Parameterized Model Checking of Ring-based Message Passing Systems

E. Allen Emerson and Vineet Kahlon

Department of Computer Sciences,
The University of Texas at Austin,
Austin, TX 78712, USA.

Abstract. The *Parameterized Model Checking Problem (PMCP)* is to decide whether a temporal property holds for a uniform family of systems, U^n, comprised of finite, but arbitrarily many, copies of a *template* process U. Unfortunately, it is undecidable in general [3]. In this paper, we consider the PMCP for systems comprised of processes arranged in a ring that communicate by passing messages via tokens whose values can be updated at most a bounded number of times. Correctness properties are expressed using the stuttering-insensitive linear time logic LTL$\backslash$X. For bidirectional rings we show how to reduce reasoning about rings with an arbitrary number of processes to rings with up to a certain finite *cutoff* number of processes. This immediately yields decidability of the PMCP at hand. We go on to show that for unidirectional rings small cutoffs can be achieved, making the decision procedure provably efficient. As example applications, we consider protocols for the leader election problem.

1 Introduction

The *Parameterized Model Checking Problem (PMCP)* is to decide whether a temporal property holds for a uniform family of systems U^n comprised of finite, but arbitrarily many, copies of a *template* process U. Unfortunately, PMCP is undecidable because a system of size n can simulate a Turing machine for n steps [3]. The Halting problem for Turing Machines can then be easily formulated as a PMCP for reachability of the halting state, viz., $\mathsf{EF}halt$. This argument can be refined even in the case where the parameterized system is a unidirectional ring [17]. It follows from a result by Shannon [16] that the undecidability result holds even when the head (token circulating in the ring) can have only two possible states [16]. An essential part of the undecidability proof of the latter is that the message token changes value an arbitrary number of times.

We show in this paper that if there is a bound b on the number of times the token changes value during a run of the system, then the PMCP is decidable. This boundedness assumption can be justified by the fact that protocols for a number of ring based applications have the property that the value of each message bearing token can be changed only a bounded number of times in any run of the protocol. For instance, in standard protocols for the leader election problem [15], every token makes at most one value change during any run of each of the protocols.

We express correctness properties using the stuttering-insensitive linear temporal logic LTL$\backslash$X. The basic assertions are of the form $\mathsf{A}h$, or $\mathsf{E}h$, where formula h is built using F "sometimes", G "always", U "until" but without X "next-time"; and A "for all futures" and E "for some future" are the usual path quantifiers. Use of stuttering-insensitive logics is natural when model checking parameterized systems as the next-time operator X gives us the ability to count, often leading to undecidability of the PMCP [10].

In the case of unidirectional (or certain restricted bidirectional) rings, we argue that arbitrarily "large" systems of size n can be imitated up to stuttering by a small system of a certain cutoff size c, where $c = O(b)$. Thus to solve PMCP, checking correctness over all sizes n, it is necessary and sufficient to check all sizes m up to c. In the context of rings, this style of "cutoff" argument has been used in [11], where it was shown how to reduce reasoning about properties expressed using the branching time temporal logic CTL$^*\backslash$X from a system with an arbitrary number of processes to systems with up to a small cutoff number of processes. However, the results were established only for unidirectional rings where the token could not carry values, viz., processes could not exchange messages among themselves, resulting in a framework with limited modeling power. For example, it is not clear how standard protocols for the Leader Election problem (see, for example, [15]) that require tokens to change values, viz., messages to be exchanged, can be encoded in this framework. Our unidirectional ring framework has a broader modeling power but with an efficiently decidable PMCP.

The case of bidirectional rings is more involved. Here we find it convenient to exploit the viewpoint that a ring of many (n) similar processes is tantamount to a Turing machine on a circular tape (CTM for short) with n tape cells. To see this, we note that a token in a ring can be viewed as the head of the CTM, with the value of the token representing the control state of the head. Cell i of the circular tape corresponds to P_i, the ith process in the ring, with the tape symbol in cell i representing the local state of P_i. This, in effect, reduces the PMCP for bidirectional rings in which the token makes only a bounded number of value changes to the study of the PMCP for CTMs in which the head only makes a bounded number of state changes. To analyze the behavior of CTMs we in turn study (Linear Tape) Turing Machines with bounded state changes to the head. For an arbitrary Turing machine, the associated PMCP again amounts to the halting problem and is undecidable. However, we demonstrate that for a Turing machine that can make at most a bounded number b of state changes, the halting problem is decidable, and, hence for the associated ring system where token values change at most b times, the PMCP is decidable. The latter result is established by induction on b. The base case $b = 1$ represents a Turing machine with a single (non-halting) state.

The rest of the paper is organized as follows. The unidirectional (or restricted bidirectional) ring model is introduced and the related cutoff results shown in section 2 while the cutoff results for bidirectional rings are given in section 3. Applications are handled in section 4 and we conclude with some remarks in section 5.

2 Unidirectional Rings

Communication in computer networks is usually carried out via message passing using packets or value-bearing tokens in which the sender puts the data and the address of the intended receiver. However, apart from data transfer, tokens also play a crucial role in the *implementation* of network protocols. In a typical network protocol, a process sends out a token *owned* by it to gather information about other processes in the network. In leader election [15], for example, a process sends out a token bearing its identifier to find out whether there exists another process with an identifier of greater value. In this role tokens play a passive role in that they do not cause any state change in processes other than the ones owning it but are used merely for information gathering. A key reason for this might be that most protocols are data independent. In this section, we propose a simple framework to model such protocols and show how to reduce reasoning about linear time properties for such a system with an arbitrary number of processes to one with a few.

The Process Framework. We consider systems comprised of processes arranged in the form of a ring communicating using multiple message-bearing tokens, each of whose value can be modified at most a bounded number of times (see remark 2.2), say b. All tokens move in the same direction, say clockwise. In a ring $\mathcal{R}$ comprised of the n processes $P_0, ..., P_{n-1}$ listed in clockwise order of occurrence around the ring, the ith process, P_i, is given by a tuple of the form $(Q_i, \Sigma_i, T_i, R_i, \mathsf{i}_i)$, where Q_i is the finite set of states of P_i, Σ_i the set of labels of P_i, T_i the set of tokens owned by P_i, R_i its transition relation and i_i the initial state. Let $\mathsf{T} = \cup_i T_i$. Each token in T can take on values from the set V. In any global state of the ring, a token is in the *possession* of exactly one process. A process may, however, possess multiple tokens.

