Refining the Undecidability Frontier of Hybrid Automata

Venkatesh Mysore[1] and Amir Pnueli[1,2]

[1] Courant Institute of Mathematical Sciences, NYU, New York, NY, U.S.A.
{mysore,amir}@cs.nyu.edu
[2] The Weizmann Institute of Science, Rehovot, Israel

Abstract. Reachability becomes undecidable in hybrid automata (HA) that can simulate a Turing (TM) or Minsky (MM) machine. Asarin and Schneider have shown that, between the decidable 2-dim Piecewise Constant Derivative (PCD) class and the undecidable 3-dim PCD class, there lies the "open" class 2-dim Hierarchical PCD (HPCD). This class was shown to be equivalent to the class of 1-dim Piecewise Affine Maps (PAM). In this paper, we first explore 2-dim HPCD's proximity to decidability, by showing that they are equivalent to 2-dim PCDs with translational resets, and to HPCDs without resets. A hierarchy of intermediates also equivalent to the HPCD class is presented, revealing semblance to timed and initialized rectangular automata. We then explore the proximity to the undecidability frontier. We show that 2-dim HPCDs with zeno executions or integer-checks can simulate the 2-counter MM. We conclude by retreating HPCDs as PAMs, to derive a simple over-approximating algorithm for reachability. This also defines a decidable subclass 1-dim Onto PAM (oPAM). The novel non-trivial transformation of 2-dim HPCDs into "almost decidable" systems, is likely to pave the way for approximate reachability algorithms, and the characterization of decidable subclasses. It is hoped that these ideas eventually coalesce into a complete understanding of the reachability problem for the class 2-dim HPCD (1-dim PAM).

1 Introduction

Reachability – the problem of deciding whether a certain continuous state is reachable from a given initial state, becomes undecidable if the dynamical system specifications allow a Turing Machine (TM) to be simulated. This is because of Alan Turing's seminal proof, that the problem of deciding whether a given TM will halt on a given input is in general undecidable [20]. Another convenient formalization is the 2-counter Minsky Machine (MM) [13], which has been shown to be able to simulate a TM. Hence reachability is undecidable for an MM, and any dynamical system that can simulate an MM as well. Hybrid Automata (HA), which can have arbitrary discrete transitions and continuous flows, correspond to a class of immense computational power. HA very easily become undecidable for the reachability query, with only extremely stringent restrictions

R. Ramanujam and S. Sen (Eds.): FSTTCS 2005, LNCS 3821, pp. 261–272, 2005.

leading to decidability. Timed automata [2], multirate automata [1], initialized rectangular automata [16,7], controllable linear systems [18], some families of linear vector fields [11] and o-minimal HA [10] have been shown to be decidable for the reachability query.

The fundamental question continues to be: "What is the simplest class of dynamical systems for which reachability is undecidable ?". The conventional answers to this question have involved proving that a certain decidable class becomes undecidable, when given some additional computational power. For instance, 2-dimensional Piecewise Constant Derivative (PCD) systems [12] and Simple Planar Differential Inclusions (SPDIs) [5] are decidable, while 3-dimensional PCDs are undecidable [3]. This paper focuses on the 2-dim Hierarchical PCD (HPCD) class introduced by Asarin and Schneider [4]. This intermediate class, between decidable 2-dim PCDs and undecidable 3-dim PCDs, is not known to be provably decidable or undecidable! Asarin and Schneider proved that 2-dim HPCDs are *equivalent* to 1-dim Piecewise Affine Maps (PAM). Since the reachability problem for 1-dim PAMs is an open question [9], 2-dim HPCD-reachability is also open. They went a step further, and proved that the HPCD class, when endowed with a little additional computational power, becomes undecidable. Thus, the HPCD [3] class (and equivalently the PAM class) is clearly on the boundary between decidable and undecidable subclasses of HA.

This paper presents new developments in the analysis of the HPCD class, a sequel to Asarin and Schneider's work [4]. We begin this analysis of the proximity to decidability and undecidability in *Section 2*, with the definitions of the various subclasses of HA we will encounter in this paper. In *Section 3*, we present our main result: 2-dim PCDs with translational resets can simulate a PAM. We then construct several very interesting subclasses of HPCDs, which also simulate PAMs. Since PAMs have been shown to be equivalent to HPCDs [4], it proves that surprisingly, these subclasses are just as powerful as the HPCD class itself. This reveals the redundancy in the expressive power of the HPCD, and shows how even closer HPCDs are to decidable systems. In *Section 4*, we present some undecidable extensions of HPCDs, very different from Asarin and Schneider's constructions. They reveal new dimensions of the fineness of the line separating HPCDs from undecidability. We present a simple algorithm for over-approximating reachability in PAMs in *Section 5*, and show how decidable subclasses can be identified. We summarize our contributions in *Section 6* and discuss several open research questions.

