Reachability in multi-agent transfer systems (Extended Version)

Nathalie Bertrand[1] [0000−0002−9957−5394], Loïc Hélouët[1] [0000−0001−7056−2672], Engel Lefaucheux[2] [0000−0003−0875−300X], and Luca Paparazzo[1] [0009−0007−1134−000X]

[1] University of Rennes, Inria, CNRS, Rennes, France
[2] Université de Lorraine, CNRS, Inria, LORIA, Nancy, France
{nathalie.bertrand,loic.helouet,engel.lefaucheux,luca.paparazzo}@inria.fr

Abstract. This paper introduces collaborative reachability games with energy constraints. In the considered arenas, agents can spend or gain energy during moves, or share it with their peers if their current position allows it. We study several variants of energy reachability games where agents move either synchronously or asynchronously, and with/without constraints on energy transfers among peers. We show that these problems have different complexities ranging from NP to EXPSPACE.

Keywords: Multi-Agent Systems · Quantitative verification · VASS · Collaborative games · Planning

1 Introduction

Cooperation of several agents occurs in a variety of applications, such as robotics, traffic control and aviation to name a few. In contrast to adversarial games, in such cooperative settings, the agents collaborate to achieve a common goal. A typical instantiation of this general framework is the multi-agent path finding problem [11], in which one aims at designing a plan to move multiple agents while avoiding collisions to perform a global task. Beyond Boolean objectives such as coverage of an area, or reachability of a position for each agent, introducing quantities in models for multi-agent systems is crucial to represent energy or financial cost. Quantitative settings where multiple agents interact are for instance useful formalisms to find optimal management strategies to control cyber physical systems (CPS) where objectives are not purely Boolean, but also aim at optimizing some measure. A variety of settings of quantitative multi-player games have been proposed in the literature [4,5,6,12]. Game concepts such as the famous Nash equilibria [20] can then be studied, for instance to efficiently distribute energy in smart grids [5].

In this paper, we introduce a new quantitative multi-agent model, in which agents move on their own local arena and are given a goal, *i.e.*, a particular vertex to reach. Local arenas are equipped with integer weights on edges to represent energy variations. Each agent stores energy and, when moving from a vertex to a consecutive one, gains energy if the weight is positive, or loses energy if the weight is negative. Interestingly, agents may cooperate by sending some or all of their stored energy to other agents. The objective is to design a collaborative plan moving each agent to its target vertex while staying within the energy available in the system, possibly using transfers among peers. We coin this model *multi-agent transfer systems*, or simply *transfer systems*.

Several semantics can be considered for transfer systems: either agents move synchronously or asynchronously. In any case, they can only take an edge if their stored energy is sufficient, as their energy level cannot drop below 0. We consider the natural question of global reachability objectives, where all agents must reach their assigned target simultaneously. Our setting thus shares the objective of multi-agent path finding [11]. There are however several crucial differences: first, in transfer systems, agents move on their respective local arenas rather than on a common space; second, transfer systems are equipped with energy variations and agents must move within energy budget; finally, in transfer systems agents can transfer energy one to another.

Transfer systems form a particularly suited model for modern urban transport networks equipped with regenerative braking systems. In these CPSs, the kinetic energy of a braking vehicle can be converted into electric energy, transferred to the power network and used by other close vehicles. Another possible application is the study of the logistics of complex systems in which resources must be provided at specific locations and times for the success of a mission.

Multi-weighted energy games [10] are close to our transfer systems. In multi-weighted energy games, stored quantities are k-vectors of integers and moves are also labeled by integer vectors of same dimension. Different to our setting, the number of players is fixed to at most two. The objective in these games is to play infinitely while respecting energy bounds on each coordinate: a lower bound or a combination of lower and a weak/strong upper bounds. With a single player, the problem with a lower bound is NP-hard and k-EXPTIME already, and with two players, the complexity is EXPTIME-hard and in k-EXPTIME. These complexity proofs build on results of [3]. One can consider transfer systems with n agents as a reachability question in a multi-weighted game with a single player (representing the coalition of agents) of dimension n, one dimension for each agent that must remain non-negative. For an arbitrary dimension, existence of an infinite run in multiweighted games with lower bounds is EXPSPACE-complete, and becomes PSPACE-complete if integral upper bound are set for each dimension. Notice that this setting has several differences with our questions in transfer systems; one of the main differences is that [10] considers the existence of infinite runs, while the questions addressed in this paper would be encoded as coverability questions. Most importantly, transfer systems are given succinctly by local arenas for each agent, while multi-weighted energy games are monolithic.

As our model deals with transfer of energy, and is close to Petri nets, a natural question is whether reachability in transfer systems is equivalent to a reachability or coverability in transfer Petri nets [8]. Transfer Petri nets extend Petri nets with flow relations that can transfer the whole contents of a place p to another place p' when firing a transition. Our complexity results on transfer systems prove that reachability for transfer systems and coverability/reachability for transfer nets are different questions. Indeed, transfer Petri nets can easily simulate Reset Petri nets a model where reachability is undecidable [1], and coverability is Ackermann-hard [23]. In contrast, our reachablity problems on transfer systems remain decidable in almost all cases, and have at worst complexities in EXPSPACE when decidable. From a modeling perspective, transfers in Petri nets and in transfer systems are quite different: in Petri nets the whole contents of a place is transferred in one step while in our model, an agent can share only a part of its energy.

The semantics of transfer systems can be captured by vector addition systems with states (VASS) [14], or equivalently by Petri nets, and our reachability problems as coverability questions. EXPSPACE-hardness for coverability in VAS was shown by[18], and the matching EXPSPACE upper bound was shown by [21]. A natural question is whether one has to pay the full complexity of VASS to solve our reachability problems in transfer systems. We show in this paper that the answer depends on the chosen characteristics of the model. For instance, reachability for transfer systems with energy transfers always enabled and under asynchronous semantics lies between NP and PSPACE. Also, under asynchronous semantics with arbitrary transfer groups, the complexity lies between PSPACE and EXPSPACE. More surprisingly, if one relaxes synchronicity by allowing agents that lack energy to idle (resulting in the so-called weak synchronous semantics), reachability becomes undecidable.

The rest of the paper is organized as follows. Section 2 presents the model and the notations that will be used throughout the paper. Section 3 details the different possible semantics of the model: asynchronous, strongly synchronous, and weakly synchronous and shows the relations between these semantics. Section 4 studies the complexity of reachability under all semantics when transfer of energy can occur at any time between agents. Section 5 considers reachability for systems with restricted local transfers, that can occur only in some states. Due to space constraints, some proofs are omitted postponed and can be found in appendix.

2 Transfer systems

Transfer systems are multi-agent systems, in which every agent plays on a local weighted graph, and the communication between agents is limited.

Definition 1. *A* local arena $A = (V, E)$ *is a directed weighted graph where* V *is a finite set of vertices,* $E \subseteq V \times \mathbb{Z} \times V$ *describes the edges of the arena.*

Intuitively, the weight on an edge represents the amount of energy an agent gains (if positive) or loses (if negative) while traversing that edge. Communication between agents is limited to energy transfers, and is formalised by transfer groups that specify conditions on the vertices of the agents to enable transfers.

Definition 2. *Let* $n \in \mathbb{N}$*,* $\mathcal{A} = \{A_1, \dots, A_n\}$ *be a set of local arenas with* $A_i = (V_i, E_i)$ *for every* $i \in [\![1, n]\!]$ *(assuming all* V_i *are disjoint sets) and* $\mathcal{T} : \bigcup_{i \in 1..n} V_i \to \mathbb{N}$ *is a partial map defining* transfer groups*.*

$\mathcal{A}$ *and* $\mathcal{T}$ *induce the* transfer system $\mathsf{TS} = \langle \mathcal{A}, \mathcal{T} \rangle$*.*

We will say that vertices v, v' belong to the same transfer group if $\mathcal{T}(v) = \mathcal{T}(v')$, and impose that transfer groups are not singletons, are disjoint sets and contain states from at least two arenas. For convenience, we will often define transfer groups as sets of states $T_1, \dots, T_k$ where $T_i = \{v \mid \mathcal{T}(v) = i\}$. Writing $V = \bigcup_{i \in 1..n} V_i$, the size of a transfer arena is defined as $|\mathsf{TS}| = |V| \cdot (|\mathcal{T}| + 1) + |V|^2. \log(w_{max})$ where w_{max} is the largest absolute value of a weight appearing in an arena, and $|\mathcal{T}|$ is bounded $|V| \cdot \log(|V|)$.

The semantics of a transfer system $\mathsf{TS} = \langle \mathcal{A}, \mathcal{T} \rangle$ is given in terms of a transition system. A *configuration* of TS consists of the current vertex of each agent and their energy level: we write C, C', etc. for a configuration, and $\Gamma = \left(\prod_{i=1}^n V_i\right) \times \mathbb{N}^n$ for the set of all configurations. For a configuration $C = (S, \overrightarrow{e})$, $S \in \prod_{i=1}^n V_i$ is referred to as the *global state* (or simply state) and $\overrightarrow{e}$ as the *energy vector*. When the dimension n is clear from the context, we use $\overrightarrow{0}$ to denote the null energy vector $(0, \dots, 0) \in \mathbb{N}^n$.

Transitions between configurations are induced by moves of the agents on their local arenas, or energy transfers between agents when permitted by the transfer groups. An agent A_i cannot move along an edge $q_i \xrightarrow{-w} q'_i$ with negative weight $-w$ if its energy level e_i is lower than w. We will say that edge $q_i \xrightarrow{w} q'_i$ is *enabled* if $e_i + w \geq 0$. For move transitions, we distinguish several semantics, depending on whether the agents move simultaneously or not.

Definition 3. *Consider two configurations* $C = \langle (q_1, \dots, q_n), (e_1, \dots, e_n) \rangle$ *and* $C' = \langle (q'_1, \dots, q'_n), (e'_1, \dots, e'_n) \rangle$*.*

move *There is a* move transition *from* C *to* C' *if one of the following holds*

- **asynchronous** $\exists i \in [\![1, n]\!] : q_i \xrightarrow{w} q'_i \in E_i$*,* $e'_i = e_i + w \geq 0$ *and* $\forall j \neq i, (q'_j, e'_j) = (q_j, e_j)$*, corresponding to the single agent* A_i *moving along an edge of its local arena. This results in an* asynchronous *move transition, and is denoted* $C \longrightarrow_m^a C'$*.*
- **synchronous** $\forall i \in [\![1, n]\!]$*,* $q_i \xrightarrow{w_i} q'_i \in E_i$ *and* $e'_i = e_i + w_i \geq 0$*, corresponding to all agents moving simultaneously in their respective local arenas. This results in a* strongly synchronous *move transition, denoted* $C \longrightarrow_m^s C'$*.*
- **weakly synchronous** $\forall i \in [\![1, n]\!]$*, either* $q_i \xrightarrow{w_i} q'_i \in E_i$ *and* $e'_i = e_i + w_i \geq 0$ *or* $(q'_i, e'_i) = (q_i, e_i)$ *and* $\forall q_i \xrightarrow{w_i} q''_i \in E_i, e_i + w_i < 0$*, corresponding to a synchronous move of all agents that have an enabled edge. This results in a* weakly synchronous *move transition, denoted* $C \longrightarrow_m^w C'$*.*

transfer *There is a* transfer transition *from* C *to* C' *if* $\forall i, q_i = q'_i$*, and* $\exists i, j \in [\![1, n]\!]$*,* $\mathcal{T}(q_i) = \mathcal{T}(q_j)$*,* $e_i + e_j = e'_i + e'_j$ *and* $\forall k \notin \{i, j\}$*,* $e_k = e'_k$*, corresponding to a transfer between agents* A_i *and* A_j *on vertices of a same transfer group. This transition is denoted* $C \longrightarrow_t C'$*.*

We use $\longrightarrow^a$ (resp. $\longrightarrow^s$, resp. $\longrightarrow^w$) to denote a transition that is either a transfer or an asynchronous (resp. strongly synchronous, resp. weakly synchronous) move and call it an asynchronous (resp. strongly synchronous, resp. weakly synchronous) transition for short. For instance $\longrightarrow^a = \longrightarrow^a_m \cup \longrightarrow_t$. We also refer to any type of move transition with $\longrightarrow_m$: $\longrightarrow_m = \longrightarrow^a_m \cup \longrightarrow^s_m \cup \longrightarrow^w_m$. Finally, an arbitrary transition is simply denoted $\longrightarrow$. Notice that agents change their local vertex in their arena during moves, and stay on the same vertex during transfers.

As usual, sequences of transitions define runs of the transfer system. A finite/infinite *asynchronous run* (resp. strongly synchronous run, resp. weakly synchronous run) over TS is a finite/infinite sequence of asynchronous (resp. strongly synchronous, resp. weakly synchronous) transitions. We will write $C \rightsquigarrow^a C'$ (resp. $C \rightsquigarrow^s C'$, $C \rightsquigarrow^w C'$) when there exists an asynchronous (resp. synchronous, weakly synchronous) run from C to C'. The set of asynchronous (resp. strongly synchronous, resp. weakly synchronous) runs over TS is denoted $\mathsf{Runs}^a(\mathsf{TS})$ (resp. $\mathsf{Runs}^s(\mathsf{TS})$, resp. $\mathsf{Runs}^w(\mathsf{TS})$). We refer to them as the asynchronous, strongly synchronous and weakly synchronous semantics of the transfer system, respectively, that we sometimes abbreviate into a-semantics, s-semantics and w-semantics.

We observe the following relations between runs of transfer systems under the various semantics. First of all, $\mathsf{Runs}^s(\mathsf{TS}) \subseteq \mathsf{Runs}^w(\mathsf{TS})$. Indeed, by definition, for every two configurations C, C', if $C \longrightarrow^s_m C'$ then $C \longrightarrow^w_m C'$. One can also notice that if $C \longrightarrow^w_m C'$, then there exists a sequence of move transitions $C \longrightarrow^a_m C_1 \longrightarrow^a_m \cdots \longrightarrow^a_m C'$. Thus, a run in the weakly synchronous semantics can be simulated by a run in the asynchronous semantics.

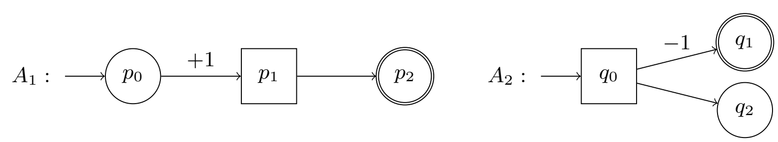

Fig. 1. Transfer system with two agents and a single transfer group with p_1 and q_0: $\mathcal{T} = \{\{p_1, q_0\}\}$. Null weights are omitted.

