

CONTEXT-BOUNDED ANALYSIS FOR CONCURRENT PROGRAMS WITH DYNAMIC CREATION OF THREADS

MOHAMED FAOUZI ATIG^a, AHMED BOUAJJANI^b, AND SHAZ QADEER^c

^a Uppsala University, Sweden
e-mail address: mohamed_faouzi.atig@it.uu.se

^b LIAFA, University Paris Diderot, France
e-mail address: abou@liafa.jussieu.fr

^c Microsoft Research, Redmond, WA, USA
e-mail address: qadeer@microsoft.com

ABSTRACT. Context-bounded analysis has been shown to be both efficient and effective at finding bugs in concurrent programs. According to its original definition, context-bounded analysis explores all behaviors of a concurrent program up to some fixed number of context switches between threads. This definition is inadequate for programs that create threads dynamically because bounding the number of context switches in a computation also bounds the number of threads involved in the computation. In this paper, we propose a more general definition of context-bounded analysis useful for programs with dynamic thread creation. The idea is to bound the number of context switches for each thread instead of bounding the number of switches of all threads. We consider several variants based on this new definition, and we establish decidability and complexity results for the analysis induced by them.

INTRODUCTION

The verification of multithreaded programs is a challenging problem both from the theoretical and the practical point of view. (We consider here programs with parallel threads which may use local variables as well as shared (global) variables.) Assuming that the variables of the program range over a finite domain (which can be obtained using some abstraction on the manipulated data), there are several aspects in multithreaded programs which make their analysis complex or even undecidable in general [Ram00].

Indeed, it is well known that for instance in the case where each thread can be modeled as a finite-state system, the state space of the program grows exponentially w.r.t. the number of threads, and the reachability problem is PSPACE-hard. Moreover, if threads are modeled as pushdown systems, which corresponds to allowing unbounded depth (recursive)

1998 ACM Subject Classification: D.2.4, D.3.1, F.4.3, I.2.2.

Key words and phrases: Pushdown Systems, Program Verification, Reachability Analysis.

A shorter version of this paper has been published in the Proceedings of TACAS 2009, LNCS 5505.

procedure calls in the program, then the reachability problem becomes undecidable as soon as two threads are considered.

Context-bounding has been proposed in [QR05] as a suitable technique for the analysis of multithreaded programs. The idea is to consider only the computations of the program that perform at most some fixed number of context switches between threads. (At each point only one thread is active and can modify the global variables, and a context-switch happens when the active thread terminates or is interrupted, and a pending one is activated.) The state space which must be explored may still be unbounded in presence of recursive procedure calls, but the context-bounded reachability problem is decidable even in this case. In fact, context-bounding provides a very useful tradeoff between computational complexity and verification coverage. This tradeoff is based on three important properties. First, context-bounded verification can be performed more efficiently than unbounded verification. From the complexity-theoretic point of view, it can be seen that context-bounded reachability is an NP-complete problem (even in the case of pushdown threads). Second, many concurrency errors, such as data races and atomicity violations, are manifested in executions with few context switches [MQ07]. Finally, verifying all executions of a concurrent program up to a context bound provides an intuitive and meaningful notion of coverage to the programmer.

While the concept of context-bounding is adequate for multithreaded programs with a (fixed) finite number of threads, the question we consider in this paper is whether this concept is still adequate when dynamic creation of threads is considered.

Dynamic thread creation is useful for modeling several important aspects, e.g., (1) unbounded number of concurrent executions of software modules such as file systems, device drivers, non-blocking data structures etc., or (2) creation of asynchronous activity such as forking a thread, queuing a closure to a threadpool with or without timers, callbacks, etc. Both these sources are very important for modeling operating system components; they are likely to become important even for application software as it becomes increasingly parallel in order to harness the power of multi-core architectures.

We argue that the “classical” notion of context-bounding which has been used so far in the existing work is actually too restrictive in this case. Indeed, bounding the number of context switches in a computation also bounds the number of threads involved. In this paper, we propose a more general definition of context-bounded analysis useful for programs with dynamic thread creation. The idea is to bound the number of context switches for each thread instead of bounding the number of switches of all threads. We consider several variants based on this new definition, and we establish decidability and complexity results for the analysis induced by them.

We introduce a notion of K -bounded computations where each of the involved threads can be interrupted and resumed at most K times. In fact, we consider that when a thread is created, the number of context switches it can perform is the one of its ancestor (at the moment of the creation) minus 1. Notice that the number of context switches by all threads in a computation is not bounded since the number of threads involved is not bounded.

In the case of finite-state threads, we prove that this problem is as hard as the coverability problem for vector addition systems with states (or, Petri nets) (which is EXPSPACE-complete). The reduction from our problem to the coverability problem of vector addition systems with states is based on the simple idea of counting the number of pending threads for different values of the global and local states, as well as of the number of switches that

these threads are allowed to perform. Conversely, we prove that the coverability problem of vector addition systems with states can be reduced to the 2-bounded reachability problem. These results show that in the case of dynamic thread creation, considering the notion of context-bounding for each individual thread makes the complexity jump from NP-completeness to EXPSPACE-completeness, even in the case of finite-state threads. Then, an interesting question is whether it is possible to have a notion of context-bounding with a lower complexity. We propose for that the notion of stratified context-bounding. The idea is to consider computations where the scheduling of the threads is ordered according to their number of allowed switches: First, threads of level K (the level means here the number of allowed switches) are scheduled generating threads of level $K - 1$, then threads of level $K - 1$ are scheduled, and so on. Again, notice that K -stratified computations may have an unbounded number of context switches since it is possible to schedule an unbounded number of threads at each level. This concept generalizes obviously the “classical” notion of context-bounding. We prove that, for finite-state threads, the K -stratified context-bounded reachability problem is NP-complete (i.e., it matches the complexity of the “classical” context-bounded reachability problem). The proof is by a reduction to the satisfiability problem of existential Presburger formulas.

Then, we consider the case of dynamic creation of pushdown threads. We prove that, surprisingly, the K -bounded reachability problem is in fact decidable, and that the same holds also for the K -stratified context-bounded reachability problem. To establish these results, we prove that these problems (for pushdown threads) can be reduced to their corresponding problems for finite-state threads. This reduction is not trivial. The main ideas behind the reduction are as follows: First, the K -bounded behaviors of each single thread can be represented by a labeled pushdown system which (1) makes visible (as labels) on its transitions the created threads, and (2) guesses points of interruption-resumption and the corresponding values of the global states. (These guesses are also made visible on the transitions.) Then, the main problem is to “synchronize” these labeled pushdown systems so that their guesses can be validated. The key observation is that it is possible to abstract these systems without loss of preciseness by finite-state systems. This is due to the fact that we can consider that some of the generated threads can be lost (since they can be seen as threads that are never activated), and therefore we can reason about the downward closure of the languages of the labeled pushdown systems mentioned above (w.r.t. suitable sub-word relation). This downward closure is in fact always regular and effectively constructible.

Related work. In the last few years, several implementations and algorithmic improvements have been proposed for context-bounded verification [BESS05, MQ07, SES08, LTKR08, LR08, LMP09]. For instance, context-bounded verification has been implemented in explicit-state model checkers such as CHESS [MQ07] and SPIN [ZJ08]; it has also been implemented in symbolic model checkers such as SLAM [QW04], jMoped [SES08], and in [LR08]. In this paper, we propose more general definitions of context-bounded analysis useful for programs with dynamic thread creation.

Several models based on rewriting systems or networks of pushdown systems have been considered to model multithreaded programs [LS98, EP00, SS00, Mo02, BT03, BT05]. While these models allow to model dynamic thread creation, they only allow communication between processes in a very restrictive way.

In [BMOT05], a model based on networks of pushdown systems called CDPN was proposed. While this model allows dynamic creation of processes, it allows only a restricted form of synchronization where a process has the right to read only the control states of its immediate children (i.e., the processes it has created).

A symbolic algorithm for over-approximating reachability in Boolean programs with unboundedly many threads was given in [CKS06, CKS07]. Our approach complements these techniques since they are able to prove that a safety property of interest holds. While our work is useful for effectively detecting bad behaviors of the analyzed programs.

A recent paper proposes an algorithm for the verification problem for parametrized concurrent programs with procedural calls under a k-round-robin schedule [LMP10]. Our work is more powerful than this framework as long as the data domain is bounded.

1. PRELIMINARY DEFINITIONS AND NOTATIONS

In this section, we introduce some basic definitions and notations that will be used in the rest of the paper.

1.1. Integers, functions, and vectors.

Integers. Let \mathbb{Z} be the set of integers and \mathbb{N} be the set of positive integers (or natural numbers). For every $i, j \in \mathbb{Z}$ such that $i \leq j$, we use $[i, j]$ and $[i, j[$ to denote respectively the sets $\{k \in \mathbb{Z} \mid i \leq k \leq j\}$ and $\{k \in \mathbb{Z} \mid i \leq k < j\}$.

Functions. Let A and B be two sets. We denote by $[A \rightarrow B]$ the set of all functions from A to B . If f, g are two functions from A to \mathbb{N} , then we write $g \leq f$ if and only if $g(a) \leq f(a)$ for all $a \in A$. We use $f + g$ (resp. $f - g$ if $g \leq f$) to denote the function from A to \mathbb{N} defined as follows: $(f + g)(a) = f(a) + g(a)$ (resp. $(f - g)(a) = f(a) - g(a)$) for all $a \in A$. For every subset $C \subseteq A$, we use ld_A^C to denote the function from A to \mathbb{N} defined as follows:

$$\text{ld}_A^C(a) = \begin{cases} 1 & \text{if } a \in C \\ 0 & \text{if } a \in (A \setminus C) \end{cases} \quad (1.1)$$

In particular, ld_A^\emptyset denotes the function that maps any element of A to 0.

Vectors. Let n be a natural number and A be a set. An n -dim vector \mathbf{v} over A is an element of A^n . For every $i \in [1, n]$, we denote by $\mathbf{v}[i] \in A$ the i^{th} component of \mathbf{v} . Given $j \in [1, n]$ and $a \in A$, we denote by $\mathbf{v}[j \leftrightarrow a]$ the n -dim vector \mathbf{v}' over A such that $\mathbf{v}'[j] = a$ and $\mathbf{v}'[k] = \mathbf{v}[k]$ for all $k \in [1, n]$ and $k \neq j$.

Vectors of integers. The order relation \leq between integers is generalized in a pointwise manner to vectors of integers. We write $\mathbf{0}^n$ to denote the n -dim vector \mathbf{v} over \mathbb{Z} such that $\mathbf{v}[i] = 0$ for all $i \in [1, n]$. We trivially extend the addition and subtraction operations over integers to vectors of integers.

1.2. Words and languages. Given a finite set Σ called an alphabet and whose elements are called letters or symbols, a word u over Σ is either a finite sequence of letters in Σ or the empty word ϵ . The length of u is denoted by $|u|$. (We assume that $|\epsilon| = 0$.) For every $a \in \Sigma$, we use $|u|_a$ to denote the number of occurrences of a in u . For every $j \in [1, |u|]$, we use $u(j)$ to denote the j^{th} letter of u .

A language L over Σ is a (possibly infinite) set of words over Σ . We adopt the widespread notations Σ^* and Σ^+ to represent respectively the languages containing all words and all non-empty words over Σ . We use also Σ_ϵ to denote the set $\Sigma \cup \{\epsilon\}$.

We denote by $\preceq \subseteq \Sigma^* \times \Sigma^*$ the *subword relation* defined as follows: For every $u, v \in \Sigma^*$, $u \preceq v$ if and only if: (1) $u = \epsilon$, or (2) there are $i_1, i_2, \dots, i_{|u|} \in [1, |v|]$ such that $i_1 < i_2 < \dots < i_{|u|}$ and $u(j) = v(i_j)$ for all $j \in [1, |u|]$. Given a language $L \subseteq \Sigma^*$, the *downward closure* of L is the language $L \downarrow = \{u \in \Sigma^* \mid \exists v \in L, u \preceq v\}$.

Let Θ be a subset of Σ . Given a word $u \in \Sigma^*$, we denote by $u|_\Theta$ the projection of u over Θ , i.e., the word obtained from u by erasing all the symbols that are not in Θ . This definition is extended to languages as follows: If L is a language over Σ , then $L|_\Theta = \{u|_\Theta \mid u \in L\}$.

The *Parikh image* of a word $u \in \Sigma^*$ is a function from Σ to \mathbb{N} such that: For every $a \in \Sigma$, $\text{Parikh}(u)(a) = |u|_a$. Accordingly, the Parikh image of a language $L \subseteq \Sigma^*$, written $\text{Parikh}(L)$, is the set of Parikh images of $u \in L$.

Let Σ_1 and Σ_2 be two alphabets. A homomorphism h is a function from Σ_1^* to Σ_2^* such that $h(\epsilon) = \epsilon$ and $h(uv) = h(u)h(v)$ for all $u, v \in \Sigma_1^*$. By definition, the homomorphism h is completely characterized by the function $f_h : \Sigma_1 \rightarrow \Sigma_2^*$ s.t. for any $a \in \Sigma_1$, $f_h(a) = h(a)$.

1.3. Transition systems. A transition system is a triplet $\mathcal{T} = (C, \Sigma, \rightarrow)$ where: (1) C is a (possibly infinite) set of configurations (also called states), (2) Σ is a finite set of labels (or actions), and (3) $\rightarrow \subseteq C \times \Sigma_\epsilon \times C$ is a transition relation.

Given two configurations $c, c' \in C$ and an action $a \in \Sigma$, we write $c \xrightarrow{a} c'$ if $(c, a, c') \in \rightarrow$. A finite run ρ of \mathcal{T} from c to c' is a finite sequence $c_0 a_1 c_1 a_2 \dots a_n c_n$, for some $n \geq 1$, such that: (1) $c_0 = c$ and $c_n = c'$, and (2) $c_i \xrightarrow{a_{i+1}} c_{i+1}$ for all $i \in [0, n[$. In this case, we say that ρ has length n and is labelled by the word $a_1 a_2 \dots a_n$.

Let $u \in \Sigma^*$ be an input word. We write $c \xrightarrow[u]{u} c'$ if one of the following two cases holds: (1) $n = 0$, $c = c'$, and $u = \epsilon$, and (2) there is a run ρ of length n from c to c' labelled by u . We also write $c \xrightarrow[u]{*} c'$ to denote that $c \xrightarrow[u]{u} c'$ for some $n \geq 0$. Finally, for every $C_1, C_2 \subseteq C$, we have $\text{Traces}_{\mathcal{T}}(C_1, C_2) = \{u \in \Sigma^* \mid \exists (c_1, c_2) \in C_1 \times C_2, c_1 \xrightarrow[u]{*} c_2\}$.

1.4. Finite state automata. A finite state automaton (FSA for short) is a quintuple $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ where: (1) Q is the finite non-empty set of states, (2) Σ is the finite set of input symbols (called also the input alphabet), (3) $\Delta \subseteq (Q \times \Sigma_\epsilon \times Q)$ is the transition relation, (4) $I \subseteq Q$ is the set of initial states, and (5) $F \subseteq Q$ is the set of final states. We use $q \xrightarrow{a} q'$ to denote that (q, a, q') is in Δ .

The size of \mathcal{A} , denoted by $|\mathcal{A}|$, is defined by $(|Q| + |\Sigma|)$. We denote by $\mathcal{T}(\mathcal{A}) = (Q, \Sigma, \Delta)$ the transition system associated to \mathcal{A} . The language accepted (or recognized) by \mathcal{A} is defined as follows $L(\mathcal{A}) = \text{Traces}_{\mathcal{T}(\mathcal{A})}(I, F)$.

It is well known that the class of languages accepted by finite state automata (the class of rational (or regular) languages) is effectively closed under union, intersection, homomorphism, and projection operations [HU79].

1.5. Pushdown automata. A pushdown automaton (PDA for short) is a 7-tuple $\mathcal{P} = (P, \Sigma, \Gamma, \Delta, p_0, \gamma_0, F)$ where:

- P is the finite non-empty set of states,
- Σ is the finite set of input symbols (called also the input alphabet),
- Γ is the finite set of stack symbols (called also the stack alphabet),
- $\Delta \subseteq ((P \times \Gamma) \times \Sigma_\epsilon \times (P \times \Gamma^{\leq 2}))$ is the transition relation (where $\Gamma^{\leq 2} = \Gamma_\epsilon \cup \Gamma^2$).
- $p_0 \in P$ is the initial state,
- $\gamma_0 \in \Gamma$ is the initial stack symbol, and
- $F \subseteq P$ is the set of final states.

The size of \mathcal{P} , denoted by $|\mathcal{P}|$, is defined as $(|P| + |\Sigma| + |\Gamma|)$. We use $\langle p, \gamma \rangle \xrightarrow{a}_{\mathcal{P}} \langle p', u \rangle$ to denote that $((p, \gamma), a, (p', u))$ is in Δ .

A configuration of \mathcal{P} is a pair (p, w) where $p \in P$ and $w \in \Gamma^*$. The set of all configurations of \mathcal{P} is denoted by $\text{Conf}(\mathcal{P})$. The transition system associated to \mathcal{P} , denoted by $\mathcal{T}(\mathcal{P})$, is given by the tuple $(\text{Conf}(\mathcal{P}), \Sigma, \rightarrow)$ where \rightarrow is the smallest transition relation such that: if $\langle p, \gamma \rangle \xrightarrow{a}_{\mathcal{P}} \langle p', u \rangle$, then $(p, \gamma w) \xrightarrow{a}_{\mathcal{T}(\mathcal{P})} (p', uw)$ for all $w \in \Gamma^*$. The language of \mathcal{P} is defined as follows $L(\mathcal{P}) = \text{Traces}_{\mathcal{T}(\mathcal{P})}(\{(p_0, \gamma_0)\}, F \times \Gamma^*)$.

It is well known that the class of context-free languages (i.e., accepted by pushdown automata) are closed under concatenation, union, Kleene star, homomorphism, projection, and intersection with a rational language. However, context-free languages are not closed under complement and intersection [HU79].

Let us recall now that the downward closure of a context-free language, with respect to the subword relation, is effectively a rational language.

Theorem 1.1 ([Cou91]). *If \mathcal{P} is a PDA, then, it is possible to construct, in time and space exponential in $|\mathcal{P}|$, a finite state automaton \mathcal{A} such that $L(\mathcal{A}) = L(\mathcal{P}) \downarrow$ and the size of $|\mathcal{A}|$ is exponential in $|\mathcal{P}|$ in the worst case.*

We can prove that the exponential blow-up in Theorem 1.1 can not be avoided. This is due to the fact that pushdown automata are more succinct than finite state automata. To show that, let us consider the following pushdown automaton $\mathcal{P} = (\{p_0, p_1, p_2\}, \{a\}, \{\perp, \gamma_0, \dots, \gamma_n\}, \Delta, p_0, \perp, \{p_2\})$ where $n \in \mathbb{N}$ and Δ is the transition relation composed from the following transitions:

- (1) $\langle p_0, \perp \rangle \xrightarrow{\epsilon}_{\mathcal{P}} \langle p_1, \gamma_0 \perp \rangle$,
- (2) for every $i \in [0, n[$, $\langle p_1, \gamma_i \rangle \xrightarrow{\epsilon}_{\mathcal{P}} \langle p_1, \gamma_{i+1} \gamma_{i+1} \rangle$,
- (3) $\langle p_1, \gamma_n \rangle \xrightarrow{a}_{\mathcal{P}} \langle p_1, \epsilon \rangle$, and
- (4) $\langle p_1, \perp \rangle \xrightarrow{\epsilon}_{\mathcal{P}} \langle p_2, \epsilon \rangle$.

It is easy to observe that $L(\mathcal{P}) = \{a^{2^n}\}$ and therefore the minimal finite state automaton \mathcal{A} recognizing $L(\mathcal{P}) \downarrow$ has at least 2^n states whereas the size of \mathcal{P} is $(n + 5)$.

2. DYNAMIC NETWORK OF CONCURRENT PUSHDOWN SYSTEMS

In this section, we introduce dynamic network of concurrent pushdown systems. Intuitively, a dynamic network of concurrent pushdown systems \mathcal{M} models dynamic multithreaded programs with (potentially) recursive procedure calls. Threads are modeled as pushdown processes which may spawn new threads (or processes). Each thread may have its local variables and has also access to global variables. The values of local variables are modeled using the stack alphabet Γ , whereas the values of the global variables are modeled using a

finite non-empty set of states Q . Transitions of the form $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \epsilon$ correspond to standard transitions of pushdown systems (popping γ and then pushing u while changing the state from q to q'). Transitions of the form $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \gamma'$ correspond to standard transitions of pushdown systems with a creation of a thread whose initial stack content is $\gamma' \in \Gamma$. Transitions of the form $\langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$ correspond to interrupt the execution of the active thread after the performing the standard pushdown operations, and transitions of the form $q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$ correspond to start/resume the execution of a pending thread with topmost stack symbol $\gamma' \in \Gamma$ after changing the state from q to q' .

2.1. Syntax.

Definition 2.1 (DCPS). A dynamic network of concurrent pushdown system (DCPS for short) is a tuple $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$ where:

- Q is the finite non-empty set of states,
- Γ is a finite set of stack symbols (called also stack alphabet),
- $\Delta = \Delta_{\text{cr}} \cup \Delta_{\text{in}} \cup \Delta_{\text{rs}}$ where:
 - $\Delta_{\text{cr}} \subseteq ((Q \times \Gamma) \times (Q \times \Gamma^{\leq 2}) \times \Gamma_{\epsilon})$ is a finite set of (*creation*) transitions.
 - $\Delta_{\text{in}} \subseteq ((Q \times \Gamma) \times (Q \times \Gamma^{\leq 2}))$ is a finite set of (*interruption*) transitions.
 - $\Delta_{\text{rs}} \subseteq (Q \times \Gamma \times Q)$ is a finite set of (*resumption*) transitions.
- q_0 is the initial state, and
- γ_0 is the initial stack symbol.

In the rest of the paper, we adopt the following notations: (1) $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \alpha$ to denote that $((q, \gamma), (q', u), \alpha) \in \Delta_{\text{cr}}$, (2) $\langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$ to denote that $((q, \gamma), (q', u)) \in \Delta_{\text{in}}$, and (3) $q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$ to denote that $(q, \gamma, q') \in \Delta_{\text{rs}}$. The size of \mathcal{M} is given by $|\mathcal{M}| = |Q| + |\Gamma|$.

When unbounded recursion is not considered, threads can be modeled as finite state processes instead of pushdown systems. This corresponds to the special case where, for all $((q, \gamma), (q', u), \alpha) \in \Delta_{\text{cr}}$ and $((q, \gamma), (q', u)) \in \Delta_{\text{in}}$, the pushed word u is of length at most 1.

Definition 2.2 (DCFS). A dynamic concurrent finite-state systems (DCFS for short) is a DCPS $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$ where, for all $((q, \gamma), (q', u), \alpha) \in \Delta$ and $((q, \gamma), (q', u)) \in \Delta$, we have $|u| \leq 1$.

2.2. Semantics.

Definition 2.3 (Local configurations of a DCPS). Let $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$ be a DCPS. A local configuration of a thread of \mathcal{M} is a pair (w, i) where $w \in \Gamma^*$ is its call stack and $i \in \mathbb{N}$ is its switch number. Let $Loc(\mathcal{M})$ denote the set of local configurations of \mathcal{M} .

Intuitively, the switch number of a thread is the number of interruptions/resumptions together with the switch number of its creator (at the moment of the creation) plus one.

Definition 2.4 (Configurations of a DCPS). Let $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$ be a DCPS. A configuration c of a \mathcal{M} is an element of $Q \times (Loc(\mathcal{M}) \cup \{\perp\}) \times [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$. We use $Conf(\mathcal{M})$ to denote the set of all configurations of \mathcal{M} .

A configuration of the form $(q, (w, i), Val)$ (resp. (q, \perp, Val)) of \mathcal{M} means that: (1) $q \in Q$ is the value of the global store, (2) (w, i) is the local configuration of the active thread (resp. there is no active thread), and (3) $Val : Loc(\mathcal{M}) \rightarrow \mathbb{N}$ is a function that associates for each $(w', i') \in Loc(\mathcal{M})$, the number of pending threads with local configuration (w', i') .

Given a configuration $c = (q, \eta, Val) \in Conf(\mathcal{M})$, let $State(c) = q$, $Active(c) = \eta$, and $Idle(c) = Val$. We use $c_{\mathcal{M}}^{\text{init}} = (q_0, \perp, \text{ld}_{Loc(\mathcal{M})}^{\{(\gamma_0, 0)\}})$ to denote the initial configuration of \mathcal{M} .

Definition 2.5 (Transition system of a DCPS). Let $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$ be a DCPS. The transition system associated with \mathcal{M} is given by $\mathcal{T}(\mathcal{M}) = (Conf(\mathcal{M}), \Sigma, \rightarrow)$ where $\Sigma = \Delta$ and \rightarrow is the smallest relation such that:

- if $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \alpha$, then $(q, (\gamma w, i), Val) \xrightarrow{t}_{\mathcal{T}(\mathcal{M})} (q', (uw, i), Val')$ for all $w \in \Gamma^*$, $i \in \mathbb{N}$, and $Val, Val' \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ such that:
 - If $\alpha \in \Gamma$, then $Val' = Val + \text{ld}_{Loc(\mathcal{M})}^{\{(\alpha, i+1)\}}$.
 - If $\alpha = \epsilon$, then $Val' = Val$.
- if $t = \langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$, then $(q, (\gamma w, i), Val) \xrightarrow{t}_{\mathcal{T}(\mathcal{M})} (q', \perp, Val + \text{ld}_{Loc(\mathcal{M})}^{\{(uw, i+1)\}})$ for all $w \in \Gamma^*$, $i \in \mathbb{N}$, and $Val \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$.
- if $t = q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$, then $(q, \perp, Val + \text{ld}_{Loc(\mathcal{M})}^{\{(\gamma w, i)\}}) \xrightarrow{t}_{\mathcal{T}(\mathcal{M})} (q', (\gamma w, i), Val)$ for all $w \in \Gamma^*$, $i \in \mathbb{N}$, and $Val \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$.

where for every sets A and C such that $C \subseteq A$, ld_A^C denotes the function from A to \mathbb{N} such that $\text{ld}_A^C = 1$ if $a \in C$ and $\text{ld}_A^C(a) = 0$ if $a \in (A \setminus C)$ (see Equation. 1.1).

The transition $(q, (\gamma w, i), Val) \xrightarrow{t}_{\mathcal{T}(\mathcal{M})} (q', (uw, i), Val')$, with $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \alpha$, corresponds to the execution of pushdown operation (pop or push) with the possibility of a creation of a new thread (if $\alpha \in \Gamma$) which is added to the set of pending threads. The created thread gets the switch number $i + 1$. The transition $(q, (\gamma w, i), Val) \xrightarrow{t}_{\mathcal{T}(\mathcal{M})} (q', \perp, Val')$, with $t = \langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$, corresponds to interrupt the execution of the current active thread after performing the pushdown operation: The local configuration (uw, i) of the active thread is added to the set of the idle threads after incrementing its switch number. The transition $(q, \perp, Val) \xrightarrow{t}_{\mathcal{T}(\mathcal{M})} (q', (\gamma w, i), Val')$, with $t = q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$, corresponds to start/resume (from the state q') the execution of a pending thread with local configuration $(\gamma w, i)$.

2.3. Bounded semantics. Let $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$ be a DCPS. For every $I \subseteq \mathbb{N}$, let $Conf_I(\mathcal{M})$ denote the set of configurations of \mathcal{M} such that $c \in Conf_I(\mathcal{M})$ if and only if $Active(c) \in \Gamma^* \times I$. In the following, we restrict the behavior of $\mathcal{T}(\mathcal{M})$ to the set of runs where the switch numbers of the active threads are always in I .

Definition 2.6 (Bounded transition system of a DCPS). For every $I \subseteq \mathbb{N}$, $\mathcal{T}_I(\mathcal{M})$ denotes the transition system $(Conf(\mathcal{M}), \Delta, \rightarrow_I)$ where: For every $c, c' \in Conf(\mathcal{M})$, $c \xrightarrow{t}_{\mathcal{T}_I(\mathcal{M})} c'$ if and only if: (1) $c \xrightarrow{t}_{\mathcal{T}(\mathcal{M})} c'$, and (2) $c \in Conf_I(\mathcal{M})$ or $c' \in Conf_I(\mathcal{M})$.

2.4. Reachability problems. Let $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$ be a DCPS. We consider the following three notions of reachability:

Definition 2.7 (The state reachability problem). A state $q \in Q$ is reachable by \mathcal{M} if and only if there are $c \in \text{Conf}(\mathcal{M})$ and $\tau \in \Delta^*$ such that $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{M})} c$, $\text{Active}(c) = \perp$, and $\text{State}(c) = q$. The state reachability (SR for short) problem for \mathcal{M} consists in deciding, for a given set $F \subseteq Q$, whether there is a state $q \in F$ such that q is reachable by \mathcal{M} .

Notice that we consider, in the definition of the state reachability problem, that the set of reachable configurations that we are interested in are those with no active thread. This is only for the sake of simplicity and does not constitute at all a restriction. Indeed, we can show that the problem of checking whether there are $c \in \text{Conf}(\mathcal{M})$ and $\tau \in \Delta^*$ such that $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{M})} c$ and $\text{State}(c) \in F$ can be reduced to the state reachability problem for a DCPS $\mathcal{M}' = (Q, \Gamma, \Delta', q_0, \gamma_0)$ built up from \mathcal{M} by adding to Δ some transition rules that interrupt the execution of the active thread when the current state is in F .

Definition 2.8 (The k -bounded state reachability problem). Let $k \in \mathbb{N}$. A state $q \in Q$ is k -bounded reachable by \mathcal{M} if and only if there are $c \in \text{Conf}(\mathcal{M})$ and $\tau \in \Delta^*$ such that $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}_{[0,k]}(\mathcal{M})} c$, $\text{Active}(c) = \perp$, and $\text{State}(c) = q$. The k -bounded state reachability (BSR[k] for short) problem for \mathcal{M} consists in deciding, for a given set $F \subseteq Q$, whether there is a state $q \in F$ such that q is k -bounded reachable by \mathcal{M} .

Observe that, in BSR[k] problem, a bound $k + 1$ is imposed on the number of switches (interruptions/resumptions) performed by each thread (together with the switch number of its ancestor (at the moment of its creation) plus one). However, due to dynamic creation of threads, bounding the number of switches of each thread does not bound the number of switches in the whole computation of the system (since an arbitrary large number of threads can be involved in these computations).