2 Background: Hybrid Automata and Subclasses

An HA approximates a complicated non-linear system in terms of a model that is partly discrete and partly continuous [8,15]. An HA is a directed graph of discrete states and transitions, which allows arbitrary: (1) "invariant" expressions dictating when the system can be in this state; (2) differential equations in the

[3] Henceforth, "HPCD" refers to 2-dim HPCD, "PAM" to 1-dim PAM, and "decidability" to decidability of reachability, unless explicitly stated otherwise.

"flow" expressions, in each discrete state (continuous evolution with time); (3) conditions controlling when a transition can be taken, in the "guard"; (2) equations that change the values of the variables, in the "reset" expressions during each discrete state transition (instantaneous discrete evolution). A computation of an HA is a series of continuous evolution steps of arbitrary time-length each, interspersed with an arbitrary number of zero time-length discrete transition steps.

Before introducing the subclasses, we quickly review some terms frequently used to describe different restrictions. Let a, b, c, d stand for numerical constants, and p, q stand for the HA variables. A *rectangular guard* refers to an expression of the form $a < p < b$, while a *non-rectangular* or *comparative guard* is of the form $ap + bq + c < 0$. A *rectangular invariant* is of the form $a < p < b \bigwedge c < q < d$ i.e. the state represents a rectangular region in the $p - q$ plane. State invariants are said to be *non-overlapping* if the regions they represent in their variable-space do not intersect. A *constant reset* refers to $p' = c$, a *translational reset* refers to $p' = p + c$ and an *affine reset* to $p' = ap + b$. An "initialized" automaton is one where all variables, whose flow changes after a discrete state transition, are reset to a constant. An automaton is "timed" if all flow-derivatives are 1.

Among the several formalizations that simplify the reachability problem by curbing the computational power of the HA, we dwell on the PCD construct. A 2-dim PCD [12] is an HA in two continuous variables, where (1) All flow-derivatives are constants; (2) The guards are rectangular i.e. $p \in I$, where I is a numerical interval; (3) No variable can be reset during transitions i.e., $p' = p \wedge q' = q$; (4) The discrete states (invariants) correspond to *non-overlapping* rectangles in the real plane with non-empty interiors. The trajectories of a 2-dim PCD are restricted to be broken straight lines, with slopes changing only when a different polygonal region (new discrete state) is entered. Maler and Pnueli [12] used the property of planar systems to prove that reachability is decidable for 2-dim PCDs. 3-dim PCDs are the natural extension of 2-dim PCDs with a third dynamic variable (dimension). Asarin, Maler and Pnueli [3] proved that 3-dim PCDs are undecidable.

Subsequently, Asarin and Schneider set out to discover an "open" class in between 2-dim and 3-dim PCDs. They proceeded by studying HA that could simulate a known open problem - the 1-dim PAM. To understand their equivalence result, we first introduce PAMs, where computation is modeled as iterative function evaluation. A PAM [9] is of the form $f(x) = a_i x + b_i\ ,\ x \in I_i\ ,\ i = 1, 2, \cdots, n$, where all a_i, b_i and the ends of the non-overlapping intervals I_i are rational. f is closed i.e. $\forall x, i\ (x \in I_i) \Rightarrow (\exists j,\ f(x) \in I_j)$. Further, the intervals are in ascending order. In other words, there are n non-overlapping partitions of the real line (which may not cover it entirely). The current value of the variable x decides which interval I_i it falls in, and hence its next value $f(x)$ is uniquely defined. The reachability problem is also defined in the natural way: "Is the point x_f reachable from the point x_0 by repeated application of the piece-wise affine maps ?". Note that unlike HA, there is no non-determinism or choice – the starting point defines a unique trajectory.

