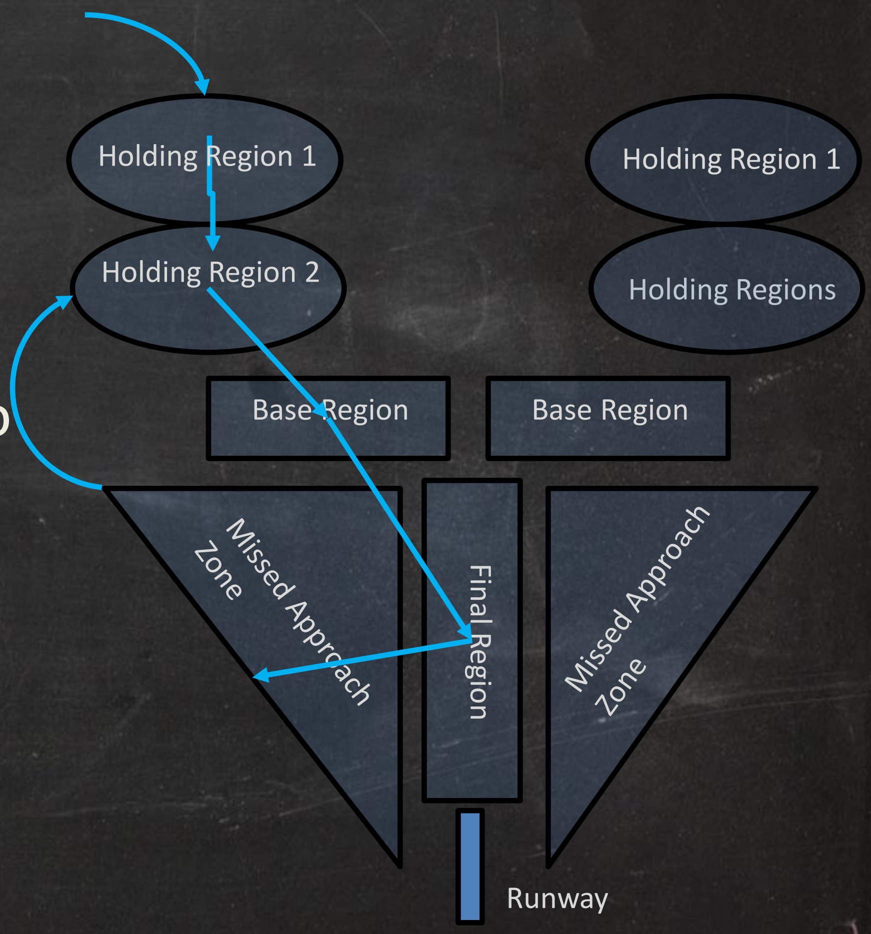
Parameterized Verification of Distributed Cyber-Physical Systems
An Aircraft Landing Protocol Case Study

Taylor Johnson and **Sayan Mitra**
University of Illinois at Urbana-Champaign

ICCPS 2012, CPSWeek, Beijing

Distributed air-traffic control protocol

- The Small Aircraft Transportation protocol (SATS) [Abbott et al. NASA Report 2002]
- Distributed traffic control for increasing general aviation access to small airports with minimal centralized infrastructure
- Features of the system/model
  - (Cyber) Location, sequence of agents
  - (Physical) Motion of agents
  - (Distributed) Ordering data-structure is spread across multiple aircrafts



Parameterized Systems and Verification

- Goal: Verify that SATS guarantees safety and progress even if **arbitrarily many aircraft participate in the protocol**

- Parameterized verification
  - For every instantiation of such a system, verify some property P regardless of the number of agents
  - $\forall N \in \mathbb{N}. \mathcal{A}(N) \triangleq \mathcal{A}_1 \parallel \mathcal{A}_2 \parallel \cdots \parallel \mathcal{A}_N \vDash P(\mathrm{N})$
  - Example: $\mathrm{P(N)}$: $\forall i, j \in [N],\ x_i - x_j > S$
    - No two aircraft ever collide, no two processes enter a critical section simultaneously