Transitions of P_i can be classified as either *internal* or *token dependent*. An internal transition of P_i is of the general form $a \xrightarrow{l} b$, and can always be fired irrespective of the current global state of the system. A token dependent transition of process P_i, on the other hand, is of the general form $tr : a \xrightarrow{l:g \to A} b$, where $g : V \to \{true, false\}$ is a boolean valued function and action A is either the expression *skip* or of the form $\mathsf{t} := v$. Token dependent transition tr can be fired only if P possesses a token t from the subset $T_{tr} \subseteq T$ with a value that enables guard g. We then say that transition tr *involves* token t. After tr is fired, process P_i transits to local state b and t is passed on to the clockwise neighbor. If action A is the expression *skip*, then the token is simply passed on with its value unchanged, else if A is the expression $\mathsf{t} := v$, then the token is passed with its value updated to v.

For each token dependent transition tr of process P_i, the set T_{tr} is either T_i or $\mathsf{T} \setminus T_i$. If $T_{tr} = T_i$, viz., tr involves tokens owned by it, then tr is termed an *endogenous* token dependent transition, else if $T_{tr} = \mathsf{T} \setminus T_i$, viz., tr involves tokens not owned by it, then tr is termed an *exogenous* token dependent transition. Exogenous transitions of process P_i can be thought of as constituting the communication layer of P_i responsible for handling tokens owned by other processes but causing no change in the local state of P_i. On executing an exogenous transition involving token t, it is passed on to the clockwise neighbor with a possible change in the value of t but without changing the local state of P_i. Thus every exogenous transition is of the general form $a \xrightarrow{l:g \to A} a$. We assume that the action A of exogenous transitions is oblivious of the current local state of P_i and depends only on the value of the token. Thus if $a \xrightarrow{l:g \to A} a$ is a exogenous transition, then for each $b \in Q_i$, there exists an exogenous transition in R_i of the form $b \xrightarrow{l:g \to A} b$. To prevent a process from indefinitely taking possession of a token not owned by it, we assume that from any local state of P_i, for any possible value of $t \notin T_i$, there always exists an exogenous transition of P_i that is enabled. We use $\mathcal{R} = (S^n, \Sigma, \mathsf{T}, R^n, \mathsf{i}^n)$, to denote the ring comprised of the n processes $P_0, ..., P_{n-1}$ executing asynchronously with interleaving semantics and is defined in the usual way.

Reduction Result. We show a one way reduction for properties of the form $\mathsf{A}h(i,j)$, where $h(i,j)$ is a LTL$\backslash$X formula with atomic propositions over the local states of processes P_i and P_j, from a ring $\mathcal{R}$ of arbitrary size comprised of possibly distinct non-isomorphic processes to a ring of size at most $\mathsf{b}(|T_i| + |T_j|)$, where b is the bound on the number of times the value of each token of $\mathcal{R}$ can be modified. We assume that each process P_i of $\mathcal{R}$ is *deterministic*, viz., for every local state a of P_i, the following conditions hold

(i) for every possible value of a token not owned by P_i, there is a unique exogenous transition of P_i from a that is enabled, and

(ii) there is either an internal transition or for every possible value of token $t \in T_i$, a unique endogenous transition that is enabled from a, but not both.

Using the fact that exogenous transitions are state oblivious and the deterministic nature of processes it can be shown that $P(t)$ is independent of the global computation $\mathcal{R}$ executes. Thus we have the following.

Lemma 2.0 *$P(t)$ is well-defined.*

For set T of tokens, we let $P(T)$ denote $\bigcup_{t \in T} P(t)$. Let P_i and P_j be processes belonging to ring $\mathcal{R}$. We let $\mathcal{R}(i,j)$ denote the ring comprised of the processes $\{P_i, P_j\} \cup P(T_{P_i}) \cup P(T_{P_j})$ occurring in the same relative clockwise order as along $\mathcal{R}$.

Proposition 2.1 (Reduction Result) *Let $\mathcal{R}$ be a ring with processes P_i and P_j. Then $\mathcal{R} \models \mathsf{E}h(i,j)$ implies that $\mathcal{R}(i,j) \models \mathsf{E}h(i,j)$, where $h(i,j)$ is a LTL$\backslash$X formula over the local states of P_i and P_j.*

Proof Idea Given a computation x of $\mathcal{R}$, we construct a computation y of $R(i,j)$ such that $x[i,j]$, viz., x projected onto processes P_i and P_j, is a stuttering of $y[i,j]$. $\square$

Remark 2.2 (Boundedness) In general, the number of value changes for tokens of a given ring might not be bounded and hence the above result may not yield any reduction. However, for special cases we can deduce from merely a static analysis of the syntax of the processes that each token undergoes only a bounded number of value changes. One such useful case results by treating each token t as essentially a counter with an integer value which decreases each time t is updated. This gives rise to a ring model where we have integer-valued tokens such that for each local state a of a process the token dependent transitions from a involving t are of the form $a \xrightarrow{l:t>c \to t:=d} b$, where $c > d$, and $a \xrightarrow{l':t \leq c \to skip} f$. Thus token t can be thought of as a counter that is set initially and each time a token dependent transition modifies the value of t there is a decrease in its value. Once t is modified by a transition of the form $tr : a \xrightarrow{l:t>c \to t:=d} b$ of process P_k, then the next token dependent transition to modify the value of t is the next transition of the form $tr' : a' \xrightarrow{l':t>c' \to t_k:=d'} b$, where $c' < d$, that is encountered as we traverse the ring in a clockwise direction from process P_k. This transition is either an exogenous transition of a process other than P_i (which is the same for each local state of the process) or an endogenous transition from the current local state of P_i. We call such a pair of transitions 'adjacent'. Thus the maximum number of times the value of t can be updated is the maximal length of a sequence of adjacent transitions. For the LCR protocol (section 4), the maximum length of such a sequence for each token is 1.

Extensions. The results also hold for the following two extensions of our model.