Example 1. Consider the transfer system with two agents depicted in Figure 1, where boxed vertices belong to the same transfer group. Let $C_0 = \langle (p_0, q_0), (0, 0) \rangle$ be the initial configuration. Then, there exists an asynchronous run from C_0 to the target state (p_2, q_1), namely $\langle (p_0, q_0), (0, 0) \rangle \longrightarrow^a_m \langle (p_1, q_0), (1, 0) \rangle \longrightarrow_t \langle (p_1, q_0), (0, 1) \rangle \longrightarrow^a_m \langle (p_1, q_1), (0, 0) \rangle \longrightarrow^a_m \langle (p_2, q_1), (0, 0) \rangle$. Yet, there are no strongly nor weakly synchronous runs to the target.

Consider now the transfer system depicted in Figure 2. Again, there exists an asynchronous execution that reaches the target state : $\langle (p_0, q_0), (0, 0) \rangle \longrightarrow^a_m \langle (p_1, q_0), (0, 0) \rangle \longrightarrow^a_m \langle (p_1, q_0), (1, 0) \rangle \longrightarrow^a_m \langle (p_1, q_1), (1, 1) \rangle \longrightarrow_t \langle (p_1, q_1), (0, 2) \rangle$

$\longrightarrow_m^a \langle (p_1, q_3), (0,0)\rangle \longrightarrow_m^a \langle (p_2, q_3), (0,0)\rangle$. Also, since the only transition fireable from $(q_1, 1)$ for A'_2 leads to q_2, there is no synchronous run to (p_2, q_3). Finally, even though A'_2 has a move transition available from $(q_1, 1)$ leading to q_2, there exists a weakly synchronous run that avoids the sink state q_2 and reaches the global target state: $\langle (p_0, q_0), (0,0)\rangle \longrightarrow_m^w \langle (p_1, q_1), (0,1)\rangle \longrightarrow_t \langle (p_1, q_1), (1,0)\rangle \longrightarrow_m^w \langle (p_1, q_1), (2,0)\rangle \longrightarrow_t \langle (p_1, q_1), (0,2)\rangle \longrightarrow_m^w \langle (p_2, q_3), (0,0)\rangle$. Interestingly, in q_1 it is in A'_2's interest to send energy to A'_1, thus not being able to fire any move transition while waiting for A'_1 to accumulate enough energy and send it back so that both reach their target.

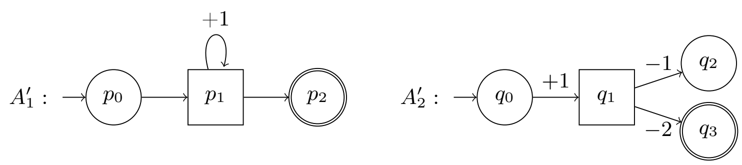

Fig. 2. Transfer system with two agents and a single transfer group with p_1 and q_1: $\mathcal{T} = \{\{p_1, q_1\}\}$. Null weights are omitted.

Reachability in transfer systems The transfer system is equipped with one reachability goal for each agent, given by a set of initial vertices and a set of final vertices. The global objective is then to find a run in which each agent reaches its final vertex, starting from its initial vertex and with initial energy 0. Similarly to vector addition systems with states (VASS), recall that the energy level of agents cannot drop below 0. As agents can reach their final vertices with an arbitrary energy level, we naturally introduce a coverability relation on configurations.

For two configurations $C = \langle S, (e_1, \dots, e_n)\rangle$ and $C' = \langle S', (e'_1, \dots, e'_n)\rangle$, we say C' *covers* C, written $C \lhd C'$, if $S = S'$ and $\forall i \in [\![1, n]\!], e_i \leq e'_i$. For any global state S we define *the covering* of S as $S\!\uparrow = \{C \mid \langle S, \overrightarrow{0}\rangle \lhd C\}$. We are now ready to define the verification problems of interest for transfer systems:

Problem x-Reach
Input: A transfer system TS, an initial state S_0 and a final state S_f
Question: Does there exist $C_f \in S_f\!\uparrow$ and a run $\rho : \langle S_0, \overrightarrow{0}\rangle \rightsquigarrow^x C_f$?

Note that this defines three decision problems, when one varies the semantics (parameter x): asynchronous, strongly synchronous or weakly synchronous. Moreover, we consider arbitrary transfer groups, as well as the special case of a unique trivial transfer group $T_\top = \bigcup_{i=1}^n V_i$. We later use ℓx-Reach and ux-Reach to highlight the *transfer group type* and respectively denote the variant with arbitrary transfer groups or a unique trivial transfer group.

Example 2. Back to the example of Figures 1 and 2, $\langle (A_1, A_2), \{\{p_1, q_0\}\}\rangle$ is a positive instance of ℓa-Reach, and $\langle (A'_1, A'_2), \{T_\top\}\rangle$ is a positive instance of

uw-REACH. but there exists only a single positive run for uw-REACH which is the same as for ℓw-REACH.

In the rest of the paper, we study the complexity of all variants of the tx-REACH problem. The following table summarizes the obtained results:

	semantics		
transfer	asynchronous	strongly synchronous	weakly synchronous
unique group	NP-hard (Th. 3) in PSPACE (Cor. 1)	PSPACE-c. (Th. 4 & Th. 5)	
arbitrary groups	PSPACE-c. (Th. 6 & Th. 7)	PSPACE-hard (Cor. 3) in EXPSPACE (Th. 8)	undecidable (Th. 9)

3 Relationships between the different semantics

A first, immediate, observation is that the unique variants of our decision problem are special cases of the local ones. Indeed, any instance of a ux-REACH with a single trivial transfer group is also an instance of ℓx-REACH. There is thus an immediate polynomial reduction from one to the other:

Proposition 1. *For every semantics $x \in \{a, s, w\}$, ux-REACH $\preceq_P$ ℓx-REACH.*

The following theorems relate to the asynchronous, strongly synchronous and weakly synchronous semantics (for a fixed transfer group type):

Theorem 1. *For every transfer group type $t \in \{u, \ell\}$, ta-REACH $\preceq_P$ ts-REACH.*

Proof. Let $\mathsf{TS} = \langle \{A_1, \dots, A_n\}, \mathcal{T} \rangle$ be a transfer system, and S_0, S_f initial and final global states. From TS, we build the transfer system TS' in which every vertex of every agent is added a self-loop with weight 0. Formally, $\mathsf{TS}' = \langle \{B_1, \dots, B_n\}, \mathcal{T} \rangle$ where for every $i \in [\![1, n]\!]$, if $A_i = (V_i, E_i)$ then $B_i = (V_i, F_i)$ with $F_i = E_i \cup \{q \xrightarrow{0} q \mid q \in V_i\}$. We claim that

$$\exists \langle S_0, \overrightarrow{0} \rangle \rightsquigarrow^a_{\mathsf{TS}} \langle S_f, \overrightarrow{e} \rangle \in \mathsf{Runs}^a(\mathsf{TS}) \quad \text{iff} \quad \exists \langle S_0, \overrightarrow{0} \rangle \rightsquigarrow^s_{\mathsf{TS}'} \langle S_f, \overrightarrow{e} \rangle \in \mathsf{Runs}^s(\mathsf{TS}').$$

Note that the difference between asynchronous and strongly synchronous semantics only lies in move transitions (and do not concern transfer transitions). Intuitively, the 0-self-loops in TS' are used to simulate an asynchronous run over TS by a synchronous run over TS'. Reciprocally, synchronous transitions over TS' can be serialized (and 0-self-loops can be removed) to obtain an asynchronous run over TS. □

Theorem 2. *For every transfer group type $t \in \{u, \ell\}$, ts-REACH $\preceq_P$ tw-REACH.*

Proof. Let $\mathsf{TS} = \langle \{A_1, \dots, A_n\}, \mathcal{T} \rangle$ be a transfer system, and S_0, S_f initial and final global states. From TS, we build the transfer system TS' in which each local arena A_i is augmented with a fresh vertex $\textsc{Bad}_i$ and additional edges from every vertex to $\textsc{Bad}_i$ with weight 0. Formally, $\mathsf{TS}' = \langle \{B_1, \dots, B_n\}, \mathcal{T} \rangle$

where for every $i \in [\![1,n]\!]$, if $A_i = (V_i, E_i)$ then $B_i = (V_i \cup \{\textsc{Bad}_i\}, F_i)$ with $F_i = E_i \cup \{q \xrightarrow{0} \textsc{Bad}_i \mid q \in V_i \cup \{\textsc{Bad}_i\}\}$. We claim that

$$\exists \langle S_0, \overrightarrow{0} \rangle \rightsquigarrow^s_{\mathsf{TS}} \langle S_f, \overrightarrow{e} \rangle \in \mathsf{Runs}^s(\mathsf{TS}) \quad \text{iff} \quad \exists \langle S_0, \overrightarrow{0} \rangle \rightsquigarrow^w_{\mathsf{TS}'} \langle S_f, \overrightarrow{e} \rangle \in \mathsf{Runs}^w(\mathsf{TS}').$$

Note that the difference between strongly and weakly synchronous semantics only lies in move transitions (and do not concern transfer transitions). Since the additional edges have weight 0, every agent always has an enabled edge. This means that for TS', the strongly synchronous runs and weakly synchronous runs coincide. Moreover, every vertex $\textsc{Bad}_i$ is a sink. Hence, a run reaching the global final states cannot visit $\textsc{Bad}_i$. Finally, by construction, the runs of $\mathsf{Runs}^w(\mathsf{TS}')$ that avoid all vertices $\textsc{Bad}_i$ are exactly the runs of $\mathsf{Runs}^s(\mathsf{TS})$. □

Remark 1 (Transfer systems and VASS.). In general, reachability in transfer systems can be cast into state-coverability of VASS. The state-space of the VASS is the product of sets of vertices for each agent, thus exponential in the transfer system size. The VASS transitions are induced by transfer transitions and move transitions, and their precise definition depends on the semantics of move transitions. In the case of a unique transfer group, for asynchronous and strongly synchronous semantics, 1-dim VASS even suffice, since intuitively, a unique counter is needed to store the total energy amount shared by the agents. For the weakly synchronous semantics however, it is less obvious how to represent the energy levels of agents with a single counter. Indeed, a global energy level exceeding the energies required to allow one move per agent is not a sufficient condition for all agents to move. For instance, an agent may have the incentive to transfer energy to another agent in order to be temporarilly blocked (see Examples 1 and 2). This suggests that the encoding in 1-dim VASS is not immediate for transfer systems under the weakly synchronous semantics, and that 1 dimension per agent may be nedded. Further, up to our knowledge, the state of the art on state-coverability in 1-dim VASS [13,17] yields worse complexity results than the direct proofs we present in the coming section, since the obtained VASS is exponential in the transfer system size. For arbitrary transfer groups, the situation is even worse since the reduction would be to an exponential size n-dim VASS.

4 Unique trivial transfer group

Let us start with the particular case of a unique and trivial transfer group: $\mathsf{TS} = \langle \{A_1, \dots, A_n\}, \{T_\top\} \rangle$. In such transfer systems, in every configuration, agents can transfer energy to others, regardless of their respective local vertices.

4.1 Asynchronous Semantics

For the asynchronous semantics, we prove the following complexity lower-bound:

Theorem 3. *ua-*$\textsc{Reach}$ *is* NP*-hard.*

Proof. We perform a reduction from the SubsetSum problem, that we recall now. Given a finite set of integers $\mathcal{S} = \{n_1, \ldots, n_m\}$ and a target integer $K \in \mathbb{N}$, the subset sum problem consists in determining whether there exists a subset $I \subseteq [\![1, m]\!]$ such that $\sum_{i \in I} n_i = K$. This problem is known to be NP-complete [15].

From an instance $\mathcal{S} = \langle \{n_1, \ldots, n_m\}, K\rangle$ of SubsetSum, we build a transfer system $\mathsf{TS} = \langle (A_C, A_1, \ldots, A_m), \{T_\top\}\rangle$ together with initial and final states S_0, S_f as represented in Figure 3, and such that $\mathcal{S}$ is a positive instance of SubsetSum iff there exists $C_f \in S_f\!\uparrow$ and $\langle S_0, \overrightarrow{0}\rangle \rightsquigarrow^a C_f$ in $\mathsf{Runs}^a(\mathsf{TS})$.

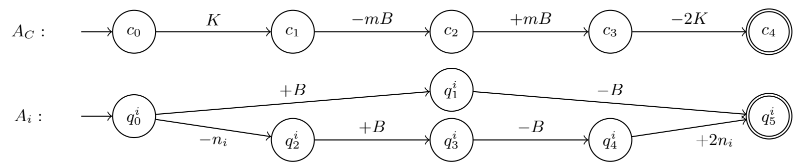

Fig. 3. Transfer system for the NP-hardness of *ua*-Reach. Incoming arrows point to initial vertices, and doubly circled vertices are final. $B = 2mK$.

Before giving the formal proof, let us explain the intuition of the reduction. The system is formed of one "control" agent A_C, as well as one agent A_i for each integer n_i. The transfer system starts with K energy units (gained by the controller A_C). In a first step, the agents A_i will have to select whether or not the solution set I contains i. This choice corresponds to two branches of the local arena A_i. If they decide $i \in I$, they have to pay n_i, which ensures that the sum of all the values selected by the agents is at most K. After this choice, each A_i agent receives a very large value B. This is a synchronization point: A_C needs all the agents to have received B energy units before it can progress, thus ensuring that every agent made their choice. In the third step, each agent A_i which decided that $i \in I$ receives $2n_i$ before reaching its target. As A_C needs $2K$ to reach its target, this will require that the sum of all the values selected by the agents is at least K. All in all, the sum has to be exactly K. Note that without the synchronization point, some agent A_j could wait for the $2n_i$ to be produced by A_i before making their initial choice.

Let us now formally describe the reduction and establish its correctness. Without loss of generality we assume that for every $i \in [\![1, m]\!]$, $n_i \leq K$, and we set $B = 2mK$. The local arenas are defined by:

- $A_C = (\{c_i \mid i \in [\![0, 4]\!]\}, E)$ with $E = \left\{c_j \xrightarrow{w_j} c_{j+1} \mid j \in [\![0, 3]\!]\right\}$, and $w_0 = K, w_1 = -mB, w_2 = mB, w_3 = -2K$.
- for every $i \in [\![1, m]\!]$, $A_i = \left(\left\{q_j^i \mid j \in [\![0, 5]\!]\right\}, E_i\right)$ with :

$$E_i = \left\{(q_0^i \xrightarrow{w_{01}^i} q_1^i), (q_1^i \xrightarrow{w_{15}^i} q_5^i)\right\}$$
$$\cup \left\{(q_0^i \xrightarrow{w_{02}^i} q_2^i), (q_2^i \xrightarrow{w_{23}^i} q_3^i), (q_3^i \xrightarrow{w_{34}^i} q_4^i), (q_4^i \xrightarrow{w_{45}^i} q_5^i)\right\};$$

and $w_{01}^i = w_{23}^i = B, w_{15}^i = w_{34}^i = -B, w_{02}^i = -n_i$ and $w_{45}^i = 2n_i$.