Definition 2.9 (The k -stratified state reachability problem). Let $k \in \mathbb{N}$. A state $q \in Q$ is k -stratified reachable by \mathcal{M} if and only if there are $\tau_0, \tau_1, \dots, \tau_k \in \Delta^*$, and $c_1, \dots, c_{k+1} \in \text{Conf}(\mathcal{M})$ such that $\text{State}(c_{k+1}) = q$, $\text{Active}(c_{k+1}) = \perp$, and we have:

$$c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau_0}^*_{\mathcal{T}_{\{0\}}(\mathcal{M})} c_1 \xrightarrow{\tau_1}^*_{\mathcal{T}_{\{1\}}(\mathcal{M})} \dots \xrightarrow{\tau_{k-1}}^*_{\mathcal{T}_{\{k-1\}}(\mathcal{M})} c_k \xrightarrow{\tau_k}^*_{\mathcal{T}_{\{k\}}(\mathcal{M})} c_{k+1}$$

The k -stratified state reachability (SSR[k] for short) problem for \mathcal{M} consists in deciding, for a given $F \subseteq Q$, whether there is a state $q \in F$ s.t. q is k -stratified reachable by \mathcal{M} .

In the SSR[k] problem, a special kind of k -bounded computations (called stratified computations) are considered: In such a computation, threads are scheduled according to their increasing switch number (from 0 to k): First, threads with switch number 0 are scheduled generating threads with switch number 1, then threads with switch number 1 are scheduled generating threads with switch number 2, and so on.

Observe that even in the case of stratified computations, an arbitrarily large number of context switches may occur along a computation due to dynamic creation of threads. Very particular stratified computations are those where the whole number of context switches is bounded [QR05].

3. THE SR PROBLEM AND THE BSR[k] PROBLEM FOR DCFSS

In the following, we show that the SR problem and the BSR[k] problem for dynamic networks of concurrent finite-state systems are as hard as the coverability problem for vector addition systems with states (which is EXPSPACE-complete).

Theorem 3.1. *The SR problem and the BSR[k] problem, with $k \geq 2$, for DCFSS are EXPSPACE-complete.*

Next, we recall some basic definitions and notations about vector addition systems with states (or equivalently, Petri nets). Then, this proof of Theorem 3.1 is structured as follows: First, we show that the BSR[k] problem for DCFSS is polynomially reducible to the SR problem for DCFSS (Proposition 3.4). Then, we show that the SR problem for DCFSS is polynomially reducible to the coverability problem for VASSs (Proposition 3.6). Finally, we prove that the coverability problem for VASSs is polynomially reducible to the BSR[2] problem for DCFSS (Proposition 3.8). As an immediate consequence of these results and Theorem 3.2, we obtain that the SR problem and the BSR[k] problem for DCFSS are EXPSPACE-complete.

3.1. Vector addition systems with states. A vector addition system with states (VASS for short) is a tuple $\mathcal{V} = (n, Q, \Sigma, \delta, q_0, \mathbf{u}_0)$ where:

- $n \in \mathbb{N}$ is the dimension,
- Q is the finite non-empty set of states,
- Σ is the finite set of actions (or labels),
- $\delta : Q \times \Sigma \rightarrow Q \times ([-1, 1])^n$ is the displacement function,
- $q_0 \in Q$ is the initial state, and
- \mathbf{u}_0 is the initial n -dim vector over \mathbb{N} such that $0 \leq \mathbf{u}_0(i) \leq 1$ for all $i \in [1, n]$.

The size of \mathcal{V} , denoted by $|V|$, is defined as $(n + |Q| + |\Sigma|)$. A configuration of \mathcal{V} is a pair (q, \mathbf{u}) where $q \in Q$ and $\mathbf{u} \in \mathbb{N}^n$. Given a configuration $c = (q, \mathbf{u})$, we let $State(c) = q$ and $Val(c) = \mathbf{u}$. The set of all configurations of \mathcal{V} is denoted by $Conf(\mathcal{V})$.

The transition system associated to \mathcal{V} , denoted by $\mathcal{T}(\mathcal{V})$, is given by $(Conf(\mathcal{V}), \Sigma, \rightarrow)$, where \rightarrow is the smallest transition relation satisfying the following condition: For every $q_1, q_2 \in Q$ and $\mathbf{u}_1, \mathbf{u}_2 \in \mathbb{N}^n$, $(q_1, \mathbf{u}_1) \xrightarrow{a} \mathcal{T}(\mathcal{V})(q_2, \mathbf{u}_2)$ if and only if $\delta((q_1, a)) = (q_2, \mathbf{u}_2 - \mathbf{u}_1)$.

A state $q \in Q$ is reachable by \mathcal{V} if and only if there are $w \in \Sigma^*$ and $c \in Conf(\mathcal{V})$ such that $(q_0, \mathbf{u}_0) \xrightarrow{w}^* \mathcal{T}(\mathcal{V}) c$ and $State(c) = q$. The coverability problem for \mathcal{V} consists in deciding, for a given set $F \subseteq Q$, whether there is $q \in F$ such that q is reachable by \mathcal{V} .

Theorem 3.2 ([Lip76, Rac78]). *The coverability problem for vector addition systems with states is EXPSPACE-complete.*

3.2. From the BSR[k] problem for DCFSSs to the SR problem for DCFSSs. In the following, we show that, for every $k \in \mathbb{N}$, the BSR[k] for DCFSSs is polynomially reducible to the SR problem for DCFSSs. Intuitively, given a DCFS $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$ and a natural number k , we construct a DCFS \mathcal{M}' that records for each thread its switch number and can execute only threads with recorded switch number less than k . Formally, the DCFS $\mathcal{M}' = (Q', \Gamma', \Delta', q'_0, \gamma'_0)$ is defined as follows:

- $Q' = Q$ is a finite set of states,

- $\Gamma' = \Gamma_\epsilon \times [0, k + 1]$ is a finite set of stack symbols. A stack symbol (α, i) corresponds to a thread with stack content α and switch number i .
- Δ' is the smallest transition relation satisfying the following conditions:
 - For every $i \in [0, k]$ and $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \epsilon$, then $\langle q, (\gamma, i) \rangle \rightarrow_{\mathcal{M}'} \langle q', (u, i) \rangle \triangleright \epsilon$.
 - For every $i \in [0, k]$ and $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \alpha$ for some stack symbol $\alpha \in \Gamma$, then $\langle q, (\gamma, i) \rangle \rightarrow_{\mathcal{M}'} \langle q', (u, i) \rangle \triangleright (\alpha, i + 1)$.
 - For every $i \in [0, k]$ and $\langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$, then $\langle q, (\gamma, i) \rangle \mapsto_{\mathcal{M}'} \langle q', (u, i + 1) \rangle$.
 - For every $i \in [0, k]$ and $q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$, then $q \mapsto_{\mathcal{M}'} q' \triangleleft (\gamma, i)$.
- $q'_0 = q_0$ is the initial state, and
- $\gamma'_0 = (\gamma_0, 0)$ is the initial stack symbol.

Observe that the size of the DCFS \mathcal{M}' is polynomial in the size of \mathcal{M} . Moreover, the relation between \mathcal{M} and \mathcal{M}' is given by the following lemma:

Lemma 3.3. *Let $q \in Q$. q is k -bounded reachable by \mathcal{M} iff q is reachable by \mathcal{M}' .*

The proof of Lemma 3.3 is done by induction on the length of the runs and is given in Appendix A.

As an immediate consequence of Lemma 3.3, we obtain the following result:

Proposition 3.4. *Let $k \geq 1$. The BSR[k] problem for DCFSs is polynomially reducible to the SR problem for DCFSs.*

3.3. From the SR problem for DCFSs to the coverability problem for VASSs.

In the following, we show that the SR problem for DCFSs is polynomially reducible to the coverability problem for VASSs. For a given DCFS $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$, with $\Gamma = \{\gamma_0, \dots, \gamma_n\}$, we can construct a VASS $\mathcal{V} = (m, P, \Sigma, \delta, p_0, \mathbf{u}_0)$ which has the following structure:

- $m = n + 2$ is the dimension of \mathcal{V} . It is easy to observe that the dimension of \mathcal{V} is equal to $|\Gamma_\epsilon|$ which is the number of all possible stack contents of threads of \mathcal{M} .
- $P = (Q \times (\Gamma_\epsilon \cup \{\perp\})) \cup \{p_{\text{halt}}\}$ is the set of states of \mathcal{V} (with $p_{\text{halt}} \notin Q$). A state of the form $(q, w) \in Q \times \Gamma_\epsilon$ (resp. (q, \perp)) of \mathcal{V} means that the state of \mathcal{M} is q and that the stack content of the active thread is w (resp. there is no active thread). The state p_{halt} is used in order to interrupt the simulation of \mathcal{M} by \mathcal{V} .
- $\Sigma = \Delta$ is the input alphabet of \mathcal{V} .
- $\delta : P \times \Sigma \rightarrow P \times ([-1, 1])^m$ is the transition function of \mathcal{V} defined as follows: For every $p \in P$ and $t \in \Sigma$, we have:
 - $\delta(p, t) = (p', \mathbf{0}^m)$ if there are $q, q' \in Q$, $\gamma \in \Gamma$, and $u \in \Gamma_\epsilon$ such that $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \epsilon$, $p = (q, \gamma)$, and $p' = (q', u)$. This corresponds to the simulation of a transition rule of \mathcal{M} without thread creation.
 - $\delta(p, t) = (p', \mathbf{0}^m[i \leftarrow 1])$ if $i \in [1, m]$ and there are $q, q' \in Q$, $\gamma \in \Gamma$, and $u \in \Gamma_\epsilon$ such that $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \gamma_{i-1}$, $p = (q, \gamma)$, and $p' = (q', u)$. This corresponds to the simulation of a transition rule of \mathcal{M} with thread creation.
 - $\delta(p, t) = (p', \mathbf{0}^m[j \leftarrow 1])$ if $j \in [1, m]$, and there are $q, q' \in Q$, $\gamma \in \Gamma$, and $u \in \Gamma_\epsilon$ such that $t = \langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$, $p = (q, \gamma)$, $p' = (q', \perp)$, $u = \epsilon$ if $j = m$, and $u = \gamma_{j-1}$ if $j < m$. This corresponds to the interruption of the execution of the current active thread.

- $\delta(p, t) = (p', \mathbf{0}^m[i \leftarrow -1])$ if $i \in [1, m[$, and there are $q, q' \in Q$, such that $t = q \mapsto_{\mathcal{M}} q' \triangleleft \gamma_{i-1}$, $p = (q, \perp)$, and $p' = (q', \gamma_{i-1})$. This corresponds to the execution of a pending thread with topmost stack symbol γ_{i-1} .
- $\delta(p, t) = (p_{\text{halt}}, \mathbf{0}^m)$ otherwise. This indicates the end of the simulation of \mathcal{M} by \mathcal{V} whenever the transition t can not be applied from the state p .
- $\mathbf{u}_0 = (1, 0, \dots, 0)$. This corresponds to the initial pending thread of \mathcal{M} (i.e., initially \mathcal{M} has one pending thread with local configuration $(\gamma_0, 0)$).
- $p_0 = (q_0, \perp)$ is the initial state of \mathcal{V} . This corresponds to the initial state q_0 of \mathcal{M} .

Observe that the size of \mathcal{V} is polynomial in the size of \mathcal{M} . Moreover, the relation between \mathcal{M} and \mathcal{V} is given by the following lemma:

Lemma 3.5. *Let $q \in Q$. q is reachable by \mathcal{M} if and only if (q, \perp) is reachable by \mathcal{V} .*

The proof of Lemma 3.5 is done by induction on the length of the runs and is given in Appendix B.

As an immediate consequence of Lemma 3.5, we obtain the following result:

Proposition 3.6. *The SR problem for DCFSSs is polynomially reducible to the coverability problem for VASSs.*

3.4. From the coverability problem for VASSs to the BSR[2] problem for DCFSSs.

In the following, we prove that the coverability problem for VASSs is polynomially reducible to the BSR[2] for DCFSSs. Given a VASS $\mathcal{V} = (n, Q, \Sigma, \delta, q_0, \mathbf{u}_0)$, we construct a DCFS \mathcal{M} such that the coverability problem for \mathcal{V} is reducible to the BSR[2] problem for \mathcal{M} . We assume w.l.o.g that for every $q \in Q$ and $a \in \Sigma$, $\delta(q, a) \in Q \times \{\mathbf{u} \in \mathbb{N}^n \mid \sum_{i=1}^n \text{abs}(\mathbf{u}[i]) \leq 1\}$ and $\mathbf{u}_0 = \mathbf{0}^n$. Intuitively, \mathcal{M} has, for each $i \in [1, n]$, a stack symbol γ_i such that the number of pending threads with local configuration $(\gamma_i, 2)$ denotes the current value of the i -th counter of \mathcal{V} . The system \mathcal{M} has also a special stack symbol γ'_0 such that the pending threads with local configuration $(\gamma'_0, 1)$ are used to create threads with local configuration $(\gamma_i, 2)$ where $i \in [1, n]$ (which corresponds to the increment of the value of a counter of \mathcal{V}). We now sketch the behavior of \mathcal{M} . First, \mathcal{M} creates an arbitrary number of threads with local configuration $(\gamma'_0, 1)$ from the initial configuration. Then, the simulation of a rule $\delta(q, a) = (q', \mathbf{u})$ depends on the value of the vector \mathbf{u} : (1) If $\mathbf{u} = \mathbf{0}^n$, then \mathcal{M} moves its state from q to q' , (2) If $\mathbf{u} = \mathbf{0}^n[i \leftarrow 1]$ for some $i \in [1, n]$, then \mathcal{M} uses a thread with local configuration $(\gamma'_0, 1)$ to create a thread with local configuration $(\gamma_i, 2)$ while moving its state from q to q' , and (3) If $\mathbf{u} = \mathbf{0}^n[i \leftarrow -1]$ for some $i \in [1, n]$, then \mathcal{M} transforms the local configuration of a pending thread from $(\gamma_i, 2)$ to $(\epsilon, 3)$. Formally $\mathcal{M} = (P, \Gamma, \Delta, p_0, \gamma_0)$ is built from \mathcal{V} as follows:

- $P = \{p_0\} \cup Q$ is the set of states such that $p_0 \notin Q$. p_0 is the initial state. A state $q \in Q$ represents the current state of \mathcal{V} .
- $\Gamma = \{\gamma_0, \gamma_1, \dots, \gamma_n\} \cup \{\gamma'_0\}$ is the finite set of stack symbols. The symbol γ_0 represents the initial stack symbol. The symbol γ'_0 represents the stack content of auxiliary threads that are “consumed” in order to simulate an operation of \mathcal{V} . For every $i \in [1, n]$, the number of pending threads with stack content γ_i denotes the current value of the i -th counter of \mathcal{V} .
- Δ is the smallest transition relation satisfying the following conditions:

- $\langle p_0, \gamma_0 \rangle \rightarrow_{\mathcal{M}} \langle p_0, \gamma_0 \rangle \triangleright \gamma'_0$ and $\langle p_0, \gamma_0 \rangle \mapsto_{\mathcal{M}} \langle q_0, \epsilon \rangle$. These transitions create an arbitrary number of threads with local configuration $(\gamma'_0, 1)$ before moving the state from p_0 to q_0 .
- For every $q \in Q$, we have that $q \mapsto_{\mathcal{M}} q \triangleleft \gamma'_0$. This transition corresponds to start the execution of a pending thread with stack content γ'_0 to simulate an operation of \mathcal{V} that increments the value of a counter.
- For every $q \in Q$, we have that $\langle q, \gamma'_0 \rangle \mapsto_{\mathcal{M}} \langle q, \epsilon \rangle$. This transition corresponds to the interruption of the execution of the current active thread with stack content γ'_0 in order to permit the simulation by \mathcal{M} of an operation of \mathcal{V} that decrements a counter.
- For every $q, q' \in Q$ and $a \in \Sigma$, if $\delta(q, a) = (q', \mathbf{0}^n)$, then $\langle q, \gamma'_0 \rangle \rightarrow_{\mathcal{M}} \langle q', \gamma'_0 \rangle \triangleright \epsilon$. This transition simulates an operation of \mathcal{V} that moves the state from q to q' .
- For every $q, q' \in Q$, $a \in \Sigma$, and each $i \in [1, n]$, if $\delta(q, a) = (q', \mathbf{0}^n[i \leftrightarrow 1])$, then $\langle q, \gamma'_0 \rangle \rightarrow_{\mathcal{M}} \langle q', \gamma'_0 \rangle \triangleright \gamma_i$. This transition simulates an operation that increments the i -th counter of \mathcal{V} . Notice that the switch number of the created thread with stack content γ_i is 2 since the switch number of the active thread (with stack content γ'_0) is always equal to 1.
- For every $q, q' \in Q$, $a \in \Sigma$, and $i \in [1, n]$, if $\delta(q, a) = (q', \mathbf{0}^n[i \leftrightarrow -1])$, then $q \mapsto_{\mathcal{M}} q' \triangleleft \gamma_i$, and $\langle q', \gamma_i \rangle \mapsto_{\mathcal{M}} \langle q', \epsilon \rangle$. These transitions simulate an operation that decrements the value of the i -th counter of \mathcal{V} .

Observe that the size of \mathcal{M} is polynomial in the size of \mathcal{V} . Moreover, the relation between \mathcal{V} and \mathcal{M} is given by the following lemma:

Lemma 3.7. *Let $q \in Q$. q is reachable by \mathcal{V} if and only if q is 2-bounded reachable by \mathcal{M} .*

The proof of Lemma 3.7 is done by induction on the length of the runs and is given in Appendix C.

As an immediate consequence of Lemma 3.7, we obtain the following result:

Proposition 3.8. *The coverability problem for VASSs is polynomially reducible to the BSR[2] for DCFSSs.*

4. THE SSR[k] PROBLEM FOR DCFSS

In this section, we consider the problem SSR[k] for $k \in \mathbb{N}$. We show that the problem SSR[k] for DCFSSs is NP-complete. But before going into the details, let us recall the definition of the existential Presburger arithmetic and some related results.

4.1. Existential Presburger arithmetic. Let \mathcal{V} be a set of variables. We use x, y, \dots to range over variables in \mathcal{V} . The set of terms of the Presburger arithmetic is defined by:

$$t ::= 0 \mid 1 \mid x \mid t + t$$

Then, the class of existential formulae is defined as follows:

$$\varphi ::= t \leq t \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x. \varphi$$

The length of a Presburger formula φ , denoted by $|\varphi|$, is the number of letters used in writing φ . The notion of free variables for an existential Presburger formula is defined as usual. We write $FV(\varphi) \subseteq \mathcal{V}$ to denote that the formula φ has $FV(\varphi)$ as a set of

free variables. The semantics of existential Presburger formulae is defined in the standard way. Given a function f from $\text{var}(\varphi)$ to \mathbb{N} , we write $f \models \varphi$ if φ holds for f (in the obvious sense) and, in this case, we say that f satisfies φ . We use $\llbracket \varphi \rrbracket$ to denote the set $\{f \in [FV(\varphi) \rightarrow \mathbb{N}] \mid f \models \varphi\}$.

An existential Presburger formula φ is satisfiable if and only if $\llbracket \varphi \rrbracket \neq \emptyset$. The satisfiability problem for φ consists in checking whether φ is satisfiable. It is well-known that the satisfiability problem for existential Presburger formulae is NP-complete [VSS05].

Theorem 4.1. *The satisfiability problem for existential Presburger formulae is NP-complete.*

We recall that the Parikh image of a context-free language is definable by an existential Presburger formula.

Theorem 4.2 ([SSMH04]). *If \mathcal{P} is a PDA with input alphabet Σ , then, it is possible to construct, in time and space polynomial in $|\mathcal{P}|$, an existential Presburger formula φ with free variables Σ such that $\llbracket \varphi \rrbracket = \text{Parikh}(L(\mathcal{P}))$.*

4.2. The $\text{SSR}[k]$ problem for DCFSs is NP-complete. In this section, we mainly prove the following result:

Theorem 4.3. *For every $k \in \mathbb{N}$, the problem $\text{SSR}[k]$ for DCFSs is NP-complete.*

The NP-hardness is proved by a reduction from the coverability problem of acyclic Petri nets [Ste95] to $\text{SSR}[k]$. This is done by a simple adaptation of the construction given in Section 3.4. The upper-bound is obtained by a reduction to the satisfiability problem for existential Presburger formulae.

Let $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$ be a DCFS, k be a natural number, and $F \subseteq Q$ be a set of target states. To reduce the k -stratified state reachability problem for \mathcal{M} to the satisfiability problem of an existential formula φ , we proceed in two steps: First, we construct a bounded stack pushdown automaton \mathcal{P} that simulates the k -stratified computations of \mathcal{M} without taking into account the causality constraints. (The use of a pushdown automaton here is for technical convenience. In principle, \mathcal{P} can be encoded as a finite state automaton, but this will make the construction cumbersome.) In fact, \mathcal{P} assumes that there is an unbounded number of pending threads for any local configurations in $\Gamma_\epsilon \times [0, k]$. Intuitively \mathcal{P} performs the same pushdown operations as the ones specified by Δ while making visible as transition labels: (1) $(\gamma, i, \triangleright)$ if the local configuration of the created (or the interrupted) thread is (γ, i) , (2) $(\gamma, i, \triangleleft)$ if the local configuration of the pending thread that has been activated is (γ, i) , and (3) $(\epsilon, i, -)$ if there no thread creation and the switch number of the current active thread is i .

Then, we show that there is a k -stratified computation of \mathcal{M} if and only if there is a computation π of \mathcal{P} that satisfies the following two conditions:

- *The stratified condition:* Threads in π are scheduled according to their increasing switch number (from 0 to k).
- *The flow condition:* For every stack content $\gamma \in \Gamma$ and switch number $i \in [0, k]$, the number of occurrences of $(\gamma, i, \triangleright)$ in π is greater than the number of occurrences of $(\gamma, i, \triangleleft)$ in π (i.e., the number of created (or interrupted) threads with local configuration (γ, i) is greater than the number of threads with local configuration (γ, i) that has been activated).

Since the set of traces that satisfies the stratified condition is a regular one, we can construct a pushdown automaton \mathcal{P}' (of bounded stack depth) that recognizes the set of traces of \mathcal{P} that satisfies the first condition. Therefore, we can use Theorem 4.2 to construct an existential Presburger formula φ' that characterizes the Parikh image of the set of traces of \mathcal{P}' . On the other hand, the flow condition can be expressed as an existential Presburger formula φ'' over the set of variables $\{(\gamma, i, \triangleright) \mid \gamma \in \Gamma, i \in [0, k]\}$ and $\{(\gamma, i, \triangleleft) \mid \gamma \in \Gamma, i \in [0, k]\}$. Armed with these results, we can show that the k -stratified state reachability problem for \mathcal{M} is reducible to the satisfiability problem of the existential formula $\varphi = \varphi' \wedge \varphi''$.

Let us give more details about the constructions described above.

From the DCFS \mathcal{M} to the pushdown automaton \mathcal{P} : The pushdown automaton $\mathcal{P} = (P, \Sigma, \Gamma_{\mathcal{P}}, \Delta_{\mathcal{P}}, p_0, \gamma_{\mathcal{P}}, F_{\mathcal{P}})$ is built up from \mathcal{M} as follows:

- $P = Q$ is the finite set of states. A state q represents the global state of \mathcal{M} .
- $\Sigma = \bigcup_{i=0}^k \Sigma_i$ is the finite set of input symbols where $\Sigma_i = \Sigma_i^{\text{cr}} \cup \Sigma_i^{\text{r}} \cup \Sigma_i^{\text{l}}$ with $\Sigma_i^{\text{cr}} = \Gamma_{\epsilon} \times \{i+1\} \times \{\triangleright\}$, $\Sigma_i^{\text{r}} = \Gamma \times \{i\} \times \{\triangleleft\}$, and $\Sigma_i^{\text{l}} = \{(\epsilon, i, -)\}$ for all $i \in [0, k]$. A transition labeled with $(\alpha, i, \triangleright)$ corresponds to a rule of \mathcal{M} that: (1) creates a thread with local configuration (α, i) , or (2) interrupts the execution of the active thread with stack content is α . A transition labeled with $(\alpha, i, \triangleleft)$ corresponds to a rule of \mathcal{M} that activates a pending thread with local configuration (α, i) . A transition labeled with $(\epsilon, i, -)$ corresponds to a rule of \mathcal{M} without thread creation and where the switch number of the current active thread is i .
- $\Gamma_{\mathcal{P}} = (\Gamma_{\epsilon} \times [0, k]) \cup \{\perp\}$ is the finite set of stack symbols. Each symbol in $\Gamma_{\mathcal{P}}$ corresponds to the local configuration of the active thread of \mathcal{M} .
- $\Delta_{\mathcal{P}}$ is the smallest transition relation satisfying the following conditions:
 - For every $i \in [0, k]$ and $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \epsilon$, $\langle q, (\gamma, i) \rangle \xrightarrow{(\epsilon, i, -)}_{\mathcal{P}} \langle q', (u, i) \rangle$. This transition corresponds to the simulation of a transition of \mathcal{M} without thread creation.
 - For every $i \in [0, k]$ and $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \alpha$ with $\alpha \in \Gamma$, $\langle q, (\gamma, i) \rangle \xrightarrow{(\alpha, i+1, \triangleright)}_{\mathcal{P}} \langle q', (u, i) \rangle$. This corresponds to the simulation of a transition of \mathcal{M} with thread creation.
 - For every $i \in [0, k]$ and $\langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$, $\langle q, (\gamma, i) \rangle \xrightarrow{(u, i+1, \triangleright)}_{\mathcal{P}} \langle q', \perp \rangle$. This corresponds to the interruption of the execution of the active thread of \mathcal{M} .
 - For every $i \in [0, k]$ and $q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$, $\langle q, \perp \rangle \xrightarrow{(\gamma, i, \triangleleft)}_{\mathcal{P}} \langle q', (\gamma, i) \rangle$. This corresponds to the activation of a pending thread of \mathcal{M} with local configuration (γ, i) .
- $p_0 = q_0$ is the initial state.
- $\gamma_{\mathcal{P}} = \perp$ is the initial stack symbol.
- $F_{\mathcal{P}} = F$ is the set of final states.

Observe that the size of the pushdown automaton \mathcal{P} is polynomial in the size of the DCFS \mathcal{M} . Moreover, the depth of the stack of \mathcal{P} is always bounded by one.

The relation between the DCFS \mathcal{M} and the pushdown automaton \mathcal{P} is established by Lemma 4.4 which states that there is a state $q \in F$ such that q is k -stratified reachable by \mathcal{M} if and only if there is a computation π of \mathcal{P} that satisfies the stratified condition and the flow condition.

Lemma 4.4. *A state $q \in F$ is k -stratified reachable by \mathcal{M} if and only if there is $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that:*

- $\sigma_0 \sigma_1 \cdots \sigma_k \in \text{Traces}_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$, and

- $|\sigma_i|_{(\gamma, i, \triangleleft)} \leq |\sigma_{i-1}|_{(\gamma, i, \triangleright)}$ for all $\gamma \in \Gamma$ and $i \in [0, k]$ where $\sigma_{-1} = (\gamma_0, 0, \triangleright)$.

The proof of Lemma 4.4 is done by induction and is given in the Appendix D.

From the PDA \mathcal{P} to the existential Presburger formula φ : In the following, we show that the problem of checking whether there is $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that $\sigma_0\sigma_1 \cdots \sigma_k \in \text{Traces}_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$ and $|\sigma_i|_{(\gamma, i, \triangleleft)} \leq |\sigma_{i-1}|_{(\gamma, i, \triangleright)}$ for all $\gamma \in \Gamma$ and $i \in [0, k]$ with $\sigma_{-1} = (\gamma_0, 0, \triangleright)$ is polynomially reducible to the satisfiability problem of an existential Presburger formula φ . This implies that the $\text{SSR}[k]$ problem for \mathcal{M} is polynomially reducible to the satisfiability problem for φ (see Lemma 4.4).

Lemma 4.5. *It is possible to construct an existential Presburger formula φ with $\llbracket \varphi \rrbracket \neq \emptyset$ if and only if there is $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that $\sigma_0\sigma_1 \cdots \sigma_k \in \text{Traces}_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$ and $|\sigma_i|_{(\gamma, i, \triangleleft)} \leq |\sigma_{i-1}|_{(\gamma, i, \triangleright)}$ for all $\gamma \in \Gamma$ and $i \in [0, k]$ with $\sigma_{-1} = (\gamma_0, 0, \triangleright)$.*

Proof. Let \mathcal{P}' be the pushdown automaton such that $L(\mathcal{P}') = \text{Traces}_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\}) \cap (\Sigma_0^* \cdot \Sigma_1^* \cdots \Sigma_k^*)$. Such pushdown automaton \mathcal{P}' is effectively constructible from \mathcal{P} since the class of pushdown automata is closed under intersection with a regular language.

Now, we can use Theorem 4.2 to construct a Presburger formula φ' with free variables Σ such that $\llbracket \varphi \rrbracket = \text{Parikh}(L(\mathcal{P}'))$. In addition, for every $i \in [1, k]$, we construct an existential Presburger formula φ_i with free variables Σ such that $\varphi_i = \bigwedge_{\gamma \in \Gamma} ((\gamma, i, \triangleleft) \leq (\gamma, i, \triangleright))$.

Let $\varphi_0 = (\bigwedge_{\gamma \in \Gamma \setminus \{\gamma_0\}} ((\gamma, 0, \triangleleft) \leq 0)) \wedge ((\gamma_0, 0, \triangleleft) \leq 1)$ and $\varphi'' = \bigwedge_{i=0}^k \varphi_i$.

Then, it is not hard to see that the existential Presburger formula $\varphi = \varphi' \wedge \varphi''$ is satisfiable if and only if for every $i \in [0, k]$, there are there are $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that $\sigma_0\sigma_1 \cdots \sigma_k \in \text{Traces}_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$ and $|\sigma_i|_{(\gamma, i, \triangleleft)} \leq |\sigma_{i-1}|_{(\gamma, i, \triangleright)}$ for all $\gamma \in \Gamma$ and $i \in [0, k]$ with $\sigma_{-1} = (\gamma_0, 0, \triangleright)$. \square

As an immediate consequence of Theorem 4.1 and Lemma 4.5, we obtain the following result:

Lemma 4.6. *For every $k \in \mathbb{N}$, the problem $\text{SSR}[k]$ for DCFSs is in NP.*

5. REACHABILITY ANALYSIS FOR DYNAMIC NETWORKS OF CONCURRENT PUSHDOWN SYSTEMS

In this section, we consider the case of DCPSs. It is well-known that the SR problem is undecidable already for networks with two concurrent pushdown processes. We show however that both problems $\text{BSR}[k]$ and $\text{SSR}[k]$ are decidable, for any given bound $k \in \mathbb{N}$. For that, we prove the following fact.

