SWAP] rule does not affect the supply of minted tokens. $\qquad\square$

Proof of Lemma 4.4. Let $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$, where $\{r_0 : \tau_0, r_1 : \tau_1\} \in \Gamma$ and $\{r'_0 : \tau_0, r'_1 : \tau'_1\} \in \Gamma'$. If $\mathsf{T} = \mathsf{A} : \mathsf{dep}(v_0 : \tau_0, v_1 : \tau_1)$, then by the [DEP] rule it must be $r'_i = r_i + v_i$ for $i \in \{0, 1\}$. Furthermore, by the premises of [DEP], we obtain:

$$r_1 v_0 = r_1 v \cdot RX_\Gamma^0(\tau_0, \tau_1) = v \cdot \frac{r_0 r_1}{S_\Gamma\{\tau_0, \tau_1\}} = r_0 v \cdot RX_\Gamma^1(\tau_0, \tau_1) = r_0 v_1$$

Therefore:

$$\frac{r_1 + v_1}{r_0 + v_0} = \frac{(\frac{r_0 v_1}{v_0}) + v_1}{r_0 + v_0} = \frac{r_0 v_1 + v_0 v_1}{(r_0 + v_0)v_0} = \frac{(r_0 + v_0)v_1}{(r_0 + v_0)v_0} = \frac{v_1}{v_0} = \frac{r_1}{r_0} \qquad (\text{B.1})$$

If $\mathsf{T} = \mathsf{A} : \mathsf{rdm}(v : \{\tau_0, \tau_1\})$, then by rule [RDM] it must be, for $i \in \{0, 1\}$:

$$r'_i = r_i - v_i = r_i - vRX_\Gamma^i(\tau_0, \tau_1) = r_i - v\frac{r_i}{S_\Gamma\{\tau_0, \tau_1\}}$$

Therefore, since $S_\Gamma\{\tau_0, \tau_1\} = S_{\Gamma'}\{\tau_0, \tau_1\} + v$:

$$\frac{r_1 - v_1}{r_0 - v_0} = \frac{r_1 - v\frac{r_1}{S_\Gamma\{\tau_0,\tau_1\}}}{r_0 - v\frac{r_0}{S_\Gamma\{\tau_0,\tau_1\}}} = \frac{r_1(S_{\Gamma'}\{\tau_0, \tau_1\} + v) - vr_1}{r_0(S_{\Gamma'}\{\tau_0, \tau_1\} + v) - vr_0} = \frac{r_1}{r_0} \qquad (\text{B.2})$$

Summing up, (B.1) and (B.2) give item (a).

For item (b), if $\mathsf{T} = \mathsf{A} : \mathsf{dep}(v_0 : \tau_0, v_1 : \tau_1)$, then by the [DEP] rule it must be $r_i' = r_i + v_i$ for $i \in \{0, 1\}$, and $S_{\Gamma'}\{\tau_0, \tau_1\} = S_{\Gamma}\{\tau_0, \tau_1\} + \frac{v_i}{r_i} S_{\Gamma}\{\tau_0, \tau_1\}$. Therefore:

$$RX_{\Gamma'}^i(\tau_0, \tau_1) = \frac{r_i + v_i}{S_{\Gamma'}\{\tau_0, \tau_1\}} = \frac{r_i + v_i}{S_{\Gamma}\{\tau_0, \tau_1\}(1 + \frac{v_i}{r_i})} = \frac{(r_i + v_i)r_i}{S_{\Gamma}\{\tau_0, \tau_1\}(r_i + v_i)} = RX_{\Gamma}^i(\tau_0, \tau_1)$$

Otherwise, if $\mathsf{T} = \mathsf{A} : \mathsf{rdm}(v : \{\tau_0, \tau_1\})$, then by rule [RDM] it must be, for $i \in \{0, 1\}$:

$$r_i' \;=\; r_i - v_i \;=\; r_i - v RX_{\Gamma}^i(\tau_0, \tau_1) \;=\; r_i - v \frac{r_i}{S_{\Gamma}\{\tau_0, \tau_1\}}$$

Therefore:

$$RX_{\Gamma'}^i(\tau_0, \tau_1) = \frac{r_i - v_i}{S_{\Gamma}\{\tau_0, \tau_1\} - v} = \frac{r_i - v \frac{r_i}{S_{\Gamma}\{\tau_0, \tau_1\}}}{S_{\Gamma}\{\tau_0, \tau_1\} - v} = \frac{r_i S_{\Gamma}\{\tau_0, \tau_1\} - v r_i}{(S_{\Gamma}\{\tau_0, \tau_1\} - v) S_{\Gamma}\{\tau_0, \tau_1\}}$$

$$= \frac{r_i}{S_{\Gamma}\{\tau_0, \tau_1\}} = RX_{\Gamma}^i(\tau_0, \tau_1) \qquad\qquad\qquad \square$$

For item (c), if $\mathsf{T} = \mathsf{A} : \mathsf{dep}(v_0 : \tau_0, v_1 : \tau_1)$, we have that:

$$P_{\Gamma'}\{\tau_0, \tau_1\} = \frac{r_0' \cdot P\tau_0 + r_1' \cdot P\tau_1}{S_{\Gamma'}\{\tau_0, \tau_1\}} \qquad\qquad \text{by Equation (3.1)}$$

$$= \frac{(1 + \frac{v_0}{r_0}) \cdot r_0 \cdot P\tau_0 + (1 + \frac{v_1}{r_1}) \cdot r_1 \cdot P\tau_1}{S_{\Gamma}\{\tau_0, \tau_1\} + \frac{v_i}{r_i} S_{\Gamma}\{\tau_0, \tau_1\}}$$

$$= \frac{(1 + \frac{v_i}{r_i}) \cdot r_0 \cdot P\tau_0 + (1 + \frac{v_i}{r_i}) \cdot r_1 \cdot P\tau_1}{\left(1 + \frac{v_i}{r_i}\right) \cdot S_{\Gamma}\{\tau_0, \tau_1\}} \qquad\qquad \text{since } \frac{v_0}{r_0} = \frac{v_1}{r_1}$$

$$= P_{\Gamma}\{\tau_0, \tau_1\} \qquad\qquad \text{by Equation (3.1)}$$

The proof for the case $\mathsf{T} = \mathsf{A} : \mathsf{rdm}(v : \{\tau_0, \tau_1\})$ is similar.

Proof of Lemma 4.5. Let $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$. We first prove item (a). Depending on the rule used to fire the transition, we have the following cases:

- [DEP0]. Let $\mathsf{T} = \mathsf{B} : \mathsf{dep}(v_0 : \tau_0, v_1 : \tau_1)$. We have that:

$$\Gamma = \mathsf{B}[\sigma] \mid \Gamma_0$$

$$\Gamma' = \mathsf{B}[\sigma - v_0 : \tau_0 - v_1 : \tau_1 + v_0 : \{\tau_0, \tau_1\}] \mid \{v_0 : \tau_0, v_1 : \tau_1\} \mid \Gamma_0$$