Recall that a class A simulates a class B if every computational trajectory of B has a unique counterpart in A. Two classes are equivalent if they simulate each other: thus if the reachability problem is (un)decidable for one class, so it is for the other. Asarin and Schneider characterized the PCD extensions necessary to simulate a PAM, keeping in mind that the resulting HA subclass in turn needed to be expressible as a PAM. The result was the HPCD class which augmented a PCD, by allowing comparative guards and affine resets in overlapping regions of the plane. A 2-dim HPCD [4] is an HA in 2 continuous variables where, (1) All flow-derivatives are constants; (2) The guards are of the form $(ax + by + c = 0 \wedge x \in I \wedge y \in J)$ where I and J are intervals and a, b, c and the extremities of I and J are rational-valued; (3) The reset functions are affine functions: $x' = ax+b$; (4) The state invariant, which could overlap with other state invariants, is the negation of the union of the guards. The term hierarchical was used originally, to indicate that an HPCD could also be thought of as a PCD with overlapping state invariants, where each state was actually a PCD.

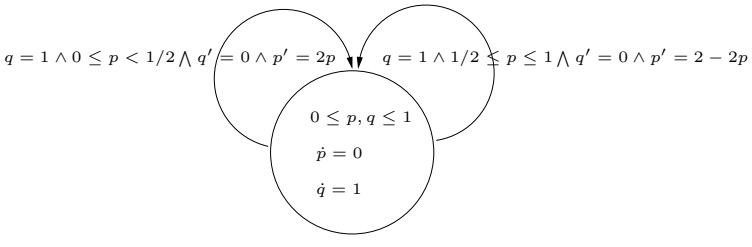

Fig. 1. One-State Tent Map HPCD

Example 1. Consider the PAM describing the Tent Map [19]:
$f(x) = 2x + 0,\ x \in [0, 1/2) (\equiv I_1)$
$f(x) = -2x + 2,\ x \in [1/2, 1] (\equiv I_2)$
The HPCD simulating this PAM is shown in *Figure* 1. □

We now summarize the results of Asarin and Schneider [4]. The HPCD class is equivalent to the PAM class. The restricted class $HPCD_{iso}$, with translational instead of affine resets, is also equivalent to the PAM class. The extended classes $HPCD_{1c}$ (with an additional counter), $HPCD_{\infty}$ (with infinite partitions) and $HPCD_x$ (origin-dependent rates) are undecidable, as they can simulate a TM.

3 Open HPCD Subclasses

Asarin and Schneider's results thus have two implications: (1) a PAM can capture an HPCD; (2) an HPCD can capture a PAM. Their first result is clearly the more significant one, also involving a non-trivial construction. It demonstrates that a 2-dim HPCD, which seems dramatically more complex than a 1-dim PAM,

actually has no additional computational power ! Their second result seems simple in comparison, as HPCDs seem to have more expressivity than necessary, to capture a PAM. A PAM can be trivially captured by a PCD with just 1 state, with all computations done only using affine resets along self-loops (see for example, the Tent Map HPCD in *Figure* 1). Schneider has proved [17] that these affine resets can be made translational ($HPCD_{iso}$). However, that construction uses all the other enhancements. To summarize, PCDs with just *affine* resets can simulate a PAM. However, multiple states with overlapping invariants and comparative guards seem necessary, when only *translational* resets are allowed.

In this section, we first prove our main result: a 1-dim PAM can be simulated using a 2-dim PCD, with just translational resets of the form $x' = x + c_i$. Comparative guards and affine resets can be done away with by making the two PCD variables (p and q) take turns simulating the PAM variable (x), while non-overlapping state invariants become sufficient because the PCD variables are guaranteed to lie in a bounded region. We then show how 1-dim PAMs can be simulated by other simple subclasses of HPCDs, each revealing proximity to a different decidable HA subclass. The following lemma simplify the proof:

Lemma 1. *A 1-dim PAM is bounded.* □

Lemma 2. *Every 1-dim PAM is equivalent to a 1-dim "positive" PAM where all intervals are positive.* □

Now we are ready to prove our main result:

Theorem 1. *A 1-dim PAM can be simulated by a 2-dim PCD with translational resets.*

Proof. Consider an equivalent 1-dim positive PAM $f(x) = a_i x + b_i \ , \ x \in I_i (\equiv [l_i, r_i)) \ , \ i = 1, 2, \cdots, n$. Let L be a number such that $L > r_n \wedge \forall i, \ L > b_i$. Corresponding to the i-th function of the PAM, we will have two states P_i and Q_i. In P_i, p flows from $p_0 = b_i$ to $x_{n+1} (\equiv b_i + a_i x_n)$ at the rate $\dot{p} = a_i$. q drops from $q_0 = x_n$ to 0 at the rate $\dot{q} = -1$. The guard $q = 0$ thus ensures that the system spends $t = q_0$ time in this state. This allows the affine term $a_i x_n$ to be computed, without using comparative guards or affine resets. In the "Q" states, the roles of p and q are reversed i.e., q uses p's value to grow to the next iterate, while p just drops to 0, effectively keeping track of time.