Theorem 5.1. *For every $k \in \mathbb{N}$, the problems $\text{BSR}[k]$ and the $\text{SSR}[k]$ for DCPS are exponentially reducible to the corresponding problems for DCFS.*

A corollary of Theorem 3.1, Theorem 4.3, and Theorem 5.1, we obtain the following results:

Corollary 5.2. *For every $k \in \mathbb{N}$, the $\text{BSR}[k]$ problem for DCPSs is in 2-EXPSpace, and the $\text{SSR}[k]$ problem for DCPSs is in NEXPTIME.*

The rest of this section is devoted to the proof of Theorem 5.1. Let us fix a DCPS $\mathcal{M} = (Q, \Gamma, \Delta, q_0, \gamma_0)$. We show that it is possible to construct a DCFS \mathcal{M}_{fs} such that the problems $\text{BSR}[k]$ and $\text{SSR}[k]$ for \mathcal{M} can be reduced to their corresponding problems for \mathcal{M}_{fs} . Let us present the main steps of this construction. For that, let us consider the problem $\text{BSR}[k]$, for some fixed $k \in \mathbb{N}$. Then, let us concentrate on the computations of one thread, and assume that this thread will be interrupted i times (with $i \leq k + 1$) during its execution starting from some initial global state q and initial local state γ . The computations of such a thread correspond to runs of a pushdown automaton, built out of \mathcal{M} , which (1) performs the same operations on the stack and the global state as the ones specified by Δ , (2) makes visible as transition labels the local state (element of Γ) of spawned threads, and (3) nondeterministically guesses jumps from a global state to another one corresponding to the effect of context switches. These jumps are also made visible as transition labels under the form of $(q, \alpha, q') \in (Q \times \Gamma_\epsilon \times Q)$ (meaning that the computation of the thread is interrupted at the state q with stack content αw for some $w \in \Gamma^*$, and is resumed at the state q'). In fact, if a thread fires a transition labeled by a symbol of the form (q, ϵ, q') then its execution will be definitely interrupted (i.e., the execution of this thread will never be resumed again). The number of such jumps in each run is precisely i .

Then, the problem is to handle the composition of all the computations of the generated threads and to make sure that the guesses made by each one of them (on their control state jumps due to context switches) are correct. In fact, handling this composition is very a hard task in general when threads are modeled as pushdown automata. To overcome this difficulty, the key observation is that it is possible to assume without loss of preciseness that some of the generated threads can be ignored (or lost). Indeed, these threads can always be considered as threads which will never be scheduled. Therefore, the behaviors of each thread can be modeled using a finite-state automaton which recognizes the downward closure of the language of the pushdown automaton of a thread with respect to the subword relation. We know by Theorem 1.1 that this automaton is effectively constructible. So, let $\mathcal{A}_{(q,\gamma)}$ be the automaton modeling the computations of threads starting from the state q and initial stack content γ , and performing at most $k + 1$ interruptions. We assume w.l.o.g that $\mathcal{A}_{(q,\gamma)}$ has no ϵ -transitions.

The next step is to synchronize the so-defined finite-state automata in order to represent valid computations of the whole system. For that, we define a DCFS \mathcal{M}_{fs} which simulates the composition of these automata as follows:

- A pending thread with stack content γ which has never been activated can be dispatched by \mathcal{M}_{fs} at the moment of a context switch. For that, \mathcal{M}_{fs} has a rule $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}_{\text{fs}}} \langle \sharp, s_0 \rangle \triangleright \epsilon$ where s_0 is the initial state of $\mathcal{A}_{(q,\gamma)}$, for every possible starting q and every stack symbol $\gamma \in \Gamma$. This rule allows to check that the control state is q , and to move the system to a special state \sharp corresponding to the simulation of a phase without context switches.
- During the simulation, when a transition $s \xrightarrow{\gamma}_{\mathcal{A}_{(q,\gamma)}} s'$, with $\gamma \in \Gamma$, is encountered, a new thread is spawned by \mathcal{M}_{fs} with initial stack content γ . This is done using a rule of the form $\langle \sharp, s \rangle \rightarrow_{\mathcal{M}_{\text{fs}}} \langle \sharp, s' \rangle \triangleright \gamma$. The new thread will stay pending until \mathcal{M}_{fs} can dispatch it.
- Encountering a transition $s \xrightarrow{(q_1, \alpha, q_2)}_{\mathcal{A}_{(q,\gamma)}} s'$ means that the computation of the simulated thread is interrupted at the global store q_1 with stack content αw for some $w \in \Gamma^*$, and will be resumed later when the global state will become q_2 (due to the execution of some other threads). Then, \mathcal{M}_{fs} moves from its global state \sharp to the global state q_1 so that the control can be taken by another pending thread), and transforms the stack configuration

of the current thread (which may be interrupted) to $(q_2, (s', \alpha))$. This is done by a rule of the form $\langle \sharp, s \rangle \mapsto_{\mathcal{M}_{\text{fs}}} \langle q_1, (q_2, (s', \alpha)) \rangle$.

- To simulate a transition $q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$ that starts/resumes the execution of a pending thread with topmost stack symbol $\gamma \in \Gamma$, \mathcal{M}_{fs} has the rules $q \mapsto_{\mathcal{M}_{\text{fs}}} q' \triangleleft \gamma$ and $q \mapsto_{\mathcal{M}_{\text{fs}}} q' \triangleleft (q', (s, \gamma))$. In this case, we observe that the only action that can be done by \mathcal{M}_{fs} after executing these rules is to activate some pending thread with topmost stack symbol γ' (either dispatched for the first time, or resumed after some interruption).

We have seen above how \mathcal{M}_{fs} dispatches pending threads for the first time. The resumption of threads at state q' is done by having rules of the form $\langle q', (q', (s, \gamma)) \rangle \mapsto_{\mathcal{M}_{\text{fs}}} \langle \sharp, s \rangle \triangleright \epsilon$. Such a rule means that if a pending thread $(q', (s, \gamma))$ exists, then it can be activated and the simulation of its behaviors is resumed from the state s (at which it was stopped at the last interruption).

Let us give in more details the construction described above.

5.1. Simulation of threads of \mathcal{M} with finite-state automata. Next, we give the construction of the finite state automaton $\mathcal{A}_{(q, \gamma)}$ for some given $q \in Q$ and $\gamma \in \Gamma$. For that, we start by considering a pushdown automaton $\mathcal{P}_{(q, \gamma)}$ simulating the behaviors of a thread that starts its execution from the global state q and the initial stack configuration γ after some number of jumps in the global state (representing guesses on the effect of context switches). The spawned thread as well as the guesses on the global state jumps made during the computation are made visible as transition labels.

Then, let $\mathcal{P}_{(q, \gamma)} = (P, \Sigma, \Gamma, \Delta_{\mathcal{P}}, q, \gamma, Q)$ be the pushdown automaton where:

- $P = Q \cup (Q \times \Gamma)$ is the finite set of states,
- $\Sigma = \Gamma \cup \Sigma_{\text{sw}} \cup \Sigma_{\text{intr}}$ is the finite set of input symbols with $\Sigma_{\text{sw}} = Q \times \Gamma \times Q$ and $\Sigma_{\text{intr}} = Q \times \{\epsilon\} \times Q$,
- $\Delta_{\mathcal{P}}$ is the smallest transition relation such that:
 - For every $\langle q_1, \gamma_1 \rangle \mapsto_{\mathcal{M}} \langle q_2, u \rangle \triangleright \alpha$, $\langle q_1, \gamma_1 \rangle \xrightarrow{\alpha}_{\mathcal{P}_{(q, \gamma)}} \langle q_2, u \rangle$. This rule simulates a push-down operation on the active thread with the possibility of a thread creation.
 - For every $\langle q_1, \gamma_1 \rangle \mapsto_{\mathcal{M}} \langle q_2, u \rangle$ and $q'_2 \in Q$, $\langle q_1, \gamma_1 \rangle \xrightarrow{(q_2, \epsilon, q'_2)}_{\mathcal{P}_{(q, \gamma)}} \langle q'_2, u \rangle$. This rule corresponds to interrupt the execution of the active thread at the state q_2 . In addition, the execution of this thread will never be resumed again.
 - For every $\langle q_1, \gamma_1 \rangle \mapsto_{\mathcal{M}} \langle q_2, u \rangle$, $q'_2 \in Q$, and $\gamma' \in \Gamma$, $\langle q_1, \gamma_1 \rangle \xrightarrow{(q_2, \gamma', q'_2)}_{\mathcal{P}_{(q, \gamma)}} \langle (q'_2, \gamma'), u \rangle$ and $\langle (q'_2, \gamma'), \gamma' \rangle \xrightarrow{\epsilon}_{\mathcal{P}_{(q, \gamma)}} \langle q'_2, \gamma' \rangle$. This rule simulates the interruption of the execution of the active thread at the state q_2 . In addition, the execution of this thread will be resumed at the state q'_2 with topmost stack symbol γ' .

Then, the set of behaviors represented by this pushdown automaton which correspond to precisely $i \geq 1$ context switches (or interruptions) is given by the following language:

$$L'_{((q, \gamma), i)} = L(\mathcal{P}_{(q, \gamma)}) \cap ((\Gamma^* \cdot \Sigma_{\text{sw}})^{i-1} (\Gamma^* \cdot \Sigma_{\text{intr}}))$$

The set $L'_{((q, \gamma), i)}$ is a context-free language in general (since it is the intersection of a context-free language with a regular one). Due to the fact that some of the generated threads can be ignored (or lost), we can consider without loss of preciseness the downward closure of $L'_{(q, \gamma)}$ w.r.t. the sub-word relation corresponding to the deletion of symbols in Γ while preserving all symbols in $\Sigma_{\text{sw}} \cup \Sigma_{\text{intr}}$, i.e., the set

$$L'_{(q,\gamma)} = \bigcup_{i=1}^{k+1} \left(L'_{((q,\gamma),i)} \downarrow \cap ((\Gamma^* \cdot \Sigma_{\text{sw}})^{i-1} (\Gamma^* \cdot \Sigma_{\text{inr}})) \right)$$

By Theorem 1.1, the language $L'_{(q,\gamma)}$ is regular and can be effectively represented by a finite-state automaton $\mathcal{A}_{(q,\gamma)} = (S_{(q,\gamma)}, \Sigma, \Delta_{(q,\gamma)}, I_{(q,\gamma)}, F_{(q,\gamma)})$. We assume w.l.o.g that all the states in the automaton $\mathcal{A}_{(q,\gamma)}$ are co-reachable from the final states. We assume also that $\Delta_{(q,\gamma)} \subseteq S_{(q,\gamma)} \times \Sigma \times S_{(q,\gamma)}$ (i.e., there is no transition of $\mathcal{A}_{(q,\gamma)}$ labeled by the empty word).

5.2. From the DCPS \mathcal{M} to the DCFS \mathcal{M}_{fs} . In the following, we give the formal definition of the DCFS \mathcal{M}_{fs} . The system \mathcal{M}_{fs} is defined by the tuple $(Q_{\text{fs}}, \Gamma_{\text{fs}}, \Delta_{\text{fs}}, q_0, \gamma_0)$ where:

- $Q_{\text{fs}} = Q \cup \{\#\}$ is the finite set of states.
- $\Gamma_{\text{fs}} = \Gamma \cup S_{\text{fs}}^{\text{sm}} \cup S_{\text{fs}}^{\text{sw}}$ is the finite set of stack alphabet where $S_{\text{fs}}^{\text{sm}} = \bigcup_{(q,\gamma) \in Q \times \Gamma} S_{(q,\gamma)}$ and $S_{\text{fs}}^{\text{sw}} = Q \times S_{\text{fs}}^{\text{sm}} \times \Gamma_{\epsilon}$.
- Δ_{fs} is the smallest set of transitions such that
 - *Initialize:* For every $\gamma \in \Gamma$ and $q \in Q$, we have $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}_{\text{fs}}} \langle \#, s_0 \rangle \triangleright \epsilon$ where s_0 is the initial state of $\mathcal{A}_{(q,\gamma)}$.
 - *Spawn:* For every $q \in Q$, $\gamma \in \Gamma$, and $s \xrightarrow{\alpha}_{\mathcal{A}_{(q,\gamma)}} s'$, we have $\langle \#, s \rangle \rightarrow_{\mathcal{M}_{\text{fs}}} \langle \#, s' \rangle \triangleright \alpha$. (Notice that, from the definition of $\mathcal{A}_{(q,\gamma)}$, α is necessarily in Γ .)
 - *Interrupt:* For every $q \in Q$, $\gamma \in \Gamma$, and $s \xrightarrow{(q_1, \alpha, q_2)}_{\mathcal{A}_{(q,\gamma)}} s'$, we have $\langle \#, s \rangle \mapsto_{\mathcal{M}_{\text{fs}}} \langle q_1, (q_2, (s', \alpha)) \rangle$.
 - *Dispatch:* For every $s \in S_{\text{fs}}^{\text{sm}}$ and $q_1 \mapsto_{\mathcal{M}} q_2 \triangleleft \gamma'$, we have $q_1 \mapsto_{\mathcal{M}_{\text{fs}}} q_2 \triangleleft \gamma'$ and $q_1 \mapsto_{\mathcal{M}_{\text{fs}}} q_2 \triangleleft (q_2, (s, \gamma'))$.
 - *Resume:* For every $q \in Q$, $\gamma \in \Gamma$, and $s \in S_{\text{fs}}^{\text{sm}}$, we have $\langle q, (q, (s, \gamma)) \rangle \rightarrow_{\mathcal{M}_{\text{fs}}} \langle \#, s \rangle \triangleright \epsilon$.

Theorem 5.1 is an immediate consequence of Lemma 5.3.

Lemma 5.3. *For every $k \in \mathbb{N}$, a control state $q \in Q$ is k -bounded reachable (resp. k -stratified) reachable by \mathcal{M} iff q is k -bounded (resp. k -stratified) reachable by \mathcal{M}_{fs} .*

The proof of Lemma 5.3 is given in Appendix E.

6. CONCLUSION

We have proposed new concepts for context-bounded verification we believe that are natural and suitable for programs with dynamic thread creation. These concepts are based on the idea of bounding the number of switches for each thread and not for all the threads in a computation.

First, we have proved that even for finite-state threads, adopting such a notion of context-bounding leads in general to a problem which is as hard as the coverability problem of Petri nets. This means that, in theory, the complexity of this problem is high, but in practice, there are quite efficient techniques (based on iterative computation of under/upper approximations) developed recently for solving this problem which have been implemented and used successfully in [GRB06b, GRB06a]. Moreover, we have proposed a notion of stratified context-bounding for which the verification is in NP, i.e., as hard as in the case

without dynamic thread creation. An interesting question is how to implement efficiently the analysis in this case using clever encodings in SMT solvers.

Moreover, we have proved that the considered problems are still decidable for the case of pushdown threads. This is done by a nontrivial reduction to the corresponding problems for finite-state threads. This reduction is based on computing the regular downward closure of context-free languages w.r.t. the sub-word relation. The downward closure computation may lead in general to an unavoidable exponential blow-up. This is due to the succinctness of context-free grammars w.r.t. finite state automata: For instance, the finite language $\{a^{2^N}\}$, for a fixed $N \geq 1$, can be defined with a context-free grammar of size N whereas a finite-state automaton representing it (or its downward closure) is necessarily of size at least 2^N . An interesting open problem is whether there is an alternative proof technique which allows to avoid the downward closure construction. In practice, we believe that it would be possible to overcome this problem by for instance designing algorithms allowing to generate efficiently and incrementally (parts of the) downward closure.

Finally, in our models, we consider that each created thread inherits a switch number from its father (the one of its father plus 1). An alternative definition can be obtained by considering that each created thread is given the switch number 0. (Therefore, each thread can perform up to k switches.) However, the problem $\text{SSR}[k]$ for finite state threads (resp. pushdown threads) becomes EXPSPACE -complete (in 2-EXPSPACE) instead of NP -complete (NEXPTIME) for this definition.

REFERENCES

- [BESS05] A. Bouajjani, J. Esparza, S. Schwoon, and J. Strejcek. Reachability analysis of multithreaded software with asynchronous communication. In *FSTTCS'05*, LNCS 3821, pages 348–359. Springer, 2005.
- [BMOT05] Ahmed Bouajjani, Markus Müller-Olm, and Tayssir Touili. Regular symbolic analysis of dynamic networks of pushdown systems. In *CONCUR'05*, LNCS, 2005.
- [BT03] Ahmed Bouajjani and Tayssir Touili. Reachability Analysis of Process Rewrite Systems. In *FSTTCS'03*. LNCS 2914, 2003.
- [BT05] Ahmed Bouajjani and Tayssir Touili. On Computing Reachability Sets of Process Rewrite Systems. In *RTA'05*. LNCS, 2005.
- [CKS06] Byron Cook, Daniel Kroening, and Natasha Sharygina. Over-approximating boolean programs with unbounded thread creation. *Formal Methods in Computer Aided Design*, 0:53–59, 2006.
- [CKS07] Byron Cook, Daniel Kroening, and Natasha Sharygina. Verification of boolean programs with unbounded thread creation. *Theoretical Computer Science*, 388(1-3):227 – 242, 2007.
- [Cou91] Bruno Courcelle. On construction obstruction sets of words. *EATCS'91*, 44:178–185, June 1991.
- [EP00] J. Esparza and A. Podelski. Efficient algorithms for pre^* and post^* on interprocedural parallel flow graphs. In *POPL'00*. ACM, 2000.
- [GRB06a] P. Ganty, J. F. Raskin, and L. Van Begin. A complete abstract interpretation framework for coverability properties of WSTS. In *VMCAI'06*, LNCS 3855, pages 49–64. Springer, 2006.
- [GRB06b] G. Geeraerts, J. F. Raskin, and L. Van Begin. Expand, enlarge and check: New algorithms for the coverability problem of WSTS. *J. Comput. Syst. Sci.*, 72(1):180–203, 2006.
- [HU79] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.
- [Lip76] R. Lipton. The reachability problem requires exponential time. Technical Report TR 66, 1976.
- [LMP09] Salvatore La Torre, P. Madhusudan, and Gennaro Parlato. Reducing context-bounded concurrent reachability to sequential reachability. In *CAV*, volume 5643 of *Lecture Notes in Computer Science*, pages 477–492. Springer, 2009.

- [LMP10] Salvatore La Torre, P. Madhusudan, and Gennaro Parlato. Model-checking parameterized concurrent programs using linear interfaces. In *CAV*, volume 6174 of *Lecture Notes in Computer Science*, pages 629–644. Springer, 2010.
- [LR08] A. Lal and T. W. Reps. Reducing concurrent analysis under a context bound to sequential analysis. In *CAV'08*, LNCS 5123, pages 37–51. Springer, 2008.
- [LS98] D. Lugiez and Ph. Schnoebelen. The regular viewpoint on PA-processes. In *Proc. 9th Int. Conf. Concurrency Theory (CONCUR'98), Nice, France, Sep. 1998*, volume 1466, pages 50–66. Springer, 1998.
- [LTKR08] A. Lal, T. Touili, N. Kidd, and T. W. Reps. Interprocedural analysis of concurrent programs under a context bound. In *TACAS'08*, LNCS 4963, pages 282–298. Springer, 2008.
- [Mo02] M. Muller-olm. Variations on constants. Habilitation thesis, Dortmund University, 2002.
- [MQ07] M. Musuvathi and S. Qadeer. Iterative context bounding for systematic testing of multithreaded programs. In *PLDI'07*, pages 446–455. ACM, 2007.
- [QR05] S. Qadeer and J. Rehof. Context-bounded model checking of concurrent software. In *TACAS'05*, LNCS 3440, pages 93–107. Springer, 2005.
- [QW04] S. Qadeer and D. Wu. KISS: keep it simple and sequential. In *PLDI'04*, pages 14–24. ACM, 2004.
- [Rac78] Charles Rackoff. The covering and boundedness problems for vector addition systems. *Theor. Comput. Sci.*, 6:223–231, 1978.
- [Ram00] G. Ramalingam. Context-sensitive synchronization-sensitive analysis is undecidable. *ACM Trans. Program. Lang. Syst.*, 22(2):416–430, 2000.
- [SES08] D. Suwimonterabuth, J. Esparza, and S. Schwoon. Symbolic context-bounded analysis of multithreaded java programs. In *SPIN'08*, LNCS 5156, pages 270–287. Springer, 2008.
- [SS00] Helmut Seidl and Bernhard Steffen. Constraint-based inter-procedural analysis of parallel programs. In *9th European Symposium on Programming (ESOP)*, 2000.
- [SSMH04] H. Seidl, T. Schwentick, A. Muscholl, and P. Habermehl. Counting in trees for free. In *ICALP'04*, LNCS 3142, pages 1136–1149. Springer, 2004.
- [Ste95] Iain A. Stewart. Reachability in some classes of acyclic petri nets. *Fundam. Inform.*, 23(1):91–100, 1995.
- [VSS05] Kumar Neeraj Verma, Helmut Seidl, and Thomas Schwentick. On the complexity of equational Horn clauses. In *CADE'05*, LNCS 3632, pages 337–352. Springer, 2005.
- [ZJ08] A. Zaks and R. Joshi. Verifying multi-threaded C programs with SPIN. In *SPIN'08*, LNCS 5156, pages 325–342. Springer, 2008.

APPENDIX A. THE PROOF OF LEMMA 3.3

Lemma 3.3 Let $q \in Q$. q is k -bounded reachable by \mathcal{M} iff q is reachable by \mathcal{M}' .

Proof. To prove Lemma 3.3 we proceed as follows: First, we show that for every reachable configuration c by \mathcal{M}' , the local configuration $((w', i'), j') \in Loc(\mathcal{M}')$ of any thread satisfies the condition that the switch number j' is equal to the recorded switch number i' (i.e., $i' = j'$). This property is established by Lemma A.1. Then, we prove that if a state q is k -bounded reachable by \mathcal{M} , then q is reachable by \mathcal{M}' (see Lemma A.2). Finally, we show that if a state q is reachable by a computation of \mathcal{M}' , then q is k -bounded reachable by \mathcal{M} (see Lemma A.3).

The switch number of any thread of \mathcal{M}' is equal to its recorded switch number: In the following, we show that for every reachable configuration c by \mathcal{M}' , the local configuration $((w', i'), j') \in Loc(\mathcal{M}')$ of any thread satisfies the condition that the switch number j' is equal to the recorded switch number i' .

Lemma A.1. *If $c_{\mathcal{M}'}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{M}')} c$, then $\text{Active}(c) \in (\{\perp\} \cup \{(w, i), i \mid w \in \Gamma_\epsilon, i \in [0, k]\})$, $\text{Idle}(c)((\epsilon, l)) = 0$ for all $l \in \mathbb{N}$, and $\text{Idle}(c)((w', i'), j') = 0$ for all $w' \in \Gamma_\epsilon$ and $i', j' \in \mathbb{N}$ such that $i' \neq j'$.*

Proof. Assume that $c_{\mathcal{M}'}^{\text{init}} \xrightarrow[n]{\tau} \mathcal{T}(\mathcal{M}') c$ for some $n \in \mathbb{N}$. We proceed by induction on n .

Basis. $n = 0$. Then $c_{\mathcal{M}'}^{\text{init}} = c = (q_0, \perp, \text{ld}_{\text{Loc}(\mathcal{M}')}^{\{((\gamma_0, 0), 0)\}})$. Hence, Lemma A.1 holds.

Step. $n > 0$. Then, there is a configuration $c' \in \text{Conf}(\mathcal{M}')$, $\tau' \in (\Delta')^*$, and $t \in \Delta'$ such that $\tau = \tau' t$, and $c_{\mathcal{M}'}^{\text{init}} \xrightarrow[n-1]{\tau'} \mathcal{T}(\mathcal{M}') c' \xrightarrow{t} \mathcal{T}(\mathcal{M}') c$.

Now, we apply the induction hypothesis to the run $c_{\mathcal{M}'}^{\text{init}} \xrightarrow[n-1]{\tau'} \mathcal{T}(\mathcal{M}') c'$, and we obtain $\text{Active}(c') \in (\{\perp\} \cup \{(w, i), i \mid w \in \Gamma_\epsilon, i \in [0, k]\})$, $\text{Idle}(c')((\epsilon, l)) = 0$ for all $l \in \mathbb{N}$, and $\text{Idle}(c')((w', i'), j') = 0$ for all $w' \in \Gamma_\epsilon$ and $i', j' \in \mathbb{N}$ such that $i' \neq j'$.

Since $c' \xrightarrow{t} \mathcal{T}(\mathcal{M}') c$, then there are four cases to study depending on the type of the transition $t \in \Delta'$:

- **Case 1:** $t = \langle q, (\gamma, r) \rangle \rightarrow_{\mathcal{M}'} \langle q', (u, r) \rangle \triangleright \epsilon$ with $r \in [0, k]$. Then, $\text{Active}(c') = ((\gamma, r), r)$ (using the induction hypothesis). This implies that $\text{Active}(c) = ((u, r), r)$ and $\text{Idle}(c) = \text{Idle}(c')$. Hence, all the conditions of Lemma A.1 are satisfied.
- **Case 2:** $t = \langle q, (\gamma, r) \rangle \rightarrow_{\mathcal{M}'} \langle q', (u, r) \rangle \triangleright (\alpha, r + 1)$ with $r \in [0, k]$ and $\alpha \in \Gamma$. Then, $\text{Active}(c') = ((\gamma, r), r)$, $\text{Active}(c) = ((u, r), r)$, and $\text{Idle}(c) = \text{Idle}(c') + \text{ld}_{\text{Loc}(\mathcal{M}')}^{\{((\alpha, r+1), r+1)\}}$. This implies that all the conditions of Lemma A.1 are satisfied.
- **Case 3:** $t = \langle q, (\gamma, r) \rangle \mapsto_{\mathcal{M}'} \langle q', (u, r + 1) \rangle$ with $r \in [0, k]$. Then, $\text{Active}(c') = ((\gamma, r), r)$, $\text{Active}(c) = \perp$, and $\text{Idle}(c) = \text{Idle}(c') + \text{ld}_{\text{Loc}(\mathcal{M}')}^{\{((u, r+1), r+1)\}}$. This implies that all the conditions of Lemma A.1 are satisfied.
- **Case 4:** $t = q \mapsto_{\mathcal{M}'} q' \triangleleft (\gamma, r)$ with $r \in [0, k]$ and $\gamma \in \Gamma$. Then, there is $j \in \mathbb{N}$ such that $\text{Active}(c') = \perp$, $\text{Active}(c) = ((\gamma, r), j)$, $\text{Idle}(c')((\gamma, r), j) \geq 1$, and $\text{Idle}(c) = \text{Idle}(c') - \text{ld}_{\text{Loc}(\mathcal{M}')}^{\{((\gamma', r), j)\}}$. Since $\text{Idle}(c')((\gamma, r), j) \geq 1$, this implies that necessarily we have $r = j$ (from the induction hypothesis). Thus, all the conditions of Lemma A.1 are satisfied.

□

The Only if direction of Lemma 3.3: In the following, we show that if a state q is k -bounded reachable by \mathcal{M} , then q is also reachable by \mathcal{M}' .

Lemma A.2. *If $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{T_{[0, k]}(\mathcal{M})} c$, then there is $\tau' \in (\Delta')^*$ such that $c_{\mathcal{M}'}^{\text{init}} \xrightarrow{\tau'}^*_{\mathcal{T}(\mathcal{M}')} c'$ where the configuration $c' \in \text{Conf}(\mathcal{M}')$ is defined as follows:*

- $\text{State}(c') = \text{State}(c)$.
- If $\text{Active}(c) = \perp$, then $\text{Active}(c') = \perp$.
- If $\text{Active}(c) = (w, i)$ for some $w \in \Gamma_\epsilon$ and $i \in [0, k]$, then $\text{Active}(c') = ((w, i), i)$.
- $\text{Idle}(c')$ is defined from $\text{Idle}(c)$ as follows:
 - (1) $\text{Idle}(c')((w', j'), j') = \text{Idle}(c)((w', j'))$ for all $w' \in \Gamma_\epsilon$ and $j' \in [0, k + 1]$, and
 - (2) 0 otherwise.

Proof. First, we observe that $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{T_{[0,k]}(\mathcal{M})} c$ implies $\text{Active}(c) = \perp$ or $\text{Active}(c) = (w, i)$ for some $w \in \Gamma_\epsilon$ and $i \in [0, k]$ by definition. Let us assume that $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}_n T_{[0,k]}(\mathcal{M}) c$ for some $n \in \mathbb{N}$. We proceed by induction on n .

Basis. $n = 0$. This implies that $\tau = \epsilon$ and $c_{\mathcal{M}}^{\text{init}} = c = (q_0, \perp, \text{ld}_{\text{Loc}(\mathcal{M})}^{\{(\gamma_0, 0)\}})$. Then, by taking $c' = c_{\mathcal{M}'}^{\text{init}}$ and $\tau' = \epsilon$, all the conditions of Lemma A.2 are satisfied.

Step. $n > 0$. Then there are $c_1 \in \text{Conf}(\mathcal{M})$, $\tau_1 \in \Delta^*$, and $t \in \Delta$ such that:

$$c_{\mathcal{M}}^{\text{init}} \xrightarrow[n-1]{\tau_1} T_{[0,k]}(\mathcal{M}) c_1 \xrightarrow{t} \tau_{[0,k]}(\mathcal{M}) c \quad (\text{A.1})$$

We apply the induction hypothesis to the run $c_{\mathcal{M}}^{\text{init}} \xrightarrow[n-1]{\tau_1} T_{[0,k]}(\mathcal{M}) c_1$, and we obtain that there are $c'_1 \in \text{Conf}(\mathcal{M}')$ and $\tau'_1 \in (\Delta')^*$ such that:

- $c_{\mathcal{M}'}^{\text{init}} \xrightarrow{\tau'_1}^*_{T(\mathcal{M}')} c'_1$.
- $\text{State}(c'_1) = \text{State}(c_1)$.
- If $\text{Active}(c_1) = \perp$, then $\text{Active}(c'_1) = \perp$.
- If $\text{Active}(c_1) = (w, i)$ for some $w \in \Gamma_\epsilon$ and $i \in [0, k]$, then $\text{Active}(c'_1) = ((w, i), i)$.
- The function $\text{Idle}(c'_1)$ is defined from $\text{Idle}(c_1)$ as follows:
 - (1) $\text{Idle}(c'_1)((w', j'), j') = \text{Idle}(c_1)((w', j'))$ for all $w' \in \Gamma_\epsilon$ and $j' \in [0, k+1]$, and
 - (2) 0 otherwise.