  If $\mathsf{B} \neq \mathsf{A}$, then A's net worth is unaffected. Otherwise, if $\mathsf{B} = \mathsf{A}$, then:

$$W_{\mathsf{A}}(\Gamma') = W_{\mathsf{A}}(\Gamma) - v_0 P\tau_0 - v_1 P\tau_1 + v_0 P_{\Gamma}\{\tau_0, \tau_1\}$$

$$= W_{\mathsf{A}}(\Gamma) - v_0 P\tau_0 - v_1 P\tau_1 + v_0 \frac{v_0 P\tau_0 + v_1 P\tau_1}{v_0} \qquad \text{by Equation (3.1)}$$

$$= W_{\mathsf{A}}(\Gamma)$$

- [DEP]. Let $\mathsf{T} = \mathsf{B} : \mathsf{dep}(v_0 : \tau_0, v_1 : \tau_1)$. We have that:

$$\Gamma = \mathsf{B}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Gamma_0$$

$$\Gamma' = \mathsf{B}[\sigma - v_0 : \tau_0 - v_1 : \tau_1 + v : \{\tau_0, \tau_1\}] \mid \{r_0 + v_0 : \tau_0, r_1 + v_1 : \tau_1\} \mid \Gamma_0$$

  where:

$$v = \frac{v_0 \cdot S_{\Gamma}\{\tau_0, \tau_1\}}{r_0}$$

If $B \neq A$, then A's net worth is unaffected (note that the value of minted tokens in A's wallet is preserved by deposits, by Lemma 4.4(c)). Otherwise, if $B = A$, then:

$$
\begin{aligned}
W_A(\Gamma') &= W_A(\Gamma) - v_0 P \tau_0 - v_1 P \tau_1 + v P_\Gamma \{\tau_0, \tau_1\} \\
&= W_A(\Gamma) - v_0 P \tau_0 - v_1 P \tau_1 + v \frac{r_0 P \tau_0 + r_1 P \tau_1}{S_\Gamma \{\tau_0, \tau_1\}} && \text{by Equation (3.1)} \\
&= W_A(\Gamma) - v_0 P \tau_0 - v_1 P \tau_1 + \frac{v_0}{r_0}\Big(r_0 P \tau_0 + r_1 P \tau_1 \Big) \\
&= W_A(\Gamma) - v_1 P \tau_1 + \frac{v_0}{r_0} r_1 P \tau_1 \\
&= W_A(\Gamma) && \text{since } r_1 v_0 = r_0 v_1
\end{aligned}
$$

- [Swap]. This case cannot happen, since we are assuming $type(\mathsf{T}) \neq \mathsf{swap}$.
- [Rdm]. Let $\mathsf{T} = B : \mathsf{rdm}(v : \{\tau_0, \tau_1\})$. We have that:

$$
\Gamma = B[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Gamma_0
$$
$$
\Gamma' = B[\sigma + v_0 : \tau_0 + v_1 : \tau_1 - v : \{\tau_0, \tau_1\}] \mid \{r_0 - v_0 : \tau_0, r_1 - v_1 : \tau_1\} \mid \Gamma_0
$$

where:

$$
v_0 = \frac{v \cdot r_0}{s} \qquad v_1 = \frac{v \cdot r_1}{s} \qquad s = S_\Gamma\{\tau_0, \tau_1\}
$$

If $B \neq A$, then A's net worth is unaffected (note that the value of minted tokens in A's wallet is preserved by redeems, by Lemma 4.4(c)). Otherwise, if $B = A$, then:

$$
\begin{aligned}
W_A(\Gamma') &= W_A(\Gamma) + v_0 P \tau_0 + v_1 P \tau_1 - v P_\Gamma \{\tau_0, \tau_1\} \\
&= W_A(\Gamma) + v_0 P \tau_0 + v_1 P \tau_1 - \frac{v \cdot r_0}{s} P \tau_0 - \frac{v \cdot r_1}{s} P \tau_1 \\
&= W_A(\Gamma) + \frac{v \cdot r_0}{s} P \tau_0 + \frac{v \cdot r_1}{s} P \tau_1 - \frac{v \cdot r_0}{s} P \tau_0 - \frac{v \cdot r_1}{s} P \tau_1 \\
&= W_A(\Gamma)
\end{aligned}
$$

We now prove item (b), i.e. that the *global* net worth is preserved by *any* transactions. First, we recall from section 3 the definition of global net worth. Let:

$$
\Gamma = A_1[\sigma_1] \mid \cdots \mid A_n[\sigma_n] \mid \{r_1 : \tau_1, r_1' : \tau_1'\} \mid \cdots \mid \{r_k : \tau_k, r_k' : \tau_k'\}
$$

Then, the global net worth of Γ is:

$$
W(\Gamma) = \sum_{i=1}^{n} W_{A_i}(\Gamma)
$$

We have the following cases:

- [Dep0], [Dep], [Rdm]. These rules affect the token reserves in AMMs, which do not contribute to the global net worth, and the balances of users, which we know to be preserved. Therefore, the global net worth is preserved.
- [Swap]. Let $A : \mathsf{swap}(v, \tau_0, \tau_1)$ be the fired transaction. We have that:

$$
\Gamma = A[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Gamma_0
$$
$$
\Gamma' = A[\sigma - v : \tau_0 + v' : \tau_1] \mid \{r_0 + v : \tau_0, r_1 - v' : \tau_1\} \mid \Gamma_0
$$

The global net worth in Γ' can be computed in terms of the global net worth in Γ, by removing the value of the $v : \tau_0$ paid by A to the AMM, adding the value of the $v_1 : \tau_1$

obtained by A through the swap, and then adding the difference between the value of the minted tokens in Γ' and in Γ, i.e.:

$$S_{\Gamma'}\{\tau_0, \tau_1\} P_{\Gamma'}\{\tau_0, \tau_1\} - S_{\Gamma}\{\tau_0, \tau_1\} P_{\Gamma}\{\tau_0, \tau_1\}$$

By Lemma 4.3, we have that $S_{\Gamma'}\{\tau_0, \tau_1\} = S_{\Gamma}\{\tau_0, \tau_1\}$. Therefore:

$$
\begin{aligned}
W(\Gamma') &= W(\Gamma) - vP\tau_0 + v'P\tau_1 + S_{\Gamma}\{\tau_0, \tau_1\}\big(P_{\Gamma'}\{\tau_0, \tau_1\} - P_{\Gamma}\{\tau_0, \tau_1\}\big) \\
&= W(\Gamma) - vP\tau_0 + v'P\tau_1 \\
&\quad + S_{\Gamma}\{\tau_0, \tau_1\} \cdot \Big(\frac{r_0 P\tau_0 + r_1 P\tau_1 + vP\tau_0 - v'P\tau_1}{S_{\Gamma}\{\tau_0, \tau_1\}} - \frac{r_0 P\tau_0 + r_1 P\tau_1}{S_{\Gamma}\{\tau_0, \tau_1\}}\Big) \\
&= W(\Gamma) \quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\square
\end{aligned}
$$

Proof of Lemma 4.6. Direct consequence of Lemma 4.5(a) and of the hypothesis that A does not hold minted tokens in Γ'. $\quad\square$

Proof of Lemma 4.8. Let $\Gamma^0 = \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta^0$ be a reachable state. We define below a procedure to construct a sequence of transitions:

$$\Gamma^0 \xrightarrow{\mathsf{T}_1} \cdots \xrightarrow{\mathsf{T}_n} \Gamma^n \quad\quad \text{where} \quad \Gamma^n = \{r_0^i : \tau_0, r_1^i : \tau_1\} \mid \Delta^n$$

By Lemma 4.2, we have that $r_0^i > 0$, $r_1^i > 0$, and $S_{\Gamma^i}\{\tau_0, \tau_1\} > 0$ for all i. At step i:

(1) Let $x = r_0^i - r_0'$ be the amount of τ_0 that users must redeem from the AMM, and let:

$$v = \frac{x}{r_0^i} S_{\Gamma^i}\{\tau_0, \tau_1\}$$

(2) if there exists some $\mathsf{A}[\sigma] \in \Gamma^i$ such that $\sigma(\{\tau_0, \tau_1\}) \geq v$, then A can fire $\mathsf{A} : \mathsf{rdm}(v : \{\tau_0, \tau_1\})$, obtaining, for some $r_1' \leq r_1$:

$$
\begin{aligned}
\{r_0^i : \tau_0, r_1^i : \tau_1\} \mid \Delta^i \to \Gamma' &= \Big\{r_0^i - v\frac{r_0^i}{S_{\Gamma^i}\{\tau_0, \tau_1\}}, r_1' : \tau_1\Big\} \mid \cdots \\
&= \{r_0' : \tau_0, r_1' : \tau_1\} \mid \cdots
\end{aligned}
$$

(3) otherwise, pick an $\mathsf{A}[\sigma] \in \Gamma^i$ such that $\sigma(\{\tau_0, \tau_1\}) = v' \geq 0$, fire $\mathsf{A} : \mathsf{rdm}(v' : \{\tau_0, \tau_1\})$.