(a) **Adding FIFO queues.** Queues may be necessary to ensure that tokens sent to a process are handled in the order received. This guarantees weak and strong fairness requirements are met for the verification of liveness properties.

(b) **Restricted Bidirectional Tokens.** The model can also be generalized by allowing restricted bidirectional tokens where instead of always moving in a fixed clockwise direction we can allow a token to be able to change direction when it is assigned a new value. For a fixed value, however, the token always moves in the same direction.

3 Bidirectional Rings

We present a generalization of the unidirectional ring model proposed in [11] by allowing (a) bidirectional rings, and (b) the token to carry values. We consider systems comprised of finite, but arbitrarily many, copies of a single process template P arranged in the form of a ring executing concurrently, viz., with interleaving semantics. We only consider the case where processes communicate using a *solitary* token t that is allowed to carry values. Template process P has two types of transitions: (1) *token dependent* that require P to possess t in order to fire, and (2) *internal* that can be fired irrespective of whether P possesses t or not. In addition, P uses transitions labeled with the *receive* action to take possession of t from its counterclockwise neighbor and transitions labeled with *send* actions to relinquish possession of t to its clockwise neighbor. In any computation, the system is allowed to change the value of the token at most a bounded number of times, say b. Allowing an unbounded number of value changes to the token could, in general, make a family of such systems Turing-powerful [17] and hence the corresponding PMCP undecidable.

Formally, process P is defined to be a labeled transition system given by the tuple $(S \times (V \cup \{\bot\}), \Sigma, R, (\mathsf{i}, \bot))$, where

- V is the finite set of values that t can take, with $\bot \notin V$.
- $S \times (V \cup \{\bot\})$ is the set of states of P with pair $(a, v) \in S \times (V \cup \{\bot\})$ indicating that P is in local state a; and v is the value of t in case P possesses t, else $v = \bot$.
- $(\mathsf{i}, \bot)$ is the initial state of P.
- Σ, the set of actions, is the disjoint union of the set of "internal" actions Σ_i, the set of "token dependent" actions Σ_{td} and the set of token transfer actions $\bigcup_{v \in V} \{snd_v, rcv_v\}$.
- R, the transition relation, is the set of all transitions $(a, v) \xrightarrow{l} (b, v', D)$, with $D \in \mathsf{Dir} = \{\mathsf{counterclockwise}, \mathsf{clockwise}, \mathsf{undefined}\}$, where
  - $l \in \Sigma_i$ implies that $v = v'$ and $D = \mathsf{undefined}$.

- $l \in \Sigma_{td}$ implies that $v = v'$, $v \in V$ and $D = \mathsf{undefined}$.
- $l = rcv_u$ implies that $v = \perp$, $v' = u$ and $D = \mathsf{undefined}$.
- $l = snd_u$ implies that $v \in V$, $v' = \perp$ and $D \in \{\mathsf{clockwise}, \mathsf{counterclockwise}\}$.
- if $a = \mathsf{i}$ and $v = \perp$, then $l = rcv_u$, for some $u \in V$, viz., the only possible initial action is a receive. We also assume that along any path of P send and receive actions alternate.

In this paper, for simplicity we consider only bidirectional rings where the processes are deterministic. A bidirectional ring system comprised of n copies of a process template P is denoted by P^n and is represented as $(P_0, ..., P_{n-1})$ to emphasize the fact that process P_{i+1}[1] has P_i as its counterclockwise and P_{i+2} its clockwise neighbor. Analogously, $(s_0, ..., s_{n-1})$, where for each i, $s_i \in S_P$, represents a global 'cyclic' state of P^n, with process P_i in local state s_i. We assume that the send and receive actions of two neighboring processes synchronize when transferring a token.

The (Single Index) PMCP for Bidirectional Rings. To decide whether for all n, $P^{n+l}, (xa^n) \models h(\mathsf{m})$, where (xa^n), the initial configuration of P^{n+l}, is such that x is a fixed sequence of local states of P of length $l \geq \mathsf{m}$ and $h(\mathsf{m})$ is a LTL\X formula over the local states of process P_{m}. We assume that initially the token is in the possession of process P_0.

Linear Tape and Circular Tape Turing Machines. A (linear tape) Turing Machine M is defined to be a tuple of the form $M = (Q, \Sigma \cup \{\sqcup, \Gamma\}, \delta, q_0)$ where,

- Q is the set of states of M
- $q_0 \in Q$ is the initial state of M
- $\Sigma \cup \{\sqcup, \Gamma\}$ is the set of tape symbols with '$\sqcup$' being the blank symbol and 'Γ' the left end tape marker such that $\Sigma \cap \{\sqcup, \Gamma\} = \emptyset$.
- $\delta \subseteq Q \times \Sigma \cup \{\sqcup, \Gamma\} \times Q \times \Sigma \times \{L, R\}$ is the transition relation. Since Γ is the left-end tape marker, we assume that if $(p, \Gamma, q, b, D) \in \delta$, then $b = \Gamma$ and $D = R$, i.e., cell 0, containing Γ, always reflects back the head to the right.

In this paper, for Turing Machines with linear tapes, the cell containing Γ will be referred to as cell 0 while the ith cell to its right is referred to as cell i.

Analogously we define a Circular Tape Turing machine (CTM), $M = (Q, \Sigma, \delta, q_0)$ on the tape cells $0, ..., m$ where, transition relation $\delta \subseteq Q \times \Sigma \times Q \times \Sigma \times \{L, R\}$, has the property that on a right move from cell m the head ends up at cell 0 and on a left move from cell 0 the head ends up at cell m. Note that in this case because of the circular topology of the system, the left and right directions are not well defined but we interpret them as the clockwise and counterclockwise directions, respectively.