Since $c_1 \xrightarrow{t} \tau_{[0,k]}(\mathcal{M}) c$, one of the following four cases holds:

- **Case 1:** $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \epsilon$. Then, there is $i \in [0, k]$ such that $\text{State}(c_1) = q$, $\text{State}(c) = q'$, $\text{Active}(c_1) = (\gamma, i)$, $\text{Active}(c) = (u, i)$, and $\text{Idle}(c_1) = \text{Idle}(c)$. From the definition of \mathcal{M}' , $t' = \langle q, (\gamma, i) \rangle \rightarrow_{\mathcal{M}'} \langle q', (u, i) \rangle \triangleright \epsilon$. Moreover, we have $\text{State}(c'_1) = \text{State}(c_1) = q$ and $\text{Active}(c_1) = ((\gamma, i), i)$. Then, by taking $c' = (q', ((u, i), i), \text{Idle}(c'_1))$ and $\tau' = \tau'_1 t'$, we can show that Lemma A.2 holds.
- **Case 2:** $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \alpha$ with $\alpha \in \Gamma$. Then, there is $i \in [0, k]$ such that $\text{State}(c_1) = q$, $\text{State}(c) = q'$, $\text{Active}(c_1) = (\gamma, i)$, $\text{Active}(c) = (u, i)$, and $\text{Idle}(c) = \text{Idle}(c_1) + \text{ld}_{\text{Loc}(\mathcal{M})}^{\{(\alpha, i+1)\}}$. From the definition of \mathcal{M}' , we have $t' = \langle q, (\gamma, i) \rangle \rightarrow_{\mathcal{M}'} \langle q', (u, i) \rangle \triangleright (\alpha, i+1)$. Then, by taking $c' = (q', ((u, i), i), \text{Idle}(c'_1) + \text{ld}_{\text{Loc}(\mathcal{M}')}^{\{((\alpha, i+1), i+1)\}})$ and $\tau' = \tau'_1 t'$, we can show that Lemma A.2 holds.
- **Case 3:** $t = \langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$. Then, there is $i \in [0, k]$ such that $\text{State}(c_1) = q$, $\text{State}(c) = q'$, $\text{Active}(c_1) = (\gamma, i)$, $\text{Active}(c) = \perp$, and $\text{Idle}(c) = \text{Idle}(c_1) + \text{ld}_{\text{Loc}(\mathcal{M})}^{\{(u, i+1)\}}$. From the definition of \mathcal{M}' , we have $t' = \langle q, (\gamma, i) \rangle \mapsto_{\mathcal{M}'} \langle q', (u, i+1) \rangle$. Then, by taking $c' = (q', \perp, \text{Idle}(c'_1) + \text{ld}_{\text{Loc}(\mathcal{M}')}^{\{((u, i+1), i+1)\}})$ and $\tau' = \tau'_1 t'$, we can show that Lemma A.2 holds.
- **Case 4:** $t = q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$ with $\gamma \in \Gamma$. Then, there is $i \in [0, k]$ such that $\text{State}(c_1) = q$, $\text{State}(c) = q'$, $\text{Active}(c_1) = \perp$, $\text{Active}(c) = (\gamma, i)$, $\text{Idle}(c_1)((\gamma, i)) \geq 1$, and $\text{Idle}(c) = \text{Idle}(c_1) - \text{ld}_{\text{Loc}(\mathcal{M})}^{\{(\gamma, i)\}}$. From the definition of \mathcal{M}' , we have $t' = q \mapsto_{\mathcal{M}'} q' \triangleleft (\gamma, i)$. Then, by taking $c' = (q', ((\gamma, i), i), \text{Idle}(c'_1) - \text{ld}_{\text{Loc}(\mathcal{M}')}^{\{((\gamma, i), i)\}})$ and $\tau' = \tau'_1 t'$, we can show that all the conditions of Lemma A.2 are satisfied. This is possible since $\text{Idle}(c'_1)((\gamma, i), i) = \text{Idle}(c_1)((\gamma, i)) \geq 1$.

□

The If direction of Lemma 3.3 : In the following, we shows that if a state q is reachable by a computation of \mathcal{M}' , then q is k -bounded reachable by \mathcal{M} .

Lemma A.3. *If $c_{\mathcal{M}'}^{\text{init}} \xrightarrow{\tau'}^*_{\mathcal{T}(\mathcal{M}')} c'$, then there is $\tau \in \Delta^*$ such that $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}_{[0,k]}(\mathcal{M})} c$ where the configuration $c \in \text{Conf}(\mathcal{M})$ is defined as follows:*

- $\text{State}(c) = \text{State}(c')$.
- If $\text{Active}(c') = \perp$, then $\text{Active}(c) = \perp$.
- If $\text{Active}(c') = ((w, i), i)$ for some $w \in \Gamma_\epsilon$ and $i \in [0, k]$, then $\text{Active}(c) = (w, i)$.
- $\text{Idle}(c)$ is defined from $\text{Idle}(c')$ as follows:
 - (1) $\text{Idle}(c)((w', j')) = \text{Idle}(c')(((w', j'), j'))$ for all $w' \in \Gamma_\epsilon$ and $j' \in [0, k + 1]$, and
 - (2) 0 otherwise.

Proof. First, we observe that if $c_{\mathcal{M}'}^{\text{init}} \xrightarrow{\tau'}^*_{\mathcal{T}(\mathcal{M}')} c'$, then by Lemma A.1 $\text{Active}(c') = \perp$ or $\text{Active}(c') = ((w, i), i)$ for some $w \in \Gamma_\epsilon$ and $i \in [0, k]$. Let us assume that $c_{\mathcal{M}'}^{\text{init}} \xrightarrow{\tau'}^*_n \mathcal{T}(\mathcal{M}') c'$ for some $n \in \mathbb{N}$. We proceed by induction on n .

Basis. $n = 0$. Then, $\tau' = \epsilon$ and $c_{\mathcal{M}'}^{\text{init}} = c' = (q_0, \perp, \text{ld}_{\text{Loc}(\mathcal{M}')}^{\{((\gamma_0, 0), 0)\}})$. By taking $c = c_{\mathcal{M}}^{\text{init}} = (q_0, \perp, \text{ld}_{\text{Loc}(\mathcal{M})}^{\{(\gamma_0, 0)\}})$ and $\tau = \epsilon$, we can show that all the conditions of Lemma A.3 are fulfilled.

Step. $n > 1$. Then, there are $\tau'_1 \in (\Delta')^*$, $t' \in \Delta'$, and $c'_1 \in \text{Conf}(\mathcal{M}')$ such that:

$$c_{\mathcal{M}'}^{\text{init}} \xrightarrow{\tau'_1}_{n-1} \mathcal{T}(\mathcal{M}') c'_1 \xrightarrow{t'} \mathcal{T}(\mathcal{M}') c' \quad (\text{A.2})$$

We apply Lemma A.1 to $c_{\mathcal{M}'}^{\text{init}} \xrightarrow{\tau'_1}_{n-1} \mathcal{T}(\mathcal{M}') c'_1$ and $c_{\mathcal{M}'}^{\text{init}} \xrightarrow{\tau'_1}_n \mathcal{T}(\mathcal{M}') c'$, and we obtain that:

- $\text{Active}(c')$, $\text{Active}(c'_1) \in (\{\perp\} \cup \{((w, i), i) \mid w \in \Gamma_\epsilon, i \in [0, k]\})$,
- $\text{Idle}(c'_1)((\epsilon, l)) = \text{Idle}(c')((\epsilon, l)) = 0$ for all $l \in \mathbb{N}$, and
- $\text{Idle}(c'_1)((w', i'), j') = \text{Idle}(c')((w', i'), j') = 0$ for all $w' \in \Gamma_\epsilon$ and $i' \neq j'$.

We apply also the induction hypothesis to $c_{\mathcal{M}'}^{\text{init}} \xrightarrow{\tau'_1}_{n-1} \mathcal{T}(\mathcal{M}') c'_1$, and we obtain that there are $\tau_1 \in \Delta^*$ and $c_1 \in \text{Conf}(\mathcal{M})$ such that:

- $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau_1}^*_{\mathcal{T}_{[0,k]}(\mathcal{M})} c_1$.
- $\text{State}(c_1) = \text{State}(c'_1)$.
- If $\text{Active}(c'_1) = \perp$, then $\text{Active}(c_1) = \perp$.
- If $\text{Active}(c'_1) = ((w, i), i)$ for some $w \in \Gamma_\epsilon$ and $i \in [0, k]$, then $\text{Active}(c_1) = (w, i)$.
- The function $\text{Idle}(c_1)$ is defined from $\text{Idle}(c'_1)$ as follows:
 - (1) $\text{Idle}(c_1)((w', j')) = \text{Idle}(c'_1)(((w', j'), j'))$ for all $w' \in \Gamma_\epsilon$ and $j' \in [0, k + 1]$, and
 - (2) 0 otherwise.

On the other hand, $c'_1 \xrightarrow{t'} \mathcal{T}(\mathcal{M}') c'$ implies that one of the following four cases holds:

- **Case 1:** $t' = \langle q, (\gamma, i) \rangle \rightarrow_{\mathcal{M}'} \langle q', (u, i) \rangle \triangleright \epsilon$ with $i \in [0, k]$. Then, $\text{State}(c'_1) = q$, $\text{State}(c') = q'$, $\text{Active}(c'_1) = ((\gamma, i), i)$, $\text{Active}(c') = ((u, i), i)$, and $\text{Idle}(c'_1) = \text{Idle}(c')$. We can use the definition of \mathcal{M}' to show that $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \epsilon$. Then, by taking $c = (q', (u, i), \text{Idle}(c_1))$ and $\tau = \tau_1 t$, we can show that Lemma A.3 holds.

- **Case 2:** $t' = \langle q, (\gamma, i) \rangle \rightarrow_{\mathcal{M}'} \langle q', (u, i) \rangle \triangleright (\alpha, i + 1)$ with $i \in [0, k]$ and $\alpha \in \Gamma$. Then, $State(c'_1) = q$, $State(c') = q'$, $Active(c'_1) = ((\gamma, i), i)$, $Active(c') = ((u, i), i)$, and $Idle(c') = Idle(c'_1) + \text{Id}_{Loc(\mathcal{M}')}^{\{((\alpha, i+1), i+1)\}}$. The definition of \mathcal{M}' implies $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \alpha$. Then, by taking $c = (q', (u, i), \text{Idle}(c_1) + \text{Id}_{Loc(\mathcal{M})}^{\{(\alpha, i+1)\}})$ and $\tau = \tau_1 t$, we can show that Lemma A.3 holds.
- **Case 3:** $t' = \langle q, (\gamma, i) \rangle \mapsto_{\mathcal{M}'} \langle q', (u, i+1) \rangle$ with $i \in [0, k]$. Then, $State(c'_1) = q$, $State(c') = q'$, $Active(c'_1) = ((\gamma, i), i)$, $Active(c') = \perp$, and $Idle(c') = Idle(c'_1) + \text{Id}_{Loc(\mathcal{M}')}^{\{((u, i+1), i+1)\}}$. We can use the definition of \mathcal{M}' to show that $t = \langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$. Then, by taking $c = (q', \perp, \text{Idle}(c_1) + \text{Id}_{Loc(\mathcal{M})}^{\{(u, i+1)\}})$ and $\tau = \tau_1 t$, we can show that all the conditions of Lemma A.3 are fulfilled.
- **Case 4:** $t' = q \mapsto_{\mathcal{M}'} q' \triangleleft (\gamma, i)$ with $i \in [0, k]$. Then, $State(c'_1) = q$, $State(c') = q'$, $Active(c'_1) = \perp$, $Active(c') = ((\gamma, i), i)$, $Idle(c'_1)((\gamma, i), i) \geq 1$, and $Idle(c') = Idle(c'_1) - \text{Id}_{Loc(\mathcal{M}')}^{\{((\gamma, i), i)\}}$. This is due to the fact that $Idle(c'_1)((\gamma, i), j) = 0$ for all $j \in \mathbb{N}$ such that $i \neq j$. We can use the definition of \mathcal{M}' to show that $t = q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$. Then, by taking $c = (q', (\gamma, i), \text{Idle}(c_1) - \text{Id}_{Loc(\mathcal{M})}^{\{(\gamma, i)\}})$ and $\tau = \tau_1 t$, we can easily show that all the conditions of Lemma A.3 are fulfilled. This is possible since $Idle(c_1)((\gamma, i)) = Idle(c'_1)((\gamma, i), i)$ and $Idle(c'_1)((\gamma, i), i) \geq 1$.

□

As an immediate consequence of Lemma A.2 and A.3, we obtain that for every state $q \in Q$, q is k -bounded reachable by \mathcal{M} iff q is reachable by \mathcal{M}' . □

APPENDIX B. THE PROOF OF LEMMA 3.5

Lemma 3.5 Let $q \in Q$. q is reachable by \mathcal{M} if and only if (q, \perp) is reachable by \mathcal{V} .

Proof. To prove Lemma 3.5, we proceed as follows: First, we introduce the function μ which defines a simulation relation between \mathcal{M} and \mathcal{V} (see Definition B.1). Then, we show that if a state q is reachable by \mathcal{M} , then (q, \perp) is also reachable by \mathcal{V} (see Lemma B.2). Finally, we prove that if (q, \perp) is reachable by \mathcal{V} , then q is reachable by \mathcal{M} (see Lemma B.3).

The simulation relation between \mathcal{V} and \mathcal{M} : Let us define the function μ which maps every configuration of \mathcal{M} to a configuration of \mathcal{V} .

Definition B.1. Let μ be a function from $Conf(\mathcal{M})$ to $Conf(\mathcal{V})$ such that: For every $c \in Conf(\mathcal{M})$, we have $\mu(c) = ((q, \eta), \mathbf{u})$ where:

- $q = State(c)$,
- $\eta = \perp$ if $Active(c) = \perp$,
- $\eta = w$ if $Active(c) = (w, i)$ for some $w \in \Gamma_\epsilon$ and $i \in \mathbb{N}$,
- $\mathbf{u}[i] = \sum_{j \in \mathbb{N}} Idle(c)((\gamma_{i-1}, j))$ for all $i \in [1, m[$, and (2) $\mathbf{u}[m] = \sum_{j \in \mathbb{N}} Idle(c)((\epsilon, j))$.

The Only if direction of Lemma 3.5 : In the following, we show that if a state q is reachable by \mathcal{M} , then (q, \perp) is also reachable by \mathcal{V} .

Lemma B.2. *If $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{M})} c$, then $((p_0, \perp), \mathbf{u}_0) \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{V})} \mu(c)$.*

Proof. We use induction on the length of the run $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{M})} c$. For some $\ell \in \mathbb{N}$ assume that $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\ell}^{\tau} \mathcal{T}(\mathcal{M}) c$. We proceed by induction on ℓ .

Basis. $\ell = 0$. Then $c = c_{\mathcal{M}}^{\text{init}}$ and $\tau = \epsilon$. Moreover, we have $\mu(c_{\mathcal{M}}^{\text{init}}) = ((q_0, \perp), \mathbf{u}_0)$. This implies that $((q_0, \perp), \mathbf{u}_0) \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{V})} \mu(c)$ holds.

Step. $\ell > 0$. Then there are $c' \in \text{Conf}(\mathcal{M})$, $\tau' \in \Sigma^*$, and $t \in \Sigma$ such that $\tau = \tau' t$ and:

$$c_{\mathcal{M}}^{\text{init}} \xrightarrow{\ell-1}^{\tau'} \mathcal{T}(\mathcal{M}) c' \xrightarrow{t} \mathcal{T}(\mathcal{M}) c \quad (\text{B.1})$$

We apply the induction hypothesis to the run $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\ell-1}^{\tau'} \mathcal{T}(\mathcal{M}) c'$, and we obtain:

$$((q_0, \perp), \mathbf{u}_0) \xrightarrow{\tau'}^*_{\mathcal{T}(\mathcal{V})} \mu(c') \quad (\text{B.2})$$

Let us assume that $\mu(c') = \varsigma'$ and $\mu(c) = \varsigma$. Since $c' \xrightarrow{t} \mathcal{T}(\mathcal{M}) c$, then one of the following cases holds:

- **Case 1:** If $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \epsilon$. Then, there is $i \in \mathbb{N}$ such that $\text{State}(c') = q$, $\text{State}(c) = q'$, $\text{Active}(c') = (\gamma, i)$, $\text{Active}(c) = (u, i)$, and $\text{Idle}(c) = \text{Idle}(c')$. We can use the definition of μ to show that $\text{State}(\varsigma') = (q, \gamma)$, $\text{State}(\varsigma) = (q', u)$, and $\text{Val}(\varsigma) = \text{Val}(\varsigma')$. Moreover, from the definition of \mathcal{V} , we have $\delta((q, \gamma), t) = ((q', u), \mathbf{0}^m)$. This implies that $\varsigma' \xrightarrow{t} \mathcal{T}(\mathcal{V}) \varsigma$, and so we obtain $((p_0, \perp), \mathbf{u}_0) \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{V})} \mu(c)$.
- **Case 2:** If $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \gamma_{j-1}$ for some $j \in [1, m[$. Then, there is $i \in \mathbb{N}$ such that $\text{State}(c') = q$, $\text{State}(c) = q'$, $\text{Active}(c') = (\gamma, i)$, $\text{Active}(c) = (u, i)$, and $\text{Idle}(c) = \text{Idle}(c') + \text{Id}_{\text{Loc}(\mathcal{M})}^{\{(\gamma_{j-1}, i+1)\}}$. We can use the definition of μ to show that $\text{State}(\varsigma') = (q, \gamma)$, $\text{State}(\varsigma) = (q', u)$, and $\text{Val}(\varsigma) = \text{Val}(\varsigma')[j \leftrightarrow (\text{Val}(\varsigma')[j] + 1)]$. Moreover, from the definition of \mathcal{V} , we have $\delta((q, \gamma), t) = ((q', u), \mathbf{0}^m[j \leftrightarrow 1])$. This implies that $\varsigma' \xrightarrow{t} \mathcal{T}(\mathcal{V}) \varsigma$, and so we obtain $((p_0, \perp), \mathbf{u}_0) \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{V})} \mu(c)$.
- **Case 3:** If $t = \langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', \epsilon \rangle$. Then, there is $i \in \mathbb{N}$ such that $\text{State}(c') = q$, $\text{State}(c) = q'$, $\text{Active}(c') = (\gamma, i)$, $\text{Active}(c) = \perp$, and $\text{Idle}(c) = \text{Idle}(c') + \text{Id}_{\text{Loc}(\mathcal{M})}^{\{(\epsilon, i+1)\}}$. We can use the definition of μ to show that $\text{State}(\varsigma') = (q, \gamma)$, $\text{State}(\varsigma) = (q', \perp)$, and $\text{Val}(\varsigma) = \text{Val}(\varsigma')[m \leftrightarrow (\text{Val}(\varsigma')[m] + 1)]$. Moreover, from the definition of \mathcal{V} , we have $\delta((q, \gamma), t) = ((q', \perp), \mathbf{0}^m[m \leftrightarrow 1])$. This implies that $\varsigma' \xrightarrow{t} \mathcal{T}(\mathcal{V}) \varsigma$, and so we obtain $((p_0, \perp), \mathbf{u}_0) \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{V})} \mu(c)$.
- **Case 4:** If $t = \langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', \gamma_{j-1} \rangle$ for some $j \in [1, m[$. Then, there is $i \in \mathbb{N}$ such that $\text{State}(c') = q$, $\text{State}(c) = q'$, $\text{Active}(c') = (\gamma, i)$, $\text{Active}(c) = \perp$, and $\text{Idle}(c) = \text{Idle}(c') + \text{Id}_{\text{Loc}(\mathcal{M})}^{\{(\gamma_{j-1}, i+1)\}}$. We can use the definition of μ to show that $\text{State}(\varsigma') = (q, \gamma)$, $\text{State}(\varsigma) = (q', \perp)$, and $\text{Val}(\varsigma) = \text{Val}(\varsigma')[j \leftrightarrow (\text{Val}(\varsigma')[j] + 1)]$. Moreover, from the definition of \mathcal{V} , we have $\delta((q, \gamma), t) = ((q', \perp), \mathbf{0}^m[j \leftrightarrow 1])$. This implies that $\varsigma' \xrightarrow{t} \mathcal{T}(\mathcal{V}) \varsigma$, and so we obtain $((p_0, \perp), \mathbf{u}_0) \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{V})} \mu(c)$.

- **Case 5:** If $t = q \mapsto_{\mathcal{M}} q' \triangleleft \gamma_{j-1}$ for some $j \in [1, m[$. Then, there is $i \in \mathbb{N}$ such that $State(c') = q$, $State(c) = q'$, $Active(c') = \perp$, $Active(c) = (\gamma_{j-1}, i)$, $Idle(c')((\gamma_{j-1}, i)) \geq 1$, and $Idle(c) = Idle(c') - \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma_{j-1}, i)\}}$. We can use the definition of μ to show $State(c') = (q, \perp)$, $State(c) = (q', \gamma_{j-1})$, $Val(c')[j] \geq 1$, and $Val(c) = Val(c')[j] \leftarrow (Val(c')[j] - 1)$. Moreover, from the definition of \mathcal{V} , we have $\delta((q, \perp), t) = ((q', \gamma_{j-1}), \mathbf{0}^m[j \leftarrow -1])$. This implies that $c' \xrightarrow{t} \mathcal{T}(\mathcal{V}) c$, and so we obtain $((p_0, \perp), \mathbf{u}_0) \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{V})} \mu(c)$.

□

The If direction of Lemma 3.5: In the following, we prove that if (q, \perp) is reachable by \mathcal{V} , then q is reachable by \mathcal{M} .

Lemma B.3. *Let $\varsigma \in (Q \times (\Gamma_\epsilon \cup \{\perp\})) \times \mathbb{N}^m$ and $\tau \in \Sigma^*$. If $((q_0, \perp), \mathbf{u}_0) \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{V})} \varsigma$, then there is $c \in Conf(\mathcal{M})$ such that $\varsigma = \mu(c)$ and $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{M})} c$.*

Proof. We use induction on the length of the run $p_0 \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{V})} \varsigma$. For some $\ell \in \mathbb{N}$ assume $((q_0, \perp), \mathbf{u}_0) \xrightarrow[\ell]{\tau} \mathcal{T}(\mathcal{V}) \varsigma$. We proceed by induction on ℓ .

Basis. $\ell = 0$. Then, $((q_0, \perp), \mathbf{u}_0) = \varsigma$ and $\tau = \epsilon$. By taking $c = c_{\mathcal{M}}^{\text{init}}$, we have $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{M})} c$.

Moreover, using the definition of μ , we have $\mu(c_{\mathcal{M}}^{\text{init}}) = \mu(c) = ((q_0, \perp), \mathbf{u}_0) = \varsigma$.

Step. $\ell > 0$. Then, there are $\varsigma' \in Conf(\mathcal{V})$, $\tau' \in \Sigma^*$, and $t \in \Sigma$ such that $\tau = \tau' t$ and $((q_0, \perp), \mathbf{u}_0) \xrightarrow[\ell-1]{\tau'} \mathcal{T}(\mathcal{V}) \varsigma' \xrightarrow{t} \mathcal{T}(\mathcal{V}) \varsigma$. Moreover, we can assume that $State(\varsigma') \in Q \times (\Gamma_\epsilon \cup \{\perp\})$ since $State(\varsigma) \in Q \times (\Gamma_\epsilon \cup \{\perp\})$ (see the definition of the transition function of \mathcal{V}).

We apply now the induction hypothesis to the run $((q_0, \perp), \mathbf{u}_0) \xrightarrow[\ell-1]{\tau'} \mathcal{T}(\mathcal{V}) \varsigma'$, and we obtain that there is a configuration $c' \in Conf(\mathcal{M})$ such that $\mu(c') = \varsigma'$ and $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau'}^*_{\mathcal{T}(\mathcal{M})} c'$. On the other hand, the run $\varsigma' \xrightarrow{t} \mathcal{T}(\mathcal{M}) \varsigma$ implies that one of the following cases holds:

- **Case 1:** If $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \epsilon$. Then, from the definition of \mathcal{V} , we have $State(\varsigma') = (q, \gamma)$, $State(\varsigma) = (q', u)$, and $Val(\varsigma) = Val(\varsigma')$. Moreover, from the definition of the function μ , we know that there is $i \in \mathbb{N}$ such that $State(c') = q$ and $Active(c') = (\gamma, i)$. Let $c = (q, (u, i), Idle(c'))$. Then, $c' \xrightarrow{t} \mathcal{T}(\mathcal{M}) c$ and $\mu(c) = \varsigma$. So, we obtain $\varsigma = \mu(c)$ and $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{M})} c$.
- **Case 2:** If $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \gamma_{j-1}$ for some $j \in [1, m[$. Then, from the definition of \mathcal{V} , we have $State(\varsigma') = (q, \gamma)$, $State(\varsigma) = (q', u)$, and $Val(\varsigma) = Val(\varsigma')[j] \leftarrow (Val(\varsigma')[j] + 1)$. Moreover, from the definition of the function μ , we know that there is $i \in \mathbb{N}$ such that $State(c') = q$ and $Active(c') = (\gamma, i)$. Let $c = (q, (u, i), Idle(c') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma_{j-1}, i+1)\}})$. Then, $c' \xrightarrow{t} \mathcal{T}(\mathcal{M}) c$ and $\mu(c) = \varsigma$. So, we obtain $\varsigma = \mu(c)$ and $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{M})} c$.
- **Case 3:** If $t = \langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', \epsilon \rangle$. Then, from the definition of \mathcal{V} , we have $State(\varsigma') = (q, \gamma)$, $State(\varsigma) = (q', \perp)$, and $Val(\varsigma) = Val(\varsigma')[m] \leftarrow (Val(\varsigma')[m] + 1)$. Moreover, from the definition of the function μ , we know that there is $i \in \mathbb{N}$ such that $State(c') = q$ and $Active(c') = (\gamma, i)$. Let $c = (q, \perp, Idle(c') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\epsilon, i+1)\}})$. Then, $c' \xrightarrow{t} \mathcal{T}(\mathcal{M}) c$ and $\mu(c) = \varsigma$. So, we obtain $\varsigma = \mu(c)$ and $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}(\mathcal{M})} c$.
- **Case 4:** If $t = \langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', \gamma_{j-1} \rangle$ for some $j \in [1, m[$. Then, from the definition of \mathcal{V} , we have $State(\varsigma') = (q, \gamma)$, $State(\varsigma) = (q', \perp)$, and $Val(\varsigma) = Val(\varsigma')[j] \leftarrow (Val(\varsigma')[j] + 1)$.

Moreover, from the definition of the function μ , we know that there is $i \in \mathbb{N}$ such that $State(c') = q$ and $Active(c') = (\gamma, i)$. Let $c = (q, \perp, Idle(c') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma_{j-1}, i+1)\}})$. Then, $c' \xrightarrow{t} \mathcal{T}(\mathcal{M}) c$ and $\mu(c) = \varsigma$. So, we obtain $\varsigma = \mu(c)$ and $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^* \mathcal{T}(\mathcal{M}) c$.

- **Case 5:** If $t = q \mapsto_{\mathcal{M}} q' \triangleleft \gamma_{j-1}$ for some $j \in [1, m[$. Then, from the definition of \mathcal{V} , we have $State(c') = (q, \perp)$, $State(\varsigma) = (q', \gamma_{j-1})$, $Val(c')[j] \geq 1$, and $Val(\varsigma) = Val(c')[j \leftarrow (Val(c')[j] - 1)]$. Moreover, from the definition of the function μ , we know that there is $i \in \mathbb{N}$ such that $State(c') = q$, $Active(c') = \perp$, and $Idle(c')((\gamma_{j-1}, i)) \geq 1$. Let $c = (q, (\gamma_{j-1}, i), Idle(c') - \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma_{j-1}, i)\}})$. Then, c is well defined, $c' \xrightarrow{t} \mathcal{T}(\mathcal{M}) c$, and $\mu(c) = \varsigma$. So, we obtain $\varsigma = \mu(c)$ and $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^* \mathcal{T}(\mathcal{M}) c$. \square

Hence Lemma 3.5 is an immediate consequence of Lemma B.2 and Lemma B.3. \square

APPENDIX C. THE PROOF OF LEMMA 3.7

Lemma 3.7 Let $q \in Q$. q is reachable by \mathcal{V} if and only if q is 2-bounded reachable by \mathcal{M} .

Proof. To prove Lemma 3.7, we proceed as follows: First, we prove that if $q \in Q$ is reachable by \mathcal{V} , then q is 2-bounded reachable by \mathcal{M} (see Lemma C.1). Then, we show that if $q \in Q$ is 2-bounded reachable by \mathcal{M} , then q is reachable by \mathcal{V} (see Lemma C.2).

The If direction of Lemma 3.7: In the following, we show if $q \in Q$ is reachable by \mathcal{V} , then q is 2-bounded reachable by \mathcal{M} .

Lemma C.1. *If $(q_0, \mathbf{0}^n) \xrightarrow{\sigma}^* \mathcal{T}(\mathcal{V}) (q, \mathbf{u})$, then for every $m \in \mathbb{N}$, there are $\tau \in \Delta^*$ and $Val \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ such that: (1) $Val((\gamma_i, 2)) = \mathbf{u}[i]$ for all $i \in [1, n]$, (2) $Val((\gamma'_0, 1)) = m$, and (3) $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^* \mathcal{T}_{[0,2]}(\mathcal{M}) (q, \perp, Val)$.*

Proof. We use induction on the length of the run $(q_0, \mathbf{0}^n) \xrightarrow{\sigma}^* \mathcal{T}(\mathcal{V}) (q, \mathbf{u})$. For some $\ell \in \mathbb{N}$ assume that $(q_0, \mathbf{0}^n) \xrightarrow{\sigma}_{\ell} \mathcal{T}(\mathcal{V}) (q, \mathbf{u})$. We proceed by induction on ℓ .

Basis. $\ell = 0$. Then $\sigma = \epsilon$, $q = q_0$ and $\mathbf{u} = \mathbf{0}^n$. It is easy to observe that for every $m \in \mathbb{N}$, $\mathcal{T}_{[0,2]}(\mathcal{M})$, from the initial configuration $c_{\mathcal{M}}^{\text{init}}$, can apply m -times the transition $t_0 = \langle p_0, \gamma_0 \rangle \rightarrow_{\mathcal{M}} \langle p_0, \gamma_0 \rangle \triangleright \gamma'_0$ followed by the transition $t'_0 = \langle p_0, \gamma_0 \rangle \mapsto_{\mathcal{M}} \langle q_0, \epsilon \rangle$ to reach the configuration (q_0, \perp, Val) (i.e., $c_{\mathcal{M}}^{\text{init}} \xrightarrow{t_0^m, t'_0}^* \mathcal{T}_{[0,2]}(\mathcal{M}) (q_0, \perp, Val)$) with $Val((\gamma'_0, 1)) = m$ and $Val((\gamma_i, 2)) = \mathbf{u}[i]$ for all $i \in [1, n]$.