Since we consider ua-REACH, the only transfer group is $T_\top$ that contains every vertex of every agent. The initial and target states are $S_0 = (c_0, q_0^1, \cdots q_0^m)$ and $S_f = (c_4, q_5^1, \cdots, q_5^m)$.

We claim that some $C_f \in S_f\!\uparrow$ is reachable from $\langle S_0, \overrightarrow{0}\rangle$ by an asynchronous run if and only if there exists $I \subseteq [\![1, m]\!]$ with $\sum_{i\in I} n_i = K$.

As a first step, let us show that if an agent A_i takes an edge with cost $-B$ before A_C reaches c_2, then the controller cannot reach its target. The maximum amount of energy that can be collected in the system before A_C reaches c_2 is bounded by $mB + K + \sum_{i=1}^{m} 2n_i$. Assume an edge with weight $-B$ is taken, the total energy is now bounded by $(m-1)B + K + \sum_{i=1}^{m} 2n_i$. Due to our choice of B, this value is strictly less than mB and A_C cannot take the edge to c_2.

The previous reasoning implies that every available edge with weight $+B$ must be taken by the agents A_i before A_C can reach c_2. Following this observation, fix a run and assume that every agent took an edge with weight $+B$ but did not take its edge with weight $-B$ yet; assume further that A_C has not reached c_2 yet; finally, without loss of generality, assume that A_C reached c_1. For every $i \in [\![1, m]\!]$, agent A_i is thus either in q_1^i or in q_3^i. Let H be defined as the set of indices i such that A_i is in q_3^i.
Assume that $D = \sum_{i\in H} n_i > K$, then the current total energy is $mB + K - D < mB$, thus A_C cannot reach c_2 and will never reach its target. So $D \leq K$. From this point, as A_C needs to traverse to c_2 not to be blocked, we can assume it goes immediately to c_3. Agent A_C and the agents A_i with $i \notin H$ will no longer gain energy before reaching their target, so we can assume the agents in H act first. They lose an amount of energy of $|H|B$ and gain $2D$. Thus, the total energy in the system is $(m - |H|)B + 2D + (K - D)$. Exactly $(m - |H|)B + 2K$ energy units are required for the remaining agents to reach their target. This is only possible if $D \geq K$. As we already showed that $D \leq K$, this means that $D = K$.

This concludes the proof. □

Note that the hardness proof of Theorem 3 uses acyclic arenas. Moreover, given a transfer system where each local arena is acyclic, the reachability problems under each semantics is in NP. Indeed, the length of a path for each agent from its initial vertex to a final vertex is bounded by the number of vertices. To derive a non-deterministic polynomial time algorithm, one can thus guess for each agent a linear length path, and then check whether they can be combined into a complete run of the transfer system. The latter can be done in polynomial time by checking that at every step the global energy exceeds the one needed for the next transition. Therefore, for acyclic local arenas, ua-REACH is NP-complete.

Towards a complexity upper-bound beyond the acyclic case, we observe that thanks to Theorem 1, ua-REACH reduces in polynomial time to us-REACH. We will state in Theorem 4 that the latter is solvable in polynomial space.

Corollary 1. *ua-REACH is in PSPACE.*

4.2 Strongly and Weakly Synchronous Semantics

Theorem 4. *uw*-Reach *is in* PSPACE.

Proof. To prove membership in PSPACE, we show a small witness property. Precisely, in the *uw*-semantics, there exist exponential bounds $B_{\max}$ and $L_{\max}$ such that: if there is a run to the target global state, then there is one (1) of length at most $L_{\max}$ and (2) along which if the energy level of the agents reach $B_{\max}$, then the energy requirements can be ignored for the rest of the run. Further, both bounds are exponential in the size of the input transfer system.

Consider the instance $\mathsf{TS} = \langle \{A_1, \ldots, A_n\}, \{T_\top\} \rangle$ of *uw*-Reach together with initial and final states S_0 and S_f. Denote by $w_{\max}$ the largest absolute value among the weights of the edges in all A_i's. Suppose agent A_k has $B_{\max} = n \cdot |\mathsf{TS}|^n \cdot w_{\max}$ energy units. It can transfer $K = |\mathsf{TS}|^n \cdot w_{\max}$ energy units to each other agent, and still have energy level K. Now, focusing on states only, not on energy vectors, the length of a cycle-free path in the transfer system is at most $\prod_{i=1}^{n} |V_i| \leq |\mathsf{TS}|^n$. With K energy units, each agent is hence able to take an acyclic path to its target, losing at most $w_{\max}$ energy units at each step. Hence, from a configuration storing $B_{\max}$ energy units, the energy can be distributed in such a way that each agent reaches its goal assuming it is reachable from its current vertex.

When exploring runs with bounded energy levels, one thus only needs to look for relatively short runs. The number of useful configurations is bounded by $L_{\max} = \prod_{i=1}^{n} |V_i| \cdot (B_{\max} + 1)$, and each of these is visited at most once in a useful run. Therefore, useful runs are then of length at most $L_{\max}$, a value that is exponential in $|\mathsf{TS}|$.

Let us thus first consider runs with energy levels bounded by B_{max}. The number of configurations such runs visit is bounded by $\prod_{i=1}^{n} |V_i| \cdot (B_{\max} + 1)$. One can also notice that a run from S_0 to S_f does not need to contain cycles. Hence configurations need only be visited at most once. This induces a bound on the length of relevant runs: $L_{\max} = \prod_{i=1}^{n} |V_i| \cdot (B_{\max} + 2)$ (notice here that $\prod_{i=1}^{n} |V_i|$ steps can be required to reach the target when energy level is above $B_{\max}$).

Using this bound on the maximum length of runs to reach the goal, we can design a non-deterministic algorithm that starts from the initial configuration, and explores runs of length at most $L_{\max}$ among configurations that store at most B_{max} energy. The algorithm returns yes if the final state S_f is reached, and fails if the length of the run exceeds L_{max} or if the current configuration is a deadlock. It requires polynomial space to store a configuration and the step-counter. Indeed, a configuration is represented by storing for each agent its vertex and and its energy level. When energy levels are bounded by $B_{\max}$, the space needed to store them is logarithmic in $B_{\max}$. Moreover, the length of runs can be encoded by a counter taking values up to $L_{\max}$, which can be encoded in space logarithmic in $L_{\max}$. Since both bound are exponential in $\mathcal{A}$, polynomial space in $|\mathsf{TS}|$ is sufficient. By Savitch's theorem [22], this non-deterministic polynomial space algorithm proves membership in PSPACE. □

Theorem 5. *us-*Reach *is* PSPACE*-hard.*

Proof. To prove PSPACE-hardness, we reduce the reachability problem for 1-safe Petri nets which is PSPACE-complete. More precisely, w.l.o.g., we consider 1-safe Petri nets in which no place is in the postset and the preset of the same transition; reachability is known to be PSPACE-complete for this class [7].

A Petri net is a tuple $\mathcal{N} = \langle P, T; F\rangle$ where $P = \{p_1, \ldots, p_n\}$ is a set of places, $T = \{t_1, \ldots, t_m\}$ is a set of transitions, and $F \subseteq P \times T \cup T \times P$ is a flow relation. A marking of a Petri net is a map $M : P \to \mathbb{N}$ that associates a number of tokens to each place. The preset of a transition is the set of places ${}^\bullet t = \{p \in P \mid (p,t) \in F\}$ and the postset of t is the set of places $t^\bullet = \{p \in P \mid (t,p) \in F\}$. A transition is firable from marking M if, for every place $p \in {}^\bullet t$, $M(p) > 0$. Firing a transition t from marking M decrements $M(p)$ by 1 for every place of p the preset, and increments $M(p')$ for each p' in its postset. We write $M[t\rangle M'$ when M' is the marking obtained by firing t from M. Given a Petri net $\mathcal{N}$, one can define the set of reachable markings $\mathsf{Reach}(\mathcal{N}, M_0)$ that are reachable from M_0. The net $\mathcal{N}$ is *1-safe* if, for every marking M in $\mathsf{Reach}(\mathcal{N}, M_0)$ and every place p, $M(p) \leq 1$. The reachability problem for Petri nets consists in deciding whether a given input marking M belongs to $\mathsf{Reach}(\mathcal{N}, M_0)$.

Let $\mathcal{N} = \langle P, T; F\rangle$ be a 1-safe Petri net. Consider the transfer system with a trivial transfer group $\mathsf{TS} = \langle \{A_C, A_1, \ldots, A_n\}, \{T_\top\}\rangle$ represented in Figures 4 and 5. This arena is composed of one control agent A_C, and one agent A_i per place $p_i \in P$.

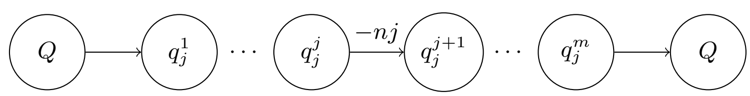

Fig. 4. Local arena of agent A_C. Only vertices relevant for transition t_j are depicted.

Formally, the control agent is $A_C = (V_C, E_C)$ with

$$V_C = \{Q\} \cup \left\{q_j^{j'} \mid j, j' \in [\![1, m]\!]\right\};$$
$$E_C = \left\{q_j^{j'} \xrightarrow{w_j^{j'}} q_j^{j'+1} \mid j' \in [\![1, m-1]\!], j \in [\![1, m]\!]\right\}$$
$$\cup \ \left\{Q \xrightarrow{0} q_j^1, q_j^m \xrightarrow{w_j^m} Q \mid j \in [\![1, m]\!]\right\}$$

and $\forall j, w_j^j = -n \cdot j$ and all other weights are null.

Intuitively, the controller chooses to fire t_j by moving to vertex q_j and expects to get enough energy in time. We will see that the only way for A_C to pay nj energy units at step j is if other agents have also chosen to fire t_j.

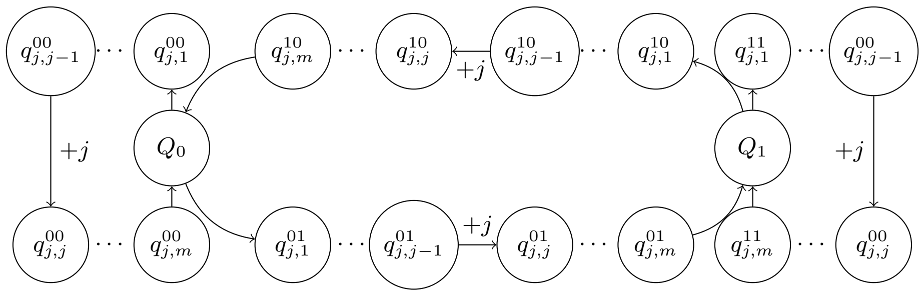

Fig. 5. Local arena of agent A_i. Only edges relevant for transition t_j are depicted.

Then, for every $i \in [\![1,n]\!]$, there is an agent $A_i = (V_i, E_i)$ with $V_i = \{Q_0, Q_1\} \cup \left\{q^{AB}_{j,j'} \mid j, j' \in [\![1,m]\!]\ A, B \in \{0,1\}\right\}$, and

$$E_i = \left\{ q^{AB}_{j,j'-1} \xrightarrow{w^{AB}_{j,j'}} q^{AB}_{j,j'} \mid j' \in [\![2,m]\!], j \in [\![1,m]\!], A, B \in \{0,1\} \right\}$$
$$\cup \left\{ Q_A \xrightarrow{w^{AB}_{j,1}} q^{AB}_{j,1}, q^{AB}_{j,m} \xrightarrow{0} Q_B \mid j \in [\![1,m]\!], A, B \in \{0,1\} \right\}$$

and for every transition $t_j \in T$, if $p_i \in {}^\bullet t_j$, then $w^{00}_{jj} = w^{01}_{jj} = w^{11}_{jj} = 0$ and $w^{10}_{jj} = j$; else if $p_i \in t_j^\bullet$, then $w^{00}_{jj} = w^{10}_{jj} = w^{11}_{jj} = 0$ and $w^{01}_{jj} = j$; otherwise $w^{10}_{jj} = w^{01}_{jj} = 0$ and $w^{11}_{jj} = w^{00}_{jj} = j$. All other weights are null. Intuitively, agent A_i at Q_0 represents place p_i having no tokens and agent A_i at Q_1 represents p_i having one token. A_i can provide for the j energy units needed by A_C through simulating the firing of t_j only if A_i is on a path corresponding to the effect of t_j on p_i: if p_i loses one token ($p_i \in {}^\bullet t_j$), the only correct path is through the 10-vertex; if p_i gains one token ($p_i \in t_j^\bullet$), the only correct path is via the 01-vertex; otherwise t_j has no effect on p_i : the correct paths are via the 00- or 11-vertex depending on the current marking.

Obviously, if firing transition t_j from the marking M leads to the marking M', there is a run from C(M) to C(M') : For all $p_i \in {}^\bullet t_j$, A_i follows the path 10, for all $p_i \in t_j^\bullet$, A_i follows the path 01 and each other agent A_i follows the path $M(p_i)M(p_i)$. Note that each agent A_i follows the path $M(p_i)M'(p_i)$. With these choices, before states q_{jj} each agent except A_C gains j energy units. On q_{jj} they all transfer that amount to A_C which can leave q_{jj} with exactly enough energy to pay the $-nj$.

Now suppose that there is a run $\rho : C(M) \rightsquigarrow^a_{\mathsf{TS}} C'$ with $m+1$ move transitions. We show that there exists M' such that $C' = C(M')$. If agent A_C follows states q_j. Suppose agents A_i follow states q_{j_i}, they will receive at most j_i energy units through the j_i-th edge of their path. The amount of energy available to A_C before going through its $j+1$-th edge is at most $\sum_{j_i \leq j} j_i$ because if $j_i > j$ this energy has not been gained before the $j+1$-th edge. The only way for $\sum_{j_i \leq j} j_i$ to be greater than or equal to nj is if for all i, $j_i = j$. Thus agents A_i follow

q_j states. But because A_i must gain j energy units, it must follow a path that represents a possible behavior of the firing of t_j on place p_i. Thus ρ is exactly as described in the first part of this proof, t_j is enabled by M and C' represents the marking M' resulting from firing t_j in M: $C' = C(M')$.