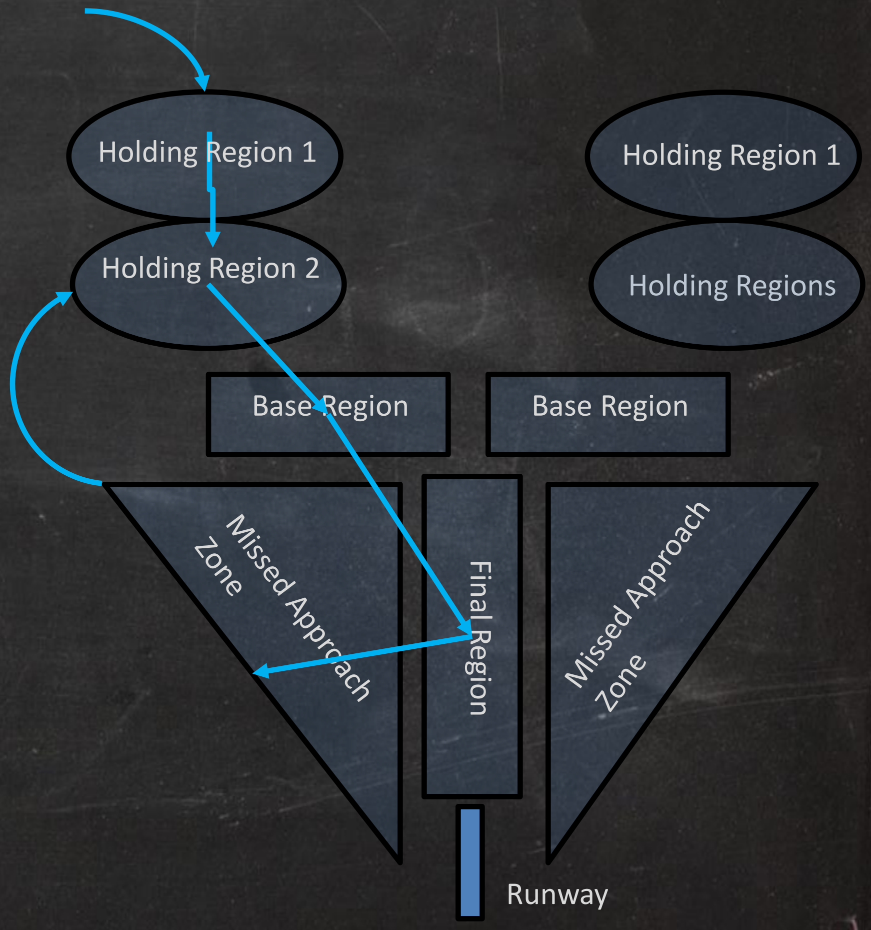
- Parameterized systems are all around …
  - Aircraft and vehicles in distributed air traffic control
  - Collaborative Apps on Mobile phones, e.g., Geocasting and Sensing
  - Robotic swarms (platooning, flocking)
  - Networked medical devices