Modeling Bidirectional Token Rings as Circular Tape Turing Machines. Consider the sequence of transitions of $a_0 \xrightarrow{snd_u} a_1 \xrightarrow{l_1} ... \xrightarrow{l_{k-1}} a_k \xrightarrow{rcv_v} a_{k+1}$ of process P where for each $i \in [1 : k-1]$, l_i is an internal or a token dependent transition. Note that since we consider only deterministic systems, it is clear that after firing the send transition $a_0 \xrightarrow{snd_u} a_1$, a process has to execute all the actions $l_1, ..., l_{k-1}, rcv_v$ in the order listed to

[1] Here '+' denotes addition modulo n.

receive the token again. Thus we can, in effect, replace the firing of the above sequence of transitions with the firing of just one receive transition $a_1 \overset{l_1...l_{k-1}rcv_v}{\longrightarrow} a_{k+1}$. A similar observation holds for all internal and token dependent transitions sandwiched between a receive and a send transition in which case we can replace all these transitions with a single send transition. Thus, it suffices to consider processes P where each transition of P is either a send or a receive transition with send and receive transitions alternating along any path in the transition diagram of P.

Using this assumption, we can now readily see that the ring $P^n = (P_0...P_{n-1})$ with token t comprised of n copies of process template $P = (S \times (V \cup \{\perp\}), \Sigma, R, (\mathsf{i}, \perp))$ can be looked upon as the CTM, $C^n = (V, S, \delta, \mathsf{i})$ with one head and tape cells $0, ..., n-1$. Here cell i corresponds to process P_i with the local state of P_i being looked upon as the tape symbol in cell i. The token t can be thought of as the head of the CTM with the value of t being the state of the head. Transition $(p, a) \rightarrow (q, c, D) \in \delta$ iff for some $b \in S$, both the transitions $(a, \perp) \overset{rcv_p}{\rightarrow} (b, p)$ and $(b, p) \overset{snd_q}{\rightarrow} (c, \perp, D)$ are in R.

Thus the PMCP defined before can now be reformulated as follows: To decide whether for all n, $C^{n+l}, (xa^n) \models h(\mathsf{m})$, where x is a sequence of tape symbols of S of length l and $h(\mathsf{m})$ is a LTL\X formula over the tape alphabet of cell m, with $\mathsf{m} \leq l$. We assume that for each n, in the initial cyclic configurations (xa^n), the head is placed at cell 0.

3.1 Linear Tape Turing Machines

We begin by showing that the behavior of a given deterministic one state Turing Machine M can be deduced from an analysis of the structure of the transition diagram of the control state of M.

Let $M = (Q, \Sigma \cup \{\Gamma, \sqcup\}, \delta, q_0)$ be a given deterministic Linear Tape Turing Machine. We assume that M has just one control state, say q, and that the head of M is initially placed at cell 0 with the rest of the tape cells each containing the empty symbol '$\sqcup$'.

We define the *transition graph* of M as the directed graph $G = (V, E)$, where $V = \Sigma \cup \{\sqcup\}$ and $E = \{(a, b) | a, b \in \Sigma \cup \{\sqcup\}, \delta(q, a) = (q, b, D), \textit{with } D \in \{L, R\}\}$. Since we are considering a Turing machine with a solitary control state, in any configuration, the direction in which the head of M moves depends only on the symbol it is currently reading. Thus each tape symbol in $\Sigma \cup \{\Gamma, \sqcup\}$ can be characterized as either a *left-symbol* or a *right-symbol* depending on whether the head moves left or right upon reading it. Given symbol $a \in \Sigma \cup \{\sqcup\}$, let G_a denote the subgraph of G induced by the set of symbols reachable from a in G. We say that symbol $a \in \Sigma$ is *writable* iff M starting at cell 0 on the empty input, with each non-zero cell containing $\sqcup$, writes a in some tape cell in finitely many moves. Symbol $a \in \Sigma \cup \{\sqcup\}$ is *readable* iff M, starting on the empty input, reaches a configuration in finitely many steps in which the head is positioned at a cell containing a.

Since M is a deterministic Turing Machine, each node of G has out-degree at most one. To start with, each non-zero tape cell contains $\sqcup$ and so for a symbol to be writable it has to be reachable from $\sqcup$ in G. We may therefore assume, without loss of generality, that all symbols are reachable from $\sqcup$ in G. Thus G is either a simple path starting at $\sqcup$

or a 'lollipop' of the form $a_0 \rightarrow ... \rightarrow a_\mathsf{k} \rightarrow ... \rightarrow a_\mathsf{d} \rightarrow a_\mathsf{k}$, where $a_0 = \sqcup$. We begin by considering the case where G is the simple path $a_0 \rightarrow ... \rightarrow a_\mathsf{k}$ starting at $\sqcup$. Later we show how to reduce the analysis for the case where G is a lollipop to this case.

Definitions and Notation. Let $a, b \in \Sigma \cup \{\sqcup\}$ be such that there is a path from a to b in G. We define the *depth* of b with respect to a, denoted by $d(b, a)$, to be the number of states, not including b, in the unique path from a to b. Analogously, the *left-depth* (*right-depth*) of symbol b with respect to a, denoted by $d_L(b, a)$ ($d_R(b, a)$), are defined to be the number of left (right) symbols, not including b, along the path from a to b in G. We abbreviate $d(a, \sqcup)$ as $d(a)$ and refer to it simply as the *depth* of a. Similarly, $d_L(a, \sqcup)$ ($d_R(a, \sqcup)$) is abbreviated by $d_L(a)$ ($d_R(a)$) and called the *left-depth* (*right-depth*) of a. We write $a < b$ to mean $d(a) < d(b)$.

The content of the ith tape cell after the nth move of M is denoted by $t(i, n)$. For $j \geq 1$, we call the portion of the tape comprised of cells numbered greater than or equal to j, the *interval* starting with j and denote it as $I(j)$. For each interval $I(j)$, we define the *traversal number* of $I(j)$ after move n of M, denoted by $trav(j, n)$, as the ordered pair (k, l), where k is the number of times the head moved from cell $j - 1$ to j, viz., *entered* interval $I(j)$, among the first n moves of M, and l is the number of moves made by the head from a cell inside the interval, viz., the cells $j, j + 1, ...$, after it entered the interval for the kth (last) time.

Key Results. The analysis of the behavior of a single state deterministic Turing machine rests on the following two facts:

1. If in the initial configuration of M, the head is placed at cell 0 and each cell of the tape contains the empty symbol $\sqcup$, then after finitely many steps of M the contents of the tape form a non-increasing (depth wise) sequence of tape symbols.
2. Symbol $a \in \Sigma$ is readable iff for each $b \leq a$, $d_R(b) \leq d_L(b)$.