From P_i, there are transitions to each possible state Q_j. p retains the value it just computed, while q is reset to the constant portion (b_j) of the next iterate of x. In Q_j, q will accumulate the rest of its target value ($a_j \times x$) by flowing for time x (stored in p) at the rate a_j. Similarly, from Q_i, there are transitions to each possible state P_j, while there are no transitions within P-states or within Q-states.

The above expressions are adjusted, now assuming that each state is associated with a different large constant "base". In a state, all numbers are represented with respect to this base. Thus, x becomes $L_{S_i} + x$ in state S_i, where L_{S_i} is the base. Even if x increases or decreases to its maximum / minimum possible value,

p and q will not cross over to an adjoining state. This is because the different base constants are themselves very far apart. This base-adjustment creates the translational resets, when the current iterate needs to be remembered and passed on to a new state. It is to be noted that just constant resets suffice if the state invariants are allowed to overlap.

We now construct the PCD with translational resets and $2n$ states:

- Corresponding to the i-th function of the PAM, we have two states P_i and Q_i associated with the constants $L_{P_i} = 4iL - 3L$ and $L_{Q_i} = 4iL - L$.
- In P_i, p grows at rate $\dot{p} = a_i$ from $L_{P_i} + p_0 (= b_i)$ to $a_i q_0 (= x_n) + b_i + L_{P_i}$, while q drops from $q_0 + L_{P_i}$ to L_{P_i} at the rate $\dot{q} = -1$. q_0 denotes the unscaled previous iterate x_n, using which x_{n+1} is being computed by spending exactly $t = q_0$ time in this state.
- Q_i behaves exactly as above with p and q swapped i.e., this corresponds to the case where q grows to the next iterate, while p just drops to L_{Q_i}.
- In P_i and Q_i, the values of p and q are both bounded by $\{(L_{P_i/Q_i} - L, L_{P_i/Q_i} + L)\}$, which is equal to $\{(4iL - 4L, 4iL - 2L)\}$ in P_i and $\{(4iL - 2L, 4iL)\}$ in Q_i. Clearly, none of rectangular regions can overlap.
- From P_i, there are transitions to each possible state Q_j with guard $q = L_{P_i} \wedge p \in I_j$ i.e., "p has reached the next iterate of x" *and* "p is in the interval corresponding to the j-th PAM function". The reset (note: constant or translational) is $p' = p - L_{P_i} + L_{Q_j} \wedge q' = L_{Q_j} + b_j$ i.e., "p, which holds the current value of x, is translated to the range of the destination state (to prevent overlap)" *and* "q is reset to the constant portion (b_j) of the next iterate of x". The portion proportional to x_n ($a_j \times x_n$) will be gained by flowing for time x_n (stored in p) with slope a_j.
- Similarly, from Q_i, there are transitions to each possible state P_j. There are no transitions within P-states or within Q-states.

This PCD with translational resets simulates the PAM, as p and q take turns simulating x. It can be seen that x_f is reachable from x_0: (i) if $(p = x_f + L_{P_i}, q = L_{P_i})$ and $(p = x_f + L_{Q_j}, q = L_{Q_j} + b_j)$ are reachable; or (ii) if $(p = L_{Q_i}, q = x_f + L_{Q_i})$ and $(p = L_{P_j}, q = x_f + L_{P_j})$ are reachable. This needs to hold for some i and j, such that $x_f \in I_j$ and one of the pre-images of x_f lies in I_i. The "and" terms are necessary to eliminate intermediate points during continuous evolution from satisfying the query. The "or" term is necessary because p reaches only even iterates and q reaches only the odd iterates of x_0. The starting state is $(p = x_0 + L_{Q_k}, q = L_{Q_k} + b_k)$ (or $(p = L_{P_k} + b_k, q = L_{Q_k} + x_0)$), where $x_0 \in I_k$. □

Example 2. We will now construct a PCD with translational resets that simulates the Tent Map, in two variables p and q and $2 \times 2 = 4$ states. Setting $L = 3 (> max(r_n, b_i) = 2)$, we get $L_{P_1} = 3, L_{Q_1} = 9, L_{P_2} = 15, L_{Q_2} = 21$. The result is presented in *Figure* 2. □

Various other intermediates – subclasses of HPCDs, simulate a 1-dim PAM. We now present some of the interesting cases, which extend known decidable systems.