Step. $\ell > 0$. Then, there are $q' \in Q$, $\mathbf{u}' \in \mathbb{N}^n$, $\sigma' \in \Sigma^*$, and $a \in \Sigma$ such that $\sigma = \sigma'a$ and:

$$(q_0, \mathbf{0}^n) \xrightarrow{\sigma'}_{\ell-1} \mathcal{T}(\mathcal{V}) (q', \mathbf{u}') \xrightarrow{a} \mathcal{T}(\mathcal{V}) (q, \mathbf{u}) \quad (\text{C.1})$$

We apply the induction hypothesis to $(q_0, \mathbf{0}^n) \xrightarrow{\sigma'}_{\ell-1} \mathcal{T}(\mathcal{V}) (q', \mathbf{u}')$, and we obtain that:

$\forall m' \in \mathbb{N}$, $\exists \tau' \in \Delta^*$ and $\exists Val' \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ s.t.:

$$\begin{aligned} c_{\mathcal{M}}^{\text{init}} &\xrightarrow{\tau'}^* \mathcal{T}_{[0,2]}(\mathcal{M})(q', \perp, Val') \\ Val'((\gamma'_0, 1)) &= m' \\ Val'((\gamma_i, 2)) &= \mathbf{u}'[i], \forall i \in [1, n] \end{aligned} \quad (\text{C.2})$$

Moreover, we have $(q', \mathbf{u}') \xrightarrow{a} \mathcal{T}_{(\mathcal{V})}(q, \mathbf{u})$. This implies that $\delta(q', a) = (q, \mathbf{u} - \mathbf{u}')$, and one of the following cases holds:

- **Case 1:** If $\mathbf{u}' = \mathbf{u}$, then $t = q' \mapsto_{\mathcal{M}} q' \triangleleft \gamma'_0$, $t' = \langle q', \gamma'_0 \rangle \rightarrow_{\mathcal{M}} \langle q, \gamma'_0 \rangle \triangleright \epsilon$, and $t'' = \langle q, \gamma_0 \rangle \mapsto_{\mathcal{M}} \langle q, \epsilon \rangle$. This implies that for every $Val \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$, $\mathcal{T}_{[0,2]}(\mathcal{M})$ can move from the configuration $(q', \perp, Val + \text{Id}_{Loc(\mathcal{M})}^{\{(\gamma'_0, 1)\}})$ to the configuration $(q, \perp, Val + \text{Id}_{Loc(\mathcal{M})}^{\{(\epsilon, 2)\}})$. Now, we can use Equations C.2, to show that:

$\forall m \in \mathbb{N}$, $\exists \tau' \in \Delta^*$ and $\exists Val \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ s.t.:

$$\begin{aligned} c_{\mathcal{M}}^{\text{init}} &\xrightarrow{\tau' t t''}^* \mathcal{T}_{[0,2]}(\mathcal{M})(q, \perp, Val) \\ Val((\gamma'_0, 1)) &= m \\ Val((\gamma_i, 2)) &= \mathbf{u}[i], \forall i \in [1, n] \end{aligned} \quad (\text{C.3})$$

- **Case 2:** If $\mathbf{u} = \mathbf{u}'[j \leftrightarrow (\mathbf{u}'[j] + 1)]$ for some $j \in [1, n]$, then we have that $t = q' \mapsto_{\mathcal{M}} q' \triangleleft \gamma'_0$, $t' = \langle q', \gamma'_0 \rangle \rightarrow_{\mathcal{M}} \langle q, \gamma'_0 \rangle \triangleright \gamma_j$, and $t'' = \langle q, \gamma_0 \rangle \mapsto_{\mathcal{M}} \langle q, \epsilon \rangle$. This implies that for every $Val'' \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$, $\mathcal{T}_{[0,2]}(\mathcal{M})$ can move from the configuration $(q', \perp, Val'' + \text{Id}_{Loc(\mathcal{M})}^{\{(\gamma'_0, 1)\}})$ to the configuration (q, \perp, Val) with $Val = Val'' + \text{Id}_{Loc(\mathcal{M})}^{\{(\epsilon, 2), (\gamma_j, 2)\}}$. Now, we can use Equations C.2, to show that:

$\forall m \in \mathbb{N}$, $\exists \tau' \in \Delta^*$ and $\exists Val \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ s.t.:

$$\begin{aligned} c_{\mathcal{M}}^{\text{init}} &\xrightarrow{\tau' t t''}^* \mathcal{T}_{[0,2]}(\mathcal{M})(q, \perp, Val) \\ Val((\gamma'_0, 1)) &= m \\ Val((\gamma_i, 2)) &= \mathbf{u}[i], \forall i \in [1, n] \end{aligned} \quad (\text{C.4})$$

- **Case 3:** If $\mathbf{u} = \mathbf{u}'[j \leftrightarrow (\mathbf{u}'[j] - 1)]$ and $\mathbf{u}'[j] \geq 1$ for some $j \in [1, n]$, then $t = q' \mapsto_{\mathcal{M}} q \triangleleft \gamma_j$, and $t' = \langle q, \gamma_j \rangle \mapsto_{\mathcal{M}} \langle q, \epsilon \rangle$. This implies that for every $Val'' \in [Conf_{\text{loc}}(\mathcal{M}) \rightarrow \mathbb{N}]$ such that $Val''((\gamma_j, 2)) \geq 1$, $\mathcal{T}_{[0,2]}(\mathcal{M})$ can move from the configuration (q', \perp, Val'') to the configuration (q, \perp, Val) with $Val = Val'' + \text{Id}_{Loc(\mathcal{M})}^{\{(\epsilon, 3)\}} - \text{Id}_{Loc(\mathcal{M})}^{\{(\gamma_j, 2)\}}$. Now, we can use Equations C.2, to show that:

$\forall m \in \mathbb{N}$, $\exists \tau' \in \Delta^*$ and $\exists Val \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ s.t.:

$$\begin{aligned} c_{\mathcal{M}}^{\text{init}} &\xrightarrow{\tau' t t''}^* \mathcal{T}_{[0,2]}(\mathcal{M})(q, \perp, Val) \\ Val((\gamma'_0, 1)) &= m \\ Val((\gamma_i, 2)) &= \mathbf{u}'[i], \forall i \in [1, n] \end{aligned} \quad (\text{C.5})$$

(This is possible since $Val'((\gamma_j, 2)) = \mathbf{u}'[j] \geq 1$.)

□

The Only if direction of Lemma 3.7: In the following, we show that if $q \in Q$ is 2-bounded reachable by \mathcal{M} , then q is reachable by \mathcal{V} .

Lemma C.2. *If $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}_{[0,2]}(\mathcal{M})} c$ for some $\tau \in \Delta^*$ and $c \in \text{Conf}(\mathcal{M})$ such that $\text{State}(c) \in Q$, then the following conditions are satisfied:*

- (1) $\text{Active}(c) \in (\{\perp\} \cup \{(\gamma'_0, 1)\}) \cup ((\Gamma \setminus \{\gamma_0, \gamma'_0\}) \times \{2\})$,
- (2) $\text{Idle}(c)((\gamma_0, i)) = 0$ for all $i \in \mathbb{N}$,
- (3) $\text{Idle}(c)((\gamma_j, i)) = 0$ for all $j \in [1, n]$ and $i \neq 2$,
- (4) $\text{Idle}(c)((\gamma'_0, i)) = 0$ for all $i \neq 1$, and
- (5) there is $\sigma \in \Sigma^*$ with $(q_0, \mathbf{0}^n) \xrightarrow{\sigma}^*_{\mathcal{T}(\mathcal{V})} (q, \mathbf{u})$, where $q = \text{State}(c)$ and $\mathbf{u}[j] = \text{Idle}(c)((\gamma_j, 2))$ for all $j \in [1, n]$.

Proof. Again, we use induction. Let us assume that $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}_{[0,2]}(\mathcal{M})} c$ for some $\tau \in \Delta^*$ and $c \in \text{Conf}(\mathcal{M})$ such that $\text{State}(c) \in Q$. Then, from the definition of \mathcal{M} , there are $\tau_1, \tau_2 \in \Delta^*$ and $m \in \mathbb{N}$ such that $\tau = \tau_1 \tau_2$ and $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau_1}^*_{\mathcal{T}_{[0,2]}(\mathcal{M})} (q_0, \perp, \text{Val}_0) \xrightarrow{\tau_2}^*_{\mathcal{T}_{[0,2]}(\mathcal{M})} c$ with $\text{Val}_0((\gamma'_0, 1)) = m$ and $\text{Val}_0((\alpha, j)) = 0$ for all $(\alpha, j) \in \Gamma \times \mathbb{N}$ such that $(\alpha, j) \neq (\gamma'_0, 1)$. Since $(q_0, \perp, \text{Val}_0) \xrightarrow{\tau_2}^*_{\mathcal{T}_{[0,2]}(\mathcal{M})} c$, then there is $\ell \in \mathbb{N}$ such that $(q_0, \perp, \text{Val}_0) \xrightarrow{\tau_2}^*_{\ell} \mathcal{T}_{[0,2]}(\mathcal{M}) c$.

To prove Lemma C.2, we proceed by induction on ℓ .

Basis. $\ell = 0$. Then, $\tau_2 = \epsilon$, $c = (q_0, \perp, \text{Val}_0)$. By taking $\sigma = \epsilon$ and $\mathbf{u} = \mathbf{0}^n$, we have $(q_0, \mathbf{0}^n) \xrightarrow{\sigma}^*_{\mathcal{T}(\mathcal{V})} (q, \mathbf{u})$ with $q = \text{State}(c)$ and $\mathbf{u}[i] = \text{Idle}(c)((\gamma_i, 2)) = 0$ for all $i \in [1, n]$. Moreover, we have $\text{Idle}(c)((\alpha, j)) = 0$ for all $(\alpha, j) \in \Gamma \times \mathbb{N}$ such that $(\alpha, j) \neq (\gamma'_0, 1)$.

Step. $\ell > 0$. Then, there are $\tau' \in \Delta^*$, $t \in \Delta$, and $c' \in \text{Conf}(\mathcal{M})$ such that $\tau_2 = \tau' t$ and $(q_0, \perp, \text{Val}_0) \xrightarrow{\tau'}^*_{\ell-1} \mathcal{T}_{[0,2]}(\mathcal{M}) c' \xrightarrow{t} \mathcal{T}_{[0,2]}(\mathcal{M}) c$.

From the definition of \mathcal{M} , it is not hard to prove that $\text{State}(c') \in Q$.

We apply the induction hypothesis to $(q_0, \perp, \text{Val}_0) \xrightarrow{\tau'}^*_{\ell-1} \mathcal{T}_{[0,2]}(\mathcal{M}) c'$, and we obtain that the following conditions are satisfied:

- $\text{Active}(c') \in (\{\perp\} \cup \{(\gamma'_0, 1)\}) \cup ((\Gamma \setminus \{\gamma_0, \gamma'_0\}) \times \{2\})$,
- $\text{Idle}(c')((\gamma_0, i)) = 0$ for all $i \in \mathbb{N}$,
- $\text{Idle}(c')((\gamma_j, i)) = 0$ for all $j \in [1, n]$ and $i \neq 2$,
- $\text{Idle}(c')((\gamma'_0, i)) = 0$ for all $i \neq 1$, and
- there is $\sigma' \in \Sigma^*$ such that:

$$(q_0, \mathbf{0}^n) \xrightarrow{\sigma'}^*_{\mathcal{T}(\mathcal{V})} (q', \mathbf{u}') \tag{C.6}$$

where $q' = \text{State}(c')$ and $\mathbf{u}'[j] = \text{Idle}(c')((\gamma_j, 2))$ for all $j \in [1, n]$.

Moreover, we have $c' \xrightarrow{t} \mathcal{T}_{[0,2]}(\mathcal{M}) c$. This implies that one of the following cases holds:

- **Case 1:** $t = \langle q', \gamma'_0 \rangle \rightarrow_{\mathcal{M}} \langle q, \gamma'_0 \rangle \triangleright \epsilon$. Then, $\text{State}(c) = q$, $\text{State}(c') = q'$, $\text{Active}(c) = \text{Active}(c') = (\gamma'_0, 1)$, and $\text{Idle}(c) = \text{Idle}(c')$. This implies that the conditions 1-4 of Lemma C.2 are satisfied. Moreover, from the definition of \mathcal{M} , there is $a \in \Sigma$ such that $\delta(q', a) = (q, \mathbf{0}^n)$ since we have $t = \langle q', \gamma'_0 \rangle \rightarrow_{\mathcal{M}} \langle q, \gamma'_0 \rangle \triangleright \epsilon$. This implies that $\mathcal{T}(\mathcal{V})$ can

reach the configuration (q, \mathbf{u}') from the configuration (q', \mathbf{u}') . I.e., we have the following computation of $\mathcal{T}(V)$:

$$(q', \mathbf{u}') \xrightarrow{a} \mathcal{T}(V) (q, \mathbf{u}) \quad (\text{C.7})$$

with $\mathbf{u} = \mathbf{u}'$.

Putting together Equation C.6 and C.7, we obtain that:

$$(q_0, \mathbf{0}^n) \xrightarrow{\sigma' a}^* \mathcal{T}(V) (q, \mathbf{u}) \quad (\text{C.8})$$

Now, we can use the fact that $\mathbf{u} = \mathbf{u}'$, $Idle(c) = Idle(c')$, and $\mathbf{u}'[j] = Idle(c')((\gamma_j, 2))$ for all $j \in [1, n]$, to show that $\mathbf{u}[j] = Idle(c)((\gamma_j, 2))$ for all $j \in [1, n]$.

- **Case 2:** $t = \langle q', \gamma'_0 \rangle \rightarrow_{\mathcal{M}} \langle q, \gamma'_0 \rangle \triangleright \gamma_k$ for some $k \in [1, n]$. Then, $State(c) = q$, $State(c') = q'$, $Active(c) = Active(c') = (\gamma'_0, 1)$, and $Idle(c) = Idle(c') + \text{Id}_{Loc(\mathcal{M})}^{\{(\gamma_k, 2)\}}$. This implies that the conditions 1-4 of Lemma C.2 are satisfied. Moreover, from the definition of \mathcal{M} , there is $a \in \Sigma$ such that $\delta(q', a) = (q, \mathbf{0}^n[k \leftarrow 1])$ since we have $t = \langle q', \gamma'_0 \rangle \rightarrow_{\mathcal{M}} \langle q, \gamma'_0 \rangle \triangleright \gamma_k$. This implies that $\mathcal{T}(V)$ can reach the configuration (q, \mathbf{u}) from the configuration (q', \mathbf{u}') with $\mathbf{u} = \mathbf{u}'[k \leftarrow (\mathbf{u}'[k] + 1)]$. I.e., we have the following computation of $\mathcal{T}(V)$:

$$(q', \mathbf{u}') \xrightarrow{a} \mathcal{T}(V) (q, \mathbf{u}) \quad (\text{C.9})$$

Putting together Equation C.6 and C.9, we obtain that:

$$(q_0, \mathbf{0}^n) \xrightarrow{\sigma' a}^* \mathcal{T}(V) (q, \mathbf{u}) \quad (\text{C.10})$$

Now, we can use the fact that $\mathbf{u} = \mathbf{u}'[k \leftarrow (\mathbf{u}'[k] + 1)]$, $Idle(c) = Idle(c') + \text{Id}_{Loc(\mathcal{M})}^{\{(\gamma_k, 2)\}}$, and $\mathbf{u}'[j] = Idle(c')((\gamma_j, 2))$ for all $j \in [1, n]$, to show that $\mathbf{u}[j] = Idle(c)((\gamma_j, 2))$ for all $j \in [1, n]$.

- **Case 3:** $t = \langle q', \gamma_k \rangle \mapsto_{\mathcal{M}} \langle q', \epsilon \rangle$ for some $k \in [1, n]$. Then, $State(c) = State(c') = q'$, $Active(c) = \perp$, $Active(c') = (\gamma_k, 2)$, and $Idle(c) = Idle(c') + \text{Id}_{Loc(\mathcal{M})}^{\{(\epsilon, 3)\}}$. This implies that the conditions 1-4 of Lemma C.2 are satisfied. Moreover, by taking $q = q'$, $\sigma = \sigma'$, and $\mathbf{u} = \mathbf{u}'$, we have:

$$(q_0, \mathbf{0}^n) \xrightarrow{\sigma}^* \mathcal{T}(V) (q, \mathbf{u}) \quad (\text{C.11})$$

Now, we can use the fact that $\mathbf{u} = \mathbf{u}'$, $Idle(c) = Idle(c') + \text{Id}_{Loc(\mathcal{M})}^{\{(\epsilon, 3)\}}$, and $\mathbf{u}'[j] = Idle(c')((\gamma_j, 2))$ for all $j \in [1, n]$, to show that $\mathbf{u}[j] = Idle(c)((\gamma_j, 2))$ for all $j \in [1, n]$.

- **Case 4:** $t = q' \mapsto_{\mathcal{M}} q \triangleleft \gamma_k$ for some $k \in [1, n]$. Then, $State(c) = q$, $State(c') = q'$, $Active(c) = (\gamma_k, 2)$, $Active(c') = \perp$, $Idle(c')((\gamma_k, 2)) \geq 1$, and $Idle(c) = Idle(c') - \text{Id}_{Loc(\mathcal{M})}^{\{(\gamma_k, 2)\}}$. This implies that the conditions 1-4 of Lemma C.2 are satisfied. Moreover, from the definition of \mathcal{M} , there is $a \in \Sigma$ such that $\delta(q', a) = (q, \mathbf{0}^n[k \leftarrow -1])$. This implies that $\mathcal{T}(V)$ can reach the configuration (q, \mathbf{u}) from the configuration (q', \mathbf{u}') with $\mathbf{u} = \mathbf{u}'[k \leftarrow (\mathbf{u}'[k] - 1)]$ since $Idle(c')((\gamma_k, 2)) = \mathbf{u}'[k] \geq 1$. I.e., we have the following computation of $\mathcal{T}(V)$:

$$(q', \mathbf{u}') \xrightarrow{a} \mathcal{T}(V) (q, \mathbf{u}) \quad (\text{C.12})$$

Putting together Equation C.6 and C.12, we obtain that:

$$(q_0, \mathbf{0}^n) \xrightarrow{\sigma' a}^*_{\mathcal{T}(\mathcal{V})} (q, \mathbf{u}) \quad (\text{C.13})$$

Now, we can use that $\mathbf{u} = \mathbf{u}'[k \leftarrow (\mathbf{u}'[k] - 1)]$, $Idle(c) = Idle(c') - \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma_k, 2)\}}$, and $\mathbf{u}'[j] = Idle(c')((\gamma_j, 2))$ for all $j \in [1, n]$, to show that $\mathbf{u}[j] = Idle(c)((\gamma_j, 2))$ for all $j \in [1, n]$.

- **Case 5:** $t = \langle q', \gamma'_0 \rangle \mapsto_{\mathcal{M}} \langle q', \epsilon \rangle$. Then, $State(c) = State(c') = q'$, $Active(c') = (\gamma'_0, 1)$, $Active(c) = \perp$, and $Idle(c) = Idle(c') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\epsilon, 2)\}}$. This implies that the conditions 1-4 of Lemma C.2 are satisfied. By taking $\mathbf{u} = \mathbf{u}'$, $\sigma = \sigma'$, and $q = q'$, we have that $\mathcal{T}(\mathcal{V})$ can reach the configuration (q, \mathbf{u}) from the configuration (q', \mathbf{u}') . I.e., we have the following computation of $\mathcal{T}(\mathcal{V})$:

$$(q_0, \mathbf{0}^n) \xrightarrow{\sigma}^*_{\mathcal{T}(\mathcal{V})} (q, \mathbf{u}) \quad (\text{C.14})$$

Now, we can use that $\mathbf{u} = \mathbf{u}'$, $Idle(c) = Idle(c') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\epsilon, 2)\}}$, and $\mathbf{u}'[j] = Idle(c')((\gamma_j, 2))$ for all $j \in [1, n]$, to show that $\mathbf{u}[j] = Idle(c)((\gamma_j, 2))$ for all $j \in [1, n]$.

- **Case 6:** $t = q' \mapsto_{\mathcal{M}} q' \triangleleft \gamma'_0$. Then, $State(c) = State(c') = q'$, $Active(c) = (\gamma'_0, 1)$, $Active(c') = \perp$, $Idle(c')((\gamma'_0, 1)) \geq 1$, and $Idle(c) = Idle(c') - \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma'_0, 1)\}}$. This implies that the conditions 1-4 of Lemma C.2 are satisfied. By taking $\mathbf{u} = \mathbf{u}'$, $\sigma = \sigma'$, and $q = q'$, we have that $\mathcal{T}(\mathcal{V})$ can reach the configuration (q, \mathbf{u}) from the configuration (q', \mathbf{u}') . I.e., we have the following computation of $\mathcal{T}(\mathcal{V})$:

$$(q_0, \mathbf{0}^n) \xrightarrow{\sigma}^*_{\mathcal{T}(\mathcal{V})} (q, \mathbf{u}) \quad (\text{C.15})$$

Now, we can use that $\mathbf{u} = \mathbf{u}'$, $Idle(c) = Idle(c') - \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma'_0, 1)\}}$, and $\mathbf{u}'[j] = Idle(c')((\gamma_j, 2))$ for all $j \in [1, n]$, to show that $\mathbf{u}[j] = Idle(c)((\gamma_j, 2))$ for all $j \in [1, n]$. \square

Hence, Lemma 3.7 is an immediate consequence of Lemma C.1 and Lemma C.2. \square

APPENDIX D. THE PROOF OF LEMMA 4.4

Lemma 4.4. A state $q \in F$ is k -stratified reachable by \mathcal{M} if and only if there is $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that:

- $\sigma_0 \sigma_1 \cdots \sigma_k \in Traces_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$, and
- $|\sigma_i|_{(\gamma, i, \triangleleft)} \leq |\sigma_{i-1}|_{(\gamma, i, \triangleright)}$ for all $\gamma \in \Gamma$ and $i \in [0, k]$ where $\sigma_{-1} = (\gamma_0, 0, \triangleright)$.

Proof. To prove Lemma 4.4, we need first to define a simulation relation μ between \mathcal{M} and \mathcal{P} that maps any configuration of \mathcal{M} to a configuration of \mathcal{P} .

Definition D.1. Let μ be a function from $Q \times \Gamma_{\mathcal{P}} \times Loc(\mathcal{M})$ to $Conf(\mathcal{P})$ such that for every $c \in Q \times \Gamma_{\mathcal{P}} \times Loc(\mathcal{M})$, $\mu(c) = (State(c), Active(c))$.

The Only if direction of Lemma 4.4: In the following, we show that if there is a state $q \in F$ such that q is k -stratified reachable by \mathcal{M} , then there is $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that $\sigma_0 \sigma_1 \cdots \sigma_k \in \text{Traces}_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$, and $|\sigma_i|_{(\gamma, i, \triangleleft)} \leq |\sigma_{i-1}|_{(\gamma, i, \triangleright)}$ for all $\gamma \in \Gamma$ and $i \in [0, k]$ where $\sigma_{-1} = (\gamma_0, 0, \triangleright)$.

To this aim, we first prove that if there is a run $c \xrightarrow{\tau_i^*}_{\mathcal{T}_{\{i\}}(\mathcal{M})} c'$ (where \mathcal{M} executes only threads with switch number $i \in [0, k]$), then there is a run $\mu(c) \xrightarrow{\sigma_i^*}_{\mathcal{T}(\mathcal{P})} \mu(c')$ of \mathcal{P} such that: (1) $\sigma_i \in \Sigma_i^*$, (2) the number of occurrences of $(\gamma, i, \triangleleft)$ in σ_i is equal to the number of activated threads by \mathcal{M} with local configuration (γ, i) , and (3) the number of created/added threads by \mathcal{M} with local configuration $(\gamma, i + 1)$ is equal to the number of occurrence of $(\gamma, i + 1, \triangleright)$ in σ_i .

Lemma D.2. *For every $i \in [0, k]$, and $c, c' \in (Q \times \Gamma_{\mathcal{P}} \times \text{Loc}(\mathcal{M}))$, if there is $\tau_i \in \Delta^*$ such that $c \xrightarrow{\tau_i^*}_{\mathcal{T}_{\{i\}}(\mathcal{M})} c'$, then there is $\sigma_i \in \Sigma_i^*$ such that:*

- (1) $\mu(c) \xrightarrow{\sigma_i^*}_{\mathcal{T}(\mathcal{P})} \mu(c')$.
- (2) $\text{Idle}(c)((\alpha, i)) \geq |\sigma_i|_{(\alpha, i, \triangleleft)}$ for all $\alpha \in \Gamma_\epsilon$.
- (3) $\text{Idle}(c)((\alpha, i)) = \text{Idle}(c)((\alpha, i)) - |\sigma_i|_{(\alpha, i, \triangleleft)}$ and $\text{Idle}(c')((\alpha, i + 1)) = \text{Idle}(c)((\alpha, i + 1)) + |\sigma_i|_{(\alpha, i + 1, \triangleright)}$ for all $\alpha \in \Gamma_\epsilon$.
- (4) $\text{Idle}(c')((\alpha, j)) = \text{Idle}(c)((\alpha, j))$ for all $(\alpha, j) \in \Gamma_\epsilon \times [0, k + 1]$ such that $j \notin \{i, i + 1\}$.

Proof. Assume that $c \xrightarrow{\tau_i}_{\ell} \mathcal{T}_{\{i\}}(\mathcal{M}) c'$ for some $\ell \in \mathbb{N}$. We proceed by induction on ℓ .

Basis. $\ell = 0$. Then, $\tau_i = \epsilon$ and $c = c'$. By taking $\sigma_i = \epsilon$, we have all the conditions of Lemma D.2 are fulfilled.

Step. $\ell > 0$. Then there are $\tau'_i \in \Delta^*$, $t \in \Delta$, $c'' \in (Q \times \Gamma_{\mathcal{P}} \times \text{Loc}(\mathcal{M}))$ such that:

$$c \xrightarrow{\tau'_i}_{\ell-1} \mathcal{T}_{\{i\}}(\mathcal{M}) c'' \xrightarrow{t}_{\mathcal{T}_{\{i\}}(\mathcal{M})} c' \quad (\text{D.1})$$

We apply the induction hypothesis to the run $c \xrightarrow{\tau'_i}_{\ell-1} \mathcal{T}_{\{i\}}(\mathcal{M}) c''$, and we obtain that there is $\sigma'_i \in \Sigma_i^*$ such that:

- $\mu(c) \xrightarrow{\sigma'_i}_{\mathcal{T}(\mathcal{P})} \mu(c'')$.
- $\text{Idle}(c)((\alpha, i)) \geq |\sigma'_i|_{(\alpha, i, \triangleleft)}$ for all $\alpha \in \Gamma_\epsilon$.
- $\text{Idle}(c'')((\alpha, i)) = \text{Idle}(c)((\alpha, i)) - |\sigma'_i|_{(\alpha, i, \triangleleft)}$ and $\text{Idle}(c'')((\alpha, i + 1)) = \text{Idle}(c)((\alpha, i + 1)) + |\sigma'_i|_{(\alpha, i + 1, \triangleright)}$ for all $\alpha \in \Gamma_\epsilon$.
- $\text{Idle}(c'')((\alpha, j)) = \text{Idle}(c)((\alpha, j))$ for all $(\alpha, j) \in \Gamma_\epsilon \times [0, k + 1]$ such that $j \notin \{i, i + 1\}$.

Since we have $c'' \xrightarrow{t}_{\mathcal{T}_{\{i\}}(\mathcal{M})} c'$, one of the followings cases holds:

- **Case 1:** $t = \langle q'', \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \epsilon$. Then $\text{State}(c'') = q''$, $\text{State}(c') = q'$, $\text{Active}(c'') = (\gamma, i)$, $\text{Active}(c') = (u, i)$, and $\text{Idle}(c') = \text{Idle}(c'')$. Moreover, from the definition of \mathcal{P} , we have $\langle q'', (\gamma, i) \rangle \xrightarrow{(\epsilon, i, -)}_{\mathcal{P}} \langle q', (u, i) \rangle$. This implies that $\mathcal{T}(\mathcal{P})$ has the following run:

$$\langle q'', (\gamma, i) \rangle \xrightarrow{(\epsilon, i, -)}_{\mathcal{T}(\mathcal{P})} \langle q', (u, i) \rangle \quad (\text{D.2})$$

We can use the definition of μ to show that $\mu(c'') = (q'', (\gamma, i))$ and $\mu(c') = (q', (u, i))$.

Then, let $\sigma_i = \sigma'_i(\epsilon, i, -)$. Putting together the equation $\mu(c) \xrightarrow{\sigma'_i}_{\mathcal{T}(\mathcal{P})} \mu(c'')$ and Equation

D.2, we obtain that:

$$\mu(c) \xrightarrow{\sigma_i^*}^*_{\mathcal{T}(\mathcal{P})} \mu(c') \quad (\text{D.3})$$

Then, we can use the fact that $Idle(c') = Idle(c'')$ and $\sigma_i = \sigma'_i(\epsilon, i, -)$ to show that all the conditions of Lemma D.2 are fulfilled.

- **Case 2:** $t = \langle q'', \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \alpha$ with $\alpha \in \Gamma$. Then $State(c'') = q''$, $State(c') = q'$, $Active(c'') = (\gamma, i)$, $Active(c') = (u, i)$, and $Idle(c') = Idle(c'') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\alpha, i+1)\}}$. Moreover, from the definition of \mathcal{P} , we have $\langle q'', (\gamma, i) \rangle \xrightarrow{(\alpha, i+1, \triangleright)}_{\mathcal{P}} \langle q', (u, i) \rangle$. This implies that $\mathcal{T}(\mathcal{P})$ has the following run:

$$\langle q'', (\gamma, i) \rangle \xrightarrow{(\alpha, i+1, \triangleright)}_{\mathcal{T}(\mathcal{P})} \langle q', (u, i) \rangle \quad (\text{D.4})$$

We can use the definition of the function μ to show that $\mu(c'') = \langle q'', (\gamma, i) \rangle$ and $\mu(c') = \langle q', (u, i) \rangle$. Then, let $\sigma_i = \sigma'_i(\alpha, i+1, \triangleright)$.