Proposition 3.1 (Monotonicity Result) *For $i \geq 1$, we have $t(i, n) \geq t(i + 1, n)$. Furthermore, if after n moves the head is positioned at cell $h < i$ and $t(i, n) \neq \sqcup$, then $t(i, n) > t(i + 1, n)$.*

An immediate consequence is the following.

Corollary 3.2 *For $i < j$, we have $t(i, n) \geq t(j, n)$.*

Using the above results, we next show that a necessary and sufficient condition for a tape symbol a to be readable is that for all $b \leq a$, we have $d_R(b) \leq d_L(b)$.

Proposition 3.3 *If $a \in \Sigma$ is readable then for all $b \leq a$, $d_R(b) \leq d_L(b)$.*

Proposition 3.4 *Let $a_i \in \Sigma$. If for all $j \leq i$, $d_R(a_j) \leq d_L(a_j)$, then a_i is readable.*

Predicting the Behavior of Linear Tape Turing Machines. Let a_j be written in cell k in move m_{j_k} and in cell $k + 1$ in move $m_{j_{k+1}}$. Consider the configuration of the tape between moves m_{j_k} and $m_{j_{k+1}}$. All the cells from 0 to k have a_j written in them. Since cell $k + 1$ gets written by a_j in finitely many steps, only finitely many, say $k + l$, cells of the tape are visited in $m_{j_{k+1}}$ moves. Then $t(j, n) = \sqcup$ for all $j \geq k + l + 1$. Consider now the configuration of the tape after execution of step $m_{j_{k+1}} - 1$. Since in the very next step a_j is written in cell $k + 1$, the head is currently at cell $k + 1$. Then using proposition 3.1, we have that $a_j = t(k, m_{j_k}) > t(k+1, m_{j_k}) \geq t(k+2, m_{j_k}) > t(k+3, m_{j_k}) > ... > t(k+l, m_{j_k})$. Therefore it follows that $l \leq \mathsf{k} + 1 = |G|$. Thus we see that configuration of the tape forms a non-increasing sequence of length $k + l$ with the remaining cells containing the blank symbol. We consider two cases.

(a) Simple Paths. First assume that G is the simple path $a_0 \to ... \to a_\mathsf{k}$. There are two sub-cases to consider:

Assume first that a_k is readable. By definition of readability, there is a reachable configuration **c** of M wherein the head after, say n moves, is at tape cell $i \geq 1$ containing a_k. Since the tape configuration forms a non-increasing sequence, it follows that if a_k is readable, then it will be read first in cell 1. Clearly, after reading a_k, the head cannot make any more moves and so M deadlocks in cell 1. Thus by the above comment, in this case only $\mathsf{k}+1$ tape cells were visited during the computation before M deadlocks in cell 1 and the visited tape cells contain a non-increasing sequence of non-empty tape symbols of length at most $\mathsf{k} + 1 = |G|$.

Next assume that a_k is not readable. In this case M cannot deadlock, for otherwise symbol a_k would be readable. Let a_j be the symbol of least depth, j, such that $d_R(a_j) > d_L(a_j)$. Clearly, a_{j-1} is a right symbol and $d_R(a_{j-1}) = d_L(a_{j-1})$ and so $d_R(a_j) = d_L(a_j) + 1$. Then using propositions 3.3 and 3.4, we have that all symbols less than or equal to a_{j-1} are readable but a_j is not. Thus a_j is writable but a_{j+1} is not. Since by corollary 3.2, for all $n, i \geq 1$, we have that $t(i, n) \geq t(i+1, n)$, we have, using the same argument as in the previous case, that the first cell into which a_j is written is cell 1. Since a_j is a right symbol, after writing a_j the head move to the right to cell 2 and then never visits cell 1 again, for otherwise a_{j+1} would be writable. Thus from the above comments we have that when a_{j+1} is written into cell k all cells $0, ..., k$ contain a_{j+1}, all cells $k+l+1, ...$ contains the blank symbols and cell $k+1, ..., k+l$ form a non-increasing sequence with $l \leq j+1$. Thus we can liken the computation to a *wave front* that moves along the tape from left to right such that to the right of the front all cells have the symbol $\sqcup$ while to the left all cell have the symbol a_j. Thus, in this case the computation is unbounded, viz., every cell of the tape is visited at least once. We say that the computation *diverges*.

(b) Lollipops. We now consider the case when G is a lollipop, say $L = a_0 \to ... \to a_\mathsf{k} \to ... \to a_\mathsf{d} \to a_\mathsf{k}$. Note that in this case the machine never deadlocks because no matter what symbol the head is currently reading, there is always a move it can make. Let L_ω be the ‘unrolling’ $\{a'_i\}_{i=0}^{\infty} = a_0 ... a_\mathsf{k}(a_{\mathsf{k}+1} ... a_\mathsf{d} a_\mathsf{k})^\omega$ of L. From the discussion in the previous section, it follows that we all we need to do is decide whether there exists an i such that $d_R(a'_i) > d_L(a'_i)$ and, if yes, find the least such i. Let C_L and C_R denote the number of left and right symbols, respectively, in the cycle $a_\mathsf{k} ... a_\mathsf{d}$ and $l_C = \mathsf{d} - \mathsf{k} + 1$ denote the length of the cycle. Then we can show that if for some i, $d_R(a'_i) > d_L(a'_i)$ then there exists such an $i \in [0 : \mathsf{k}(\mathsf{d} - \mathsf{k} + 2)]$ and hence such an i can be determined efficiently in time O($|G|^2 log(|G|)$).

Using the result for the case when G is a simple path we see that if there exists an i such that $d_R(a'_i) > d_L(a'_i)$, then a front develops writing a'_j to its left, where j is the least i with the above mentioned property. If no such i exists then no front develops and thus cell 1 is visited infinitely often during the computation. In this case if the cycle of the lollipop contains a right symbol then the computation of M on the empty string is unbounded. On the other hand, if all symbols in the cycle are left symbols then since a_k is readable and all symbols appearing after a_k in the lollipop are left symbols, so after reading a_k the head shuttles between cells 0 and 1 without visiting any other cell thereafter with tape symbols being written repeatedly in following cyclic fashion $a_{k+1} \rightarrow ... \rightarrow a_\mathsf{d} \rightarrow a_k$ in cell 1.