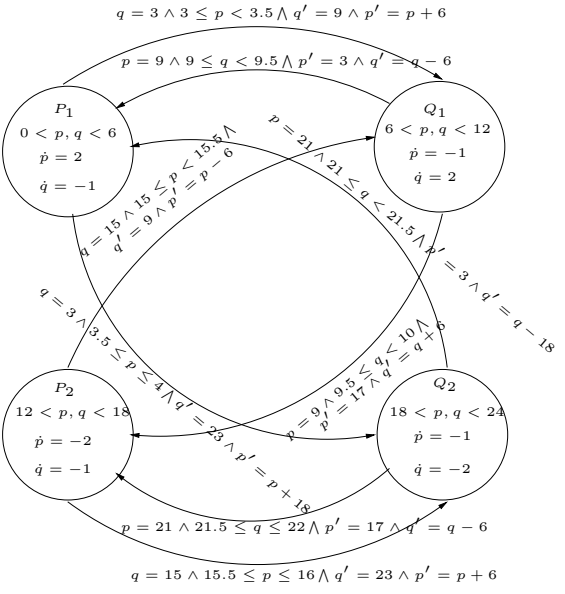

Fig. 2. PCD with Translational Resets simulating the Tent Map

Theorem 2. *A 1-dim PAM can be simulated by: (1) an HPCD with comparative guards, 3 different flows* $+1, -1, 0$ *and no resets; (2) an initialized PCD, with comparative guards; (3) an HPCD with rectangular guards i.e.* $p = 0 \wedge q \in I_i$*, when simple constant resets of the form* $q' = a_j \wedge p' = p$ *are allowed; (4) a PCD with just clocks, when translational resets and comparative guards are allowed; and (5) an HPCD with just clocks, when simple constant resets* $(p', q') = (0, q)$ *or* $(p, 0)$ *and comparative guards are allowed.*

Proof. All the proofs are based on the techniques demonstrated in the proof of *Theorem 1.* So, for brevity, we only give a flavor of results (1) and (4).

For result (1), we will construct an HPCD with $4n$ states of the form $P_j^{\pm}$ and $Q_j^{\pm}$ that simulates this PAM. We will now have p evolving from x_{n-1} to x_{n+1}, while q remains stationary at x_n. The affine guard condition $p = a_i q + b_i$ makes the HA jump to the next state at the correct time. Since x_{n+1} could be greater or less than x_{n-1}, the flow will need to be $+1$ or -1 respectively. Hence, each P (and Q) state now corresponds to two states: P^+ and P^-. In the state P_j^+, q flows from $q_0 (\in$ some $I_i)$ to $q' = a_j p_0 + b_j$, with flow is $\dot{q} = +1$. In P_j^-, $q' > q_0$ and $\dot{q} = -1$. $\dot{p} = 0$ to ensure that q flows to the correct amount. The transitions are of the form $P_j^+ \rightarrow Q_k^{\pm}$, with the guard being $q = a_j p + b_j \wedge q \in I_k \wedge p < (b_k + a_k b_j)/(1 - a_j a_k)$. The last term will be $p \geq$ if we are jumping to Q_k^-. The $Q_j^{\pm}$ states are defined exactly as above, with p and q interchanged. Clearly, the above HPCD without resets simulates the given PAM. In particular, the reachability query "*Is* x_f *reachable from* x_0" is true *iff* $(p = x_f, q = x_{f-1})$ *or* $(p = x_{f-1}, q = x_f)$ *is*

reachable from $(p = x_0, q = x_1)$, where x_{f-1} is some pre-image of x_f and x_1 is the successor of x_0.

For result (4), an HPCD with $2n$ states of the form P_j and Q_j can simulate the equivalent positive PAM. In state P_j, p flows from p_0 to $p_0 + a_j p_0 + b_j$ with $\dot{p} = +1$, while q flows from 0 to $a_j p_0 + b_j$ with $\dot{q} = +1$. The discrete transitions will be of the form $P_j \rightarrow Q_k$ with guard $a_j p - (1 + a_j)q + b_j = 0 \wedge q \in I_k$ and reset $p' = 0 \wedge q' = q$. □

4 Undecidable HPCD Extensions

True to its "open" nature, the HPCD class does not present any direct mechanism to simulate a TM / MM. Asarin and Schneider [4] have shown that the HPCD class becomes undecidable when extended with one additional counter ($HPCD_{1c}$), with infinite partition ($HPCD_{\infty}$) and with origin-dependent rates ($HPCD_x$). In this section, we present a new set of extensions of HPCDs that manage to be undecidable. We proceed by simulating the MM with the least possible additional work. Recall that an MM uses two (positive) integer counters m and n. Incrementing and decrementing a counter, and branching based on equality to zero are the operations that need to be supported.