Finally, this reduction is polynomial : $|A_C| = O(m(m + \log(nm)))$ and $\forall i, |A_i| = O(m(m + \log m))$. $|\mathsf{TS}| = O(nm^2 \log(mn))$. □

Thanks to Theorem 2, we deduce:

Corollary 2. *us*-Reach *and uw*-Reach *are* PSPACE-*complete.*

5 Arbitrary transfer groups

5.1 Asynchronous semantics

Let us now consider the complexity of ℓa-Reach, i.e., reachability for transfer systems with local transfer groups, and under asynchronous semantics. We show below that this problem is PSPACE-complete. The PSPACE membership is shown by exhibiting an algorithm that requires polynomial space to reach a target configuration. The PSPACE-hardness is proved by a reduction from a reachability problem for safe Petri nets. We give a construction that builds for each safe Petri net, a transfer system of polynomial size w.r.t. the original net, and whose runs simulate that net. We already mentioned that [7] proved PSPACE-completeness of reachability for safe Petri nets. Later, [9] has showed that reachability is NP-complete for free-choice safe Petri nets. However, the encoding shown below applies to any 1-safe Petri net. Let us start with the PSPACE membership.

Theorem 6. ℓa-Reach *is in* PSPACE.

Proof. Consider $\mathsf{TS} = \langle \{A_1, \dots, A_n\}, \mathcal{T} \rangle$. Call $e_{\max}$ the biggest weight on an edge in TS and $S_{\max}$ the biggest size among sets S_i. From each vertex of its graph, an agent with $B_{\max} = S_{\max} \cdot e_{\max}$ energy units does not need to receive energy from an other agent to reach any other vertex of its graph. In the sequel, we give an upper bound on the useful energy level of an agent, taking into account that it may transfer energy to others to help them achieve their reachability objective. We show that if TS is a positive instance of ℓa-Reach, then there exists a witness execution ρ in which the energy of each agent is bounded by $B_{\max}(2n-1)$.

Fix $\rho \in \mathsf{Runs}^a(\mathsf{TS})$. We say that A_i *helps* A_j by e energy units along ρ with 0 intermediary if A_i sends at least e energy units to A_j through a transfer transition in ρ and the energy level of A_j is at least e right after the last transfer transition involving this agent in ρ. We say that A_i helps A_j by e energy units along ρ with $k \geq 1$ intermediaries if A_i transfers at least e energy units to an other agent A_p that helps A_j along ρ by e energy units with $k-1$ intermediaries. If A_i helps A_j along ρ by e energy units with any number of intermediaries, we say that A_i helps A_j by e energy units for short. This can be thought of as if A_i has ultimately sent e energy units to A_j that it can keep for itself.

We show by induction on k that if an agent starts ρ with $(2k-1)B_{\max}$ energy units, it may help up to k other agents by $B_{\max}$ energy units and have at least $B_{\max}$ energy units after its last transfer transition. The case $k = 0$ is immediate. Suppose the property holds until $k \geq 0$. If agent A_i has $(2k+1)B_{\max}$ energy units, it may try to meet an other agent A_j at cost at most $B_{\max}$ leaving at least $2kB_{\max}$ energy units. If A_i sends $(2k'-1)B_{\max}$ energy units to A_j for some $k' \in [\![0, k]\!]$, then A_j may help up to k' other agents by $B_{\max}$ and A_i may help up to $k - k' - 1$ other agents by $B_{\max}$. Note that A_i helps all the up to k' agents that A_j helps (with an additional intermediary) and because A_j has at least $B_{\max}$ energy units after its last transfer transition, A_i also helps A_j which adds up to a total of k agents helped. As a consequence, an agent never needs to have more than $(2n-1)B_{\max}$ energy units since helping every other agents by $B_{\max}$ while still having that much afterwards is enough for TS to reach the final state.

The same way as we showed it in the proof of Theorem 4, we have now an exponential bound in $O(n|\mathsf{TS}|2^{|\mathsf{TS}|})$ on useful energy levels which results in a polynomial bound in $O(|\mathsf{TS}|\log(n|\mathsf{TS}|))$ on the space needed to store a useful configuration and an exponential bound $L_{\max} \in O(n|\mathsf{TS}|2^{|\mathsf{TS}|})$ on the length of useful runs. In the end, there exists an NPSPACE algorithm that explores runs of size at most $L_{\max}$ and either fails if the final state S_f is not reached in $L_{\max}$ steps (the current length of the run can be stored in space $O(|\mathsf{TS}|\log(n|\mathsf{TS}|)))$ or if a deadlock is reached, and succeeds otherwise. By Savitch's theorem, we get that ℓa-Reach is in PSPACE. □

Theorem 7. *ℓa-Reach is PSPACE-hard.*

Proof (sketch). We encode a reachability problem for safe Petri nets in a ℓa -Reach problem with a transfer system whose size is linear in the size of the considered net. Let $\mathcal{N} = (P, T; F)$ be a safe Petri net with initial marking M_0. We build a transfer system composed of $n+1$ agents, $\mathsf{TS}_{\mathcal{N}} = \langle\{A_C, A_1, \ldots, A_n\}, \mathcal{T}\rangle$ simulating the behavior of $\mathcal{N}$. We do not give the whole construction here, and refer to [2] for details. The first agent A_C is a controller that initiates the simulation of a transition firing. Agents of the form A_i encode the contents of place p_i through their states, and simulate the effect of a transition firing via sequences of moves. We distinguish in particular two states $A_{i,0}$ and $A_{i,1}$, used to encode $M(p_i) = 0$ and $M(p_i) = 1$ in order to represent the marking M. Then we set an ordering on places, and ensure that when the controller agent chooses a particular transition t, all place agents choose the transition they simulate, but have to wait for energy from their predecessor to progress in this simulation. If two agents choose different transitions, the system deadlocks. Upon agreement on the chosen transition to simulate, the last agent A_n eventually sends back energy to the controller, acknowledging the fact that all places are engaged in the simulation of the same transition from the same marking. The controller then launches another round among place agents (still by transferring energy) who successively update their state to encode the effect of t on their place contents before acknowledging all changes to the controller. For instance, if transition t

consumes a token from place p_i and $M(p_i) = 1$, then agent A_i will start its interactions from state $A_{i,1}$ and will end the simulation of t's firing in state $A_{i,0}$. If a transition t is chosen, p_i is in the preset of t, but A_i started from state $A_{i,0}$, then choosing to simulate t will send A_i to a deadlock state, and will prevent reaching global states that encode markings. During this simulation process, if a place agent does not transfer energy to its successor and moves to its next state, then the system necessarily deadlocks or can return to the situation where the transfer was missed. When an agent keeps energy for its future moves, it can only repeat the choice of a new transition to simulate, hence canceling its previous choice. The only way to simulate properly a Petri net transition is if all agents choose the same transition and transfer their energy to their successor. Other choices lead either to deadlocks or livelocks in configurations that do not encode markings.

Configurations of $\mathsf{TS}_{\mathcal{N}}$ of the form $\langle (q_1^C, A_{1,b_1}, \cdot, A_{n,b_n}), \overrightarrow{0} \rangle$ will be called *stable* configurations (and other configurations *unstable*). Stable configurations represent an encoding of a marking of $\mathcal{N}$. A first step in the proof is to show that, for a pair of markings M, M' of $\mathcal{N}$, such that $M[t\rangle M'$, there exists a run from $C(M) = \langle (q_1^C, A_{1,M(p_1)}, \dots, A_{n,M(p_n)}), \overrightarrow{0} \rangle$ to $C(M')$ in $\mathsf{TS}_{\mathcal{N}}$. The shape of such runs is depicted in Figure 6. In this figure, agents steps are organized as local sequences, red dashed arrows depict energy transfers, and vertices that belong to the same transfer group have identical shape and color. This first part of the proof shows that a transfer system can simulate a safe Petri net, as for $M[t\rangle M'$, there exists a "canonical" run $\rho_{M[t\rangle M'}$ from stable configuration $C(M)$ to stable configuration $C(M')$ that does not visit any other stable configuration.

It then remains to show that $\mathsf{TS}_{\mathcal{N}}$ does not allow the reachability of stable configurations that are not encodings of reachable markings. To this extent, we look at the transition system composed of possible configurations and moves of the transfer system and highlight its properties. One can show that the runs encoding firing of a transition t of a safe net follow a particular pattern: Agents choose their transition and guess their predecessor's bit. When all agents agree on a common transition, one unit of energy flows from A_c to A_1, $A_2 \dots A_n$ and then back to A_c. A second round then starts, with two units of energy transferred successively from A_c to A_1, $A_2 \dots A_n$. If two agents did incompatible choices of simulated transition or control bit, then a deadlock (without reaching any stable configuration) is unavoidable. Similarly, if an agent does not transfer all its energy to its successor , then the system either deadlocks, or enters an infinite sequence of moves that can be only exited by sending back the faulty agent to the state from which the wrong choice was performed, still without visiting a stable configuration.

We can hence conclude that a marking M of the safe Petri net $\mathcal{N}$ is reachable from marking M_0 if and only if the stable configuration $C(M)$ is reachable from the configuration $C(M_0)$ in $\mathsf{TS}_{\mathcal{N}}$. Hence, ℓa-Reach is PSPACE-hard. ◻

5.2 Strongly synchronous semantics

Theorem 8. *ℓs-Reach is in* EXPSPACE.

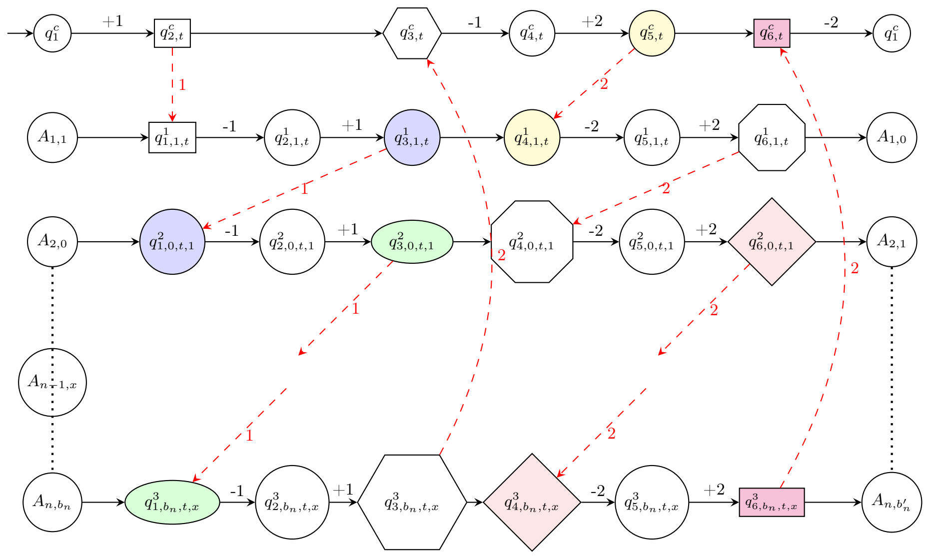

Fig. 6. Simulating a transition t moving a token from p_1 to p_2

Proof. The exponential space algorithm we exhibit to show membership in EXPSPACE involves the construction of a VASS $\mathcal{V}$ of exponential size yet polynomial dimension. Rackoff's backward algorithm for VASS coverability requires $2^{O(d)} \cdot \log|\mathcal{V}|$ space [16] where d is the dimension of $\mathcal{V}$. This results in an overall exponential space algorithm for ℓs-Reach.

The above-mentioned VASS $\mathcal{V}$ is built as follows. Given an instance $\mathsf{TS} = \langle\{A_1, \cdots, A_n\}, \mathcal{T}\rangle$ with $\forall i, A_i = (V_i, E_i)$, we define $\mathcal{V} = \langle\mathcal{S}, \delta\rangle$ such that:

- $\mathcal{S} = \prod_{i=1}^{n} V_i$
- $\forall (q_1 \xrightarrow{w_1} q'_1, \ldots, q_n \xrightarrow{w_n} q'_n) \in \prod_{i=1}^{n} E_i, (q_1, \ldots, q_n) \xrightarrow{w_1, \ldots, w_n} (q'_1, \ldots, q'_n) \in \delta$
- $\forall \overrightarrow{q} = (q_1, \ldots, q_n) \in \mathcal{S}, \forall q_i \neq q_j, \left[\exists T \in \mathcal{T} \mid q_i, q_j \in T \implies \overrightarrow{q} \xrightarrow{\overrightarrow{w_{ij}}} \overrightarrow{q} \in \delta\right]$

where $\overrightarrow{w_{ij}}$ is the vector with only zeros except $+1$ at index i and -1 at index j.

With this construction, whether there exists a run $\rho_{\mathsf{TS}} : \langle S, \overrightarrow{e}\rangle \rightsquigarrow^{s}_{\mathsf{TS}} \left\langle S', \overrightarrow{e'}\right\rangle$ is equivalent to whether there exists a run $\rho_{\mathcal{V}} : \langle S, \overrightarrow{e}\rangle \rightsquigarrow_{\mathcal{V}} \left\langle S', \overrightarrow{e'}\right\rangle$.
We show the direct implication by induction on the length of ρ_{TS}:

- If ρ_{TS} is empty, $\langle S, \overrightarrow{e}\rangle = \left\langle S', \overrightarrow{e'}\right\rangle$ and the property holds.
- Let ρ_{TS} be of length $l+1$. Let $\left\langle S^{-1}, \overrightarrow{e^{-1}}\right\rangle$ be the penultimate configuration of $\rho_{\mathcal{A}}$. By induction, there exists $\rho_{\mathcal{V}} : \langle S, \overrightarrow{e}\rangle \rightsquigarrow_{\mathcal{V}} \left\langle S^{-1}, \overrightarrow{e^{-1}}\right\rangle$. If the last transition of ρ_{TS} is a move transition, for all i there is a transition $q_i^{-1} \xrightarrow{w_i} q'_i \in E_i$ such that all energy levels $e'_i = e_i^{-1} + w_i$ are non-negative. $\rho_{\mathcal{V}}$

may reach $\left\langle S', \overrightarrow{e'} \right\rangle$ with the transition $S^{-1} \xrightarrow{w_1,\dots,w_n} S'$ induced by these transitions. Otherwise, there are some $i, j \in [\![1, n]\!]$ and some $T \in \mathcal{T}$ with $q'_i, q'_j \in T$, $e_i^{-1} + e_j^{-1} = e'_i + e'_j$ and $\forall k \notin \{i, j\}$, $e_k^{-1} = e'_k$. The transfer from coordinate i to coordinate j may be decomposed using the transition $S' \xrightarrow{\overrightarrow{w_{ij}}} S'$ since q'_i and q'_j share the same group T. In all cases, $\left\langle S', \overrightarrow{e'} \right\rangle$ is reachable.