Related Work: Automatic parameterized verification

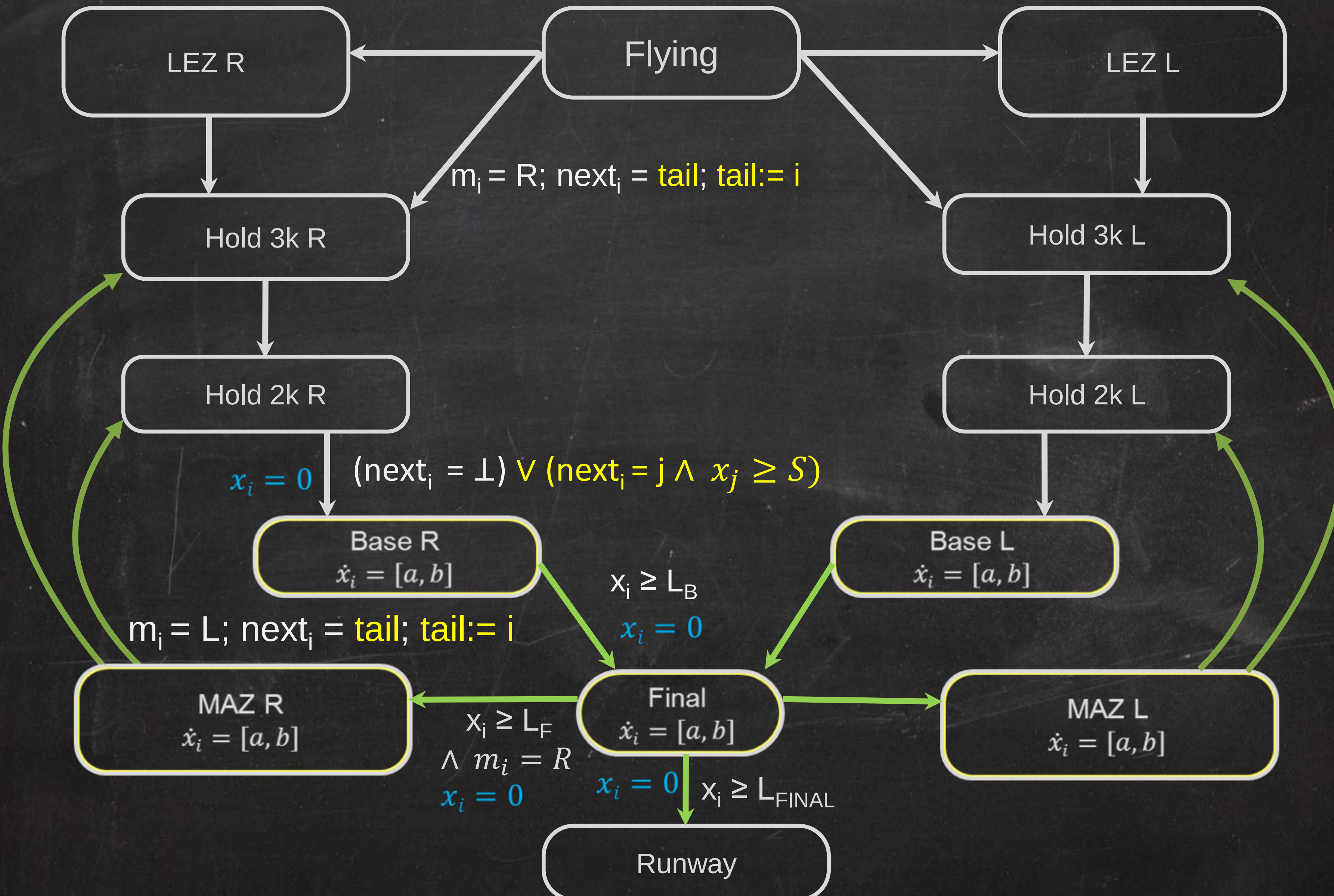
- Finite-state automata: Undecidable in general [Apt and Kozen, 1986]
- Timed automata
  - Decidable with a single real-valued clock, finite number of integer clocks [Abdulla et al. 2001-04]
  - Undecidable with two or more real-valued clocks, urgency, universal guards [Abdulla et al. 2001-07]
- Model checking:
  - Counter abstraction [Delzanno 2000], Environment abstraction [Clarke, Talupur, and Veith 2006], Network invariants [Wolper, Lovinfosse, 1990], Small model theorems [Pnueli et al. 2001] [Johnson & Mitra, FORTE 2012]
- Theorem-prover based applications
  - SATS: Discrete abstractions [Munoz, Dowek, and Carreno 2004], Hybrid versions [Munoz and Dowek 2005-06], [Umeno and Lynch 2007]
  - Adaptive cruise control [Loos, Platzer, et al. 2011]
  - Fischer's mutual exclusion [Dutertre and Sorea 2004]
- MCMT: Tool for backward reachability algorithm [Ghilardi et al. IJCAR 2008], Timed automata [Carioni et al. 2010]