Any program over a 2-counter MM can be almost trivially captured as an HPCD, using just the discrete transitions without using the flows. Recall that a "zeno" system is one where one cannot bound the number of discrete transitions i.e. potentially all the computation could be done in the resets, in zero time. Thus:

Theorem 3. *Reachability over HPCDs with zeno-paths ($HPCD_{zeno}$) is undecidable.* □

Alternatively, we can capture the value of both the counters m and n using just one continuous variable x as $x = p_1^m p_2^n$, where p_1 and p_2 are two prime numbers. Clearly, given the integer product x, there is exactly one way of factoring it and hence m and n can be extracted. The second variable y is now free to be used as a temporary variable for computations and to make the system non-zeno. Incrementing and decrementing the counter correspond respectively, to multiplying and dividing by the appropriate prime factor. The problem of simulating a 2-counter MM over a HPCD now reduces to the problem of checking if $m > 0$ given the numerical value of $x = p_1^m p_2^n$, and being able to recover the original value of x at the end of the procedure. One approach is to divide x by the prime number corresponding to the counter we wish to check for zero, and then check if the result of the division is an integer. *The problem of simulating a 2-counter MM over an HPCD thus reduces, to the problem of checking whether a given number is an integer (!) using the 2-dim HPCD infrastructure, and being able to recover the original number at the end of the procedure.* Surprisingly, there is no known way of doing this.

Theorem 4. *Reachability over the following HPCD-extensions are undecidable:*

1. *$HPCD_{fn-int}$, where the guard can include a function integer(x) that returns* true *if the parameter x is an integer*
2. *$HPCD_{zeno-int}$, where the integer-check function is now simulated by a zeno execution of repeatedly subtracting* 1 *and checking if the number equals* 0 □

5 Understanding PAMs

Having refined the decidable and undecidable frontiers of the HPCD class, we explore one last avenue – treating HPCDs as PAMs, and subjecting them to a similar extend-restrain analysis. Just like we enhanced a 2-dim HPCD to make it undecidable, we present the flavor of a similar effort for PAMs.

Theorem 5. *1-dim PAMs that can check if a given number x can be expressed as p^{-i} (the class "PAM_{pow}"), where p is a given prime number and i is an unknown positive integer, can simulate an MM.* □

We stop with this contrived extension, and move on to restricted subclasses.

The simplest PAM is one where every interval maps exactly on to another interval. Thus the mapping unwinds to a cyclical application of functions, possibly preceded by a linear sequence.

Definition 1. ***1-dim oPAM*** *A 1-dimensional Onto PAM (oPAM) is a 1-dim PAM where, for every interval I_i in the PAM definition, there is an interval I_j also in the definition such that $\{a_i x + b_i | x \in I_i\} = \{x | x \in I_j\}$.* □

Next we prove a crucial lemma:

Lemma 3. *In a 1-dim oPAM with k intervals, every point has at most $2k$ unique successors.*

Proof. If interval I_i maps on to I_j, the end points (l_i, r_i) have to map on to (l_j, r_j) or to (r_j, l_j). No other mapping is possible because of our restriction, that the affine post-image of I_i has to exactly and completely overlap with I_j. Hence, there are only two possible equations linking x_j with x_i:

1. Direct $(l_i \to l_j, r_i \to r_j)$: $x_j = l_j + \frac{x_i - l_i}{r_i - l_i}(r_j - l_j)$
2. Flipped $(l_i \to r_j, r_i \to l_j)$: $x_j = l_j + \frac{r_i - x_i}{r_i - l_i}(r_j - l_j)$

In other words, if we define $d = \frac{x_0 - l_{x_0}}{r_{x_0} - l_{x_0}}$, only the points that are $l_j + d(r_j - l_j)$ or $l_j + (1-d)(r_j - l_j)$ are ever reachable. Thus, every interval has only two possible reachable points from a given x_0. Since there are k intervals, after $2k$ iterations all possible successors would have been explored, and there will be a cycle of period $\leq 2k$ in the path. □

Using this observation about exactly onto affine maps over linear intervals, we can prove that:

Theorem 6. *Reachability is decidable for 1-dim oPAMs.* □

Example 3. $f(x) = 2x + 1/3,\ x \in [0, 1/3)(\equiv I_1)$ and $f(x) = 1/2 - x/2,\ x \in [1/3, 1](\equiv I_2)$ is a oPAM as $f([0, 1/3]) = [1/3, 1]$ and $f([1/3, 1]) = [0, 1/3]$. Thus, all points reachable from $x_0 = 1/4$ are given by $x_1 = 2/4 + 1/3 = 5/6, x_2 = 1/2 - 5/12 = 1/12, x_3 = 2/12 + 1/3 = 1/2, x_4 = 1/2 - 1/4 = 1/4 = x_0$ as expected. □

Reachability is easily semidecidable for PAMs: we just keep iterating x_0, $f(x_0)$, $f(f(x_0))$, $\cdots$ until x_f is reached. If x_f is not reachable, this algorithm will never converge. We now present a simple algorithm for over-approximating the reachable points (see box below). The idea is to repeatedly partition the intervals I_i of the PAM, until all the successors (post-images) of points in one interval map on to exactly one complete interval i.e. domain and range are fully covered (an extension of this idea was presented in [14]).

Over-Approximation of PAM Reachability

1. Let the initial set of partitions P be the set of PAM intervals $\{I_i\}$
2. Pick an interval P_i in P and calculate its post-image P_i'. Let P_i' span the intervals $P_l, P_{l+1}, \cdots, P_{r-1}, P_r$.
3. P_i' induces $r-l+1$ partitions of P_i: $P_{i_1} \cdots P_{i_{r-l+1}}$ such that P_{i_j} maps on to P_{l+j-1}. It could also partition P_l and P_r in case it maps on to a sub-interval rather than covering the whole of P_l or P_r. In all, the total number of partitions can increase by 0 to $n + 1$.
4. Update P so it now holds the newly induced partitions as well.
5. Repeat steps $2-4$ until every interval P_i maps on to exactly one interval P_j already in P

By treating each interval as a node and connecting P_i and P_j if the post image of P_i is P_j, we get a graph representation of the PAM. Thus, x_f is reachable from x_0, if there is a path from P_{x_0} to P_{x_f} in this graph (where $x_i \in P_{x_i}$). □

Clearly, the algorithm is not guaranteed to converge. However, we can terminate after a reasonable number of steps and still use the resultant graph to approximately decide reachability. Also note that the graph needs to be constructed only once no matter how many different reachability queries we need to answer. A rewarding observation is that a 1-dim oPAM is obtained, if the above partitioning algorithm converges ! This concurs with the fact that they are decidable.

6 Discussion

In this paper, we refined the decidability frontier by exploiting the expressive redundancy of the HPCD class definition. We introduced the "taking-turns" idea, that the two PCD variables could alternately compute PAM iterations.

It was pointed out by one of the reviewers that a similar idea was used by Berard and Duford to prove that the emptiness query is undecidable for timed automata with four clocks and additive clock constraints [6]. We also showed how we could exploit the finite range of the PAM to construct non-overlapping state invariants. These ideas helped show that a 1-dim PAM can be simulated by a 2-dim PCD with translational resets. Further, resets can be disposed, if we allow overlapping invariants and comparative guards. We also demonstrated how decidable classes, like timed and initialized rectangular automata, can be extended into open problems. On the undecidability front, we showed that zeno HPCD executions can naturally capture MMs. More interestingly, the ability to check if a number is an integer was seen to be the computational ability, that separates an HPCD from universal Turing computability. A simple algorithm for over-approximating reachability was presented. It revealed that the problem is decidable, for those PAMs that converge during this iteration (oPAMs). The current understanding of this undecidability frontier of HA is summarized in *Figure* 3.

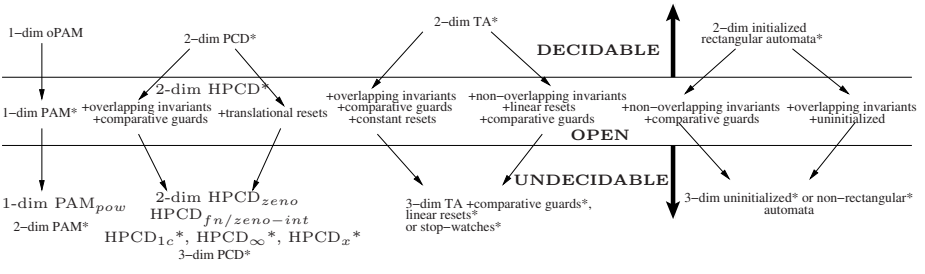

Fig. 3. Decidable, Open and Undecidable subclasses of HA ("*" indicates what was already known; unstarred results are contributions of this paper)