Conversely, by induction on the length of $\rho_{\mathcal{V}}$:

- The case of $\rho_{\mathcal{V}}$ empty is immediate.
- Let $\rho_{\mathcal{V}}$ be of length $l+1$. Let $\left\langle S^{-1}, \overrightarrow{e^{-1}} \right\rangle$ be the penultimate configuration of $\rho_{\mathcal{V}}$. By induction, there exists $\rho_{\mathsf{TS}} : \langle S, \overrightarrow{e} \rangle \rightsquigarrow^{s}_{\mathsf{TS}} \left\langle S^{-1}, \overrightarrow{e^{-1}} \right\rangle$. The last transition of $\rho_{\mathcal{V}}$ has two possible forms. Either there is $(q_1 \xrightarrow{w_1} q'_1, \dots, q_n \xrightarrow{w_n} q'_n) \in \prod_{i=1}^{n} E_i$ such that this last transition is $(q_1, \dots, q_n) \xrightarrow{w_1,\dots,w_n} (q'_1, \dots, q'_n)$, in which case ρ_{TS} can be extended by the move transition induced by these edges to reach S'. Note that because the coordinates are maintained non-negative in $\mathcal{V}$, so will the energy levels. Or, there are some coordinates $i, j \in [\![1, n]\!]$ such that the last transition of $\rho_{\mathcal{V}}$ is $S' \xrightarrow{\overrightarrow{w_{ij}}} S'$ with some transfer group $T \in \mathcal{T}$ such that $q'_i, q'_j \in T$. In that case, a transfer transition from ρ_A that sends 1 energy unit from A_i to A_j reaches $\left\langle S', \overrightarrow{e'} \right\rangle$.

  According to the previous result, we conclude that if an instance is positive for ℓs-Reach then its joined instance for VASS-cover is also positive. Furthermore, $\mathcal{V}$ is of size $O(|\mathsf{TS}|^{2n})$ since there are $O(|\mathsf{TS}|^n)$ states, $O(n^2)$ loops on each state and $O((|\mathsf{TS}|^n)^2)$ other transitions.

□

For the complexity lower-bound, recall that ℓa-Reach is PSPACE-hard (Theorem 7) and conclude with Theorem 1 that:

Corollary 3. *ℓs-Reach is PSPACE-hard.*

5.3 Weakly synchronous semantics

Perhaps surprisingly, the relaxation of agents synchronization from strong to weak synchronous semantics, *i.e.* the fact that agents with no enabled edges may not move simulaneously with other agents, leads to undecidability. The main reason is the following. Both the asynchronous and (strongly) synchronous semantics enjoy a monotonicity property: higher energy levels can only enable more transitions and allow to reach more configurations. This monotonicity does not hold under the w-semantics. Indeed, it can be profitable for an agent to reach a vertex with a low energy level, so that it is allowed to "wait" for other agents to move and later gain energy through a transfer. Intuitively, this behaviour allows one to test whether an agent has energy left, thus encoding a zero test.

Theorem 9. *ℓw-Reach is undecidable.*

Proof. We give a reduction from the termination problem of Minsky machines, which is known to be undecidable [19]. Let us start with a quick recall on Minsky machines. A Minsky machine $\mathcal{M}$ is described by two counters x and y, as well as a sequence of commands $l_0, \ldots, l_m$ where l_0 is the starting command, l_m ends the run of the system, and every command l_0 to l_{m-1} is of one of the following three types:

- increment counter $c \in \{x, y\}$, move to the next command;
- decrement counter $c \in \{x, y\}$, move to the next command[3];
- if counter $c \in \{x, y\}$ is equal to 0, move to command l_k, otherwise move to command l_j.

In summary, the machine goes through a list of commands starting with l_0, incrementing, decrementing counters, or testing whether a counter is equal to 0 in order to select the new command to jump to—and it terminates whenever it reaches l_m. The *termination problem* for Minsky machines consists in deciding, given a machine $\mathcal{M}$, whether $\mathcal{M}$ terminates.

In our reduction, we use two agents with local arenas A_x and A_y, storing as energy level the current value of each counter, one control agent A_C that encodes the control flow of the Minsky machine, and one additional agent A_i for each command l_i to simulate the effect of l_i when it is activated by A_C. Sink vertices Bad are used to punish agents that reach a vertex with an energy level that differs from the one that is expected in the Minsky machine simulation.

We now detail how to encode a decrement: assume command l_i decrements counter $z \in \{x, y\}$. Only three agents take part in the simulation of l_i: the control agent A_C, agent A_z associated to counter z and agent A_i dedicated to l_i.

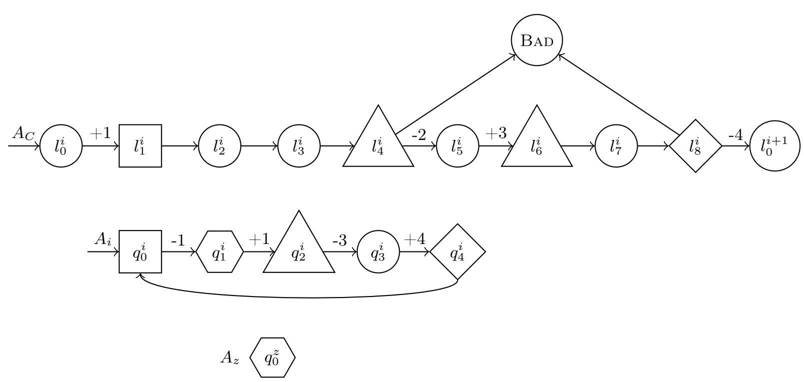

Fig. 7. Encoding a decrement. To the exception of circles, states with the same shape belong to the same transfer group.

[3] Note that the machine must be designed so that decrement can only occur when the counter value is positive.

Formally, we define the transfer system associated with command line l_i as $\mathsf{TS}_i = \langle \{A_z, A_C, A_i\}, \mathcal{T}_i \rangle$, depicted in Figure 7, with:

- $A_z = (\{q_0^z\}, \emptyset)$;
- $A_i = (V_i, E_i)$, where $V_i = \{q_j^i \mid j = 0 \dots 4\}$; $E_i = \{q_j^i \xrightarrow{w_j^i} q_k^i \mid j \in [\![0, 4]\!] \wedge (k \equiv j + 1 \pmod 5)\}$; with $w_0^i = -1, w_1^i = 1, w_2^i = -3, w_3^i = 4$ and $w_4^i = 0$.
- $A_C = (V_i^C, E_i^C)$, where $V_i^C = \{l_j^i \mid j \in [\![0, 8]\!]\} \cup \{\textsc{Bad}, l_0^{i+1}\}$ and $E_i^C = \{l_j^i \xrightarrow{w^i} l_r^i \mid j = 0 \dots 7 \wedge r = j + 1\} \cup \{l_8^i \xrightarrow{w^8} l_0^{i+1}, l_4^i \xrightarrow{0} \textsc{Bad}, l_8^i \xrightarrow{0} \textsc{Bad}\}$; and $w^0 = 1, w^4 = -2, w^5 = 3, w^8 = -4$ and other weights are null.
- $\mathcal{T}_i$ contains the groups $\{l_1^i, q_0^i\}$, $\{q_0^z, q_1^i\}$, $\{l_4^i, l_6^i, q_2^i\}$ and $\{l_8^i, q_4^i\}$.

Recall that this addresses a single command line l_i. To encode a complete Minsky machine $\mathcal{M}$ with k instructions, we assemble the arenas for all command lines into the transfer system $\mathsf{TS}_\mathcal{M} = \langle (A_C, A_x, A_y, A_1, \dots A_k), \bigcup \mathcal{T}_i \rangle$ A_C, in which A_x and A_y are the unions of sets of vertices and edges of the arenas of every command line. In particular, the vertex l_0^{i+1} is shared with the local arena associated with the command line l_{i+1}.

Let us show that if the agents A_C, A_i and A_z start in $\langle (l_0^i, q_0^i, q_0^z), (0, 0, n) \rangle$ with $n > 0$, then the only way to avoid $\textsc{Bad}$ leads them to the configuration $\langle (l_0^{i+1}, q_0^i, q_0^z), (0, 0, n - 1) \rangle$. Hence, executing the command line l_i will indeed decrement the energy of A_z by 1.

In the first step, only A_C can move, reaching l_1^i with 1 unit of energy. There it can either transfer this energy to A_i or keep it. If it chooses not to transfer, it remains the only agent able to move, and when reaching l_4^i, it will have 1 unit of energy, forcing it to go to $\textsc{Bad}$. Assume thus that A_C tranfers its energy unit to A_i. Both then move synchronously to l_2^i and q_1^i. In q_1^i, A_i can interact with A_z. Let m be the energy level of of A_i at that point. Agents A_C and A_i then reach l_3^i and q_2^i with energy levels 0 and $m + 1$. If $m + 1 \geq 3$, then both agents move to l_4^i and q_3^i and in the next step, A_C is forced to go to $\textsc{Bad}$. In order for A_C to avoid its $\textsc{Bad}$ vertex, it must be the case that $m + 1 < 3$. In this case, A_C progresses to l_4^i while A_i remains in q_2^i, because the only available edge consumes 3 units of energy. With this move, A_C can receive energy from A_i, as their current vertices belong to the same transfer group. Now A_C needs 2 energy units to avoid going to $\textsc{Bad}$, which requires $m + 1 \geq 2$. Hence $m + 1 = 2$ which implies that during their interaction, A_z transferred 1 energy unit to A_i, leaving A_z with $n - 1$ units of energy. After transferring 2 energy units to A_C, A_i remains stuck in q_2^i while A_C progresses to l_5^i and then l_6^i with 3 energy units. Again, A_C can choose to move on its own without transferring energy to A_i, but it will eventually reach vertex l_8^i, and lacking 4 energy units will be forced to go to $\textsc{Bad}$. If A_C transfers 3 energy units to A_i they both move to l_8^i and q_4^i where A_i can then transfer to A_C the 4 energy units required to avoid $\textsc{Bad}$. This then leads to a configuration where A_C is in state l_0^{i+1} with no energy left, A_i is back in vertex q_0^i also with no energy left, and A_z is in vertex q_0^z with an energy level $n - 1$. In this construction, transferring other quantities of energy always leads to deadlock configurations.

The encoding of the increment is similar. The zero test however is even more involved, allowing the agents to remain stuck in some vertices and wait for the other agents iff the counter value is 0. The constructions for these two operations are provided in details in [2]. Altogether, the three constructions ensure that in order to avoid the Bad vertices, the agents must correctly implement the command of the Minsky machine.

To complete the reduction, the reachability objective for the transfer system is defined as follows. The last command l_m of the Minsky machine, is represented by a single vertex l_0^m which is the target vertex of the control agent. The targets of other agents are the set of vertices l_0^m, q_0^i for $i \in [\![0, m-1]\!]$ and q_0^z for $z \in \{x, y\}$. As the agents must avoid the Bad vertices, the above constructions ensure that the target state is covered in the transfer system if and only if the Minsky machine terminates. □

6 Conclusion

This paper introduced and studied a cooperative game model, in which agents move on local weighted arenas, and can help each other by tranferring energy to their peers. We considered a global reachability question, *i.e.*, whether it is possible to reach a system configuration where each agent is in its goal vertex, while always keeping all energy levels non-negative. While transfer systems can be easily encoded as vector addition systems with states, and our reachability problems as a coverability question, we showed that the energy transfer feature induces a complexity drop, with complexities ranging from PSPACE to EXPSPACE. For asynchronous and strongly synchronous semantics, we exploited a form of monotonicity and a small witness property. However, monotonicity does not hold under weak synchronous semantics, leading to undecidability.

An obvious future work is to close the complexity gaps for ua-Reach and ℓs-Reach. The similarities between transfer systems and subclasses of VASS, in particular 1-dim VASS for ua-Reach might help solving this issue. Considering extensions of transfer systems is another interesting research direction, for instance with features that have been considered for Petri nets while maintaining decidability such as transfers or resets. Finally, beyond the purely cooperative question we tackled here, it would also be interesting to consider alternative problems in which the agents have conflicting objectives.

References

1. Toshiro Araki and Tadao Kasami. Some decision problems related to the reachability problem for Petri nets. *Theoretical Computer Science*, 3(1):85–104, 1976.
2. Nathalie Bertrand, Loïc Hélouët, Engel Lefaucheux, and Luca Paparazzo. Reachability in multi-agent transfer systems (extended version). Technical report, HAL Inria https://inria.hal.science/hal-05366409, 2025.