The above discussion can be summed up as follows.

Proposition 3.5 (Behavior Lemma) *Let M be a given linear tape Turing machine with only one control state. Then one of the following holds.*

- *the head of M eventually deadlocks in cell 1*
- *the head of M diverges*
- *the head of M eventually shuttles between cells 0 and 1 indefinitely.*

Furthermore, if G is the transition graph of M, then the behavior of M can be decided in time $O(|G|^2 log|G|)$.

3.2 The PMCP for Bidirectional Rings

We now show how the results for linear tape Turing machines with a solitary state can be leveraged to give decision procedures for the PMCP for bidirectional rings. The connection between Turing machines and rings is established via the *Ring Traversal Lemma* using the notion of *crossing numbers* discussed below. The PMCP for rings can equivalently be formulated as follows: given a LTL\X formula $h(\mathsf{m})$ with atomic propositions over the local states of process P_m, where $\mathsf{m} \in [0 : l-1]$, does there exist n such that $C^{l+n}, (xa^n) \models \mathsf{E}h(\mathsf{m})$? We assume that in each of the initial cyclic configurations (xa^n), the head is placed at cell 0.

Notation. We refer to the counterclockwise and clockwise directions along the circular tape of C^{n+l} as *right* and *left* directions, respectively. We assume that tape cells $0, ..., n+l-1$ of C^{n+l} are arranged in a counterclockwise direction in the order listed. For any *interval*, viz., a finite set of adjacent cells along the circular tape, when traversing the cells of the interval in the counterclockwise direction, the cell encountered first is called the *left end* of the interval whereas the cell encountered last is called the *right end* of the interval. For C^{n+l}, cells $0, ..., l-1$ containing the input sequence x is designated as interval X while the set of remaining cells, each containing the tape symbol a, is designated the *outer ring*. As for Turing machines with linear tapes, we let G denote the transition graph of C^{n+l} and G_a the subgraph of G induced by the set of all symbols reachable from a in G.

Strategy. We begin by outlining our strategy. For a ring of size $n+l$, starting at the initial cyclic configuration (xa^n), we construct a transition diagram $G_X(n)$ on the configurations of interval X, where each configuration is given by the contents of the tape cells constituting X along with the cell number of X on which the head is currently placed. If from a configuration **c** of X, the head moves outside interval X, then if the head does not re-enter X, then **c** has no successor in $G_X(n)$, else the successor is the configuration that results when the head re-enters X. In the second case, the transition that results is called an *external* transition of $G_X(n)$. All transitions of $G_X(n)$ that are not external are called *internal* and correspond to movements of the head within interval X. Since M is a deterministic Turing machine, $G_X(n)$ is either a simple path or a lollipop. Note that since we are interested in the 'behaviour' of cell m belonging to interval X, there exists n such that $M, (xa^n) \models \mathsf{E}h(\mathsf{m})$ iff there exists n such that $G_X(n) \models \mathsf{E}h(\mathsf{m})$, where in both cases the formula $h(\mathsf{m})$ is interpreted over the tape alphabets in cell m. We show the existence of a cutoff $c \geq l$ such that for all $j \geq c$, the transition diagram $G_X(j)$ is the same as $G_X(c)$. This reduces the PMCP to determining whether there exists $i \in [l : c]$ such that $M, (xa^i) \models \mathsf{E}h(\mathsf{m})$, i.e., model checking at most c finite state systems, which is clearly decidable. We point out that we do not actually construct $G_X(n)$ but merely use it to prove our cutoff result. Towards that end, however, we need to elucidate the structure of $G_X(n)$. Note that the internal transitions of $G_X(n)$ are easy to figure out as they correspond to movements of the head within interval X. But for the external transition, the key question that needs to be answered is that in case the head leaves interval X whether it re-enters X again and, if yes, then the direction from which it re-enters X and the configurations of both interval X and the outer ring on re-entry in relation to the configuration of the outer ring on the last exit. We address this issue next.

Ring Traversals. Let (xa^*) denote the set of cyclic configurations wherein all cells other than the one containing sequence x contain the tape symbol a. We now show that if M starts at the configuration (xa^*) with the head positioned inside interval X, then the above result says that if the head exits X for the kth time, then it cannot shuttle in the outer ring forever, but (a) it either re-enters X, or (b) it deadlocks outside X, and in both cases when that happens the configuration of the ring is of the form $(yx'zb^*)$ where $|x'| = |x|$ and y and z constitute the 'out-growth' of the sequence in interval X during the kth 'excursion' of the head outside X. The ring traversal lemma given below allows us to quantify the length of this outgrowth. The key idea is that starting from a cyclic configuration of the form $(yxza^n)$ with interval X containing the sequence x, if the head exits X on the right, then the head may re-enter X on the right thus completing an external transition or deadlock outside X without diverging in the outer ring. The interesting case occurs when the head diverges in the outer ring, say from the right end of interval Z (containing z) in the counterclockwise direction. Because of the circular nature of the tape, the head enters interval Y (containing sequence y) from the left end. There are three possibilities now. The head may in finitely many moves either (1) re-enter X from the left without diverging again in the outer ring again, thus completing the external transition, or (2) deadlock without re-entering interval X and without diverging again in the outer ring, or (3) diverge in the outer ring again, this time in the clockwise direction. In this fashion, we see that the head may keep on diverging back and forth in the outer ring till it either re-enters X from either the right or the left end, or it deadlocks without re-entering X. This is formalized in the ring traversal lemma, the statement of which requires the notion of crossing numbers defined next.

Crossing Numbers. Let y be a given finite string of tape symbols and let interval Y comprised of cells $1, ..., n$ of a linear tape, contain y. Let cell $n + 1$ contain Δ, where δ, the transition relation for M has the property that $\delta(q, \Delta) = (q, \Delta, L)$. Thus Δ merely 'reflects' back the head to the left into Y.