Note that the procedure always terminates: since $S_{\Gamma^i}\{\tau_0, \tau_1\} > 0$ for all i, either step (2) or (3) can be performed; further, the number of performed transactions is bounded by the number of users, which is finite. $\quad\square$

Proof of Lemma 4.9. Assume that $\Gamma \xrightarrow{\mathsf{T}_0} \Gamma_0 \xrightarrow{\mathsf{T}_1} \Gamma_{01}$. We have the following exhaustive cases on the type of the transactions T_0 and T_1:

(1) $\mathsf{T}_0 = \mathsf{A}_0 : \mathsf{dep}(v_0 : \tau_0, v_0' : \tau_0')$.
   (a) $\mathsf{T}_1 = \mathsf{A}_1 : \mathsf{dep}(v_1 : \tau_1, v_1' : \tau_1)$. Both transactions are dep, so we are in case (a) of the statement. If $\{\tau_0, \tau_0'\} \neq \{\tau_1, \tau_1'\}$, then the thesis is straightforward, since $\mathsf{T}_0, \mathsf{T}_1$ operate on different AMMs. Otherwise, let:

$$a_0 = 1 + \frac{v_0}{r_0} \quad m_0 = \frac{v_0}{r_0} S_{\Gamma}\{\tau_0, \tau_1\} \quad a_{01} = 1 + \frac{v_1}{a_0 r_0} \quad m_{01} = \frac{v_1}{a_0 r_0} S_{\Gamma_0}\{\tau_0, \tau_1\}$$

$$a_1 = 1 + \frac{v_1}{r_0} \quad m_1 = \frac{v_1}{r_0} S_{\Gamma}\{\tau_0, \tau_1\} \quad a_{10} = 1 + \frac{v_0}{a_1 r_0} \quad m_{10} = \frac{v_0}{a_1 r_0} S_{\Gamma_1}\{\tau_0, \tau_1\}$$

We have that:

$$\mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$$

$$\xrightarrow{\mathsf{T}_0} \mathsf{A}[\sigma - v_0 : \tau_0 - v_0' : \tau_1 + m_0 : \{\tau_0, \tau_1\}] \mid \{a_0 r_0 : \tau_0, a_0 r_1 : \tau_1\} \mid \Delta$$

$$\xrightarrow{\mathsf{T}_1} \mathsf{A}[\sigma - (v_0 + v_1) : \tau_0 - (v_0' + v_1') : \tau_1 + (m_0 + m_{01}) : \{\tau_0, \tau_1\}] \mid$$
$$\{a_{01} a_0 r_0 : \tau_0, a_{01} a_0 r_1 : \tau_1\} \mid \Delta$$

Inverting the two transactions, we obtain:

$$\mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$$

$$\xrightarrow{\mathsf{T}_1} \mathsf{A}[\sigma - v_1 : \tau_0 - v_1' : \tau_1 + m_1 : \{\tau_0, \tau_1\}] \mid \{a_1 r_0 : \tau_0, a_1 r_1 : \tau_1\} \mid \Delta$$

$$\xrightarrow{\mathsf{T}_0} \mathsf{A}[\sigma_1 - (v_0 + v_1) : \tau_0 - (v_0' + v_1') : \tau_1 + (m_1 + m_{10}) : \{\tau_0, \tau_1\}] \mid$$
$$\{a_{10} a_1 r_0 : \tau_0, a_{10} a_1 r_1 : \tau_1\} \mid \Delta$$

We have that $a_{01} a_0 = a_{10} a_1$, since:

$$a_{10} a_1 = \left(1 + \tfrac{v_0}{a_1 r_0}\right) a_1 = \frac{a_1 r_0 + v_0}{r_0} = \frac{\left(1 + \tfrac{v_1}{r_0}\right) r_0 + v_0}{r_0} = \frac{r_0 + v_0 + v_1}{r_0}$$

$$a_{01} a_0 = \left(1 + \tfrac{v_1}{a_0 r_0}\right) a_0 = \frac{a_0 r_0 + v_1}{r_0} = \frac{\left(1 + \tfrac{v_0}{r_0}\right) r_0 + v_1}{r_0} = \frac{r_0 + v_0 + v_1}{r_0}$$

Furthermore, we have that $m_0 + m_{01} = m_1 + m_{10}$, since:

$$m_{10} + m_1 = \frac{v_0 v_1 + a_1 r_0 v_1 + r_0 v_0}{a_1 r_0^2} S_\Gamma\{\tau_0, \tau_1\}$$

$$= \frac{v_0 v_1 + v_1(r_0 + v_1) + r_0 v_0}{(r_0 + v_1) r_0} S_\Gamma\{\tau_0, \tau_1\}$$

$$= \frac{(v_0 + v_1)(r_0 + v_1)}{(r_0 + v_1) r_0} S_\Gamma\{\tau_0, \tau_1\} \;=\; \frac{v_0 + v_1}{r_0} S_\Gamma\{\tau_0, \tau_1\}$$

$$m_{01} + m_0 = \frac{v_0 v_1 + a_0 r_0 v_0 + r_0 v_1}{a_0 r_0^2} S_\Gamma\{\tau_0, \tau_1\}$$

$$= \frac{v_0 v_1 + v_0(r_0 + v_0) + r_0 v_1}{(r_0 + v_0) r_0} S_\Gamma\{\tau_0, \tau_1\}$$

$$= \frac{(v_0 + v_1)(r_0 + v_0)}{(r_0 + v_0) r_0} S_\Gamma\{\tau_0, \tau_1\} \;=\; \frac{v_0 + v_1}{r_0} S_\Gamma\{\tau_0, \tau_1\}$$

Summing up, we have shown that $\Gamma_{01} = \Gamma_{10}$.

(b) $\mathsf{T}_1 = \mathsf{A}_1 : \mathsf{swap}(v_1, \tau_1, \tau_1')$. Then, we are in case (a) of the statement, with $tok(\mathsf{T}_0)$ disjoint from $tok(\mathsf{T}_1)$. The thesis is straightforward by analysis of the rules.

(c) $\mathsf{T}_1 = \mathsf{A}_1 : \mathsf{rdm}(v_1 : \{\tau_1, \tau_1'\})$. There are two subcases. If we are in case (a), then $\mathsf{T}_0, \mathsf{T}_1$ operate on different AMMs, and so the thesis is straightforward. Otherwise, if we are in case (b) of the statement, by hypothesis we know that $\mathsf{T}_1 \mathsf{T}_0$ is enabled in Γ, leading to a state Γ_{10}. If $\{\tau_0, \tau_0'\} \neq \{\tau_1, \tau_1'\}$, then the thesis is straightforward. Otherwise, the proof is done by computing the states Γ_{01} and Γ_{10} and showing they are equal, similarly to what we have done in case (1a).

(2) $\mathsf{T}_0 = \mathsf{A}_0 : \mathsf{rdm}(v_0 : \{\tau_0, \tau_0'\})$.

(a) $\mathsf{T}_1 = \mathsf{A}_1 : \mathsf{dep}(v_1 : \tau_1, v_1' : \tau_1')$. Symmetric to case (1c).