3. Tomás Brázdil, Petr Jancar, and Antonín Kucera. Reachability games on extended vector addition systems with states. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP'10)*, volume 6199 of *Lecture Notes in Computer Science*, pages 478–489. Springer, 2010.
4. Thomas Brihaye, Véronique Bruyère, Aline Goeminne, Jean-François Raskin, and Marie van den Bogaard. The complexity of subgame perfect equilibria in quantitative reachability games. In *Proceedings of the 30th International Conference on Concurrency Theory (CONCUR'19)*, volume 140 of *LIPIcs*, pages 13:1–13:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.
5. Thomas Brihaye, Amit Kumar Dhar, Gilles Geeraerts, Axel Haddad, and Benjamin Monmege. Efficient energy distribution in a smart grid using multi-player games. In *Proceedings of Cassting Workshop on Games for the Synthesis of Complex Systems and 3rd International Workshop on Synthesis of Complex Parameters (Cassting/SynCoP'16)*, volume 220 of *EPTCS*, pages 1–12, 2016.
6. Nils Bulling and Valentin Goranko. Combining quantitative and qualitative reasoning in concurrent multi-player games. *Autonomous Agents and Multi-Agent Systems*, 36(1):2, 2022.
7. Allan Cheng, Javier Esparza, and Jens Palsberg. Complexity results for 1-safe nets. *Theorerical Computer Science*, 147(1&2):117–136, 1995.
8. Gianfranco Ciardo. Petri nets with marking-dependent arc cardinality: Properties and analysis. In *Proceedings of the 15th International Conference on Application and Theory of Petri Nets 1994 (PetriNets'94)*, volume 815 of *Lecture Notes in Computer Science*, pages 179–198. Springer, 1994.
9. Javier Esparza. Reachability in live and safe free-choice Petri nets is NP-complete. *Theoretical Computer Science*, 198(1-2):211–224, 1998.
10. Uli Fahrenberg, Line Juhl, Kim G. Larsen, and Jirí Srba. Energy games in multiweighted automata. In *Proceedings of the 8th International Colloquium on Theoretical Aspects of Computing (ICTAC'11)*, volume 6916 of *Lecture Notes in Computer Science*, pages 95–115. Springer, 2011.
11. Ariel Felner, Roni Stern, Solomon Eyal Shimony, Eli Boyarski, Meir Goldenberg, Guni Sharon, Nathan R. Sturtevant, Glenn Wagner, and Pavel Surynek. Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges. In *Proceedings of the 10th International Symposium on Combinatorial Search (SOCS'17)*, pages 29–37. AAAI Press, 2017.
12. Julian Gutierrez, Aniello Murano, Giuseppe Perelli, Sasha Rubin, Thomas Steeples, and Michael J. Wooldridge. Equilibria for games with combined qualitative and quantitative objectives. *Acta Informatica*, 58(6):585–610, 2021.
13. Christoph Haase, Stephan Kreutzer, Joël Ouaknine, and James Worrell. Reachability in succinct and parametric one-counter automata. In *Proceeding of the 20th International Conference on Concurrency Theory (CONCUR'09)*, volume 5710 of *Lecture Notes in Computer Science*, pages 369–383. Springer, 2009.
14. John E. Hopcroft and Jean-Jacques Pansiot. On the reachability problem for 5-dimensional vector addition systems. *Theoretical Computer Science*, 8:135–159, 1979.
15. Richard M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer, 1972.
16. Marvin Künnemann, Filip Mazowiecki, Lia Schütze, Henry Sinclair-Banks, and Karol Wegrzycki. Coverability in VASS revisited: Improving Rackoff's bound to obtain conditional optimality. In *Proceeding of the 50th International Colloquium on Automata, Languages, and Programming (ICALP'23)*, volume 261 of *LIPIcs*, pages 131:1–131:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

17. Jérôme Leroux. Petri net reachability problem (invited talk). In *Proceedings of the 44th International Symposium on Mathematical Foundations of Computer Science (MFCS'19)*, volume 138 of *LIPIcs*, pages 5:1–5:3. Schloss Dagstuhl - Leibniz-Zentrum f"ur Informatik, 2019.
18. Richard Lipton. The reachability problem requires exponential space. Technical report, Yale University, 1976.
19. Marvin L. Minsky. *Computation: Finite and infinite machines*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1967.
20. John F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950.
21. Charles Rackoff. The covering and boundedness problems for vector addition systems. *Theoretical Computer Science*, 6:223–231, 1978.
22. Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970.
23. Philippe Schnoebelen. Revisiting Ackermann-hardness for lossy counter machines and reset Petri nets. In *Proceedings of the 35th International Symposium on Mathematical Foundations of Computer Science 2010 (MFCS'10)*, volume 6281 of *Lecture Notes in Computer Science*, pages 616–628. Springer, 2010.

This appendix presents full proofs, that were ommitted in the core of the paper due to space constraints.

A Missing proofs for Section 5.1

Theorem 7 ℓa-Reach is PSPACE-hard.

Proof. We encode a reachability problem for safe Petri nets as an ℓa-Reach problem in a transfer system whose size is linear in the size of the considered net. Before formalizing this encoding, we can give the general principles of the proof. For a safe Petri net with n places $\{p_1, \ldots, p_n\}$ and q transitions, the transfer system encoding a reachability question is composed of $n+1$ agents: $\{A_C\} \cup \{A_i, \mid p_i \in P\}$. The first agent A_C is a controller that initiates the simulation of a transitions's firing, and agents of the form A_i encode the contents of place p_i through their states, and simulate the effect of a transition's firing. We distinguish in particular two states, used to encode $m(p_i) = 0$ and $m(p_i) = 1$. Then we set an ordering on places, and ensure that when the controller agent chooses a particular transition t, all place agents choose the transition they simulate, but have to wait for energy from their predecessor to progress in this simulation. If two agents choose different transitions, the system deadlocks. Upon agreement on the choosen transition to simulate, the last agent A_n eventually sends back energy to the controller, acknowledging the fact that all places committed to the simulation of the same transition from the same marking. The controller then launches another round among place agents (still by transferring energy) who successively update their state to encode the effect of t on their place contents before acknowledging all changes to the controller. For instance, if transition t consumes a token from place p_i and $m(p_i) = 1$, then agent A_i will start its interactions from state $A_{i,1}$ and will end the simulation of t's firing in state $A_{i,0}$. If a transition t is chosen, p_i is in the preset of t, but A_i started from state $A_{i,0}$, then choosing to simulate t will send A_i to a deadlock state, and will prevent reaching global states that encode markings. During this transition simulation process, if a place agent does not transfer energy to its successor and moves to its next state, then the system necessarily deadlocks or can return to the situation where the transfer was missed. When an agent keeps energy for his future moves, it can only repeat the choice of a new transition to simulate, hence canceling its previous choice. The only way to simulate properly a Petri net transition is if all agents choose the same transition and transfer their energy to their successor. Other choices lead either to deadlocks or livelocks in configurations that do not encode markings.

Let us now formalize the encoding. Let $\mathcal{N} = (P, T, F, m_0)$ be a safe Petri net, were $P = \{p_1, \ldots, p_n\}$ is the set of places, $T = \{t_1, \ldots, t_q\}$ the set of transitions, $F \subseteq T \times P \cup P \times P$ the flow relation and $m_0 : P \to \{0, 1\}$ be a marking. We call the preset of a transition t the set of places $\{p \in P \mid (p, t) \in F\}$ and the postset of a transition t the set of places $\{p \in P \mid (t, p) \in F\}$. A marking in a map $m : P \to \{0, 1\}$ depicting a number of tokens in each place. A transition is firable from m if every place p in its preset is marked, i.e. $m(p) = 1$. Firing a transition t from m removes all tokens from the preset of $t$and puts a token in each place of the postset of t. We will say that a marking m is reachable by $\mathcal{N}$ from a marking m_0 if there exists a sequence of transitions firing starting from m_0 leading to m. Reachability for safe Petri nets is a PSPACE complete problem [7].

We will now build a transfer system $\mathsf{TS}_{\mathcal{N}} = (A_C, A_{p_1}, \dots A_{p_n}, \mathcal{T})$ simulating the behavior of $\mathcal{N}$. We first detail the construction of the local arena for controller agent A_C. It is composed of a set of states

$$S_C = \{q_1^C\} \cup \{q_{2,t,b_c}^C, q_{3,t,b_c}^C, q_{4,t,b_c}^C, q_{5,t,b_c}^C, q_{6,t,b_c}^C \mid b_c \in \{0,1\}, t \in T\}$$

The control bit b_c in states of the form q_{k,t,b_c}^C is a guess of making $m(p_n)$, that has to be correct to end successfully the simulation of t. Transitions of A_C are of the form

$$T_C = \{(q_1^C, +1, q_{2,t}^C), (q_{2,t}^C, 0, q_{3,t}^C), (q_{3,t}^C, -1, q_{4,t}^C), (q_{4,t}^C, +2, q_{5,t}^C), (q_{5,t}^C, 0, q_{6,t}^C), (q_{6,t}^C, -2, q_1^C) \mid t \in T\}$$

. Roughly speaking, this set of transitions corresponds to two loops of weight 0 around state q_1^C per transition of $\mathcal{N}$ (see Figure 8 for an illustration of one loop).

To simulate the contents of place p_1, we build an arena $A_{p_1} = (V_1, v_0^1, E_1, w_1)$ where V_1 is a set of states of the form

$$V_1 = \{A_{1,0}, A_{1,1}, D_1\} \cup \{q_{1,b_1,t}^1, q_{2,b_1,t}^1, q_{3,b_1,t}^1, q_{4,b_1,t}^1, q_{5,b_1,t}^1, q_{6,b_1,t}^1 \mid t \in T, b_1 \in \{0,1\}\}$$

The set of edges of the arena depict, for each transition t of the net, the effect of the firing of a transition on a place, via sequences of transitions from A_{1,b_1} to $A_{1,b_1'}$ of the form :

- Type 00 $\rho_{1,A_1,0} = A_{1,0} \xrightarrow{0} q_{1,0,t}^1 \xrightarrow{-1} q_{2,0,t}^1 \xrightarrow{+1} q_{3,0,t}^1 \xrightarrow{0} q_{4,0,t}^1 \xrightarrow{-2} q_{5,0,t}^1 \xrightarrow{+2} q_{6,0,t}^1 \xrightarrow{0} A_{1,0}$ when p_1 is not in the preset nor in the postset of t. Intuitively, place p_1 is not used by transition t so its contents is not changed.
- Type 11 A similar sequence $\rho_{1,A_1,1}$ is also part of transitions of A_{p_1}: $\rho_{1,A_1,1} = A_{1,1} \xrightarrow{0} q_{1,1,t}^1 \xrightarrow{-1} q_{2,1,t}^1 \xrightarrow{+1} q_{3,1,t}^1 \xrightarrow{0} q_{4,1,t}^1 \xrightarrow{-2} q_{5,1,t}^1 \xrightarrow{+2} q_{6,0,t}^1 \xrightarrow{0} A_{1,1}$. This type of sequence encodes situation where $m(p_1) = 1$ and either p_1 is both in the postset and in the preset of t ore in none of them.
- Type 01 Sequences of transitions from $A_{1,0}$ to $A_{1,1}$ of the form : $\rho_{2,A1} = A_{1,0} \xrightarrow{0} q_{1,0,t}^1 \xrightarrow{-1} q_{2,0,t}^1 \xrightarrow{+1} q_{3,0,t}^1 \xrightarrow{0} q_{4,0,t}^1 \xrightarrow{-2} q_{5,0,t}^1 \xrightarrow{+2} q_{6,0,t}^1 \xrightarrow{0} A_{1,1}$ when p_1 is not in the preset but is in the postset of t. These sequences represent the creation of one token in place p_1
- Type 10 Sequences of transitions from $A_{1,1}$ to $A_{1,0}$ of the form : $\rho_{3,A1} = A_{1,1} \xrightarrow{0} q_{1,1,t}^1 \xrightarrow{-1} q_{2,1,t}^1 \xrightarrow{+1} q_{3,1,t}^1 \xrightarrow{0} q_{4,1,t}^1 \xrightarrow{-2} q_{5,1,t}^1 \xrightarrow{+2} q_{6,1,t}^1 \xrightarrow{0} A_{1,0}$ when p_1 is in the preset but not in the postset of t. These sequences represent the consumption of one token in place p_1 by transition t when $m(p_1) = 1$. Notice that sequences of type 11 and of type 10 are exclusive, and depend of the flow relation of the simulated net.

– Type 0bad sequence of the form $\rho_{4,A1} = A_{1,0} \xrightarrow{0} q^1_{1,0,t} \xrightarrow{-1} q^1_{2,0,t} \xrightarrow{+1} q^1_{3,0,t} \xrightarrow{0} q^1_{4,0,t} \xrightarrow{-2} q^1_{5,0,t} \xrightarrow{+2} q^1_{6,0,t} \xrightarrow{0} D_1$ when p_1 is in the preset of t. This situation corresponds to the wrong choice of agent A_1 to simulate transition t when starting from a local state encoding $m(p) = 0$. Transition of type 01 and 0bad are exclusive in our construction.

Along these paths (as illustrated in Figure 8) a state of the form $q^1_{i,b_1,t}$ represents the i^th step of path starting from state A_{1,b_1} representing place p_1 with content $m(p_1) = b_1 \in \{0, 1\}$. Arena A_{p_1} has $|T| \cdot 12 + 2$ states, and $14 \cdot |T|$ transitions .

Then, for each agent $A_{p_2}, \dots, A_{p_n}$, we build similar sequences of type 00,01,10,11,0bad as for A_{p_1}, but these sequences are duplicated to differentiate situation where agent $k-1$ was in a state representing place p_k holding a token or not. For each agent A_{p_k}, we have a set of vertices $V_k = \{A_{k,0}, A_{k,1}, D_k\} \cup \{q^k_{i,b_k,t,0}, q^k_{i,b_k,t,1} \mid b_k \in \{0, 1\}, i \in 1..6, t \in T\}$. Intuitively, a state of the form A_{k,b_k} represents marking of place p_k with $m(p_k) = b_k$, and states of the form q^k_{i,b_k,t,b'_k} represent steps of a simulation of the effects of transition t on place p_k. Bit b_k is the marking of place b_k and bit b'_k a guess of the marking of the preceding place. This guarantees that no agent can simulate a transition twice, and hence forces all agents to perform their simulation from a single marking. We will see later that choosing to simulate the wrong transition of a transition with the wrong predecessor bit leads to deadlocks. As for A_{p_1}, the edges of the arena A_{p_k} are defined trough sequences of transitions. However, the extra bit b'_k leads to distinguishing two sequences of transitions for each situation identified above (sequences of transitions of types 00, 11, 01, 10, 0bad) for every state A_{k,b_k}. For instance, we have two sequences of type 00, namely

$$\rho_{1,Ak} = A_{k,0} \xrightarrow{0} q^k_{1,0,t,0} \xrightarrow{-1} q^k_{2,0,t,0} \xrightarrow{+1} q^k_{3,0,t,0} \xrightarrow{0} q^k_{4,0,t,0} \xrightarrow{-2} q^k_{5,0,t,0} \xrightarrow{+2} q^k_{6,0,t,0} \xrightarrow{0} A_{k,0}$$

and

$$\rho'_{1,Ak} = A_{k,0} \xrightarrow{0} q^k_{1,0,t,1} \xrightarrow{-1} q^k_{2,0,t,1} \xrightarrow{+1} q^k_{3,0,t,1} \xrightarrow{0} q^k_{4,0,t,1} \xrightarrow{-2} q^k_{5,0,t,1} \xrightarrow{+2} q^k_{6,0,t,1} \xrightarrow{0} A_{k,0}$$

when p_k is not in the preset nor in the postset of t. Sequences $\rho_{2,Ak}, \rho_{3,Ak}, \rho_{4,Ak}$ and $\rho'_{2,Ak}, \rho'_{3,Ak}, \rho'_{4,Ak}$ are built similarly. The number of states and transitions in agent A_{p_k} is linear in the number of transitions of $\mathcal{N}$.

Last, the *transfer groups* are defined as sets containing $T_{c,A_{p_1}} = \{\{q^c_{2,t,b_c}, q^1_{1,0,t}, q^1_{1,1,t}\} \mid t \in T, b_c \in \{0, 1\}\}$ to "synchronize" the choice of a transition of the controller and the choice of the same transition by agent A_1. $T_{A_{p_1},A_{p_2}} = \{\{q^1_{3,b,t}, q^2_{1,b'=0,t,b}, q^2_{1,b'=1,t,b}\} \mid b \in \{0, 1\}, t \in T\}$ to "synchronize" the choice of a transition by agent A_{p_1} holding bit b and choice of the same transition by agent A_{p_2}. $T_{A_{p_k},A_{p_{k+1}}} = \{\{q^k_{3,b,t}, q^{k+1}_{1,b'=0,t,b}, q^2_{1,b'=1,t,b}\} \mid b \in \{0, 1\}, t \in T\}$ to "synchronize" the choice of a transition by agent A_{p_k} holding bit b and choice of the same transition by agent $A_{p_{k+1}}$. $T_{A_{p_n},A_C} = \{\{q^k_{3,b_c,t,b'}, q^C_{3,b_c,t}\} \mid b, b' \in \{0, 1\}, t \in T\} \cup \{\{q^k_{6,b,t,b'}, q^C_{6,t}\} \mid b, b' \in \{0, 1\}, t \in T\}$ to acknowledge choice of t and contents of place p_n by A_C.