Putting together the equation $\mu(c) \xrightarrow{\sigma_i^*}^*_{\mathcal{T}(\mathcal{P})} \mu(c'')$ and Equation D.4, we obtain that:

$$\mu(c) \xrightarrow{\sigma_i^*}^*_{\mathcal{T}(\mathcal{P})} \mu(c') \quad (\text{D.5})$$

Then, we can use the fact that $Idle(c') = Idle(c'') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\alpha, i+1)\}}$ and $\sigma_i = \sigma'_i(\alpha, i+1, \triangleright)$ to show that all the conditions of Lemma D.2 are fulfilled.

- **Case 3:** $t = \langle q'', \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$. Then $State(c'') = q''$, $State(c') = q'$, $Active(c'') = (\gamma, i)$, $Active(c') = \perp$, and $Idle(c') = Idle(c'') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(u, i+1)\}}$. Moreover, from the definition of \mathcal{P} , we have $\langle q'', (\gamma, i) \rangle \xrightarrow{(u, i+1, \triangleright)}_{\mathcal{P}} \langle q', \perp \rangle$. This implies that $\mathcal{T}(\mathcal{P})$ has the following run:

$$\langle q'', (\gamma, i) \rangle \xrightarrow{(u, i+1, \triangleright)}_{\mathcal{T}(\mathcal{P})} \langle q', \perp \rangle \quad (\text{D.6})$$

Using the definition of the function μ , it is easy to observe that $\mu(c'') = \langle q'', (\gamma, i) \rangle$ and $\mu(c') = \langle q', \perp \rangle$. Then, let $\sigma_i = \sigma'_i(u, i+1, \triangleright)$. Putting together the equation $\mu(c) \xrightarrow{\sigma_i^*}^*_{\mathcal{T}(\mathcal{P})} \mu(c'')$ and Equation D.6, we obtain that:

$$\mu(c) \xrightarrow{\sigma_i^*}^*_{\mathcal{T}(\mathcal{P})} \mu(c') \quad (\text{D.7})$$

Then, we can use the fact that $Idle(c') = Idle(c'') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(u, i+1)\}}$ and $\sigma_i = \sigma'_i(u, i+1, \triangleright)$ to show that all the conditions of Lemma D.2 are fulfilled.

- **Case 4:** $t = q'' \mapsto_{\mathcal{M}} q' \triangleleft \gamma$. Then $State(c'') = q''$, $State(c') = q'$, $Active(c'') = \perp$, $Active(c') = (\gamma, i)$, $Idle(c'')((\gamma, i)) \geq 1$, and $Idle(c') = Idle(c'') - \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma, i)\}}$. Moreover, from the definition of \mathcal{P} , we have $\langle q'', \perp \rangle \xrightarrow{(\gamma, i, \triangleleft)}_{\mathcal{P}} \langle q', (\gamma, i) \rangle$. This implies that $\mathcal{T}(\mathcal{P})$ has the following run:

$$\langle q'', \perp \rangle \xrightarrow{(\gamma, i, \triangleleft)}_{\mathcal{T}(\mathcal{P})} \langle q', (\gamma, i) \rangle \quad (\text{D.8})$$

Using the definition of the function μ , it is easy to observe that $\mu(c'') = \langle q'', \perp \rangle$ and $\mu(c') = \langle q', (\gamma, i) \rangle$. Then, let $\sigma_i = \sigma'_i(\gamma, i, \triangleleft)$. Putting together the equation $\mu(c) \xrightarrow{\sigma_i^*}^*_{\mathcal{T}(\mathcal{P})} \mu(c'')$ and Equation D.8, we obtain that:

$$\mu(c) \xrightarrow{\sigma_i^*}^*_{\mathcal{T}(\mathcal{P})} \mu(c') \quad (\text{D.9})$$

Then, we can use the fact that $Idle(c') = Idle(c'') - \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma, i)\}}$ and $\sigma_i = \sigma'_i(\gamma, i, \triangleleft)$ to show that all the conditions of Lemma D.2 are fulfilled. \square

Now, we are ready to prove the only if direction of Lemma 4.4.

Lemma D.3. *If $q \in F$ is k -stratified reachable by \mathcal{M} , then there is $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that $\sigma_0 \sigma_1 \cdots \sigma_k \in \text{Traces}_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$, and $|\sigma_i|_{(\gamma, i, \triangleleft)} \leq |\sigma_{i-1}|_{(\gamma, i, \triangleright)}$ for all $\gamma \in \Gamma$ and $i \in [0, k]$ where $\sigma_{-1} = (\gamma_0, 0, \triangleright)$.*

Proof. Let us assume that there is a state $q \in F$ such that q is k -stratified reachability by \mathcal{M} . Then, there are $\tau_0, \tau_1, \dots, \tau_k \in \Delta^*$, and $c_1, \dots, c_{k+1} \in \text{Conf}(\mathcal{M})$ such that $\text{State}(c_{k+1}) = q$, $\text{Active}(c_{k+1}) = \perp$, and we have:

$$c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau_0}^*_{\mathcal{T}_{\{0\}}(\mathcal{M})} c_1 \xrightarrow{\tau_1}^*_{\mathcal{T}_{\{1\}}(\mathcal{M})} \cdots \xrightarrow{\tau_{k-1}}^*_{\mathcal{T}_{\{k-1\}}(\mathcal{M})} c_k \xrightarrow{\tau_k}^*_{\mathcal{T}_{\{k\}}(\mathcal{M})} c_{k+1} \quad (\text{D.10})$$

Notice that all the configurations $c_{\mathcal{M}}^{\text{init}}, c_1, c_2, \dots, c_{k+1}$ are in $(Q \times \Gamma_{\mathcal{P}} \times \text{Loc}(\mathcal{M}))$ by definition. Then, we can use Lemma D.2, to show that there are $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that:

$$\mu(c_{\mathcal{M}}^{\text{init}}) \xrightarrow{\sigma_0}^*_{\mathcal{T}(\mathcal{P})} \mu(c_1) \xrightarrow{\sigma_1}^*_{\mathcal{T}(\mathcal{P})} \cdots \xrightarrow{\sigma_{k-1}}^*_{\mathcal{T}(\mathcal{P})} \mu(c_k) \xrightarrow{\sigma_k}^*_{\mathcal{T}(\mathcal{P})} \mu(c_{k+1}) \quad (\text{D.11})$$

Then, we obtain $\sigma_0 \sigma_1 \cdots \sigma_k \in \text{Traces}_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$ since $\mu(c_{\mathcal{M}}^{\text{init}}) = (q_0, \perp)$, $\text{State}(c_{k+1}) = q \in F$, and $\text{Active}(c_{k+1}) = \perp$ (i.e., $\mu(c_{k+1}) \in F \times \{\perp\}$). Moreover, we can use the fact that $\text{Idle}(c_{\mathcal{M}}^{\text{init}}) = \text{Id}_{\text{Loc}(\mathcal{M})}^{\{(\gamma_0, 0)\}}$ and the second condition of Lemma D.2, to prove that for every $\alpha \in \Gamma_\epsilon$, we have that $|\sigma_0|_{(\alpha, i, \triangleleft)} \leq |\sigma_{-1}|_{(\alpha, i, \triangleright)}$ with $\sigma_{-1} = (\gamma_0, 0, \triangleright)$.

Conversely, we can use the conditions (3) and (4) of Lemma D.2, to prove that for every $j \in [1, k+1]$ and every $\alpha \in \Gamma_\epsilon$, we have $\text{Idle}(c_j)((\alpha, j)) = |\sigma_{j-1}|_{(\alpha, j, \triangleright)}$ and $\text{Idle}(c_j)((\alpha, j+1)) = 0$. So, for every $j \in [1, k]$ and $\alpha \in \Gamma_\epsilon$, we can use the fact that $\text{Idle}(c_j)((\alpha, j)) = |\sigma_{j-1}|_{(\alpha, j, \triangleright)}$ and the second condition of Lemma D.2 (i.e., $\text{Idle}(c_j)((\alpha, j)) \geq |\sigma_j|_{(\alpha, j, \triangleleft)}$), to prove that $|\sigma_{j-1}|_{(\alpha, j, \triangleright)} \geq |\sigma_j|_{(\alpha, j, \triangleleft)}$. \square

The if direction of Lemma 4.4: In the following, we prove that if there is $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that $\sigma_0 \sigma_1 \cdots \sigma_k \in \text{Traces}_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$, and $|\sigma_i|_{(\gamma, i, \triangleleft)} \leq |\sigma_{i-1}|_{(\gamma, i, \triangleright)}$ for all $\gamma \in \Gamma$ and $i \in [0, k]$ where $\sigma_{-1} = (\gamma_0, 0, \triangleright)$, then there is a state $q \in F$ such q is k -stratified reachable by \mathcal{M} .

To this aim, we first show that for every configuration $c \in (Q \times \Gamma_{\mathcal{P}} \times \text{Loc}(\mathcal{M}))$ and $\sigma_i \in \Sigma_i^*$, if there is a run $\mu(c) \xrightarrow{\sigma_i}^*_{\mathcal{T}(\mathcal{P})} c'$ for some $c' \in \text{Conf}(\mathcal{P})$ and the number of occurrences of $(\gamma, i, \triangleleft)$ in σ_i is less than the number of pending thread in c with local configuration (γ, i) , then there are $c' \in \text{Conf}(\mathcal{M})$ and a run $c \xrightarrow{\tau_i}^*_{\mathcal{T}_{\{i\}}(\mathcal{M})} c'$ such that: (1) $\mu(c') = c'$, (2) the number of occurrences of $(\gamma, i, \triangleleft)$ in σ_i is equal to the number of activated threads by \mathcal{M} with local configuration (γ, i) , and (3) the number of created/added threads by \mathcal{M} with local configuration $(\gamma, i+1)$ is equal to the number of occurrence of $(\gamma, i+1, \triangleright)$ in σ_i .

Lemma D.4. *For every $i \in [0, k]$, $\varsigma, \varsigma' \in \text{Conf}(\mathcal{P})$, $c \in (Q \times \Gamma_{\mathcal{P}} \times \text{Loc}(\mathcal{M}))$, and $\sigma_i \in \Sigma_i^*$, if $\varsigma \xrightarrow{\sigma_i}^*_{\mathcal{T}(\mathcal{P})} \varsigma'$, $\mu(c) = \varsigma$, and $\text{Idle}(c)((\alpha, i)) \geq |\sigma_i|_{(\alpha, i, \triangleleft)}$ for all $\alpha \in \Gamma_\epsilon$, then there are $\tau_i \in \Delta^*$ and $c' \in (Q \times \Gamma_{\mathcal{P}} \times \text{Loc}(\mathcal{M}))$ such that:*

- (1) $\mu(c') = \varsigma'$.
- (2) $c \xrightarrow{\tau_i}^*_{\mathcal{T}_{\{i\}}(\mathcal{M})} c'$.
- (3) $\text{Idle}(c')((\alpha, i)) = \text{Idle}(c)((\alpha, i)) - |\sigma_i|_{(\alpha, i, \triangleleft)}$ and $\text{Idle}(c')((\alpha, i+1)) = \text{Idle}(c)((\alpha, i+1)) + |\sigma_i|_{(\alpha, i+1, \triangleright)}$ for all $\alpha \in \Gamma_\epsilon$.
- (4) $\text{Idle}(c')((\alpha, j)) = \text{Idle}(c)((\alpha, j))$ for all $(\alpha, j) \in \Gamma_\epsilon \times [0, k+1]$ such that $j \notin \{i, i+1\}$.

Proof. Assume that $\varsigma \xrightarrow[\ell]{\sigma_i} \mathcal{T}(\mathcal{P}) \varsigma'$ for some $\ell \in \mathbb{N}$, $\mu(c) = \varsigma$, and $Idle(c)((\alpha, i)) \geq |\sigma_i|_{(\alpha, i, \triangleleft)}$ for all $\alpha \in \Gamma_\epsilon$. The proof is done by induction on ℓ .

Basis. $\ell = 0$. Then, $\varsigma = \varsigma'$, $\sigma_i = \epsilon$. By taking $c' = c$ and $\tau_i = \epsilon$, all the conditions of Lemma D.4 are fulfilled.

Step. $\ell > 0$. Then, there are $\sigma'_i \in \Sigma_i^*$, $a \in \Sigma_i$, and $\varsigma'' \in Conf(\mathcal{P})$ such that $\sigma_i = \sigma'_i a$ and $\varsigma \xrightarrow[\ell]{\sigma'_i} \mathcal{T}(\mathcal{P}) \varsigma'' \xrightarrow{a} \mathcal{T}(\mathcal{P}) \varsigma'$.

We apply the induction hypothesis to $\varsigma \xrightarrow[\ell]{\sigma'_i} \mathcal{T}(\mathcal{P}) \varsigma''$ since $Idle(c)((\alpha, i)) \geq |\sigma_i|_{(\alpha, i, \triangleleft)} \geq |\sigma'_i|_{(\alpha, i, \triangleleft)}$ for all $\alpha \in \Gamma_\epsilon$, and we obtain that there are $\tau'_i \in \Delta^*$ and $c'' \in (Q \times \Gamma_{\mathcal{P}} \times Loc(\mathcal{M}))$ such that:

- $\mu(c'') = \varsigma''$.
- $c \xrightarrow[\tau'_i]{\tau'_i}^* \mathcal{T}_{\{i\}}(\mathcal{M}) c''$.
- $Idle(c'')((\alpha, i)) = Idle(c)((\alpha, i)) - |\sigma'_i|_{(\alpha, i, \triangleleft)}$ and $Idle(c'')((\alpha, i+1)) = Idle(c)((\alpha, i+1)) + |\sigma'_i|_{(\alpha, i+1, \triangleright)}$ for all $\alpha \in \Gamma_\epsilon$.
- $Idle(c'')((\alpha, j)) = Idle(c)((\alpha, j))$ for all $(\alpha, j) \in \Gamma_\epsilon \times [0, k+1]$ such that $j \notin \{i, i+1\}$.

Since we have $\varsigma'' \xrightarrow{a} \mathcal{T}(\mathcal{P}) \varsigma'$, one of the following cases holds:

- **Case 1:** $a = (\epsilon, i, -)$. Then, there are $q, q' \in Q$, $\gamma \in \Gamma$, and $u \in \Gamma_\epsilon$ such that $\langle q, (\gamma, i) \rangle \xrightarrow{(\epsilon, i, -)} \mathcal{P} \langle q', (u, i) \rangle$, $\varsigma'' = (q, (\gamma, i))$, and $\varsigma' = (q', (u, i))$. Since $\mu(c'') = \varsigma''$, we have $State(c'') = q$ and $Active(c'') = (\gamma, i)$. Moreover, we have $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \epsilon$. By taking $c' = (q', (u, i), Idle(c''))$, we have $c'' \xrightarrow{t} \mathcal{T}_i(\mathcal{M}) c'$.

Now, we can use the definition of the function μ to show that $\mu(c') = (q', (u, i)) = \varsigma'$. Let $\tau_i = \tau'_i t$. We can put together the equation $c \xrightarrow[\tau'_i]{\tau'_i}^* \mathcal{T}_i(\mathcal{M}) c''$ and the equation $c'' \xrightarrow{t} \mathcal{T}_i(\mathcal{M}) c'$ to obtain the following run of $\mathcal{T}(\mathcal{P})$:

$$c \xrightarrow[\tau_i]{\tau_i}^* \mathcal{T}(\mathcal{P}) c' \tag{D.12}$$

Then, we can use the fact that $Idle(c') = Idle(c'')$ and $\sigma_i = \sigma'_i(\epsilon, i, -)$ to show that the conditions 4-5 of Lemma D.4 are fulfilled.

- **Case 2:** $a = (\alpha, i+1, \triangleright)$ and $c' \notin (Q \times (\{\perp\}))$. Then, there are $q, q' \in Q$, $\gamma \in \Gamma$, and $u \in \Gamma_\epsilon$ such that $\langle q, (\gamma, i) \rangle \xrightarrow{a} \mathcal{P} \langle q', (u, i) \rangle$, $\varsigma'' = (q, (\gamma, i))$, and $\varsigma' = (q', (u, i))$. Since $\mu(c'') = \varsigma''$, we have $State(c'') = q$ and $Active(c'') = (\gamma, i)$. Moreover, we have $t = \langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \alpha$. By taking $c' = (q', (u, i), Idle(c'') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\alpha, i+1)\}})$, we have $c'' \xrightarrow{t} \mathcal{T}_i(\mathcal{M}) c'$.

Then, we can use the definition of the function μ to show that $\mu(c') = (q', (u, i)) = \varsigma'$. Let $\tau_i = \tau'_i t$. We can put together the equation $c \xrightarrow[\tau'_i]{\tau'_i}^* \mathcal{T}_i(\mathcal{M}) c''$ and the equation $c'' \xrightarrow{t} \mathcal{T}_i(\mathcal{M}) c'$ to obtain the following run of $\mathcal{T}(\mathcal{P})$:

$$c \xrightarrow[\tau_i]{\tau_i}^* \mathcal{T}(\mathcal{P}) c' \tag{D.13}$$

Then, we can use the fact that $Idle(c') = Idle(c'') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\alpha, i+1)\}}$ and $\sigma_i = \sigma'_i(\alpha, i+1, \triangleright)$ to show that the conditions 4-5 of Lemma D.4 are fulfilled.

- **Case 3:** $a = (u, i+1, \triangleright)$ and $\varsigma' \in (Q \times (\{\perp\}))$. Then, there are $q, q' \in Q$ and $\gamma \in \Gamma$ such that $\langle q, (\gamma, i) \rangle \xrightarrow{a} \mathcal{P} \langle q', \perp \rangle$, $\varsigma'' = (q, (\gamma, i))$, and $\varsigma' = (q', \perp)$. Since $\mu(c'') = \varsigma''$, we

have $State(c'') = q$ and $Active(c'') = (\gamma, i)$. Moreover, we have $t = \langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$. By taking $c' = (q', \perp, Idle(c'') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(u, i+1)\}})$, we have $c'' \xrightarrow{t}_{\mathcal{T}_i(\mathcal{M})} c'$.

Now, we can use the definition of the function μ to show that $\mu(c') = (q', \perp) = \zeta'$. Let $\tau_i = \tau'_i t$. We can put together the equation $c \xrightarrow{\tau'_i}_{\mathcal{T}_i(\mathcal{M})}^* c''$ and the equation $c'' \xrightarrow{t}_{\mathcal{T}_i(\mathcal{M})} c'$ to obtain the following run of $\mathcal{T}(\mathcal{P})$:

$$c \xrightarrow{\tau_i}_{\mathcal{T}(\mathcal{P})}^* c' \quad (\text{D.14})$$

Then, we can use the fact that $Idle(c') = Idle(c'') + \mathbf{ld}_{Loc(\mathcal{M})}^{\{(u, i+1)\}}$ and $\sigma_i = \sigma'_i(u, i+1, \triangleright)$ to show that the conditions 4-5 of Lemma D.4 are fulfilled.

- **Case 4:** $a = (\gamma, i, \triangleleft)$. Then, there are $q, q' \in Q$ such that $\langle q, \perp \rangle \xrightarrow{a}_{\mathcal{P}} \langle q', (\gamma, i) \rangle$, $\zeta'' = (q, \perp)$, and $\zeta' = (q', (\gamma, i))$. Since $\mu(c'') = \zeta''$, $State(c'') = q$ and $Active(c'') = \perp$. In addition, we have $Idle(c'')((\gamma, i)) \geq 1$ since $idle(c'')((\gamma, i)) = Idle(c) - |\sigma'_i|_{(\gamma, i, \triangleright)}$, $Idle(c) \geq |\sigma_i|_{(\gamma, i, \triangleleft)}$, and $|\sigma_i|_{(\gamma, i, \triangleleft)} = |\sigma'_i|_{(\gamma, i, \triangleleft)} + 1$. Moreover, we have $t = q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$. Then, by taking $c' = (q', (\gamma, i), Idle(c'') - \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma, i)\}})$, we have $c'' \xrightarrow{t}_{\mathcal{T}_i(\mathcal{M})} c'$.

Now, we can use the definition of the function μ to show that $\mu(c') = (q', \perp) = \zeta'$. Let $\tau_i = \tau'_i t$. Then, we can put together the equation $c \xrightarrow{\tau'_i}_{\mathcal{T}_i(\mathcal{M})}^* c''$ and the equation $c'' \xrightarrow{t}_{\mathcal{T}_i(\mathcal{M})} c'$ to obtain the following run of $\mathcal{T}(\mathcal{P})$:

$$c \xrightarrow{\tau_i}_{\mathcal{T}(\mathcal{P})}^* c' \quad (\text{D.15})$$

Then, we can use the fact that $Idle(c') = Idle(c'') - \mathbf{ld}_{Loc(\mathcal{M})}^{\{(\gamma, i)\}}$ and $\sigma_i = \sigma'_i(\gamma, i, \triangleleft)$ to show that the conditions 4-5 of Lemma D.4 are fulfilled. \square

Now, we are ready to prove the if direction of Lemma D.4:

Lemma D.5. *There is $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that $\sigma_0 \sigma_1 \cdots \sigma_k \in Traces_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$, and $|\sigma_i|_{(\gamma, i, \triangleleft)} \leq |\sigma_{i-1}|_{(\gamma, i, \triangleright)}$ for all $\gamma \in \Gamma$ and $i \in [0, k]$ with $\sigma_{-1} = (\gamma_0, 0, \triangleright)$, then there is a state $q \in F$ such that q is k -stratified reachable by \mathcal{M} .*

Proof. Let us assume now that there are $\sigma_i \in \Sigma_i^*$ for all $i \in [0, k]$ such that $\sigma_0 \sigma_1 \cdots \sigma_k \in Traces_{\mathcal{T}(\mathcal{P})}(\{(q_0, \perp)\}, F \times \{\perp\})$, and $|\sigma_i|_{(\gamma, i, \triangleleft)} \leq |\sigma_{i-1}|_{(\gamma, i, \triangleright)}$ for all $\gamma \in \Gamma$ and $i \in [0, k]$ with $\sigma_{-1} = (\gamma_0, 0, \triangleright)$. Then, there are $\varsigma_0, \varsigma_1, \dots, \varsigma_{k+1} \in Conf(\mathcal{P})$ such that: (i) $\varsigma_0 = (q_0, \perp)$ and $\varsigma_{k+1} \in F \times \{\perp\}$, and (ii) we have the following run of $\mathcal{T}(\mathcal{P})$:

$$\varsigma_0 \xrightarrow{\sigma_0}_{\mathcal{T}(\mathcal{P})}^* \varsigma_1 \xrightarrow{\sigma_1}_{\mathcal{T}(\mathcal{P})}^* \varsigma_2 \cdots \varsigma_{k-1} \xrightarrow{\sigma_{k-1}}_{\mathcal{T}(\mathcal{P})}^* \varsigma_k \xrightarrow{\sigma_k}_{\mathcal{T}(\mathcal{P})}^* \varsigma_{k+1}$$

Then, we can apply Lemma D.4 to $\mu(c_{\mathcal{M}}^{\text{init}}) = \varsigma_0$, $\varsigma_0 \xrightarrow{\sigma_0}_{\mathcal{T}(\mathcal{P})}^* \varsigma_1$, and $Idle(c_{\mathcal{M}}^{\text{init}})((\alpha, 0)) \geq |\sigma_0|_{(\alpha, 0, \triangleleft)}$ for all $\alpha \in \Gamma_\epsilon$, to prove that there are $\tau_0 \in \Delta^*$ and $c_1 \in (Q \times \Gamma_{\mathcal{P}} \times Loc(\mathcal{M}))$ such that:

- $\mu(c_1) = \varsigma_1$.
- $c_{\mathcal{M}}^{\text{init}} \xrightarrow{\tau_0}_{\mathcal{T}_{\{0\}}(\mathcal{M})}^* c_1$.
- $Idle(c_1)((\alpha, 0)) = Idle(c_{\mathcal{M}}^{\text{init}})((\alpha, 0)) - |\sigma_0|_{(\alpha, 0, \triangleleft)}$ and $Idle(c_1)((\alpha, 1)) = |\sigma_0|_{(\alpha, 1, \triangleright)}$ for all $\alpha \in \Gamma_\epsilon$.
- $Idle(c_1)((\alpha, j)) = Idle(c_{\mathcal{M}}^{\text{init}})((\alpha, j)) = 0$ for all $(\alpha, j) \in \Gamma_\epsilon \times [0, k+1]$ such that $j \notin \{0, 1\}$.

Now, we can apply Lemma D.4 to $\mu(c_1) = \varsigma_1$, $\varsigma_1 \xrightarrow{\sigma_1}^*_{\mathcal{T}(\mathcal{P})} \varsigma_2$, and $Idle(c_1)((\alpha, 1)) = |\sigma_0|_{(\alpha, 1, \triangleright)} \geq |\sigma_1|_{(\alpha, 1, \triangleleft)}$ for all $\alpha \in \Gamma_\epsilon$, to show that there are $\tau_1 \in \Delta^*$ and $c_2 \in (Q \times \Gamma_{\mathcal{P}} \times Loc(\mathcal{M}))$ such that:

- $\mu(c_2) = \varsigma_2$.
- $c_1 \xrightarrow{\tau_1}^*_{\mathcal{T}_{\{1\}}(\mathcal{M})} c_2$.
- $Idle(c_2)((\alpha, 1)) = Idle(c_1)((\alpha, 1)) - |\sigma_1|_{(\alpha, 1, \triangleleft)}$ and $Idle(c_2)((\alpha, 2)) = |\sigma_1|_{(\alpha, 2, \triangleright)}$ for all $\alpha \in \Gamma_\epsilon$.
- $Idle(c_2)((\alpha, j)) = Idle(c_1)((\alpha, j))$ for all $(\alpha, j) \in \Gamma_\epsilon \times [0, k+1]$ such that $j \notin \{1, 2\}$.

So, we can apply step by step Lemma D.4 to prove that there are $\tau_0, \dots, \tau_k \in \Delta^*$ and $c_0, c_1, \dots, c_{k+1} \in (Q \times \Gamma_{\mathcal{P}} \times Loc(\mathcal{M}))$ such that: (1) $c_0 = c_{\mathcal{M}}^{\text{init}}$, (2) $\mu(c_i) = \varsigma_i$ for all $i \in [0, k+1]$, and (3) $c_0 \xrightarrow{\tau_0}^*_{\mathcal{T}_{\{0\}}(\mathcal{M})} c_1 \xrightarrow{\tau_1}^*_{\mathcal{T}_{\{1\}}(\mathcal{M})} \dots \xrightarrow{\tau_{k-1}}^*_{\mathcal{T}_{\{k-1\}}(\mathcal{M})} c_k \xrightarrow{\tau_k}^*_{\mathcal{T}_{\{k\}}(\mathcal{M})} c_{k+1}$. Moreover, we have $State(c_{k+1}) \in F$ and $Active(c_{k+1}) = \perp$ since $\varsigma_{k+1} \in F \times \{\perp\}$ and $\mu(c_{k+1}) = \varsigma_{k+1}$. This implies that $State(c_{k+1}) \in F$ is k -stratified reachable by \mathcal{M} . \square

Lemma 4.4 is an immediate consequence of Lemma D.3 and Lemma D.5. \square

APPENDIX E. THE PROOF OF LEMMA 5.3

The proof of Lemma 5.3 is structured as follows: First, we establish the relation between a computation of a thread of \mathcal{M}_{fs} and a run of $\mathcal{S}_{(q, \gamma)}$. Then, we give the relation between a computation of a thread of \mathcal{M} and a run of $\mathcal{P}_{(q, \gamma)}$. Due to the link between the set of runs of $\mathcal{P}_{(q, \gamma)}$ and the set of runs of $\mathcal{A}_{(q, \gamma)}$, these two relations permit us to construct for every thread computation of \mathcal{M} an “equivalent” thread computation of \mathcal{M}_{fs} and vice-versa. Then, we consider a DCPS \mathcal{M}_{\cup} which is the union of \mathcal{M} and \mathcal{M}_{fs} in the sense that for each thread T with initial configuration $\gamma \in \Gamma$, \mathcal{A}_{\cup} chooses in nondeterministic way to execute the thread T following the transition relation of \mathcal{M}_{fs} or the transition relation of \mathcal{M} .

Afterwards, we define the rank of a run of \mathcal{M}_{\cup} from the initial configuration $c_{\mathcal{M}_{\cup}}^{\text{init}}$ by the pair $(m, n) \in \mathbb{N} \times \mathbb{N}$ where m is the number of threads involved in the run following the transition relation of \mathcal{M} and n is the number of threads involved in the run following the transition relation of \mathcal{M}_{fs} . Observe that runs of rank (m, n) where $n = 0$ (resp. $m = 0$) are precisely the runs of \mathcal{M} (resp. \mathcal{M}_{fs}). Then, we prove that for any computation of \mathcal{M}_{\cup} (from the initial configuration $c_{\mathcal{M}_{\cup}}^{\text{init}}$) of rank $(m+1, n)$ (resp. $(m, n+1)$), there is a run of \mathcal{A}_{\cup} of rank $(m, n+1)$ (resp. $(m+1, n)$). This run is obtained from the original one by replacing a thread that follows the transition relation of \mathcal{M} (resp. \mathcal{M}_{fs}) by a thread that follows the transition relation of \mathcal{M}_{fs} (resp. \mathcal{M}). This is possible since any thread of \mathcal{M}_{fs} can be simulated by a thread of \mathcal{M} and vice-versa. As an immediate consequence of the following result is that, for every $m \in \mathbb{N}$, a state q is k -bounded (resp. k -stratified) reachable by a run of \mathcal{M}_{\cup} of rank $(m, 0)$ (i.e., a run of \mathcal{M}_{fs}) if and only if it is k -bounded (resp. k -stratified) reachable by a run of \mathcal{M}_{\cup} of rank $(0, m)$ (i.e., a run of \mathcal{M}). This is precisely what Lemma 5.3 says.