(b) $\mathsf{T}_1 = \mathsf{A}_1 : \mathsf{swap}(v_1, \tau_1, \tau_1')$. Then, we are in case (a) of the statement, where $\{\tau_1, \tau_1'\}$ and $\{\tau_0, \tau_0'\}$ are disjoint. Then, the thesis is straightforward.

(c) $\mathsf{T}_1 = \mathsf{A}_1 : \mathsf{rdm}(v_1 : \{\tau_1, \tau_1'\})$. Then, we are in case (a) of the statement. If $tok(\mathsf{T}_0)$ is disjoint from $tok(\mathsf{T}_1)$, then the thesis is straightforward. Otherwise, note that the tokens paid by the AMM in response of T_0 and T_1 only depend on the ratio between the amounts of τ_0 and τ_0' initially held by the AMM, which are constrained to preserve the ratio.

(3) $\mathsf{T}_0 = \mathsf{A}_0 : \mathsf{swap}(v_0, \tau_0, \tau_0')$. The only case not covered by the previous items is when $\mathsf{T}_1 = \mathsf{A}_1 : \mathsf{swap}(v_1, \tau_1, \tau_1')$. Then, we are in case (a) of the statement, where $\{\tau_1, \tau_1'\}$ and $\{\tau_0, \tau_0'\}$ are disjoint. The thesis is straightforward. $\qquad\square$

Proof of Theorem 4.10. For item 1, there are two cases, depending on whether T_0 is fired through rule [DEP0] or [DEP]. If T_0 is fired through rule [DEP], let $\Gamma = \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$. We have that:

$$\Gamma_0 = \{r_0 + v_0 : \tau_0, r_1 + v_1 : \tau_1\} \mid \Delta_0 \qquad\qquad r_1 v_0 = r_0 v_1 \qquad\qquad (\text{B.3})$$

$$\Gamma_1 = \{(r_0 + v_0) + v_0' : \tau_0, (r_1 + v_1) + v_1' : \tau_1\} \mid \Delta_1 \qquad (r_1 + v_1)v_0' = (r_0 + v_0)v_1' \qquad (\text{B.4})$$

We must just check that the premises for firing $\mathsf{A} : \mathsf{dep}(v_0 + v_0' : \tau_0, v_1' + v_1' : \tau_1)$ are satisfied:

$$
\begin{aligned}
r_1(v_0 + v_0') &= r_1 v_0 + r_1 v_0' \\
&= r_0 v_1 + r_1 v_0' && \text{by (B.3)} \\
&= r_0 v_1 + r_1 \left(\frac{r_0 + v_0}{r_1 + v_1}\right) v_1' && \text{by (B.4)} \\
&= r_0 v_1 + r_1 \frac{r_0}{r_1} v_1' && \text{by (B.1)} \\
&= r_0(v_1 + v_1')
\end{aligned}
$$

The case where T_0 is fired through rule [DEP0] is similar:

$$
\begin{aligned}
\Gamma_0 &= \{v_0 : \tau_0, v_1 : \tau_1\} \mid \Delta_0 \\
\Gamma_1 &= \{v_0 + v_0' : \tau_0, v_1 + v_1' : \tau_1\} \mid \Delta_1 \qquad\qquad v_1 v_0' = v_0 v_1'
\end{aligned}
$$

The premises of [DEP0] when firing $\mathsf{A} : \mathsf{dep}(v_0 + v_0' : \tau_0, v_1' + v_1' : \tau_1)$ are trivially satisfied, hence the thesis follows.

For item 2, let $\Gamma = \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$ let $\tau = \{\tau_0, \tau_1\}$, and let $s = S_\Gamma \tau$. By rule [RDM], we have that:

$$\Gamma_0 = \{r_0 - v_0 : \tau_0, r_1 - v_1 : \tau_1\} \mid \Delta_0 \qquad\qquad v_i = v \cdot \frac{r_i}{s} \qquad\qquad (\text{B.5})$$

$$\Gamma_1 = \{(r_0 - v_0) - v_0' : \tau_0, (r_1 - v_1) - v_1' : \tau_1\} \mid \Delta_1 \qquad\qquad v_i' = v' \cdot \frac{r_i - v_i}{s - v} \qquad\qquad (\text{B.6})$$

Therefore, for $i \in \{0, 1\}$, we have that:

$$r_i - v_i - v_i' = r_i - v \cdot \frac{r_i}{s} - v' \cdot \frac{r_i - v \cdot \frac{r_i}{s}}{s - v} \qquad \text{by (B.5), (B.6)}$$

$$= r_i - v \cdot \frac{r_i(s - v)}{s(s - v)} - v' \cdot \frac{sr_i - v \cdot r_i}{s(s - v)}$$

$$= r_i - \frac{vr_i(s - v) + v'(sr_i - vr_i)}{s(s - v)}$$

$$= r_i - \frac{vr_i(s - v) + v'r_i(s - v)}{s(s - v)}$$

$$= r_i - (v + v') \cdot \frac{r_i}{s}$$

from which the thesis follows. $\qquad\square$

Proof of Theorem 4.11. By cases on the rule used to deduce $\Gamma \xrightarrow{\mathsf{T}} \Gamma'$. The premise that $S_\Gamma \tau = 0$ implies $S_{\Gamma'} \tau = 0$ excludes the case [Dep0], so we have two cases:

- [Dep]. We have that $\mathsf{A} : \mathsf{dep}(v_0 : \tau_0, v_1 : \tau_1)$, $\Gamma = \mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$, and:

$$\Gamma' = \mathsf{A}[\sigma - v_0 : \tau_0 - v_1 : \tau_1 + v : \{\tau_0, \tau_1\}] \mid \{r_0 + v_0 : \tau_0, r_1 + v_1 : \tau_1\} \mid \Delta$$
$$= \mathsf{A}[\sigma'] \mid \{r_0' : \tau_0, r_1' : \tau_1\} \mid \Delta$$

where $v = \frac{v_i}{r_i} \cdot s$, with $s = S_\Gamma\{\tau_0, \tau_1\}$. Let $\mathsf{T}^{-1} = \mathsf{A} : \mathsf{rdm}(v : \{\tau_0, \tau_1\})$. We have that:

$$\Gamma' \xrightarrow{\mathsf{T}^{-1}} \mathsf{A}[\sigma' + v_0' : \tau_0 + v_1' : \tau_1 - v : \{\tau_0, \tau_1\}] \mid \{r_0' - v_0' : \tau_0, r_1' - v_1' : \tau_1\} \mid \Delta \; = \; \Gamma''$$

where, for $i \in \{0, 1\}$ and $s' = S_\Gamma\{\tau_0, \tau_1\} = s + v$:

$$v_i' = v \cdot \frac{r_i'}{s'} = v \cdot \frac{r_i + v_i}{s + v} = \left(\frac{v_i}{r_i} \cdot s\right) \cdot \frac{r_i + v_i}{s + \left(\frac{v_i}{r_i} \cdot s\right)} = \frac{v_i s(r_i + v_i)}{r_i s + v_i s} = v_i$$

Since $v_i = v_i'$ for $i \in \{0, 1\}$, we conclude that $\Gamma'' = \Gamma$.