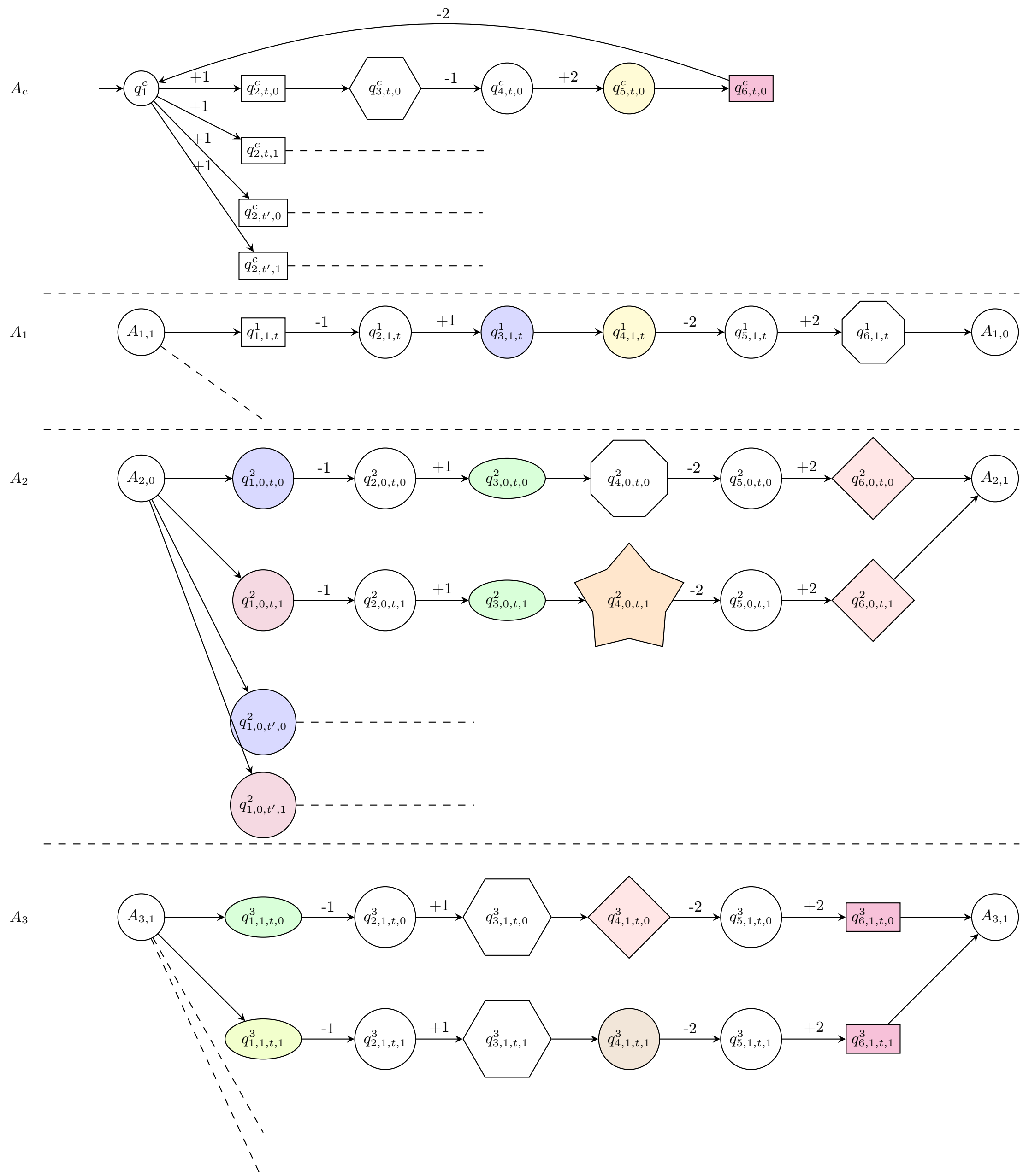

Fig. 8. Simulating a safe Petri net with a transfer system. The simulated net contains 3 places $\{p_1, p_2, p_3\}$ and two transitions $\{t, t'\}$. Transition t consumes a token from p_1, p_3 and produces a token in p_2, p_3. Most of the part of transfer system for transition t' is not represented for simplicity, hence specifying transition t' is irrelevant.

Configurations of the form $\langle (q_1^C, A_{1,b_1}, \cdot, A_{n,b_n}), 0^{n+1}\rangle$ will be called *stable* configurations (and other configurations *unstable*). Stable configurations represent an encoding of a marking of $\mathcal{N}$.

Lemma 1. *Let m be a marking of $\mathcal{N}$, and let $m[t\rangle m'$. Then the configuration $C_{m'} = \langle (q_1^C, A_{1,m'(p_1)}, \cdot, A_{n,m'(p_n)}), 0^{n+1}\rangle$ is reachable in $\mathsf{TS}_{\mathcal{N}}$ from $C_m = \langle (q_1^C, A_{1,m(p_1)}, \cdot, A_{n,m(p_n)}), 0^{n+1}\rangle$.*

Proof. Consider a subset of the system represented in Figure 6, and consider $C_m = \langle (q_1^C, A_{1,b_1=1}, A_{2,b_2=0}, \dots, A_{n,b_n}), 0^{n+1})$ as the current configuration. We will show how to simulate the execution of a transition t moving a token from place p_1 to place p_2. In C_m, all agents have an energy level of 0. Hence, agent A_C starts moving, and chooses transition t, *i.e.* moves to vertex $q^c_{2,t,b_c=m(p_n)}$ and increases its energy level to 1. Then, agents $A_{p_1}, \dots A_{p_n}$ can move to vertices $q^1_{1,t}, q^2_{0,t}, \dots q^n_{b_n=m(p_n),t}$, but are blocked in these vertices as their energy level is still 0. At this point, a transfer of one unit of energy can occur between A_C and A_{p_1} allowing A_{p_1} to move to vertex $q^1_{2,1,t}$ and immediately after to $q^1_{3,1,t}$, gaining one unit of energy. All other agents still have an energy level of 0. A_{p_2} can move to vertex $q^2_{1,0,t,1}$. Then, A_{p_1} can transfer 1 unit of energy to A_{p_2}, who can move to vertex $q^2_{2,0,t,1}$ and immediately after to $q^2_{3,0,t,1}$, get one unit of energy (again all other agents have an energy level of 0). Repeating this for all agents, the system reaches a configuration $C_{\mathtt{fwd}} = \langle (q_3^C, q^1_{3,t}, q^2_{3,0,t,1}, \dots q^n_{3,0,t,1}, (e_c, e_1, \dots e_n)\rangle$ where agent A_n is the only agent with one unit of energy. From $C_{\mathtt{fwd}}$, A_n can transfer 1 unit of energy to A_C, and unlock its move to $q^C_{4,t}$ and then reach $q^C_{4,t}$ with energy level 2. These two units of energy can be successively transferred to $A_{p_1}, \cdots A_{p_n}$ as in the preceding phase, so that the system reaches a configuration $C_{\mathtt{mov}} = (q_6^C, q^1_{6,t}, q^2_{6,0,t,1}, \dots q^n_{6,0,t,1}, 0^n.1)$ where agent A_{p_n} is the only agent with energy level 2, that they can transferred to A_C. From this new configuration, one can reach configuration $C_{m'} = \langle (q_1^C, A_{1,0}, A_{2,1}, \dots A_{n,b'_n}), 0^{n+1}\rangle$. □

Lemma 1 shows that transfer systems can simulate a safe Petri net, and that for a pair of markings $m[t\rangle m'$, there exist a "canonical" run $\rho_{m,m'}$ from stable configuration C_m to stable configuration $C_{m'}$ that does not visit any other stable configuration.

Notice however that $\rho_{m,m'}$ is not the only run from C_m to $C_{m'}$, nor the only run from C_m. Let us denote by $\rho_{m,C,t}$ the run that starts from C_m, and is composed only of successive moves of the controller that visit $q^C_{1,t}, q^C_{2,t}, q^C_{3,t}, q^C_{4,t}, q^C_{5,t}, q^C_{6,t}$ before getting back to q_1^C. Then, the concatenation of such run portions forms a legal run $\rho_{m,C,t_{i1}} \dots \rho_{m,C,t_{i1}}.\rho_{m,C,t}$ from C_m to $C_{m'}$, for every sequence of transitions $t_{i1} \dots t_{ik}$, independent of whether they are firable or not from m. However, this *stuttering* behaviour of the controller does not permit to reach other stable configurations than C_m, and can thus be ignored.

It remains to show that runs that are not sequences of canonical runs either contain a stuttering of an agent, or deadlock, and in both cases explore no stable configurations other than C_m or $C_{m'}$. To do so, we put forward the properties of configurations and transitions of the transfer system.

For a configuration $C = (q_c, q_1, \dots q_n, E)$, we will say that agent i is *committed*to the simulation of transition t with bits b_i, b'_i if its current state is of the form q^i_{s,b_i,t,b'_i}.

Starting from a configuration $C_m = \langle (q^C_1, A_{1,b_1}, \dots A_{n,b_n}), 0^{n+1}\rangle$, we can build a transition system that stores for each agent-place A_{p_i} its local state, the chosen transition that it is currently simulated, and the associated bits that memorize the marking $m(p_i)$ ansd $m(p_{i-1})$. For agent $\mathcal{A}_c$, local states will be of the form (q^c_1, e_c) or (q^c_{k,t,b_c}, e_c) where k is an integer in $[2, 6]$, b_c a bit, t a transition, e_c an integer. For agent A_{p_1}, local states will be of the form (A^1_1, b_1) or $(q^1_{k,b_1,t}, e_1)$ where k is an integer in $[1, 6]$, b_c a bit, t a transition, e_1 an integer For agents $A_{p_i}, i \in 2..n$, local states will be of the form (A^i_1, b_i) or $(q^c_{k,b_i,t,b'_i}, e_i)$ where k is an integer in $[1, 6]$, b_i, b'_i are bit, t a transition, and e_i an integer.

The definition of transfer groups in the construction of the system imposes the following constraints. A transfer necessarily occurs between agent A_{p_n} and agent A_c, of between a pair of agents $A_{p_i}, A_{p_{i+1}}, i \in 1..n-1$. Further, this transfer can occur between two agents iff they agree on the chosen transition to simulate and on the marking of the preceding place. Transfers that occur between the controller and A_{p_1} occur in transfer group $\{q^c_{2,t,j}, q^1_{1,b_1,t}\}$ and in transfer group $\{q^c_{5,t,0}, q_{5,t,1}, q^1_{4,0,t}, q^1_{4,0,t}\}$, that is if both agents have chosen the same transition. Transfers that occur between A_{p_1} and A_{p_2} occur in a transfer group of the form $\{q^1_{3,b_1,t}, q^2_{1,0,t,b_1}, q^2_{1,1,t,b_1}\}$ and in a transfer group of the form $\{q^1_{6,b_1,t}, q^2_{4,0,t,b_1}, q^2_{4,1,t,b_1}\}$, that is if A_{p_1} and A_{p_2} have chosen the same transition t, and A_{p_2} has correctly guessed $m(p_1)$. Last, transfers that occur between A_{p_i} and $A_{p_{i+1}}$ occur in a transfer group of the form $\{q^i_{3,b_1,t,b'_1}, q^{i+1}_{1,0,t,b_1}, q^{i+1}_{1,1,t,b_1}\}$ and $\{q^i_{6,b_1,t}, q^{i+1}_{4,0,t,b_1}, q^{i+1}_{4,1,t,b_1}\}$, with identical transition and correctly chosen bits.

Claim. If agents A_{p_i} and $A_{p_{i+1}}$ choose different transitions, or $A_{p_{i+1}}$ wrongly guesses p_i's marking, then the transfer system deadlocks without reaching a stable configuration.

The main principle of a run simulating a transition firing $m[t\rangle m'$ is that when all agents agree on a common transition, one unit of energy flows from A_c to A_{p_1}, $A_{p_2} \dots A_{p_n}$ and then back to A_c. A second round then starts, with two units of energy transferred successively to A_c to A_{p_1}, $A_{p_2} \dots A_{p_n}$. If two agents, say A_{p_i} and $A_{p_{i+1}}$ have done incompatible choices, then $A_{p_{i+1}}$ is blocked in state $q^i_{1,b_{i+1},t',b_{i+1}}$. On the other hand, Agent A_C is blocked in state q_3 and all other agents are blocked in state q_4.

Claim. In every stable configuration $e_c + \sum e_i = 0$

Proof. Let ρ_{C_1,C_2} be a run from a stable configuration C_1 with energy levels $E_1 = (e^1_c, e^1_1, \dots e^1_n) = 0^{n+1}$ for every agent to a stable configuration C_2 with $E_2 = (e^2_c, e^2_1, \dots e^2_n)$. Then this run can be projected on each agent $C, A_1, \dots A_n$ to get local sequences of transitions. Let $\rho^C_{C_1,C_2}$ be the projection on the controller agent. Then $\rho^C_{C_1,C_2}$ is a succession of cycles around q^C_1. Each cycle has a total weight of 0 so $\rho^C_{C_1,C_2}$ has a weight $W(\rho^C_{C_1,C_2}) = 0$. Similarly, the projection of

$\rho^C_{C_1,C_2}$ of a given agent A_i is a sequences of paths from $A_{i,b}$ to $A_{i,b'}$ of weight 0. As no extraction from the environment is performed during $\rho^C_{C_1,C_2}$, transfers just move energy from one component to another, and we have that $e_c^2 + \sum e_i^2 = W(\rho^C_{C_1,C_2}) + \sum W(\rho^i_{C_1,C_2}) = 0$. So, we also have $E_2 = 0^{n+1}$.

Claim. The total amount of energy in the system is always smaller than 2.