E.1. The language of finite state automata $\mathcal{A}_{(q, \gamma)}$. In the following, we establish the following property about the finite state automata $\mathcal{A}_{(q, \gamma)}$:

Lemma E.1. *Let $i \leq k$. If there are elements $\sigma_0, \dots, \sigma_i \in \Gamma^*$, $\gamma_0, \dots, \gamma_i \in \Gamma$, $\alpha \in \Gamma_\epsilon$, $g_0, p_0, g_1, \dots, p_i, g_{i+1} \in Q$, $s_0 \in I_{(g_0, \gamma)}$, and $s \in S_{(g_0, \gamma)}$ such that*

$$\sigma_0(p_0, \gamma_1, g_1) \sigma_1 \cdots \sigma_{i-1}(p_{i-1}, \gamma_i, g_i) \sigma_i(p_i, \alpha, g_{i+1}) \in \text{Traces}_{\mathcal{T}(\mathcal{A}_{(g_0, \gamma)})}(\{s_0\}, \{s\}),$$

then

$$\sigma_0(p_0, \gamma_1, g_1) \sigma_1 \cdots \sigma_{i-1}(p_{i-1}, \gamma_i, g_i) \sigma_i(p_i, \epsilon, g_{i+1}) \in L(\mathcal{A}_{(g_0, \gamma)}).$$

Proof. Since all the states in the automaton $\mathcal{A}_{(g_0, \gamma)}$ are co-reachable from the final states, in particular the state s , there is $\nu \in \Sigma^*$ such that:

$$\sigma_0(p_0, \gamma_1, g_1) \sigma_1 \cdots \sigma_{i-1}(p_{i-1}, \gamma_i, g_i) \sigma_i(p_i, \alpha, g_{i+1}) \nu \in L(\mathcal{A}_{(g_0, \gamma)})$$

This implies that there are $\sigma'_0, \dots, \sigma'_i \in \Gamma^*$ and $\nu' \in \Sigma^*$ such that: (1) $\sigma_l \preceq \sigma'_l$ for all $l \in [0, i]$, (2) $\nu \preceq \nu'$, and (3) we have:

$$\sigma'_0(p_0, \gamma_1, g_1) \sigma'_1 \cdots \sigma'_{i-1}(p_{i-1}, \gamma_i, g_i) \sigma'_i(p_i, \alpha, g_{i+1}) \nu' \in L(\mathcal{P}_{(g_0, \gamma)})$$

Now, we can use the definition of $\mathcal{P}_{(g_0, \gamma)}$ to show that we have:

$$\sigma'_0(p_0, \gamma_1, g_1) \sigma'_1 \cdots \sigma'_{i-1}(p_{i-1}, \gamma_i, g_i) \sigma'_i(p_i, \epsilon, g_{i+1}) \in L(\mathcal{P}_{(g_0, \gamma)})$$

In addition, we can show that $\sigma'_0(p_0, \gamma_1, g_1) \sigma'_1 \cdots \sigma'_{i-1}(p_{i-1}, \gamma_i, g_i) \sigma'_i(p_i, \epsilon, g_{i+1}) \in L'_{((g_0, \gamma), i+1)}$ since $i \leq k$. This implies that $\sigma_0(p_0, \gamma_1, g_1) \sigma_1 \cdots \sigma_{i-1}(p_{i-1}, \gamma_i, g_i) \sigma_i(p_i, \epsilon, g_{i+1}) \in L(\mathcal{A}_{(g_0, \gamma)})$ since $\sigma_l \preceq \sigma'_l$ for all $l \in [0, i]$. \square

E.2. The relation between the DCFS \mathcal{M}_{fs} and the FSA $\mathcal{A}_{(p, \gamma)}$. In the following, we establish the link between the set of runs of a thread of \mathcal{M}_{fs} without a context switches and the language generated by the finite state automaton $\mathcal{A}_{(q, \gamma)}$.

Lemma E.2. *Let $j \in \mathbb{N}$, $s, s' \in S_{\text{fs}}^{\text{sm}}$, and $Val, Val' \in [Loc(\mathcal{M}_{\text{fs}}) \rightarrow \mathbb{N}]$. There is $\tau \in \Delta_{\text{fs}}^*$ such that $(\sharp, (s, j), Val) \xrightarrow{\tau}^*_{\mathcal{T}_{\{j\}}(\mathcal{M}_{\text{fs}})} (\sharp, (s', j), Val + Val')$ if and only if there are $q \in Q$, $\gamma \in \Gamma$, and $\sigma \in \Gamma^*$ such that $s \xrightarrow{\sigma}^*_{\mathcal{T}(\mathcal{A}_{(q, \gamma)})} s'$, $Val'((\gamma', j+1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val'((w, l)) = 0$ for all $w \in \Gamma_{\text{fs}}^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$.*

Proof. The Only if direction: Assume that there is $\tau \in \Delta_{\text{fs}}^*$ such that $(\sharp, (s, j), Val) \xrightarrow{\tau}_n_{\mathcal{T}_{\{j\}}(\mathcal{M}_{\text{fs}})} (\sharp, (s', j), Val + Val')$ for some $n \in \mathbb{N}$. We proceed by induction on n .

Basis. $n = 0$. Then, $\tau = \epsilon$, $s = s'$, and $Val' = \text{ld}_{Loc(\mathcal{M}_{\text{fs}})}^0$. Since $s \in S_{\text{fs}}^{\text{sm}}$, there is $q \in Q$ and $\gamma \in \Gamma$ such that $s \in S_{(q, \gamma)}$. By taking $\sigma = \epsilon$, we have $s \xrightarrow{\sigma}^*_{\mathcal{T}(\mathcal{A}_{(q, \gamma)})} s'$, $Val'((\gamma', j+1)) = |\sigma|_{\gamma'} = 0$ for all $\gamma' \in \Gamma$, and $Val'((w, l)) = 0$ for all $w \in \Gamma_{\text{fs}}^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$.

Step. $n > 0$. Then, from the definition of \mathcal{M}_{fs} , there is $\tau' \in \Delta_{\text{fs}}^*$, $t \in \Delta_{\text{fs}}$, $s'' \in S_{\text{fs}}^{\text{sm}}$, and $Val'' \in [Loc(\mathcal{M}_{\text{fs}}) \rightarrow \mathbb{N}]$ such that $\tau = \tau' t$, and:

$$(\sharp, (s, j), Val) \xrightarrow{\tau'}_{n-1} \mathcal{T}_{\{j\}}(\mathcal{M}_{\text{fs}})(\sharp, (s'', j), Val + Val'') \xrightarrow{t} \mathcal{T}_{\{j\}}(\mathcal{M}_{\text{fs}})(\sharp, (s', j), Val + Val') \quad (\text{E.1})$$

We apply the induction hypothesis to $(\sharp, (s, j), Val) \xrightarrow{\tau'}_{n-1} \mathcal{T}_{\{j\}}(\mathcal{M}_{\text{fs}})(\sharp, (s'', j), Val + Val'')$,

and we obtain that there are $q \in Q$, $\gamma \in \Gamma$, and $\sigma' \in \Gamma^*$ such that $s \xrightarrow{\sigma'}^*_{\mathcal{T}(\mathcal{A}_{(q, \gamma)})} s''$,

$Val''((\gamma', j+1)) = |\sigma'|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val''((w, l)) = 0$ for all $w \in \Gamma_{fs}^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$.

In addition, from the definition of \mathcal{M}_{fs} , t is necessarily of the form $\langle \sharp, s' \rangle \rightarrow_{\mathcal{M}} \langle \sharp, s' \rangle \triangleright \alpha$ with $\alpha \in \Gamma$. This implies that $s' \in S_{(q, \gamma)}$ and $s'' \xrightarrow{\alpha}_{\mathcal{A}(q, \gamma)} s'$. Moreover, we have $Val' = Val'' + \text{ld}_{Loc(\mathcal{M}_{fs})}^{\{(\alpha, j+1)\}}$. Then, by taking $\sigma = \sigma'\alpha$, we can show (using the induction hypothesis) that $s \xrightarrow{\sigma}_{\mathcal{T}(\mathcal{A}(q, \gamma))}^* s'$, $Val'((\gamma', j+1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val'((w, l)) = 0$ for all $w \in \Gamma_{fs}^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$.

The If direction: Assume that there are $q \in Q$, $\gamma \in \Gamma$, and $\sigma \in \Gamma^*$ such that $s \xrightarrow{\sigma}_n \mathcal{T}(\mathcal{A}(q, \gamma)) s'$ for some $n \in \mathbb{N}$, $Val'((\gamma', j+1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val'((w, l)) = 0$ for all $w \in \Gamma_{fs}^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$. We proceed by induction on n .

Basis. $n = 0$. Then, $s = s'$ and $\sigma = \epsilon$. By taking $\tau = \epsilon$, we get $(\sharp, (s, j), Val) \xrightarrow{\tau}_{\mathcal{T}_{\{j\}}(\mathcal{M}_{fs})}^* (\sharp, (s', j), Val + Val')$ since $Val' = \text{ld}_{Loc(\mathcal{M}_{fs})}^{\emptyset}$.

Step. $n > 0$. Then, there are s'' , $\sigma' \in \Gamma^*$, and $\alpha \in \Gamma$ such that $\sigma = \sigma'\alpha$, and:

$$s \xrightarrow{\sigma'}_{n-1} \mathcal{T}(\mathcal{A}(q, \gamma)) s'' \xrightarrow{\alpha}_{\mathcal{T}(\mathcal{A}(q, \gamma))} s' \quad (\text{E.2})$$

Let $Val'' \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ such that $Val''((\gamma', j+1)) = |\sigma'|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val''((w, l)) = 0$ for all $w \in \Gamma_{fs}^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$. Then, we apply the induction hypothesis to $s \xrightarrow{\sigma'}_{n-1} \mathcal{T}(\mathcal{A}(q, \gamma)) s''$ and Val'' , and we obtain that there is $\tau \in \Delta_{fs}^*$ such that $(\sharp, (s, j), Val) \xrightarrow{\tau'}_{\mathcal{T}_{\{j\}}(\mathcal{M}_{fs})}^* (\sharp, (s'', j), Val + Val'')$.

Since $s'' \xrightarrow{\alpha}_{\mathcal{T}(\mathcal{A}(q, \gamma))} s'$, we have $t = \langle \sharp, s'' \rangle \rightarrow_{\mathcal{M}_{fs}} \langle \sharp, s' \rangle \triangleright \alpha$. Then, using the induction hypothesis, we can show that $(\sharp, (s, j), Val) \xrightarrow{\tau}_{\mathcal{T}_{\{j\}}(\mathcal{M}_{fs})}^* (\sharp, (s', j), Val + Val')$ with $\tau = \tau't$ since we have $Val' = Val'' + \text{ld}_{Loc(\mathcal{M}_{fs})}^{\{(\alpha, j+1)\}}$. \square

Next, we use Lemma E.2 to establish the relation between the set of languages accepted by the finite state automata $\mathcal{A}(q, \gamma)$ and the set of runs of \mathcal{M}_{fs} between two configurations with no active thread and without context switches.

Lemma E.3. *Let $j \in \mathbb{N}$, $p_1, p_2 \in Q$, $\lambda_1 \in (S_{fs}^{sw} \cup \Gamma)$, $\lambda_2 \in S_{fs}^{sw}$, and $Val \in [Loc(\mathcal{M}_{fs}) \rightarrow \mathbb{N}]$. There is $\tau \in \Delta_{fs}^*$ such that $|\tau| \geq 1$ and $(p_1, \perp, \text{ld}_{Loc(\mathcal{M}_{fs})}^{\{(\lambda_1, j)\}}) \xrightarrow{\tau}_{\mathcal{T}_{\{j\}}(\mathcal{M}_{fs})}^* (p_2, \perp, Val + \text{ld}_{Loc(\mathcal{M}_{fs})}^{\{(\lambda_2, j+1)\}})$ iff there are $q \in Q$, $\gamma, \gamma_1 \in \Gamma$, $\alpha \in \Gamma_{\epsilon}$, $p'_1, p'_2 \in Q$, $s, s' \in S_{(q, \gamma)}$, and $\sigma \in \Gamma^*$ such that:*

- $p_1 \mapsto_{\mathcal{M}_{fs}} p'_1 \triangleleft \lambda_1$, $\lambda_1 = (p'_1, (s, \gamma_1))$ if $\lambda_1 \in S_{fs}^{sw}$, and $p'_1 = q$ and $s \in I_{(q, \gamma)}$ if $\lambda_1 \in \Gamma$.
- $s \xrightarrow{\sigma(p_2, \alpha, p'_2)}_{\mathcal{T}(\mathcal{A}(q, \gamma))}^* s'$ and $\lambda_2 = (p'_2, (s', \alpha))$.
- $Val((\gamma', j+1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$.
- $Val((w, l)) = 0$ for all $w \in \Gamma_{fs}^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$.

Proof. The Only if direction: Assume the existence of some $\tau \in \Delta_{fs}^*$ such that $|\tau| \geq 1$ and $(p_1, \perp, \text{ld}_{Loc(\mathcal{M}_{fs})}^{\{(\lambda_1, j)\}}) \xrightarrow{\tau}_{\mathcal{T}_{\{j\}}(\mathcal{M}_{fs})}^* (p_2, \perp, Val + \text{ld}_{Loc(\mathcal{M}_{fs})}^{\{(\lambda_2, j+1)\}})$. Then, from the definition of $\mathcal{T}_{\{j\}}(\mathcal{M}_{fs})$, there are $q \in Q$, $\gamma, \gamma_1 \in \Gamma$, $p'_1 \in Q$, $\tau' \in \Delta_{fs}^*$, $s, s'' \in S_{(q, \gamma)}$, and $\sigma \in \Gamma^*$ such that:

- $(p_1, \perp, \text{ld}_{Loc(\mathcal{M}_{fs})}^{\{(\lambda_1, j)\}}) \xrightarrow{t}_{\mathcal{T}_{\{j\}}(\mathcal{M}_{fs})} (p'_1, (\lambda_1, j), \text{ld}_{Loc(\mathcal{M}_{fs})}^{\emptyset})$ such that $t = p_1 \mapsto_{\mathcal{M}_{fs}} p'_1 \triangleleft \lambda_1$.

- $(p'_1, (\lambda_1, j), \text{Id}_{Loc(\mathcal{M}_{fs})}^\emptyset) \xrightarrow{t'} \mathcal{T}_{\{j\}}(\mathcal{M}_{fs}) (\sharp, (s, j), \text{Id}_{Loc(\mathcal{M}_{fs})}^\emptyset)$ with $t' = \langle p'_1, \lambda_1 \rangle \mapsto_{\mathcal{M}_{fs}} \langle \sharp, s \rangle \triangleright \epsilon$, $\lambda_1 = (p'_1, (s, \gamma_1))$ if $\lambda_1 \in S_{fs}^{sw}$, and $p'_1 = q$ and $s \in I_{(q, \gamma)}$ if $\lambda_1 \in \Gamma$.
- $(\sharp, (s, j), \text{Id}_{Loc(\mathcal{M}_{fs})}^\emptyset) \xrightarrow{\tau'}^* \mathcal{T}_{\{j\}}(\mathcal{M}) (\sharp, (s'', j), Val)$. Then, we apply Lemma E.2, and we obtain that there is $\sigma \in \Gamma^*$ such that $s \xrightarrow{\sigma}^*_{\mathcal{T}(\mathcal{A}_{(q, \gamma)})} s''$, $Val((\gamma', j+1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$ and $Val((w, l)) = 0$ for all $w \in \Gamma_{fs}^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$.
- $(\sharp, (s'', j), Val) \xrightarrow{t''} \mathcal{T}_{\{j\}}(\mathcal{M}) (p_2, \perp, Val + \text{Id}_{Loc(\mathcal{M}_{fs})}^{\{(\lambda_2, j+1)\}})$ with $t'' = \langle \sharp, s'' \rangle \mapsto_{\mathcal{M}_{fs}} \langle p_2, \lambda_2 \rangle$. From the definition of \mathcal{M}_{fs} , this implies that there is $p'_2 \in Q$, $s' \in S_{(q, \gamma)}$, and $\alpha \in \Gamma_\epsilon$ such that $s'' \xrightarrow{(p_2, \alpha, p'_2)} s'$ and $\lambda_2 = (p'_2, (s', \alpha))$.

This terminates the proof of the Only if direction.

The If direction: Assume that there are $q \in Q$, $\gamma, \gamma_1 \in \Gamma$, $\alpha \in \Gamma_\epsilon$, $p'_1, p'_2 \in Q$, $s, s' \in S_{(q, \gamma)}$, and $\sigma \in \Gamma^*$ such that:

- $t = p_1 \mapsto_{\mathcal{M}} p'_1 \triangleleft \lambda_1$, $\lambda_1 = (p'_1, (s, \gamma_1))$ if $\lambda_1 \in S_{fs}^{sw}$, and $p'_1 = q$ and $s \in I_{(q, \gamma)}$ if $\lambda_1 \in \Gamma$.
- $s \xrightarrow{\sigma(p_2, \alpha, p'_2)}^*_{\mathcal{T}(\mathcal{A}_{(q, \gamma)})} s'$ and $\lambda_2 = (p'_2, (s', \alpha))$.
- $Val((\gamma', j+1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$.
- $Val((w, l)) = 0$ for all $w \in \Gamma_{fs}^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$.

Then, from the definition of $\mathcal{T}(\mathcal{M}_{fs})$, we have the following run:

$$(p_1, \perp, \text{Id}_{Loc(\mathcal{M}_{fs})}^{\{(\lambda_1, j)\}}) \xrightarrow{t} \mathcal{T}_{\{j\}}(\mathcal{M}_{fs}) (p'_1, (\lambda_1, j), \text{Id}_{Loc(\mathcal{M}_{fs})}^\emptyset) \quad (\text{E.3})$$

Let $t' = \langle p'_1, \lambda_1 \rangle \mapsto_{\mathcal{M}_{fs}} \langle \sharp, s \rangle \triangleright \epsilon$. Then, we have the following run of $\mathcal{T}_{\{j\}}(\mathcal{M}_{fs})$:

$$(p'_1, (\lambda_1, j), \text{Id}_{Loc(\mathcal{M}_{fs})}^\emptyset) \xrightarrow{t'} \mathcal{T}_{\{j\}}(\mathcal{M}_{fs}) (\sharp, (s, j), \text{Id}_{Loc(\mathcal{M}_{fs})}^\emptyset) \quad (\text{E.4})$$

Let $s'' \in S_{(q, \gamma)}$ such that $s \xrightarrow{\sigma}^*_{\mathcal{T}(\mathcal{A}_{(q, \gamma)})} s''$ and $s'' \xrightarrow{(p_2, \alpha, p'_2)}_{\mathcal{T}(\mathcal{A}_{(q, \gamma)})} s'$. Then, we can apply Lemma E.2, to prove that there is $\tau' \in \Delta_{fs}^*$ such that:

$$(\sharp, (s, j), \text{Id}_{Loc(\mathcal{M}_{fs})}^\emptyset) \xrightarrow{\tau'}^* \mathcal{T}_{\{j\}}(\mathcal{M}) (\sharp, (s'', j), Val) \quad (\text{E.5})$$

Since $s'' \xrightarrow{(p_2, \alpha, p'_2)}_{\mathcal{T}(\mathcal{A}_{(q, \gamma)})} s'$, we have $\langle \sharp, s'' \rangle \mapsto_{\mathcal{M}_{fs}} \langle p_2, \lambda_2 \rangle$. This implies that $\mathcal{T}_{\{j\}}(\mathcal{M}_{fs})$ has the following run:

$$(\sharp, (s'', j), Val) \xrightarrow{t''} \mathcal{T}_{\{j\}}(\mathcal{M}) (p_2, \perp, Val + \text{Id}_{Loc(\mathcal{M}_{fs})}^{\{(\lambda_2, j+1)\}}) \quad (\text{E.6})$$

This terminates the proof of the If direction. \square

E.3. The relation between the DCPS \mathcal{M} and the PDA $\mathcal{P}_{(p, \gamma)}$. In the following, we establish the link between the set of runs of a thread of \mathcal{M} and the language generated by the pushdown automaton $\mathcal{P}_{(q, \gamma)}$.

Lemma E.4. *Let $j \in \mathbb{N}$, $q \in Q$, $\gamma \in \Gamma$, $p_1, p_2 \in Q$, $w_1, w_2 \in \Gamma^*$, and $Val, Val' \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$. There is $\tau \in \Delta^*$ such that $(p_1, (w_1, j), Val) \xrightarrow{\tau}^*_{\mathcal{T}_{\{j\}}(\mathcal{M})} (p_2, (w_2, j), Val + Val')$ for some iff there is $\sigma \in \Gamma^*$ such that $(p_1, w_1) \xrightarrow{\sigma}^*_{\mathcal{P}_{(q, \gamma)}} (p_2, w_2)$, $Val'((\gamma', j+1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val'((w, l)) = 0$ for all $w \in \Gamma_\epsilon$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$.*

Proof. The Only if direction: Assume that there is $\tau \in \Delta^*$ and some $n \in \mathbb{N}$ such that $(p_1, (w_1, j), Val) \xrightarrow[n]{\tau} \tau_{\{j\}}(\mathcal{M})(p_2, (w_2, j), Val + Val')$. We proceed by induction on n .

Basis. $n = 0$. This implies that $p_1 = p_2$, $w_1 = w_2$, $\tau = \epsilon$, and $Val' = \text{Id}_{Loc(\mathcal{M})}^\emptyset$. By taking $\sigma = \epsilon$, we have $(p_1, w_1) \xrightarrow{\sigma} \mathcal{P}(q, \gamma)(p_2, w_2)$, $Val'((\gamma', j + 1)) = |\sigma|_{\gamma'} = 0$ for all $\gamma' \in \Gamma$, and $Val'((w, l)) = 0$ for all $w \in \Gamma_\epsilon$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j + 1\}$.

Step. $n > 0$. From the definition of \mathcal{M} , this implies that there are $p \in Q$, $w'_1 \in \Gamma^*$, $Val'' \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$, $\tau' \in \Delta^*$, and $t \in \Delta$ such that $\tau = \tau't$ and:

$$(p_1, (w_1, j), Val) \xrightarrow[n-1]{\tau'} \tau_{\{j\}}(\mathcal{M})(p, (w'_1, j), Val + Val'') \xrightarrow{t} \tau_{\{j\}}(\mathcal{M})(p_2, (w_2, j), Val + Val') \quad (\text{E.7})$$

We apply the induction hypothesis to $(p_1, (w_1, j), Val) \xrightarrow{\tau'} \tau_{\{j\}}^*(\mathcal{M})(p, (w'_1, j), Val + Val'')$, and we obtain that there is $\sigma' \in \Gamma^*$ such that $(p_1, w_1) \xrightarrow{\sigma'} \mathcal{T}(\mathcal{P}(q, \gamma))(p, w'_1)$, $Val''((\gamma', j + 1)) = |\sigma'|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val''((w, l)) = 0$ for all $w \in \Gamma_\epsilon$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j + 1\}$.

In addition, from the definition of \mathcal{M} , the transition t is necessarily of the form $\langle p, \gamma_1 \rangle \rightarrow_{\mathcal{M}} \langle p_2, u \rangle \triangleright \alpha$ with $\gamma_1 \in \Gamma$ and $\alpha \in \Gamma_\epsilon$ such that $w'_1 = \gamma_1 v$ and $w_2 = uv$ for some $v \in \Gamma^*$. This implies that $\langle p, \gamma_1 \rangle \xrightarrow{\alpha} \mathcal{P}(q, \gamma) \langle p_2, u \rangle$, and so, $(p, w'_1) \xrightarrow{\alpha} \mathcal{T}(\mathcal{P}(q, \gamma))(p_2, w_1)$.

Moreover, we have $Val' = Val'' + \text{Id}_{Loc(\mathcal{M})}^{\{(\alpha, j+1)\}}$. Then, by taking $\sigma = \sigma'\alpha$, we can easily show (using the induction hypothesis) that $(p_1, w_1) \xrightarrow{\sigma} \mathcal{P}(q, \gamma)(p_2, w_2)$, $Val'((\gamma', j + 1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val'((w, l)) = 0$ for all $w \in \Gamma_\epsilon$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j + 1\}$.

The If direction: Assume that there is $\sigma \in \Gamma^*$ such that $(p_1, w_1) \xrightarrow[n]{\sigma} \mathcal{P}(q, \gamma)(p_2, w_2)$, $Val'((\gamma', j + 1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val'((w, l)) = 0$ for all $w \in \Gamma_\epsilon$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j + 1\}$. We proceed by induction on n .

Basis. $n = 0$. This implies that $p_1 = p_2$, $w_1 = w_2$, $\sigma = \epsilon$, and $Val' = \text{Id}_{Loc(\mathcal{M})}^\emptyset$. By taking $\tau = \epsilon$, we have $(p_1, (w_1, j), Val) \xrightarrow{\tau} \tau_{\{j\}}^*(\mathcal{M})(p_2, (w_2, j), Val + Val')$.

Step. $n > 0$. Then, from the definition of $\mathcal{P}(q, \gamma)$, there are $p \in Q$, $w'_1 \in \Gamma^*$, $\sigma' \in \Gamma^*$, and $\alpha \in \Gamma_\epsilon$ such that $\sigma = \sigma'\alpha$, and:

$$(p_1, w_1) \xrightarrow[n-1]{\sigma'} \mathcal{T}(\mathcal{P}(q, \gamma))(p, w'_1) \xrightarrow{\alpha} \mathcal{T}(\mathcal{P}(q, \gamma))(p_2, w_2) \quad (\text{E.8})$$

Let $Val'' \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ such that $Val''((\gamma', j + 1)) = |\sigma'|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val''((w, l)) = 0$ for all $w \in \Gamma_\epsilon$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j + 1\}$.

Then, we apply the induction hypothesis to $(p_1, w_1) \xrightarrow[n-1]{\sigma'} \mathcal{T}(\mathcal{P}(q, \gamma))(p, w'_1)$ and Val'' , and we obtain that there is $\tau' \in \Delta^*$ such that $(p_1, (w_1, j), Val) \xrightarrow{\tau'} \tau_{\{j\}}^*(\mathcal{M})(p, (w'_1, j), Val + Val'')$

Since $(p, w'_1) \xrightarrow{\alpha} \mathcal{T}(\mathcal{P}(q, \gamma))(p_2, w_2)$, there are elements $\gamma_1 \in \Gamma$ and $u \in \Gamma^*$ such that $t = \langle p, \gamma_1 \rangle \rightarrow_{\mathcal{M}} \langle p_2, u \rangle \triangleright \alpha$, $w'_1 = \gamma_1 v$, and $w_2 = uv$ for some $v \in \Gamma^*$. Then, using the induction hypothesis, we can easily show that $(p_1, (w_1, j), Val) \xrightarrow{\tau} \tau_{\{j\}}^*(\mathcal{M})(p_2, (w_2, j), Val + Val')$ where $\tau = \tau't$. \square

Next, we use Lemma E.4 to establish the relation between the set of languages accepted by the pushdown automata $\mathcal{P}_{(q,\gamma)}$ and the set of runs of \mathcal{M} between two configurations with no active thread and without context switches.

Lemma E.5. *Let $j \in \mathbb{N}$, $q \in Q$, $\gamma \in \Gamma$, $p_1, p_2 \in Q$, $w_1 \in \Gamma^*$, and $Val \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$. There is $\tau \in \Delta^*$ such that $|\tau| \geq 1$ and $(p_1, \perp, \text{Id}_{Loc(\mathcal{M})}^{\{(w_1, j)\}}) \xrightarrow{\tau}^*_{\mathcal{T}_{\{j\}}(\mathcal{M})} (p_2, \perp, Val)$ iff there are $p'_1 \in Q$, $\gamma_1 \in \Gamma$, $w_2 \in \Gamma^*$, $\sigma \in \Gamma^*$, $Val' \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ such that:*

- $p_1 \mapsto_{\mathcal{M}} p'_1 \triangleleft \gamma_1$ and $w_1 = \gamma_1 v_1$ for some $v_1 \in \Gamma^*$.
- $Val = Val' + \text{Id}_{Loc(\mathcal{M})}^{\{(w_2, j+1)\}}$.
- $(p'_1, w_1) \xrightarrow{\sigma(p_2, \alpha, p'_2)}^*_{\mathcal{T}(\mathcal{P}_{(q,\gamma)})} (p'_2, w_2)$ for all $p'_2 \in Q$ and $\alpha \in \Gamma_\epsilon$ such that $w_2 = \alpha v_2$ for some $v_2 \in \Gamma^*$.
- $Val'((\gamma', j+1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$.
- $Val'((w, l)) = 0$ for all $w \in \Gamma_\epsilon$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$.

Proof. The Only if direction: Assume that $(p_1, \perp, \text{Id}_{Loc(\mathcal{M})}^{\{(w_1, j)\}}) \xrightarrow{\tau}^*_{\mathcal{T}_{\{j\}}(\mathcal{M})} (p_2, \perp, Val)$ for some $\tau \in \Delta^*$ such that $|\tau| \geq 1$. Then, from the definition of $\mathcal{T}_{\{j\}}(\mathcal{M})$, there are $p'_1, p' \in Q$, $\gamma_1, \gamma_2 \in \Gamma$, $w'_2, u \in \Gamma^*$, $\tau' \in \Delta^*$, $Val' \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ such that:

- $(p_1, \perp, \text{Id}_{Loc(\mathcal{M})}^{\{(w_1, j)\}}) \xrightarrow{t} \mathcal{T}_{\{j\}}(\mathcal{M}) (p'_1, (w_1, j), \text{Id}_{Loc(\mathcal{M})}^\emptyset)$ such that $t = p_1 \mapsto_{\mathcal{M}} p'_1 \triangleleft \gamma_1$. This implies that $w_1 = \gamma_1 v_1$ for some $v_1 \in \Gamma^*$.
- $(p'_1, (w_1, j), \text{Id}_{Loc(\mathcal{M})}^\emptyset) \xrightarrow{\tau'}^*_{\mathcal{T}_{\{j\}}(\mathcal{M})} (p', (w'_2, j), Val')$. Then, we can apply Lemma E.4, to show that there is $\sigma \in \Gamma^*$ such that $(p'_1, w_1) \xrightarrow{\sigma}^*_{\mathcal{P}_{(q,\gamma)}} (p', w'_2)$, $Val'((\gamma', j+1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$, and $Val'((w, l)) = 0$ for all $w \in \Gamma_\epsilon$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$.
- $(p', (w'_2, j), Val') \xrightarrow{t'} \mathcal{T}_{\{j\}}(\mathcal{M}) (p_2, \perp, \text{Id}_{Loc(\mathcal{M})}^{\{w_2, j+1\}} + Val')$ such that: (1) $Val = \text{Id}_{Loc(\mathcal{M})}^{\{w_2, j+1\}} + Val'$, (2) $t' = \langle p', \gamma_2 \rangle \mapsto_{\mathcal{M}} \langle p_2, u \rangle$, and (3) $w'_2 = \gamma_2 v$ and $w_2 = uv$ for some $v \in \Gamma^*$. Using the definition of $\mathcal{P}_{(q,\gamma)}$, we have $(p', w'_2) \xrightarrow{(p_2, \alpha, p'_2)}^*_{\mathcal{P}_{(q,\gamma)}} (p'_2, w_2)$ for all $p'_2 \in Q$ and $\alpha \in \Gamma_\epsilon$ such that $w_2 = \alpha v_2$ for some $v_2 \in \Gamma^*$.