- [Rdm]. We have that $\mathsf{T} = \mathsf{A} : \mathsf{rdm}(v : \{\tau_0, \tau_1\})$, $\Gamma = \mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$, and:

$$\Gamma' = \mathsf{A}[\sigma + v_0 : \tau_0 + v_1 : \tau_1 - v : \{\tau_0, \tau_1\}] \mid \{r_0 - v_0 : \tau_0, r_1 - v_1 : \tau_1\} \mid \Delta$$
$$= \mathsf{A}[\sigma'] \mid \{r_0' : \tau_0, r_1' : \tau_1\} \mid \Delta$$

where $v_i = v \cdot \frac{r_i}{s}$, for $i \in \{0, 1\}$ and $s = S_\Gamma\{\tau_0, \tau_1\}$. Let $\mathsf{T}^{-1} = \mathsf{A} : \mathsf{dep}(v_0 : \tau_0, v_1 : \tau_1)$. We have that:

$$\Gamma' \xrightarrow{\mathsf{T}^{-1}} \mathsf{A}[\sigma' - v_0 : \tau_0 - v_1 : \tau_1 + v' : \{\tau_0, \tau_1\}] \mid \{r_0' + v_0 : \tau_0, r_1' + v_1 : \tau_1\} \mid \Delta \; = \; \Gamma''$$

where $v' = \frac{v_i}{r_i'} \cdot s'$, with $s' = S_{\Gamma'}\{\tau_0, \tau_1\} = s - v$. We have that:

$$v' = \frac{v_i}{r_i'} \cdot s' = \frac{v \cdot \frac{r_i}{s}}{r_i - v \cdot \frac{r_i}{s}} \cdot (s - v) = \frac{v \cdot r_i}{sr_i - vr_i} \cdot (s - v) = \frac{v}{s - v} \cdot (s - v) = v$$

Since $v' = v$, we conclude that $\Gamma'' = \Gamma$. $\qquad\square$

APPENDIX C. PROOFS FOR SECTION 5

Proof of Lemma 5.2. The condition $S_\Gamma\{\tau_0, \tau_1\} > 0$ ensures that Γ contains an AMM for the pair τ_0, τ_1. The premise $\sigma(\tau_0) \geq x$ ensures that A has enough units of the input token τ_0. Output-boundedness implies the premise $x \cdot SX(x, r_0, r_1) < r_1$ of [SWAP]. $\qquad \square$

Proof of Lemma 5.4. Straightforward by Definition 5.3 and Lemma 3.2. $\qquad \square$

Proof of Theorem 5.6. Let $\Gamma = \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$. We have that:

$$\Gamma_0 = \{r_0 + x_0 : \tau_0, r_1 - y_0 : \tau_1\} \mid \Delta_0 \qquad\qquad y_0 = x_0 \cdot SX(x_0, r_0, r_1)$$
$$\Gamma_1 = \{r_0 + x_0 + x_1 : \tau_0, r_1 - y_0 - y_1 : \tau_1\} \mid \Delta_1 \qquad y_1 = x_1 \cdot SX(x_1, r_0 + x_0, r_1 - y_0)$$

Since SX is additive, we have that:

$$SX(x_0 + x_1, r_0, r_1) = \frac{y_0 + y_1}{x_0 + x_1}$$

Therefore, rule [SWAP] gives the thesis:

$$\Gamma \xrightarrow{\mathsf{A}:\mathsf{swap}(x_0 + x_1, \tau_0, \tau_1)} \{r_0 + x_0 + x_1 : \tau_0, r_1 - (y_0 + y_1) : \tau_1\} \mid \Delta_1 \qquad\qquad \square$$

Proof of Lemma 5.7. Since SX is output-bounded, then by Lemma 5.2, $\mathsf{T}(x_0)$ and $\mathsf{T}(x_0 + x_1)$ are enabled in Γ, and $\mathsf{T}(x_1)$ is enabled in Γ'. Let:

$$\alpha = SX(x_0, r_0, r_1) \qquad \beta = SX(x_1, r_0 + x_0, r_1 - \alpha x_0)$$

By additivity of SX (Definition 5.5), we have that:

$$\gamma = SX(x_0 + x_1, r_0, r_1) = \frac{\alpha x_0 + \beta x_1}{x_0 + x_1} \tag{C.1}$$

Therefore:

$$
\begin{aligned}
G_{\mathsf{A}}(\Gamma, &\mathsf{T}(x_0 + x_1)) - G_{\mathsf{A}}(\Gamma, \mathsf{T}(x_0)) \\
&= \gamma(x_0 + x_1)P_{\tau_1} - (x_0 + x_1)P_{\tau_0} - \alpha x_0 P_{\tau_1} + x_0 P_{\tau_0} \qquad \text{(Lemma 3.2)} \\
&= \big((\gamma(x_0 + x_1) - \alpha x_0)P_{\tau_1} - x_1 P_{\tau_0} \\
&= \big(\alpha x_0 + \beta x_1 - \alpha x_0)\big)P_{\tau_1} - x_1 P_{\tau_0} \qquad\qquad \text{(Equation C.1)} \\
&= \beta x_1 P_{\tau_1} - x_1 P_{\tau_0} \\
&= G_{\mathsf{A}}(\Gamma', \mathsf{T}(x_1)) \qquad\qquad\qquad\qquad\qquad \text{(Lemma 3.2)} \qquad \square
\end{aligned}
$$

Proof of Theorem 5.9. Let $\Gamma = \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta$, and let $y = x \cdot SX(x, r_0, r_1)$. By the
[Swap] rule, there exists Δ' such that:

$$\Gamma' = \{r_0 + x : \tau_0, r_1 - y : \tau_1\} \mid \Delta'$$

Let $\mathsf{T}^{-1} = \mathsf{A} : \mathsf{swap}(y, \tau_1, \tau_0)$, and let $x' = y \cdot SX(y, r_1 - y, r_0 + x)$. For some Δ'', we have:

$$\Gamma' \xrightarrow{\ \mathsf{T}^{-1}\ } \{r_0 + x - x' : \tau_0, r_1 - y + y : \tau_1\} \mid \Delta''$$

By reversibility of the swap rate, we have that:

$$\frac{y}{x} = SX(x, r_0, r_1) \implies SX(y, r_1 - y, r_0 + x) = \frac{x}{y}$$

from which we obtain that:

$$x' = y \cdot SX(y, r_1 - y, r_0 + x) = y \cdot \frac{x}{y} = x$$

from which we obtain the thesis. $\qquad\square$

Proof of Lemma 5.10. Straightforward from the definition of gain and from Theorem 5.9.
$\qquad\square$

Proof of Lemma 5.12. Let $\{r_0 : \tau_0, r_1 : \tau_1\} \in \Gamma$, $\{r_0' : \tau_0, r_1' : \tau_1\} \in \Gamma'$, and let $a = r_0'/r_0$.
We have that:

$$
\begin{aligned}
X_\Gamma(\tau_0, \tau_1) &= \lim_{x \to 0} SX(x, r_0, r_1) &&\text{by Equation (3.3)}\\
&= \lim_{x \to 0} SX(ax, ar_0, ar_1) &&\text{since } SX \text{ is homogeneous}\\
&= \lim_{x \to 0} SX(ax, r_0', r_1') &&\text{by Lemma 4.4(a)}\\
&= X_{\Gamma'}(\tau_0, \tau_1) &&\text{by Equation (3.3)} \qquad\square
\end{aligned}
$$

Proof of Lemma 5.13. For item (a), let $\mathsf{T} = \mathsf{A} : \mathsf{dep}(v_0 : \tau_0, v_1 : \tau_1)$. By rule [Dep],
$r_i' = r_i + v_i$ for $i \in \{0, 1\}$, with $r_0 v_1 = r_1 v_0$. By Lemma 4.4(a), $r_0 + v_0/r_1 + v_1 = r_0/r_1$. Then:

$$r_0 + v_0 = \frac{r_1 + v_1}{r_1} r_0 = a\, r_0 \qquad r_1 + v_1 = \frac{r_1 + v_1}{r_1} r_1 = a\, r_1 \qquad\qquad \text{where } a = \tfrac{r_1 + v_1}{r_1}$$

Therefore:

$$
\begin{aligned}
SX(x, r_0', r_1') &= SX(x, ar_0, ar_1)\\
&= SX(\tfrac{x}{a}, r_0, r_1) &&\text{(homogeneity)}\\
&> SX(x, r_0, r_1) &&\text{(strict monotonicity, } a > 1 \implies \tfrac{x}{a} < x)
\end{aligned}
$$