There are many related questions that need to be explored. Using the reductions of an HPCD to a PCD or initialized rectangular automaton with extensions, can we identify more interesting decidable subclasses and approximate reachability algorithms ? One suggestion we offer is to construct the "PCD-graph" of a 1-dim PAM, and then show how planarity can correspond to decidability. Using the construction in *Theorem 1*, we can build a graph with a set of nodes capturing each state. Different nodes can correspond to the different intervals, with an edge corresponding to each reset. We can then check whether it is possible to rearrange the given PCD-graph into a planar graph, by using the standard graph drawing literature. Another perspective to be explored is, how discrete chaotic dynamical systems (like the Tent Map) can have a say in the decidability of the PAM class. It is hoped that the ideas presented in this paper will aid the eventual "deciding" of the 1-dim PAM / 2-dim HPCD reachability problem.

References

1. R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The Algorithmic Analysis of Hybrid Systems. *Theoretical Computer Science*, 138:3–34, 1995.
2. R. Alur and D.L. Dill. A Theory of Timed Automata. *TCS*, 126:183–235, 1994.
3. E. Asarin, O. Maler, and A. Pnueli. Reachability analysis of dynamical systems having piecewise-constant derivatives. *Theoretical Computer Science*, 138:35–65, 1995.
4. E. Asarin and G. Schneider. Widening the boundary between decidable and undecidable hybrid systems. In *CONCUR'2002, Brno, Czech Republic*, volume 2421 of *LNCS*, pages 193–208. Springer-Verlag, August 2002.
5. E. Asarin, G. Schneider, and S. Yovine. On the decidability of the reachability problem for planar differential inclusions. In *In: Hybrid Systems: Computation and Control*, pages 89–104. LNCS 2034, March 2001.
6. B. Berard and C. Dufourd. Timed automata and additive clock constraints. *Information Processing Letter*, 75(1-2):1–7, 2000.
7. T. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya. What's Decidable about Hybrid Automata. In *Symposium on the Theory of Computing (STOC)*, pages 373–382, 1995.
8. Thomas A. Henzinger and Shankar Sastry. *Hybrid Systems-Computation and Control: Proceedings of the First International Workshop, HSCC '98. Lecture Notes in Computer Science 1386*. Springer-Verlag, 1998.
9. Pascal Koiran. My favourte problems. http://perso.ens-lyon.fr/pascal.koiran/problems.html, 1999.
10. G. Lafferiere, G. J. Pappas, and S. Sastry. O-minimal Hybrid Systems. *Mathematics of Control, Signals, and Systems*, 13(1):1–21, March 2000.
11. Gerardo Lafferriere, George J. Pappas, and Sergio Yovine. Symbolic reachability computation for families of linear vector fields. *J. Symb. Comput.*, 32(3):231–253, 2001.
12. O. Maler and A. Pnueli. Reachability analysis of planar multi-linear systems. In C. Courcoubetis, editor, *Computer Aided Verification: Proc. of the 5th International Conference CAV'93*, pages 194–209, Berlin, Heidelberg, 1993. Springer.
13. M.L. Minsky. Recursive unsolvability of post's problem of tag and other topics in theory of turing machines. *Ann. of Math.*, 74:437–455, 1961.
14. Venkatesh Mysore and Bud Mishra. Algorithmic Algebraic Model Checking III: Approximate Methods. In *Infinity*, 2005.
15. C. Piazza, M. Antoniotti, V. Mysore, A. Policriti, F. Winkler, and B. Mishra. Algorithmic Algebraic Model Checking I: The Case of Biochemical Systems and their Reachability Analysis. In *CAV*, 2005.
16. A. Puri and P. Varaiya. Decidebility of hybrid systems with rectangular differential inclusions. *Computer Aided Verification*, pages 95–104, 1994.
17. Gerardo Schneider. *Algorithmic Analysis of Polygonal Hybrid Systems. Ph.D. thesis.* VERIMAG - UJF, Grenoble, France, 2002.
18. Paulo Tabuada and George J. Pappas. Model checking ltl over controllable linear systems is decidable. *Hybrid Systems : Computation and Control, Lecture Notes in Computer Science*, 2623, April 2003.
19. Gerald Teschl. Ordinary differential equations and dynamical systems. Lecture Notes from http://www.mat.univie.ac.at/ gerald/ftp/book-ode/index.html, 2004.
20. Alan Turing. On computable numbers, with an application to the entscheidungs problem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936.