Then the *left-right crossing number* of y, denoted by $C_{LR}(y)$, is intended to capture the number of moves made by the head on the left end of interval Y, viz., from cell 1 to 0, after the head enters interval Y at the left end and before it exits Y at the right end for the first time. Formally, $C_{LR}(y)$ is defined as follows. Starting at cell 0 (containing Γ), if the head ever exits interval Y on the right, viz., makes a right move from cell n to $n+1$, then we define $C_{LR}(y)$ as the number of moves made by the head from cell 1 to 0 before it exits Y to the right for the first time. If the head never exits Y on the right, there are three possible cases (1) the head either deadlocks in Y in which case $C_{LR}(Y)$ is defined to the number of moves made by the head from cell 1 to cell 0, viz., at the *left end* of interval X, before it deadlocks, or (2) the interval Y is exited to the left an unbounded number of times in which case we define $C_{LR}(y)$ as ∞, or (3) after finitely steps the head keeps on shuttling in Y without exiting Y on either side thereafter. In that case, we define $C_{LR}(y)$ as $\perp$.

In general, for $D_1, D_2 \in \{L, R\}$, we may define $C_{D_1D_2}(y)$, to capture the number of moves made by the head on the D_2'th end of interval Y, where $D_2' \in \{L, R\} \setminus \{D_2\}$, viz., the opposite end from which the head is supposed to exit Y, after the head enters interval Y at the D_1th end.

Proposition 4.1 (Ring Traversal Lemma) *Starting at the initial configuration (xa^n) of C^{l+n}, suppose that when the head exits interval X for the kth time, the ring configuration is of the form $(yx'zb^*)$, where x' is the content of X. If n is greater than the maximum of the minimum of $C_{LL}(z)|G|$ and $C_{LR}(y)|G|$, and the minimum of $C_{RL}(z)|G|$ and $C_{RR}(y)|G|$, viz., the ring is of sufficiently large size, then one of the following holds.*

1. the head deadlocks before entering interval X again.

2. the head re-enters interval X after finitely many steps.

In both cases, the resulting configuration is of one of the two forms: $(y''y'x''z'c^*)$ or $(y'x''z'z''c^*)$, where $|x''| = |x|$, $|y'| = |y|$, $|z'| = |z|$ and $|y''|, |z''|$ are less than or equal to the minimum of $C_{RL}(z)|G|$ and $C_{RR}(y)|G|$ or the minimum of $C_{LR}(y)|G|$ and $C_{LL}(z)|G|$ accordingly as the head exits X to the left or to the right.

A crucial consequence is that the behavior of the head (as far as interval X is concerned) after exiting X for the kth time is the same for all n greater than a *threshold* value, viz., the minimum of $C_{RL}(z)|G|$ and $C_{RR}(y)|G|$ or the minimum of $C_{LR}(y)|G|$ and $C_{LL}(z)|G|$, the only difference being the number of cells in the outer ring containing the symbol c. This observation gives us the cutoff which we derive next.

Generating the Cutoff. Using the above result, we next show the existence of cutoff $c \geq l$ such that for all $j \geq c$, the transition graphs $G_X(j)$ is the same as $G_X(c)$. Let (xa^n) be the initial tape configuration with the head at cell 0. Recall that G_a is the subgraph of G induced by the set of all tape symbols reachable from a in G. Here we consider only one case where G_a is a simple path with the other being handled in a similar fashion.

Let G_a be the simple path $a_0 \to ... \to a_{\mathsf{d}}$. In this case, we have that the head can make at most $\mathsf{d}+1$ moves from any cell of the ring without deadlocking. Then, from the definition of crossing numbers, it follows that $C_{DD'}(w) \leq \mathsf{d}+1$, for any sequence w of tape symbols and any $D, D' \in \{L, R\}$. Hence from the Ring Traversal Lemma

4.1, it follows that after the head exits X in configuration $(yx'zb^*)$ the length of the newly added intervals Y' and Z' containing respectively y'' and z'' is at most $(\mathsf{d}+1)^2$. Since the head can exit interval X at most $\mathsf{d}+1$ times (without deadlocking), at most $\mathsf{d}+1$ external transitions can be fired in $G_X(n)$ for any n. Then, using proposition 4.1 repeatedly, we have that in all exits and re-entries of X, the total length of the newly added intervals is at most $(\mathsf{d}+1)^3$. Thus in this case, for each $j \geq c = l + (\mathsf{d}+1)^3$, $G_X(j)$ is the same as $G_X(c)$ and so the value of the cutoff is $c = l + (\mathsf{d}+1)^3$.

Multiple but Bounded Number of States. Using the fact that M is deterministic, we can reduce the analysis of the case where $b \geq 1$ changes are allowed to the control state to the repeated application of the case with one control case. Starting from the initial configuration (xa^*) in state q_0, the first step is to decide whether a state change occurs to the head and if, yes, then the resulting configuration $\mathbf{c}_0$ after the move in which the change occurs. If a state change does occurs then we repeat the above step but starting in $\mathbf{c}_0$ as the initial configuration. But this is just an instance of the original problem but with one lesser state change allowed. Thus to study the behavior of M we need to carry out this procedure at most b times.

Proposition 4.2 (Decidability Result) *The PMCP for LTL$\backslash$X properties is decidable for bidirectional rings with a token that is allowed to change value a bounded number of times.*

4 Applications

The framework(s) presented in this paper are broad enough to model a variety of ring based applications. Our framework can model the Leader Election Problem and Token Ring LANs, neither of which could be handled by [11]. Examples that require bidirectional rings include bidirectional variants of all applications considered in [11]. However for lack of space, we consider only the Leader Election Problem.

Leader Election Protocols. In local area token networks, a single token circulates around the ring giving it owner the sole right to initiate communication. If the token is lost, then the *Leader Election Problem* [15], is to elect a new unique leader to act as the new owner of the regenerated token.

The LCR Leader Election Protocol. The Le Lann, Chang and Roberts (LCR) protocol assumes that each process P_i in a given unidirectional ring has an integer $id_i > 0$ as a unique identifier not necessarily in increasing or decreasing order around the ring. The protocol works as follows: Each process P_i sends token t_i with its identifier value id_i around the ring. We model this by letting P_i own t_i. When a process receives a token, it compares the value of the token to its own identifier. If the value is greater than its identifier, it passes the token unchanged. If the value is less than its own it changes its value to 0. If the value is equal to its own, the process declares itself leader. The transition diagram for process P_i is shown in figure 1. Then from the discussion in section 2, it follows that $P(t_i)$ is the first (in case there exists one) process occurring along the ring in the clockwise direction with identifier greater than id_i.