The thesis $\Delta X_\Gamma(x, \tau_0, \tau_1) > \Delta X_{\Gamma'}(x, \tau_0, \tau_1)$ follows from this inequality and Lemma 5.12.
For item (b), let $\mathsf{T} = \mathsf{A} : \mathsf{rdm}(v : \{\tau_0, \tau_1\})$. By rule [Rdm], for $i \in \{0, 1\}$:

$$r_i' = r_i - v_i = r_i - v\frac{r_i}{S_\Gamma\{\tau_0, \tau_1\}} = a\, r_i \qquad\qquad \text{where } a = 1 - \frac{v}{S_\Gamma\{\tau_0, \tau_1\}}$$

Therefore:

$$
\begin{aligned}
SX\left(x, r_0', r_1'\right) &= SX\left(x, ar_0, ar_1\right) \\
&= SX\left(\tfrac{x}{a}, r_0, r_1\right) && \text{(homogeneity)} \\
&< SX\left(x, r_0, r_1\right) && \text{(strict monotonicity, } a < 1 \implies \tfrac{x}{a} < x)
\end{aligned}
$$

The thesis $\Delta X_\Gamma(x, \tau_0, \tau_1) < \Delta X_{\Gamma'}(x, \tau_0, \tau_1)$ follows from this inequality and Lemma 5.12. $\quad\square$

Proof of Theorem 5.15. For output-boundedness, let $x > 0$ and $r_0, r_1 > 0$. We have that:

$$
SX\left(x, r_0, r_1\right) = \frac{r_1}{r_0 + x} < \frac{r_1}{x}
$$

For monotonicity, Let $x' \le x$, $r_0' \le r_0$ and $r_1 \le r_1'$. We have that:

$$
SX\left(x', r_0', r_1'\right) \;=\; \frac{r_1'}{r_0' + x'} \;\ge\; \frac{r_1}{r_0 + x} \;=\; SX\left(x, r_0, r_1\right)
$$

The proof for strict monotonicity is similar.
For additivity, by Definition 2.1 we have that:

$$
\begin{aligned}
\alpha &= SX\left(x, r_0, r_1\right) = \frac{r_1}{r_0 + x} \\
\beta &= SX\left(y, r_0 + x, r_1 - \alpha x\right) = \frac{r_1 - \alpha x}{r_0 + x + y} = \frac{r_0 r_1}{(r_0 + x)(r_0 + x + y)}
\end{aligned}
$$

Therefore:

$$
\begin{aligned}
\frac{\alpha x + \beta y}{x + y} &= \frac{1}{x + y}\left(\frac{r_1 x}{r_0 + x} + \frac{r_0 r_1 y}{(r_0 + x)(r_0 + x + y)}\right) \\
&= \frac{1}{x + y}\frac{r_0 r_1 x + r_1 x^2 + r_1 xy + r_0 r_1 y}{(r_0 + x)(r_0 + x + y)} \\
&= \frac{r_1(r_0 + x)(x + y)}{(x + y)(r_0 + x)(r_0 + x + y)} \\
&= \frac{r_1}{r_0 + x + y} \\
&= SX\left(x + y, r_0, r_1\right)
\end{aligned}
$$

For reversibility, let $\alpha = SX\left(x, r_0, r_1\right)$. By Definition 2.1, we have that:

$$
SX\left(\alpha x, r_1 - \alpha x, r_0 + x\right) \;=\; \frac{r_0 + x}{(r_1 - \alpha x) + \alpha x} \;=\; \frac{r_0 + x}{r_1} \;=\; \left(\frac{r_1}{r_0 + x}\right)^{-1} = \frac{1}{\alpha}
$$

For homogeneity, we have that:

$$
SX\left(ax, ar_0, ar_1\right) \;=\; \frac{ar_1}{ar_0 + ax} \;=\; \frac{r_1}{r_0 + x} \;=\; SX\left(x, r_0, r_1\right)
$$

The computations of the internal exchange rate and of the slippage are straightforward. $\quad\square$

Proof of Theorem 5.16. Output-boundedness, monotonicity and homogeneity are straightforward. For additivity, by Definition 5.16 we have that:

$$\alpha = SX(x, r_0, r_1) = \frac{r_1}{x}\left(1 - \left(\frac{r_0}{r_0 + x}\right)^{\frac{w_0}{w_1}}\right)$$

$$\beta = SX(y, r_0 + x, r_1 - \alpha x) = \frac{r_1 - \alpha x}{y}\left(1 - \left(\frac{r_0 + x}{r_0 + x + y}\right)^{\frac{w_0}{w_1}}\right)$$

Therefore:

$$\frac{\alpha x + \beta y}{x + y} = \frac{1}{x + y}\left(\alpha x + (r_1 - \alpha x)\left(1 - \left(\frac{r_0 + x}{r_0 + x + y}\right)^{\frac{w_0}{w_1}}\right)\right)$$

$$= \frac{1}{x + y}\left(r_1 - r_1\left(\frac{r_0 + x}{r_0 + x + y}\right)^{\frac{w_0}{w_1}} + r_1\left(1 - \left(\frac{r_0}{r_0 + x}\right)^{\frac{w_0}{w_1}}\right)\left(\frac{r_0 + x}{r_0 + x + y}\right)^{\frac{w_0}{w_1}}\right)$$

$$= \frac{1}{x + y}\left(r_1 - r_1\left(\frac{r_0}{r_0 + x}\right)^{\frac{w_0}{w_1}}\left(\frac{r_0 + x}{r_0 + x + y}\right)^{\frac{w_0}{w_1}}\right)$$

$$= \frac{r_1}{x + y}\left(1 - \left(\frac{r_0}{r_0 + x + y}\right)^{\frac{w_0}{w_1}}\right)$$

$$= SX(x + y, r_0, r_1) \qquad\qquad \square$$

Appendix D. Proofs for Section 6

Proof of Lemma 6.1. Assume that $SX(x, r_0, r_1) \geq X(\tau_0, \tau_1)$. Let $\alpha(z) = SX(z, r_1, r_0)$. We have that:

$$SX(y, r_1, r_0) < \lim_{z \to 0} SX(z, r_1, r_0) \qquad\qquad \text{(strict monotonicity)}$$

$$= \lim_{z \to 0} \frac{1}{SX(\alpha(z) \cdot z, r_0 - \alpha(z) \cdot z, r_1 + z)} \qquad \text{(reversibility)}$$

$$< \frac{1}{SX(x, r_0, r_1)} \qquad\qquad \text{(strict monotonicity)}$$

$$\leq \frac{1}{X(\tau_0, \tau_1)} \qquad\qquad \text{(hypothesis)}$$

$$= X(\tau_1, \tau_0) \qquad\qquad \text{(def. of } X) \qquad \square$$

where in the second application of strict monotonicity, we have exploited the (asymptotic) inequalities $\alpha(z) \cdot z < x$ (where $\lim_{z \to 0} \alpha(z) \cdot z = 0$ follows from the existence of the internal exchange rate), $r_0 - \alpha(z) \cdot z < r_0$, and $r_1 + z > r_1$.