This terminates the proof of the only if direction of Lemma E.5

The If direction: Assume that there are $p'_1 \in Q$, $\gamma_1 \in \Gamma$, $w_2 \in \Gamma^*$, $\sigma \in \Gamma^*$, $Val' \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$ such that:

- $t = p_1 \mapsto_{\mathcal{M}} p'_1 \triangleleft \gamma_1$ and $w_1 = \gamma_1 v_1$ for some $v_1 \in \Gamma^*$.
- $Val = Val' + \text{Id}_{Loc(\mathcal{M})}^{\{(w_2, j+1)\}}$.
- $(p'_1, w_1) \xrightarrow{\sigma(p_2, \alpha, p'_2)}^*_{\mathcal{P}_{(q,\gamma)}} (p'_2, w_2)$ for all $p'_2 \in Q$ and $\alpha \in \Gamma_\epsilon$ such that $w_2 = \alpha v_2$ for some $v_2 \in \Gamma^*$.
- $Val'((\gamma', j+1)) = |\sigma|_{\gamma'}$ for all $\gamma' \in \Gamma$.
- $Val'((w, l)) = 0$ for all $w \in \Gamma_\epsilon$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j+1\}$

Since $t = p_1 \mapsto_{\mathcal{M}} p'_1 \triangleleft \gamma_1$ and $w_1 = \gamma_1 v_1$ for some $v_1 \in \Gamma^*$, we have the following run of $\mathcal{T}_{\{j\}}(\mathcal{M})$:

$$(p_1, \perp, \text{Id}_{Loc(\mathcal{M})}^{\{(w_1, j)\}}) \xrightarrow{t} \mathcal{T}_{\{j\}}(\mathcal{M}) (p'_1, (w_1, j), \text{Id}_{Loc(\mathcal{M})}^\emptyset) \quad (\text{E.9})$$

Let $p' \in Q$ and $w'_2 \in \Gamma^*$ such that $(p'_1, w_1) \xrightarrow{\sigma}^*_{\mathcal{P}_{(q,\gamma)}} (p', w'_2)$ and $(p', w'_2) \xrightarrow{(p_2, \epsilon, p'_2)}_{\mathcal{P}_{(q,\gamma)}} (p'_2, w_2)$ for some $p'_2 \in Q$. Now, we can apply Lemma E.4 to $(p'_1, w_1) \xrightarrow{\sigma}^*_{\mathcal{P}_{(q,\gamma)}} (p', w'_1)$ show that there is $\tau' \in \Delta^*$ such that:

$$(p'_1, (w_1, j), \text{ld}_{Loc(\mathcal{M})}^\emptyset) \xrightarrow{\tau'}^*_{\mathcal{T}_{\{j\}}(\mathcal{M})} (p', (w'_2, j), Val') \quad (\text{E.10})$$

Since $(p', w'_2) \xrightarrow{(p_2, \epsilon, p'_2)}_{\mathcal{P}_{(q,\gamma)}} (p'_2, w_2)$, we can use the definition of $\mathcal{P}_{(q,\gamma)}$, to show that there are $\gamma_2 \in \Gamma$ and $u \in \Gamma^*$ such that $t' = \langle p', \gamma_2 \rangle \mapsto_{\mathcal{M}} \langle p_2, u \rangle$ such that $w'_2 = \gamma_2 v$ and $w_2 = uv$ for some $v \in \Gamma^*$. So, $\mathcal{T}_j(\mathcal{M})$ has the following run:

$$(p', (w'_2, j), Val') \xrightarrow{t'}_{\mathcal{T}_{\{j\}}(\mathcal{M})} (p_2, \perp, \text{ld}_{Loc(\mathcal{M})}^{\{w_2, j+1\}} + Val') \quad (\text{E.11})$$

Now, we can put together Equation E.9, Equation E.10, and Equation E.11, and we obtain that $(p_1, \perp, \text{ld}_{Loc(\mathcal{M})}^{\{(w_1, j)\}}) \xrightarrow{\tau}^*_{\mathcal{T}_{\{j\}}(\mathcal{M})} (p_2, \perp, Val)$ with $\tau = t\tau't'$. \square

E.4. From the DCPS \mathcal{M} to the DCPS \mathcal{M}_{pf} . In order to be able to distinguish between pending threads of \mathcal{M} that has been activated at least one time from the other ones, we need to define a DCPS \mathcal{M}_{pf} (which is just a copy of \mathcal{M}) that uses, in addition to the stack alphabet Γ , a new stack alphabet Γ' , which is a copy Γ , to process threads. Let Γ' be a stack alphabet such that $\Gamma' \cap \Gamma_{\text{fs}} = \emptyset$ and there is a bijective function f from Γ to Γ' . This function f is extended to words over Γ in the natural way: $f(\epsilon) = \epsilon$ and $f(u \cdot v) = f(u) \cdot f(v)$ for all $u, v \in \Gamma^*$. Moreover, we define the function h from Γ_{pf} to Γ such that $h(\gamma) = h(f(\gamma)) = \gamma$ for all $\gamma \in \Gamma$. The function h is extended in the usual way to words.

In the following, we define the DCPS \mathcal{M}_{pf} obtained from \mathcal{M} by using Γ' to process threads instead of Γ . Let $\mathcal{M}_{\text{pf}} = (Q, \Gamma_{\text{pf}}, \Delta_{\text{pf}}, q_0, \gamma_0, F)$ be a DCPS where $\Gamma_{\text{pf}} = \Gamma \cup \Gamma'$ and Δ'_{pf} is the smallest transition relation satisfying the following conditions:

- *Initialize:* For every $q \in Q$ and $\gamma \in \Gamma$, we have $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}_{\text{pf}}} \langle q, f(\gamma) \rangle \triangleright \epsilon$.
- *Spawn:* For every $\langle q, \gamma \rangle \rightarrow_{\mathcal{M}} \langle q', u \rangle \triangleright \alpha$, we have $\langle q, f(\gamma) \rangle \rightarrow_{\mathcal{M}_{\text{pf}}} \langle q', f(u) \rangle \triangleright \alpha$.
- *Interrupt:* For every $\langle q, \gamma \rangle \mapsto_{\mathcal{M}} \langle q', u \rangle$, we have $\langle q, f(\gamma) \rangle \mapsto_{\mathcal{M}_{\text{pf}}} \langle q', f(u) \rangle$.
- *Dispatch:* For every $q \mapsto_{\mathcal{M}} q' \triangleleft \gamma$, we have $q \mapsto_{\mathcal{M}_{\text{pf}}} q' \triangleleft \gamma$ and $q \mapsto_{\mathcal{M}_{\text{pf}}} q' \triangleleft f(\gamma)$.

Then, the relation between a thread execution of \mathcal{M} and a thread execution of \mathcal{M}_{pf} is given by the following lemma:

Lemma E.6. *Let $j \in \mathbb{N}$, $p_1, p_2 \in Q$, $w_1 \in ((\Gamma')^* \cup \Gamma)$, $w_2 \in (\Gamma')^*$, $Val \in [Loc(\mathcal{M}) \rightarrow \mathbb{N}]$, and $Val' \in [Loc(\mathcal{M}_{\text{pf}}) \rightarrow \mathbb{N}]$. There is $\tau' \in \Delta^*_{\text{pf}}$ such that*

$$(p_1, \perp, \text{ld}_{Loc(\mathcal{M}_{\text{pf}})}^{\{(w_1, j)\}}) \xrightarrow{\tau'}_{\mathcal{T}_j(\mathcal{M}_{\text{pf}})} (p_2, \perp, Val' + \text{ld}_{Loc(\mathcal{M}_{\text{pf}})}^{\{(w_2, j+1)\}})$$

if and only if there is $\tau \in \Delta^$ such that*

$$(p_1, \perp, \text{ld}_{Loc(\mathcal{M}_{\text{pf}})}^{\{(h(w_1), j)\}}) \xrightarrow{\tau}_{\mathcal{T}_j(\mathcal{M})} (p_2, \perp, Val + \text{ld}_{Loc(\mathcal{M}_{\text{pf}})}^{\{(h(w_2), j+1)\}})$$

and $Val'((\gamma', j+1)) = Val((\gamma', j+1))$ for all $\gamma' \in \Gamma$, and $Val'((w', l)) = Val((w, l)) = 0$ for all $w \in \Gamma^$, $w' \in \Gamma^*_{\text{pf}}$, and $l \in \mathbb{N}$ such that $(w, l), (w', l) \notin \Gamma \times \{j+1\}$.*

Moreover, we can show that the BSR[k] problem for \mathcal{M} is reducible to its corresponding problem for \mathcal{M}_{pf} .

Lemma E.7. *For every $k \in \mathbb{N}$, a state $q \in Q$ is k -bounded reachable by \mathcal{M} if and only if q is k -bounded reachable by \mathcal{M}_{pf} .*

E.5. From the DCPS \mathcal{M}_{pf} and the DCFS \mathcal{M}_{fs} to the DCPS \mathcal{M}_{\cup} . We define the DCPS $\mathcal{M}_{\cup} = (Q_{\cup}, \Gamma_{\cup}, \Delta_{\cup}, q_0, \gamma_0, F)$ as the union of \mathcal{M}_{pf} and \mathcal{M}_{fs} where: (1) $Q_{\cup} = Q_{\text{fs}}$, (2) $\Gamma_{\cup} = \Gamma_{\text{pf}} \cup \Gamma_{\text{fs}}$ is a finite set of stack symbols, and (3) $\Delta_{\cup} = \Delta_{\text{pf}} \cup \Delta_{\text{fs}}$ is the transition relation.

Now, we are ready to define the rank of a run of $\mathcal{T}(\mathcal{M}_{\cup})$. Intuitively, the number of threads that are simulated according to Δ_{fs} (resp. Δ_{is}) is given by the number of pending threads with stack configuration in $S_{\text{fs}}^{\text{sw}} \times \mathbb{N}$ (resp. $(\Gamma')^* \times \mathbb{N}$). Formally, we have:

Definition E.8. (The rank of a run of \mathcal{M}_{\cup}) Let $\rho = c_{\mathcal{M}_{\cup}}^{\text{init}} \xrightarrow{\tau}^*_{\mathcal{T}_{[0,k]}(\mathcal{M}_{\cup})} c$ be a run of $\mathcal{T}_{[0,k]}(\mathcal{M}_{\cup})$ such that $\text{Active}(c) = \perp$. The rank of ρ , denoted by $\text{rank}(\rho)$, is defined by the pair (m, n) with $m = \sum_{(w,j) \in (\Gamma')^* \times \mathbb{N}} \text{Idle}(c)((w, j))$ and $n = \sum_{(\lambda, j) \in S_{\text{fs}}^{\text{sw}} \times \mathbb{N}} \text{Idle}(c)((\lambda, j))$.

E.6. From a run of rank $(m+1, n)$ of \mathcal{M}_{\cup} to a run of rank $(m, n+1)$ of \mathcal{M}_{\cup} . In the following, we establish that given a run of \mathcal{M}_{\cup} such that there is one thread executed following the set of transitions Δ_{pf} , we can compute a run of \mathcal{M}_{\cup} where the execution of this thread is replaced by an execution of a thread following the set of transitions Δ_{fs} . To this aim, we need first to prove Lemma E.9 which states that for any run of a thread of \mathcal{M}_{pf} , we can construct an equivalent run of a thread of \mathcal{M}_{fs} .

Lemma E.9. *Let $\gamma \in \Gamma$ and $i, j \in \mathbb{N}$ such that $i + j \leq k$. If there are $p_0, p'_0, \dots, p_i, p'_i \in Q$, $w_1, \dots, w_{i+1} \in (\Gamma')^*$, $\tau_0, \dots, \tau_i \in \Delta_{\cup}^*$, and $\text{Val}'_0, \dots, \text{Val}'_i \in [\text{Loc}(\mathcal{M}_{\cup}) \rightarrow \mathbb{N}]$ such that:*

- $(p_0, \perp, \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(\gamma, j)\}}) \xrightarrow{\tau_0}^*_{\mathcal{T}_{\{j\}}(\mathcal{M}_{\cup})} (p'_0, \perp, \text{Val}'_0 + \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(w_1, j+1)\}})$.
- For every $l \in [1, i]$, $(p_l, \perp, \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(w_l, j+l)\}}) \xrightarrow{\tau_l}^*_{\mathcal{T}_{\{j+l\}}(\mathcal{M}_{\cup})} (p'_l, \perp, \text{Val}'_l + \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(w_{l+1}, j+l+1)\}})$.

Then, there are $\lambda_1, \dots, \lambda_{i+1} \in S_{\text{fs}}^{\text{sw}}$ and $\tau'_0, \dots, \tau'_i \in \Delta_{\cup}^$ such that:*

- $(p_0, \perp, \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(\gamma, j)\}}) \xrightarrow{\tau'_0}^*_{\mathcal{T}_{\{j\}}(\mathcal{M}_{\cup})} (p'_0, \perp, \text{Val}'_0 + \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(\lambda_1, j+1)\}})$.
- For every $l \in [1, i]$, $(p_l, \perp, \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(\lambda_l, j+l)\}}) \xrightarrow{\tau'_l}^*_{\mathcal{T}_{\{j+l\}}(\mathcal{M}_{\cup})} (p'_l, \perp, \text{Val}'_l + \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(\lambda_{l+1}, j+l+1)\}})$.

Proof. Let us assume there are $p_0, p'_0, \dots, p_i, p'_i \in Q$, $w_1, \dots, w_{i+1} \in (\Gamma')^*$, $\tau_0, \dots, \tau_i \in \Delta_{\cup}^*$, and $\text{Val}'_0, \dots, \text{Val}'_i \in [\text{Loc}(\mathcal{M}_{\cup}) \rightarrow \mathbb{N}]$ such that:

- $(p_0, \perp, \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(\gamma, j)\}}) \xrightarrow{\tau_0}^*_{\mathcal{T}_{\{j\}}(\mathcal{M}_{\cup})} (p'_0, \perp, \text{Val}'_0 + \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(w_1, j+1)\}})$.
- For every $l \in [1, i]$, $(p_l, \perp, \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(w_l, j+l)\}}) \xrightarrow{\tau_l}^*_{\mathcal{T}_{\{j+l\}}(\mathcal{M}_{\cup})} (p'_l, \perp, \text{Val}'_l + \text{ld}_{\text{Loc}(\mathcal{M}_{\cup})}^{\{(w_{l+1}, j+l+1)\}})$.

From now, we confuse the system \mathcal{M}_{\cup} and \mathcal{M}_{pf} (resp. \mathcal{M}_{\cup} and \mathcal{M}_{fs}) when \mathcal{M}_{\cup} behaves according to the set of transitions Δ_{pf} (resp. Δ_{fs}).

Then, we apply Lemma E.5 and Lemma E.6 to show that there are $\sigma_0, \dots, \sigma_i \in \Gamma^*$, $\gamma_0, \dots, \gamma_i \in \Gamma$, and $g_0, \dots, g_{i+1} \in Q$ such that:

- $\gamma_0 = \gamma$.
- For every $l \in [0, i]$, $p_l \mapsto_{\mathcal{M}_{\cup}} g_l \triangleleft \gamma_l$.
- $\sigma_0(p'_0, \gamma_1, g_1)\sigma_1 \cdots \sigma_{i-1}(p'_{i-1}, \gamma_i, g_i)\sigma_i(p'_i, \epsilon, g_{i+1})$ in $L'_{((g_0, \gamma), i+1)}$.
- For every $l \in [0, i]$, $\text{Val}'_l((\gamma', j+l+1)) = |\sigma_l|_{\gamma'}$ for all $\gamma' \in \Gamma$.

- For every $l \in [0, i]$, $Val'_l((w, l)) = 0$ for all $w \in \Gamma_{\cup}^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j + l + 1\}$.

Since $L'_{((g_0, \gamma), i+1)} \subseteq L(\mathcal{A}_{(g_0, \gamma)})$, there are $s_0, \dots, s_{i+1} \in S_{(g_0, \gamma)}$ such that: (1) $s_0 \in I_{(g_0, \gamma)}$, (2) $s_l \xrightarrow{\sigma_l(p'_l, \gamma_{l+1}, g_{l+1})}^* \mathcal{T}_{(\mathcal{A}_{(g_0, \gamma)})} s_{l+1}$ for all $l \in [0, i]$, and (3) $s_i \xrightarrow{\sigma_i(p'_i, \epsilon, g_{i+1})}^* \mathcal{T}_{(\mathcal{A}_{(g_0, \gamma)})} s_{i+1}$.

Let $\lambda_l = (g_l, (s_l, \gamma_l))$ for all $l \in [1, i]$ and $\lambda_{i+1} = (g_{i+1}, (s_{i+1}, \epsilon))$. Since $p_l \mapsto_{\mathcal{M}_{\cup}} g_l \triangleleft \gamma_l$ for all $l \in [1, i]$, we can use the definition of \mathcal{M}_{fs} to show that $p_l \mapsto_{\mathcal{M}_{\cup}} g_l \triangleleft \lambda_l$ for all $l \in [1, i]$.

Now, we can apply Lemma E.2 to prove that there are $\tau'_0, \dots, \tau'_i \in \Delta_{\cup}^*$ such that:

- $(p_0, \perp, \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(\gamma, j)\}}) \xrightarrow{\tau'_0}^* \mathcal{T}_{\{j\}}(\mathcal{M}_{\cup})(p'_0, \perp, Val'_0 + \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(\lambda_1, j+1)\}})$.
- For every $l \in [1, i]$, $(p_l, \perp, \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(\lambda_l, j+l)\}}) \xrightarrow{\tau'_l}^* \mathcal{T}_{\{j+l\}}(\mathcal{M}_{\cup})(p'_l, \perp, Val'_l + \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(\lambda_{l+1}, j+l+1)\}})$.

□

Next, we show that if some state q is k -bounded reachable by a run of \mathcal{M}_{\cup} of rank $(m+1, n)$, then q is k -bounded reachable by a run of \mathcal{M}_{\cup} of rank $(m, n+1)$.

Lemma E.10. *Let $(m, n) \in \mathbb{N} \times \mathbb{N}$, $c_{\mathcal{M}_{\cup}}^{\text{init}} \xrightarrow{\tau}^* \mathcal{T}_{[0, k]}(\mathcal{M}_{\cup}) c$ be a run of rank $(m+1, n)$ such that $Active(c) = \perp$. Then, there is a run $c'_{\mathcal{M}_{\cup}} \xrightarrow{\tau'}^* \mathcal{T}_{[0, k]}(\mathcal{M}_{\cup}) c'$ of rank $(m, n+1)$ such that $Active(c') = \perp$, and $State(c') = State(c)$.*

Proof. Let us assume that $c_{\mathcal{M}_{\cup}}^{\text{init}} \xrightarrow{\tau}^* \mathcal{T}_{[0, k]}(\mathcal{M}_{\cup}) c$ is a run of rank $(m+1, n)$ with $Active(c) = \perp$. Then, by the definition of DCPSs there are $i, j \in \mathbb{N}$, $\gamma \in \Gamma$, $p_0, p'_0, \dots, p'_i, p_{i+1} \in Q$, $w_1, \dots, w_{i+1} \in (\Gamma')^*$, $\kappa_0, \tau_0, \kappa_1, \tau_1, \dots, \tau_i, \kappa_{i+1} \in \Delta_{\cup}^*$, and $Val_0, Val'_0, \dots, Val'_i, Val_{i+1} \in [Loc(\mathcal{M}_{\cup}) \rightarrow \mathbb{N}]$ such that the following conditions are satisfied:

- $i + j \leq k$.
- $\tau = \kappa_0 \tau_0 \kappa_1 \tau_1 \cdots \tau_i \kappa_{i+1}$.
- $State(c) = p_{i+1}$ and $Idle = Val_i + \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(w_{i+1}, j+l+1)\}}$.
- $c_{\mathcal{M}_{\cup}}^{\text{init}} \xrightarrow{\kappa_0}^* \mathcal{T}_{[0, k]}(\mathcal{M}_{\cup})(p_0, \perp, Val_0 + \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(\gamma, j)\}})$.
- $(p_0, \perp, \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(\gamma, j)\}}) \xrightarrow{\tau_0}^* \mathcal{T}_{\{j\}}(\mathcal{M}_{\cup})(p'_0, \perp, Val'_0 + \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(w_1, j+1)\}})$.
- For every $l \in [1, i+1]$, $(p'_{l-1}, \perp, Val'_{l-1} + Val_{l-1}) \xrightarrow{\kappa_l}^* \mathcal{T}_{[0, k]}(\mathcal{M}_{\cup})(p_l, \perp, Val_l)$.
- For every $l \in [1, i]$, $(p_l, \perp, \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(w_l, j+l)\}}) \xrightarrow{\tau_l}^* \mathcal{T}_{\{j+l\}}(\mathcal{M}_{\cup})(p'_l, \perp, Val'_l + \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(w_{l+1}, j+l+1)\}})$.

Now, we can apply Lemma E.9 to show that there are $\lambda_1, \dots, \lambda_{i+1} \in S_{\text{fs}}^{\text{sw}}$ as well as $\tau'_0, \dots, \tau'_i \in \Delta_{\cup}^*$ such that:

- $(p_0, \perp, \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(\gamma, j)\}}) \xrightarrow{\tau'_0}^* \mathcal{T}_{\{j\}}(\mathcal{M}_{\cup})(p'_0, \perp, Val'_0 + \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(\lambda_1, j+1)\}})$.
- For every $l \in [1, i]$, $(p_l, \perp, \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(\lambda_l, j+l)\}}) \xrightarrow{\tau'_l}^* \mathcal{T}_{\{j+l\}}(\mathcal{M}_{\cup})(p'_l, \perp, Val'_l + \text{Id}_{Loc(\mathcal{M}_{\cup})}^{\{(\lambda_{l+1}, j+l+1)\}})$.

Then, we can use the definition of DCPSs to show that there is a run $c'_{\mathcal{M}_{\cup}} \xrightarrow{\tau'}^* \mathcal{T}_{[0, k]}(\mathcal{M}_{\cup}) c'$ of rank $(m, n+1)$ such that $Active(c') = \perp$ and $State(c') = State(c)$. □

E.7. From a run of rank $(m, n + 1)$ of \mathcal{M}_\cup to a run of rank $(m + 1, n)$ of \mathcal{M}_\cup . In the following, we establish that given a run of \mathcal{M}_\cup such that there is one thread executed following the set of transitions Δ_{fs} , we can compute a run of \mathcal{M}_\cup where the execution of this thread is replaced by an execution of a thread following the set of transitions Δ_{pf} . To this aim, we need first to prove Lemma E.11 which states that for any run of a thread of \mathcal{M}_{fs} , we can construct an equivalent run of a thread of \mathcal{M}_{pf} .

Lemma E.11. *Let $\gamma \in \Gamma$ and $i, j \in \mathbb{N}$ such that $i + j \leq k$. If there are $p_0, p'_0, \dots, p_i, p'_i \in Q$, $\lambda_1, \dots, \lambda_{i+1} \in S_{\text{fs}}^{\text{sw}}$, $\tau'_0, \dots, \tau'_i \in \Delta_\cup^*$, and $Val'_0, \dots, Val'_i \in [\text{Loc}(\mathcal{M}_\cup) \rightarrow \mathbb{N}]$ such that:*

- $(p_0, \perp, \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(\gamma, j)\}}) \xrightarrow{\tau'_0}^* \mathcal{T}_{\{j\}}(\mathcal{M}_\cup)(p'_0, \perp, Val'_0 + \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(\lambda_1, j+1)\}})$.
- For every $l \in [1, i]$, $(p_l, \perp, \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(\lambda_l, j+l)\}}) \xrightarrow{\tau'_l}^* \mathcal{T}_{\{j+l\}}(\mathcal{M}_\cup)(p'_l, \perp, Val'_l + \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(\lambda_{l+1}, j+l+1)\}})$.

Then, there are elements $w_1, \dots, w_{i+1} \in (\Gamma')^*$, $\tau_0, \dots, \tau_i \in \Delta_\cup^*$, and $Val''_0, \dots, Val''_i \in [\text{Loc}(\mathcal{M}_\cup) \rightarrow \mathbb{N}]$ such that:

- $(p_0, \perp, \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(\gamma, j)\}}) \xrightarrow{\tau_0}^* \mathcal{T}_{\{j\}}(\mathcal{M}_\cup)(p'_0, \perp, Val''_0 + \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(w_1, j+1)\}})$.
- For every $l \in [1, i]$, $(p_l, \perp, \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(w_l, j+l)\}}) \xrightarrow{\tau_l}^* \mathcal{T}_{\{j+l\}}(\mathcal{M}_\cup)(p'_l, \perp, Val''_l + \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(w_{l+1}, j+l+1)\}})$.
- For every $l \in [0, i]$, $Val'_l \leq Val''_l$.

Proof. Let us assume that there are $p_0, p'_0, \dots, p_i, p'_i \in Q$, $\lambda_1, \dots, \lambda_{i+1} \in S_{\text{fs}}^{\text{sw}}$, $\tau'_0, \dots, \tau'_i \in \Delta_\cup^*$, and $Val'_0, \dots, Val'_i \in [\text{Loc}(\mathcal{M}_\cup) \rightarrow \mathbb{N}]$ such that:

- $(p_0, \perp, \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(\gamma, j)\}}) \xrightarrow{\tau'_0}^* \mathcal{T}_{\{j\}}(\mathcal{M}_\cup)(p'_0, \perp, Val'_0 + \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(\lambda_1, j+1)\}})$.
- For every $l \in [1, i]$, $(p_l, \perp, \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(\lambda_l, j+l)\}}) \xrightarrow{\tau'_l}^* \mathcal{T}_{\{j+l\}}(\mathcal{M}_\cup)(p'_l, \perp, Val'_l + \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(\lambda_{l+1}, j+l+1)\}})$.

Then, we apply Lemma E.2 to show that there are $\sigma_0, \dots, \sigma_i \in \Gamma^*$, $\gamma_0, \dots, \gamma_i \in \Gamma$, $\alpha \in \Gamma_\epsilon$, $g_0, \dots, g_{i+1} \in Q$, $s_0 \in I_{(g_0, \gamma)}$, and $s \in S_{(g_0, \gamma)}$ such that:

- $\gamma_0 = \gamma$.
- For every $l \in [0, i]$, $p_l \mapsto_{\mathcal{M}_\cup} g_l \triangleleft \gamma_l$.
- $\sigma_0(p'_0, \gamma_1, g_1)\sigma_1 \cdots \sigma_{i-1}(p'_{i-1}, \gamma_i, g_i)\sigma_i(p'_i, \alpha, g_{i+1})$ in $\text{Traces}_{\mathcal{T}(\mathcal{A}_{(g_0, \gamma)})}(\{s_0\}, \{s\})$.
- For every $l \in [0, i]$, $Val'_l((\gamma', j + l + 1)) = |\sigma_l|_{\gamma'}$ for all $\gamma' \in \Gamma$.
- For every $l \in [0, i]$, $Val'_l((w, l)) = 0$ for all $w \in \Gamma_\cup^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j + l + 1\}$.

On the other hand, we can use the Lemma E.1 to show that

$$\sigma_0(p'_0, \gamma_1, g_1)\sigma_1 \cdots \sigma_{i-1}(p'_{i-1}, \gamma_i, g_i)\sigma_i(p'_i, \epsilon, g_{i+1}) \in L(\mathcal{A}_{(g_0, \gamma)}).$$

Now, we can use the definition of $L(\mathcal{A}_{(g_0, \gamma)})$ to show that there are $\sigma'_0, \dots, \sigma'_i \in \Gamma^*$ such that $\sigma'_0(p'_0, \gamma_1, g_1)\sigma'_1 \cdots \sigma'_{i-1}(p'_{i-1}, \gamma_i, g_i)\sigma'_i(p'_i, \epsilon, g_{i+1}) \in L'_{((g_0, \gamma), i+1)}$ and $\sigma_l \preceq \sigma'_l$ for all $l \in [0, i]$.

Then, we can apply E.5 and Lemma E.6 to prove that there are $w_1, \dots, w_{i+1} \in (\Gamma')^*$, $\tau_0, \dots, \tau_i \in \Delta_\cup^*$, and $Val''_0, \dots, Val''_i \in [\text{Loc}(\mathcal{M}_\cup) \rightarrow \mathbb{N}]$ such that:

- $(p_0, \perp, \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(\gamma, j)\}}) \xrightarrow{\tau_0}^* \mathcal{T}_{\{j\}}(\mathcal{M}_\cup)(p'_0, \perp, Val''_0 + \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(w_1, j+1)\}})$.
- For every $l \in [1, i]$, $(p_l, \perp, \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(w_l, j+l)\}}) \xrightarrow{\tau_l}^* \mathcal{T}_{\{j+l\}}(\mathcal{M}_\cup)(p'_l, \perp, Val''_l + \text{Id}_{\text{Loc}(\mathcal{M}_\cup)}^{\{(w_{l+1}, j+l+1)\}})$.
- For every $l \in [0, i]$, $Val'_l \leq Val''_l$.
- For every $l \in [0, i]$, $Val''_l((\gamma', j + l + 1)) = |\sigma'_l|_{\gamma'}$ for all $\gamma' \in \Gamma$.
- For every $l \in [0, i]$, $Val''_l((w, l)) = 0$ for all $w \in \Gamma_\cup^*$ and $l \in \mathbb{N}$ such that $(w, l) \notin \Gamma \times \{j + l + 1\}$. \square

Next, we show that if some state q is k -bounded reachable by a run of \mathcal{M}_\cup of rank $(m, n+1)$, then q is k -bounded reachable by a run of \mathcal{M}_\cup of rank $(m+1, n)$.

Lemma E.12. *Let $(m, n) \in \mathbb{N} \times \mathbb{N}$, $c_{\mathcal{M}_\cup}^{\text{init}} \xrightarrow{\tau}^* \mathcal{T}_{[0,k]}(\mathcal{M}_\cup) c$ be a run of rank $(m, n+1)$ such that $\text{Active}(c) = \perp$. Then, there is a run $c'_{\mathcal{M}_\cup} \xrightarrow{\tau'}^* \mathcal{T}_{[0,k]}(\mathcal{M}_\cup) c'$ of rank $(m+1, n)$ such that $\text{Active}(c') = \perp$, and $\text{State}(c') = \text{State}(c)$.*

Proof. Let us assume that $c_{\mathcal{M}_\cup}^{\text{init}} \xrightarrow{\tau}^* \mathcal{T}_{[0,k]}(\mathcal{M}_\cup) c$ is a run of rank $(m, n+1)$ such that $\text{Active}(c) = \perp$. Then, we can use the definition of DCPSs and Lemma E.11 to show that there is a run $c'_{\mathcal{M}_\cup} \xrightarrow{\tau'}^* \mathcal{T}_{[0,k]}(\mathcal{M}_\cup) c'$ of rank $(m+1, n)$ such that $\text{Active}(c') = \perp$, and $\text{State}(c') = \text{State}(c)$. \square