We need to verify that for any arbitrarily large ring it is never the case that two distinct processes P_i and P_j declare themselves leaders, viz., $f = \mathsf{EF}(leader_i \wedge leader_j)$

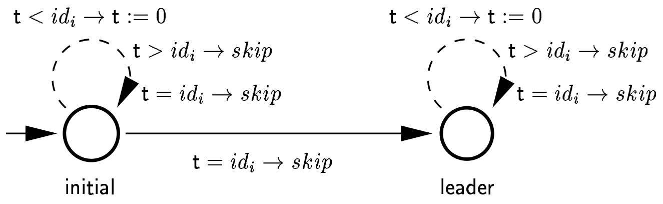

Fig 1. The LCR Protocol

is not satisfied. Since $|P(t_i)|, |P(t_j)| \leq 1$, $\mathcal{R}(i, j)$ has at most 4 processes. Then, we see using proposition 2.1, that we can reduce the reasoning of the leader election protocols for *all* rings containing processes P_i and P_j irrespective of their size to 6 canonical ring systems each with at most 4 processes. Since i and j were arbitrarily chosen, we have that since the LCR protocol is correct for the 6 canonical systems it is correct for any arbitrary ring.

5 Concluding Remarks

The generally undecidable PMCP has received a good deal of attention in the literature. A number of interesting proposals have been put forth, and successfully applied to certain examples (e.g., [1, 2, 5, 6, 14, 18]). However a lot of these methods suffer from the following drawbacks: much human ingenuity may be required to develop, e.g., network invariants; the method may not terminate; the complexity may be intractably high; and the underlying abstraction may only be conservative rather than exact.

However for frameworks that handle specialized application domains decision procedures can be given that are both sound and complete, fully automatic and in some cases efficient ([4, 7, 8, 11, 12])). In this paper, we have considered the PMCP for LTL$\backslash$X properties for parameterized families of rings wherein processes communicate using message passing via tokens. Previous work, to the best of our knowledge, has only considered unidirectional rings with a solitary token that could not carry any values [11] and so messages could not be exchanged between processes. Such systems have limited expressive power and cannot model, for instance, standard solutions for the leader election problem. We have extended the known envelope of decidability of the PMCP for ring systems to bidirectional token rings wherein the token can carry messages but only a bounded number of value changes to the token are permitted. Our reduction technique involves showing how to reduce reasoning about a ring with an arbitrary number of processes to a ring with up to a cutoff number of processes. In this paper, the reduction results were established for a bidirectional ring with a single token. A possible direction for future research is to study bidirectional rings with multiple tokens.

We have also identified a broad unidirectional ring framework which allows multiple tokens with each token being allowed a bounded number of value changes. For this framework, we have shown that small cutoffs can indeed be obtained making our technique truly efficient. For bidirectional rings, our methods are *exact*, viz., both sound and complete, and fully automated and for unidirectional rings provably efficient. Moreover the use of cutoffs has the added advantage that the reduced system is a replica of the original system but with a fewer number of processes. This is beneficial for several reasons. First it gives us a clean reduction as there is no need, e.g., to construct an abstract graph which may have a complex, non-obvious structure very different from the original system. Secondly, it caters to automatic error trace recovery.

References

1. P. Abdulla, A. Boujjani, B. Jonsson and M. Nilsson. Handling global conditions in parameterized systems verification. CAV 1999.
2. P. Abdulla and B. Jonsson. On the existence of network invariants for verifying parameterized systems. In *Correct System Design - Recent Insights and Advances*, 1710, LNCS, pp 180-197, 1999.
3. K. Apt and D. Kozen. Limits for automatic verification of finite-state concurrent systems. *Information Processing Letters*, 15, pages 307-309, 1986.
4. T. Arons, A. Pnueli, S. Ruah, J, Xu and L. Zuck. Parameterized Verification with Automatically Computed Inductive Assertions. CAV 2001, LNCS 2102, 2001.
5. M.C. Browne, E.M. Clarke and O. Grumberg. Reasoning about Networks with Many Identical Finite State Processes. *Information and Control*, 81(1), pages 13-31, April 1989.
6. E.M. Clarke, O. Grumberg and S. Jha. Verifying Parameterized Networks using Abstraction and Regular Languages. CONCUR 95. LNCS 962, pages 395-407, Springer-Verlag, 1995.
7. E.A. Emerson and V. Kahlon. Reducing Model Checking of the Many to the Few. CADE-17. LNCS , Springer-Verlag, 2000.
8. E.A. Emerson and V. Kahlon. Model Checking Large-Scale and Parameterized Resource Allocation Systems. TACAS, 2002.
9. E.A. Emerson and V. Kahlon. Rapid Parameterized Model Checking of Snoopy Cache Coherence Protocols. TACAS, 2003.
10. E.A. Emerson and V. Kahlon. Model Checking Guarded Protocols. LICS, 2003.
11. E.A. Emerson and K.S. Namjoshi. Reasoning about Rings. POPL. pages 85-94, 1995.
12. E.A. Emerson and K.S. Namjoshi. Automatic Verification of Parameterized Synchronous Systems. CAV. LNCS, Springer-Verlag, 1996.
13. S.M. German and A.P. Sistla. Reasoning about Systems with Many Processes. *J. ACM*,39(3), July 1992.
14. R.P. Khurshan and L. McMillan. A Structural Induction Theorem for Processes. PODC. pages 239-247, 1989.
15. N. Lynch. *Distributed Algorithms*, Morgan-Kaufmann, 1996.
16. C.E. Shannon, A Universal Turing Machine with Two Internal States. Automata Studies. Princeton, NJ: Princeton University Press, pp. 157-165, 1956.
17. I. Suzuki. Proving properties of a ring of finite state systems. *IPL*, 28, pages 213-314,1988.
18. P. Wolper and V. Lovinfosse. Verifying Properties of Large Sets of Processes with Network Invariants. In J. Sifakis(ed) *Automatic Verification Methods for Finite State Systems*, Springer-Verlag, LNCS 407, 1989.