Proof of Lemma 6.2. Let $y > 0$. Assume that $G_A(\Gamma, T_d(x)) > 0$. Then, $T_d(x)$ is enabled in Γ, and so by Lemma 3.3, we have that $SX(x, r_d, r_{1-d}) > X(\tau_d, \tau_{1-d})$. Then, by Lemma 6.1 it follows that $SX(y, r_{1-d}, r_d) < X(\tau_{1-d}, \tau_d)$. Since $\sigma \tau_{1-d} \geq y$ and SX is output-bounded, then Lemma 5.2 implies that $T_{1-d}(y)$ is enabled in Γ. By using again Lemma 3.3, concluding that $G_A(\Gamma, T_{1-d}(y)) < 0$. $\qquad \square$

Proof of Theorem 6.3. Let x_0 and Γ' be as in the hypotheses, i.e.:

$$\Gamma \xrightarrow{\mathsf{T}(x_0)} \Gamma' = \mathsf{A}[\sigma'] \mid \{r_0 + x_0 : \tau_0, r_1 - \alpha x_0 : \tau_1\} \mid \Delta \qquad \text{where} \quad \begin{array}{l} \alpha = SX(x_0, r_0, r_1) \\ X_{\Gamma'}(\tau_0, \tau_1) = X(\tau_0, \tau_1) \end{array}$$

We have two cases, depending on whether $x > x_0$ or $x < x_0$.

- If $x > x_0$, let $x_1 > 0$ be such that $x = x_0 + x_1$. Since SX is output-bounded and additive, then by Lemma 5.7:

$$G_{\mathsf{A}}(\Gamma, \mathsf{T}(x)) \;=\; G_{\mathsf{A}}(\Gamma, \mathsf{T}(x_0)) + G_{\mathsf{A}}(\Gamma', \mathsf{T}(x_1)) \tag{D.1}$$

  We have that:

$$\begin{aligned} SX(x_1, r_0 + x_0, r_1 - \alpha x_0) &< \lim_{z \to 0} SX(z, r_0 + x_0, r_1 - \alpha x_0) && \text{(strict monotonicity)} \\ &= X_{\Gamma'}(\tau_0, \tau_1) && \text{def. } X_{\Gamma'} \\ &= X(\tau_0, \tau_1) && \text{(hypothesis)} \end{aligned}$$

  Then, by Lemma 3.3 we obtain $G_{\mathsf{A}}(\Gamma', \mathsf{T}(x_1)) < 0$. By Equation (D.1), we conclude that $G_{\mathsf{A}}(\Gamma, \mathsf{T}(x)) < G_{\mathsf{A}}(\Gamma, \mathsf{T}(x_0))$.

- If $x < x_0$, let $x_1 > 0$ be such that $x_0 = x + x_1$. Since SX is output-bounded, then by Lemma 5.2, $\mathsf{T}(x_0)$ and $\mathsf{T}(x)$ are enabled in Γ, and $\mathsf{T}(x_1)$ is enabled in the state Γ_1 reached after performing $\mathsf{T}(x_1)$, i.e.:

$$\Gamma \xrightarrow{\mathsf{T}(x)} \Gamma_1 \xrightarrow{\mathsf{T}(x_1)} \Gamma'$$

Since SX is output-bounded and additive, then by Lemma 5.7:

$$G_{\mathsf{A}}(\Gamma, \mathsf{T}(x_0)) \;=\; G_{\mathsf{A}}(\Gamma, \mathsf{T}(x)) + G_{\mathsf{A}}(\Gamma_1, \mathsf{T}(x_1))$$

Since SX is reversible, then by Theorem 5.9, $\mathsf{T}(x_1)$ has an inverse, which has the form $\mathsf{T}^{-1}(x_1) = \mathsf{A} : \mathsf{swap}(y_1, \tau_1, \tau_0)$ for some $y_1 > 0$. Then, by Lemma 5.10, $G_{\mathsf{A}}(\Gamma_1, \mathsf{T}(x_1)) = -G_{\mathsf{A}}(\Gamma', \mathsf{T}^{-1}(y_1))$, therefore:

$$G_{\mathsf{A}}(\Gamma, \mathsf{T}(x_0)) \;=\; G_{\mathsf{A}}(\Gamma, \mathsf{T}(x)) - G_{\mathsf{A}}(\Gamma', \mathsf{T}^{-1}(y_1)) \tag{D.2}$$

We have that:

$$\begin{aligned} SX(y_1, r_1 - \alpha x_0, r_0 + x_0) &< \lim_{z \to 0} SX(z, r_1 - \alpha x_0, r_0 + x_0) && \text{(strict monotonicity)} \\ &= X_{\Gamma'}(\tau_1, \tau_0) && \text{def. } X_{\Gamma'} \\ &= \frac{1}{X_{\Gamma'}(\tau_0, \tau_1)} && \text{(Equation (5.1))} \\ &= \frac{1}{X(\tau_0, \tau_1)} && \text{(hypothesis)} \\ &= X(\tau_1, \tau_0) && \text{(def. } X) \end{aligned}$$

Then, by Lemma 3.3 we obtain $G_{\mathsf{A}}(\Gamma', \mathsf{T}^{-1}(y_1)) < 0$. By Equation (D.2), we conclude that $G_{\mathsf{A}}(\Gamma, \mathsf{T}(x)) < G_{\mathsf{A}}(\Gamma, \mathsf{T}(x_0))$.

For uniqueness, by contradiction assume that there exists $x_1 \neq x_0$ satisfying Equation (6.1). Then, it should be $G_{\mathsf{A}}(\Gamma, \mathsf{T}(x_1)) > G_{\mathsf{A}}(\Gamma, \mathsf{T}(x_0))$ — contradiction. $\qquad \square$

Proof of Lemma 6.4. Let $\Gamma \xrightarrow{\mathsf{T}} \Gamma' = \mathsf{A}[\sigma'] \mid \{r_0 + x_0 : \tau_0, r_1 - x_0 \cdot SX(x_0, r_0, r_1) : \tau_1\}$. We have that:

$$
\begin{aligned}
X_{\Gamma'}(\tau_0, \tau_1) &= \frac{r_1 - x_0 \cdot SX(x_0, r_0, r_1)}{r_0 + x_0} && \text{by Theorem 5.15} \\
&= \frac{r_1 - x_0 \cdot \frac{r_1}{r_0 + x_0}}{r_0 + x_0} && \text{by Definition 2.1} \\
&= \frac{r_0 r_1}{(r_0 + x_0)^2} && \\
&= \frac{r_0 r_1}{\frac{P\tau_1}{P\tau_0} r_0 r_1} && \text{by Equation (6.2)} \\
&= X(\tau_0, \tau_1) && \text{by Equation (3.2)}
\end{aligned}
$$

The thesis follows from Theorem 6.3. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\square$

Proof of Theorem 6.6. Let:

$$
\Gamma = \mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta \xrightarrow{\mathsf{T_{dep}}} \Gamma' = \mathsf{A}[\sigma'] \mid \{r_0' : \tau_0, r_1' : \tau_1\} \mid \Delta'
$$

The hypothesis $wal(\mathsf{T_{swap}}) = \mathsf{A} \neq wal(\mathsf{T_{rdm}})$ means that the user who performs the deposit is *not* A, hence the deposit does not affect the number of minted tokens in A's wallet. Then:

$$
\begin{aligned}
G_\mathsf{A}(&\Gamma, \mathsf{T_{dep}}\mathsf{T_{swap}}) \\
&= G_\mathsf{A}(\Gamma', \mathsf{T_{swap}}) \\
&= x \cdot \left(SX(x, r_0', r_1') P\tau_1 - P\tau_0\right) \cdot \left(1 - \frac{\sigma'\{\tau_0, \tau_1\}}{S_{\Gamma'}\{\tau_0, \tau_1\}}\right) && \text{(Lemma 3.2)} \\
&> x \cdot \left(SX(x, r_0, r_1) P\tau_1 - P\tau_0\right) \cdot \left(1 - \frac{\sigma'\{\tau_0, \tau_1\}}{S_{\Gamma'}\{\tau_0, \tau_1\}}\right) && \text{(Lemma 5.13(a))} \\
&= x \cdot \left(SX(x, r_0, r_1) P\tau_1 - P\tau_0\right) \cdot \left(1 - \frac{\sigma\{\tau_0, \tau_1\}}{S_{\Gamma'}\{\tau_0, \tau_1\}}\right) && (\sigma'\{\tau_0, \tau_1\} = \sigma\{\tau_0, \tau_1\}) \\
&> x \cdot \left(SX(x, r_0, r_1) P\tau_1 - P\tau_0\right) \cdot \left(1 - \frac{\sigma\{\tau_0, \tau_1\}}{S_{\Gamma}\{\tau_0, \tau_1\}}\right) && (S_{\Gamma'}\{\tau_0, \tau_1\} > S_\Gamma\{\tau_0, \tau_1\}) \\
&= G_\mathsf{A}(\Gamma, \mathsf{T_{swap}}) && \square
\end{aligned}
$$