This can be observed by constructing the transition system, and also considering the fact that agents play sequences of weight 0, and need to receive energy from their predecessor to get through transitions of weight -2. A consequence is that, at a given instant, at most one agent is able to get through transitions of weight -2. This leads to the following claim:

Claim. Let $C = \langle (s_c, s_1, \dots s_i \dots s_n), (e_c, e_1, \dots 2, \dots e_n)\rangle$ such that $s_i = q_{6,b_i,t,b'_i}$, $s_{i+1} = q_{4,b_{i+1},t,b'_{i+1}}$. Then, if $\mathcal{A}_i$ transfers one unit of energy to $\mathcal{A}_{i+1}$ and moves to its next state of the form $A_{i,x}$, then the system deadlocks and never reaches a stable configuration.

Proof. After transfer, two agents have one unit of energy, and necessarily meet a transition of weight -2.

Claim. Let $C = \langle (s_c, s_1, \dots s_i \dots s_n), (e_c, e_1, \dots 2, \dots e_n)\rangle$ such that $s_i = q_{6,b_i,t,b'_i}$, $s_{i+1} = q_{4,b_{i+1},t,b'_{i+1}}$. Then, if A_{p_i} does not transfer energy to $A_{p_{i+1}}$ and moves to its next state of the form $A_{i,x}$, then $A_{p_{i+1}}$ stays in s_{i+1} as long as configuration C is not visited again.

In this setting, agent A_{p_i} keeps 2 energy units and can iterate choices of bits and transitions without waiting for energy from the preceding agent. It may hence visit an arbitrary number of times a local state of the form A_i, x and choose new sequences of transitions to commit to. In the meantime, other agents can perform only a bounded number of steps of weight 0, and in any case $A_{p_{i+1}}$ cannot move. Notice that no stable configuration is met, regardless of the length of the run.

Let ρ_{m_1,m_2} be the run from C_{m1} to C_{m2} shown in the proof of lemma 1. One can remarks that one requires $2.n{+}2$ transfers, using transfer groups in the order $\mathcal{A}_c \rightarrow \mathcal{A}_1\mathcal{A}_n \rightarrow \mathcal{A}_c, \mathcal{A}_1\mathcal{A}_n \rightarrow \mathcal{A}_c$. One can find equivalent runs up to permutation of some transitions of positive weight that do not change the appearance order of transfer groups, and reach C_{m2}. Following the claims above, we can say that, in any of the unstable configurations visited in these runs, if an agent does not perform the good choice of transition, of marking bit, or transfers less energy than in the considered run ρ_{m_1,m_2}, then the system either deadlocks, or enters an infinite sequence of moves that can be only exited by sending back the faulty agent to the state from which the wrong choice was performed, without visiting a stable configuration.

We can hence conclude that a marking m of the safe Petri net $\mathcal{N}$ is reachable from marking m_0 if and only if the stable configuration C_m is reachable from the configuration C_{m_0} in $\mathsf{TS}_{\mathcal{N}}$.

B Missing proof for Section 5.3

Theorem 9 ℓw-REACH is undecidable.

Proof. We provide here the constructions for the increment and zero test gadgets.
• If the command l_i increments counter $z \in \{x, y\}$, we handle it in a very similar way as the decrement (see Figure 7): the only difference is that we add another step at the bottom so that $\mathcal{A}_i$ has one too much energy when interacting with $\mathcal{A}_z$ instead of one too little. Due to the similarity, we do not detail this case.

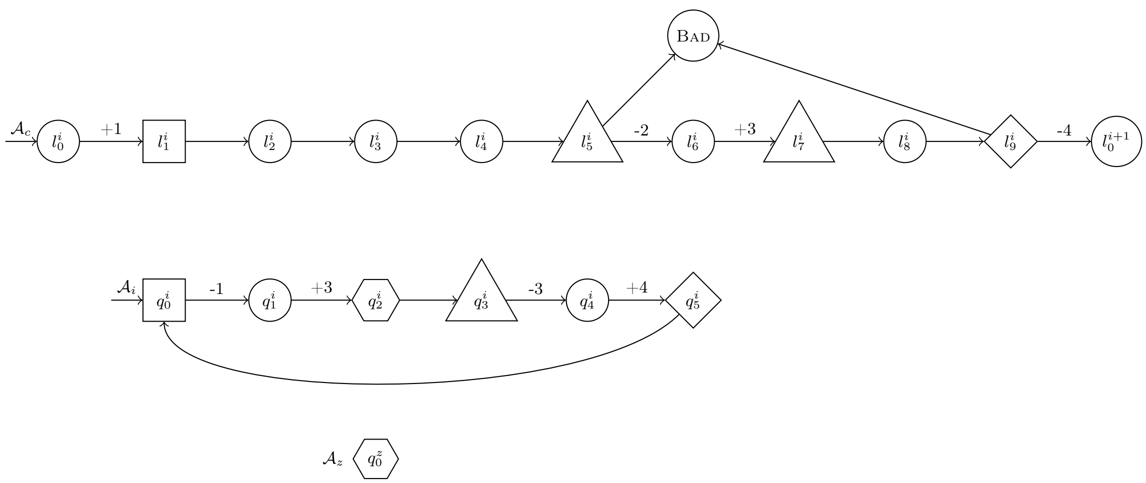

Fig. 9. Encoding an increment.

• Let us now consider the case where the command l_i tests whether the counter z is equal to 0, in which case it moves to command line l_k, and otherwise it moves to l_m. Again, in order to handle this command line, we will use three agents: the control agent $\mathcal{A}_c$, the agent associated to counter c, $\mathcal{A}_z$ and a counter dedicated to l_i, $\mathcal{A}_i$. This case is a slightly more involved, and we will in particular reuse the increment and decrement gadgets as black boxes.

The transfer system $\langle \mathcal{A}_z, \mathcal{A}_c, \mathcal{A}_i, \mathcal{T} \rangle$ is illustrated in Figure 10 and formally defined by:

– $\mathcal{A}_z = (V_z, q_0^z, E_z, w_z)$ with
  $V_z = \{q_j^z \mid j = 0 \ldots 5\} \cup \{\text{BAD}^z\}$;
  $E_z = \left\{ q_j^z \xrightarrow{w_j} l_r^z \mid j = 0 \ldots 5 \wedge r \equiv j+1 \pmod 6 \right\} \cup \{(q_5^z \xrightarrow{0} \text{BAD}^z)\}$;
  and $w_0 = 1, w_2 = -1, w_3 = 4, w_5 = -4$ and all other w_j's are equal to 0.
– $\mathcal{A}_i = (V_i, q_1^i, E_i, w_i)$ with
  $V_i = \{q_j^i \mid j = 0 \ldots 10\} \cup \{\text{BAD}^i\}$;

$E_z = \left\{q^i_j \xrightarrow{w_j} l^i_r \mid j = 0\ldots10 \wedge r \equiv j+1 \pmod{1}1\right\} \cup \{(q^i_6 \xrightarrow{0} \text{Bad}^i), (q^i_{10} \xrightarrow{0} \text{Bad}^i)\}$;
and $w_0 = -1, w_1 = 2, w_6 = -1, w_7 = 3, w_{10} = -4$ and all other w_j's are equal to 0.

– $\mathcal{A}_c = (V^c_i, l^i_0, E^c_i, w^c_i)$ with
$V^c_i = \{l^i_m \mid m = 0\ldots8\} \cup \{\text{Bad}^c, l^m_0, l^k_0,$ Decrement z,Increment z$\}$ where Decrement z and Increment z represent an entire gadget allowing to decrement or increment z;
$E_c = \left\{l^i_j \xrightarrow{w_j} l^i_{j+1} \mid j = 0\ldots7\right\} \cup \{(l^i_8 \xrightarrow{0} l^k_0), (l^i_3 \xrightarrow{0} \text{Bad}^c), (l^i_4 \xrightarrow{0} \text{Bad}^c),$ $(l^i_0 \xrightarrow{0}$Decrement z), (Decrement z$\xrightarrow{0}$Increment z), (Increment z$\xrightarrow{0} l^m_0)\}$;
and $w_0 = 1, w_3 = -2, w_4 = -1, w_5 = 1, w_6 = -2, q_7 = 4$ and all other w_j's are equal to 0.

– $\mathcal{T}$ contains the groups $\{l^i_1, q^i_0\}$, $\{l^i_3, q^i_2\}$, $\{l^i_4, q^z_1\}$, $\{l^i_6, q^i_6, q^i, 9\}$, $\{l^i_8, q^z_5\}$, $\{q^i_8, q^z_2\}$, and $\{q^i_{10}, q^z_4\}$.

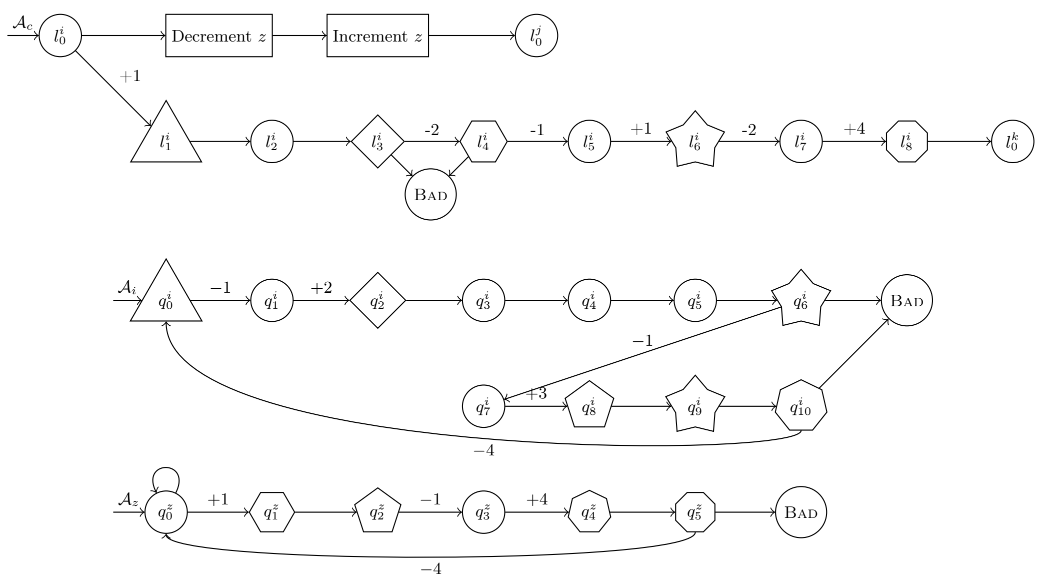

Fig. 10. Encoding a zero test.

Let us show that if the agents $\mathcal{A}_c, \mathcal{A}_i$ and $\mathcal{A}_z$ start in l^i_0, q^i_0 and q^z_0 with 0, 0 and n energy levels respectively, then the only way to avoid the Bad vertices leads them to the vertices l^m_0, q^i_0 and q^z_0 with 0, 0 and n energy units respectively if $n > 0$ and to the vertices l^k_0, q^i_0 and q^z_0 with 0, 0 and 0 energy levels respectively otherwise. Hence, executing the command line l_i indeed selects the new vertex of $\mathcal{A}_c$ depending on the energy level of $\mathcal{A}_z$.

First, consider the path from l_0^i which goes through the decrement then increment of z before reaching l_0^m. The decrement step can only be achieved without encountering a bad vertex if the energy of $\mathcal{A}_z$ is at least 1, ensuring the correction of this part of the construction. So we focus on the rest of the gadget, showing it can only be taken if the energy in $\mathcal{A}_z$ is exactly 0 at the start. As for the decrement and increment gadget, we will rely on the need to wait for the other agent to ensure the energy is low.

Let us now explain this in details. We assume $\mathcal{A}_z$ has n energy. Following the previous point, we can assume $\mathcal{A}_c$ starts by going in l_1^i, gaining 1 energy. It can either continue from this point, but without additional energy it will reach Bad from l_3^i as it cannot pay 2. So it gives 1 to q_0^i. Both then reach l_3^i and q_2^i where $\mathcal{A}_i$ must give its 2 energy so that $\mathcal{A}_c$ does not reach Bad. They then move to l_4^i and q_3^i. In l_4^i, to avoid Bad, $\mathcal{A}_c$ must receive 1 from $\mathcal{A}_z$ in q_1^z. $\mathcal{A}_z$ can pay this no matter the value of n as it just gained one energy by taking (q_0^z, q_1^z). If $n \neq 0$, $\mathcal{A}_z$ could have given more than 1 to $\mathcal{A}_c$ however. Let n_1 and n_2 be the energy amounts so that the next configuration is $((l_5^i, q_4^i, q_2^z), (n_1, 0, n_2))$. In particular, $n = n_1 + n_2$. If n_2 is at least 1, then $\mathcal{A}_z$ will be able to advance. As we will see, $\mathcal{A}_z$ will be necessary for the other agents to avoid Bad, so $n_2 = 0$ (and thus $n_1 = n$). On the next step, the agents reach $((l_6^i, q_5^i, q_2^z), (n_1 + 1, 0, 0))$. Again, if n_1 is at least 1, then $\mathcal{A}_c$ will advance on the next step, and thus $\mathcal{A}_i$ will not be able to avoid Bad from q_6^i. So $n_1 = 0$. The next configuration is thus $((l_6^i, q_6^i, q_2^z), (1, 0, 0))$ where $\mathcal{A}_c$ transfers 1 to $\mathcal{A}_i$ so that $(q_6^i \xrightarrow{-1} q_7^i)$ can be taken. The following configurations are thus $((l_6^i, q_7^i, q_2^z), (0, 0, 0))$ and then $((l_6^i, q_8^i, q_2^z), (0, 3, 0))$. There, $\mathcal{A}_i$ needs to free $\mathcal{A}_z$, or it will not have the 4 units of energy required to leave q_{10}^i while avoiding Bad. Precisely, it must give 1 to $\mathcal{A}_z$, and then 2 to $\mathcal{A}_c$ on the next step (the 4 energy $\mathcal{A}_c$ will obtain by reaching l_8^i are needed to avoid Bad in the other two agents). With those two transfers, the configuration is thus $((l_6^i, q_9^i, q_3^z), (0, 2, 0))$ and then $((l_7^i, q_{10}^i, q_4^z), (0, 0, 4))$. Then, the only configuration avoiding Bad is $((l_8^i, q_0^i, q_5^z), (4, 0, 0))$ which again offers only one option $((l_0^k, q_0^i, q_0^z), (0, 0, 0))$ which is the claimed end configuration of this gadget in the case where $\mathcal{A}_z$ initially has 0 energy.

This gadget has one specificity that must be mentionned: while every other gadget is "stuck" when $\mathcal{A}_c$ is not reading the current command line, this is not the case for $\mathcal{A}_z$ here. It can in fact loop throughout his line of vertices by itself no matter how much initial energy it has. This has no impact however as during this loop it can exchange energy only if the other agents are going through the same command line, and the entire loop does not modify its amount of energy.

□