Proof of Theorem 6.8. Let Γ and Γ_d be as in the statement. By rule [Dep], $r_i' = r_i + v_i$ for $i \in \{0, 1\}$. By Lemma 4.4(a), we have that $r_0 + v_0/r_1 + v_1 = r_0/r_1$. Then:

$$
r_0' = r_0 + v_0 = \frac{r_1 + v_1}{r_1} r_0 = a\, r_0 \qquad r_1' = r_1 + v_1 = \frac{r_1 + v_1}{r_1} r_1 = a\, r_1 \qquad \text{where } a = \frac{r_1 + v_1}{r_1}
$$

For item (1), assume that $\lambda = \mathsf{A} : \mathsf{swap}(x, \tau_0, \tau_1)$ is a solution to the arbitrage game in Γ. By Theorem 6.3, it must be:

$$
X_{\Gamma_s}(\tau_0, \tau_1) = X(\tau_0, \tau_1) \qquad \text{where } \Gamma \xrightarrow{\lambda} \Gamma_s \qquad\qquad (D.3)
$$

Let $x' = ax$, let $\mathsf{T}' = \mathsf{A} : \mathsf{swap}(x', \tau_0, \tau_1)$, and let $\Gamma_d \xrightarrow{\mathsf{T}'} \Gamma_{ds}$. We have that:

$$
\begin{aligned}
X_{\Gamma_{ds}}&(\tau_0, \tau_1) \\
&= \lim_{z \to 0} SX\left(z, r_0' + x', r_1' - x' \cdot SX\left(x', r_0', r_1'\right)\right) \\
&= \lim_{z \to 0} SX\left(z, ar_0 + ax, ar_1 - ax \cdot SX\left(ax, ar_0, ar_1\right)\right) \\
&= \lim_{z \to 0} SX\left(z, ar_0 + ax, ar_1 - ax \cdot SX\left(x, r_0, r_1\right)\right) &&\text{(homogeneity)} \\
&= \lim_{z \to 0} SX\left(z, r_0 + x, r_1 - x \cdot SX\left(x, r_0, r_1\right)\right) &&\text{(homogeneity)} \\
&= X_{\Gamma_s}(\tau_0, \tau_1) &&\text{(def. } X_{\Gamma_s}) \\
&= X(\tau_0, \tau_1) &&\text{(Equation (D.3))}
\end{aligned}
$$

Therefore, Theorem 6.3 implies that T' is a solution to the arbitrage game in Γ_d. We compute the gain of T' in Γ_d as follows:

$$
\begin{aligned}
G_\mathsf{A}(\Gamma_d, \mathsf{T}') &= x' \cdot \left(SX\left(x', r_0', r_1'\right) P\tau_1 - P\tau_0\right) \\
&= ax \cdot \left(SX\left(ax, ar_0, ar_1\right) P\tau_1 - P\tau_0\right) \\
&= ax \cdot \left(SX\left(x, r_0, r_1\right) P\tau_1 - P\tau_0\right) &&\text{(homogeneity)} \\
&= a\, G_\mathsf{A}(\Gamma, \mathsf{T})
\end{aligned}
$$

For item (2), assume that ε is a solution to the arbitrage game in Γ. By contradiction, assume that $\lambda_d = \mathsf{A} : \mathsf{swap}(x', \tau_0, \tau_1)$ is a solution in Γ_d. By Theorem 6.3, it must be:

$$
X_{\Gamma_{ds}}(\tau_0, \tau_1) = X(\tau_0, \tau_1) \tag{D.4}
$$

The chain of equations above shows that $X_{\Gamma_{ds}}(\tau_0, \tau_1) = X_{\Gamma_s}(\tau_0, \tau_1)$. By Equation (D.4), this implies that $X_{\Gamma_{ds}}(\tau_0, \tau_1) = X(\tau_0, \tau_1)$. Hence, by Theorem 6.3, ε cannot be a solution to the arbitrage game in Γ — contradiction. $\square$

Proof of Theorem 6.9. Let:

$$
\Gamma = \mathsf{A}[\sigma] \mid \{r_0 : \tau_0, r_1 : \tau_1\} \mid \Delta \xrightarrow{\mathsf{T}_{\mathsf{rdm}}} \Gamma' = \mathsf{A}[\sigma'] \mid \{r_0' : \tau_0, r_1' : \tau_1\} \mid \Delta'
$$

The hypothesis $wal(\mathsf{T}_{\mathsf{swap}}) = \mathsf{A} \neq wal(\mathsf{T}_{\mathsf{rdm}})$ means that the user who performs the redeem is *not* A, hence the redeem does not affect the number of minted tokens in A's wallet. Then:

$$
\begin{aligned}
G_\mathsf{A}&(\Gamma, \mathsf{T}_{\mathsf{rdm}}\mathsf{T}_{\mathsf{swap}}) \\
&= G_\mathsf{A}(\Gamma', \mathsf{T}_{\mathsf{swap}}) \\
&= x \cdot \left(SX\left(x, r_0', r_1'\right) P\tau_1 - P\tau_0\right) \cdot \left(1 - \frac{\sigma'\{\tau_0, \tau_1\}}{S_{\Gamma'}\{\tau_0, \tau_1\}}\right) &&\text{(Lemma 3.2)} \\
&< x \cdot \left(SX\left(x, r_0, r_1\right) P\tau_1 - P\tau_0\right) \cdot \left(1 - \frac{\sigma'\{\tau_0, \tau_1\}}{S_{\Gamma'}\{\tau_0, \tau_1\}}\right) &&\text{(Lemma 5.13(b))} \\
&= x \cdot \left(SX\left(x, r_0, r_1\right) P\tau_1 - P\tau_0\right) \cdot \left(1 - \frac{\sigma\{\tau_0, \tau_1\}}{S_{\Gamma'}\{\tau_0, \tau_1\}}\right) &&(\sigma'\{\tau_0, \tau_1\} = \sigma\{\tau_0, \tau_1\}) \\
&< x \cdot \left(SX\left(x, r_0, r_1\right) P\tau_1 - P\tau_0\right) \cdot \left(1 - \frac{\sigma\{\tau_0, \tau_1\}}{S_{\Gamma}\{\tau_0, \tau_1\}}\right) &&(S_{\Gamma'}\{\tau_0, \tau_1\} < S_{\Gamma}\{\tau_0, \tau_1\}) \\
&= G_\mathsf{A}(\Gamma, \mathsf{T}_{\mathsf{swap}})
\end{aligned}
$$
$\square$

Proof of Theorem 6.10. Let Γ and Γ_d be as in the statement. By rule [RDM], it must be, for $i \in \{0, 1\}$:

$$r'_i \;=\; r_i - v_i \;=\; r_i - vRX^i_\Gamma(\tau_0, \tau_1) \;=\; r_i - v\frac{r_i}{S_\Gamma\{\tau_0, \tau_1\}} \;=\; ar_i \qquad \text{where } a = 1 - \frac{v}{S_\Gamma\{\tau_0, \tau_1\}}$$

The rest of the proof follows exactly that of Theorem 6.8. $\qquad\square$