SATS Overview

- Automaton model of each aircraft A_i
  - Region of airspace
  - Position within region (x_i)
  - Sequence number (s_i)
  - Miss direction (m_i)
- Central coordinator assigns unique **sequence numbers**
- Aircraft coordinate with one another to make landing attempts while ensuring **separation assurance**
- Communication modeled as **synchronized transitions** that **atomically** read/write the state of two aircrafts (and coordinator)

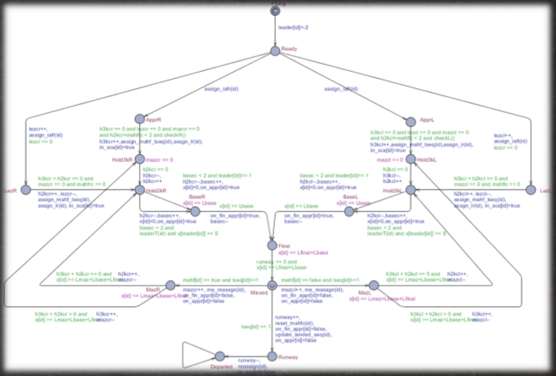


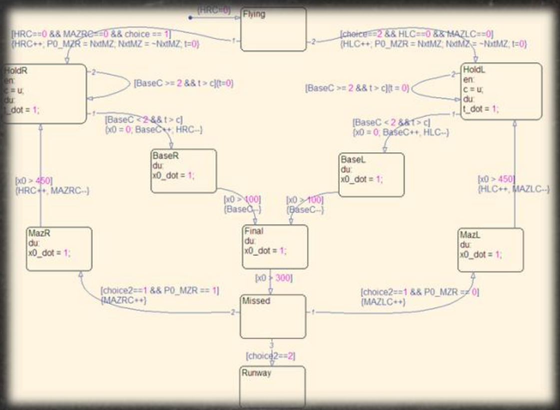
Hybrid Automaton Model for an aircraft in SATS

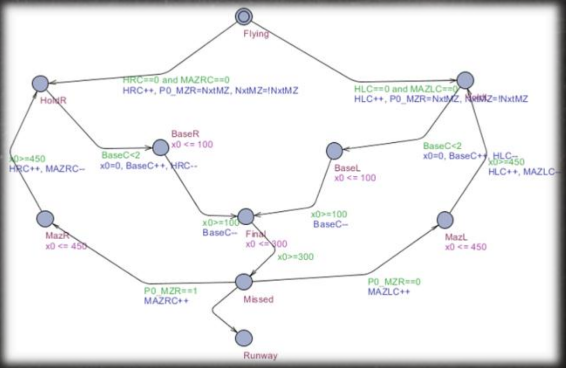


Direct verification results

SATS Constant velocity	• Automatic translation from Simulink to UPPAAL • Verification of properties using UPPAAL (4 aircrafts 10 mins; 5 aircrafts ~ 1 hour)
SATS Simplified	• Automatic translation from Simulink to HyTech for a simpler model with no sequences • Verification of properties using HyTech (4 aircrafts, 155 sec)







Parameterized verification: Symbolic computation for entire family of systems:

Symbolic representation of states, transitions, unsafe sets

Variables and Symbolic States

Variables

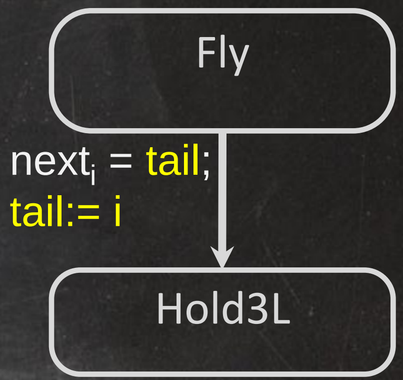
- q_i : { Fly, H3KL, BaseL, …, Runway} — Control location for A_i
- x_i : $\mathbb{R}$ — Position of A_i within q_i
- $next_i$: $\mathbb{N}_\perp$ — Sequence number
- m_i: {Left, Right} — Miss side
- tail : $\mathbb{N}_\perp$ — Global counter airport module

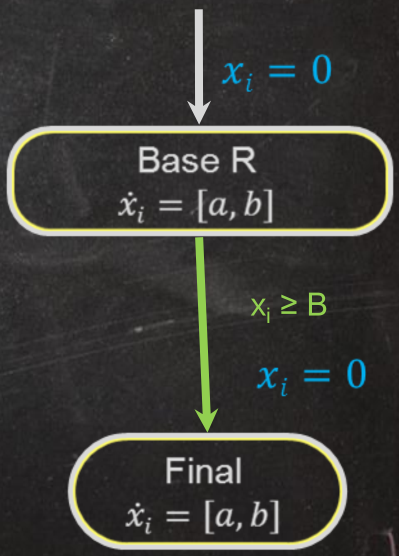
Parametric predicates

- General: $\phi_I(N) \triangleq \forall i, j, k, \ldots \in [N]. \psi_I$, where ψ_I is a propositional formula
- Initial condition: $\mathrm{Init}(N) \triangleq \forall i \in [N]. q_i = \mathrm{Fly}$
- Separation: $\mathrm{Sep(N)} \triangleq \forall i, j \in [N], i \neq j \wedge q_i, q_j \in \{\mathrm{Base, Runway, Missed}\} \wedge (next_i = j) \Rightarrow x_j - x_i \geq S$
- Unsafe (negation): $\exists i, j \in [N], (i \neq j \wedge q_i, q_j \in \{\mathrm{Base, Runway, Missed}\} \wedge (next_i = j) \wedge x_j - x_i < S$
- General: $\phi_b \triangleq \exists i, j, k, \ldots \in [N]. \psi_b$ for some propositional formula ψ_b over the variables of $\mathcal{A}_i$

Discrete Transitions

- For each location pair a to b the transition
  - $T(N, Fly, Hold3L) \triangleq \exists\, i \in [N]$
    - $q_i = \text{Fly} \wedge$
    - $q_i' = \text{Hold3L} \wedge next_i' = tail \wedge tail' = i \wedge$
    - $\forall\, j \in [N]{:}\, j \neq i \Rightarrow next_j' = next_j \wedge q_j' = q_j$
- $Trajs(N){:}\ \exists\, t > 0, \forall\, j \in [N]{:}$
  - $(q_i = \text{Base} \Rightarrow \left(\forall\, t' \leq t{:}\ x_j + t' = B \Rightarrow t' = t\right) \wedge$
  - $(q_i = \text{Final} \Rightarrow \left(\forall\, t' \leq t{:}\ x_j + t' = L \Rightarrow t' = t\right) \wedge$
  - $\ldots \wedge$
  - $x'_j = x_j + t$
- $T(\boldsymbol{x}, \boldsymbol{x}') =$
  $T(N, Fly, Hold3L) \vee T(N, Fly, Hold3R) \vee$
  $T(N, Hold3L, Hold2L) \vee \ \ldots \vee Trajs(\text{N})$





Reachability Algorithm

BR = $\neg S$ **// S: property**

While

If $BR \wedge Init$ is SAT **then return UNSAFE**

// Safety check

Else $\mathrm{BR} = \mathrm{BR} \vee \exists\, x' : \mathrm{T}(x, x') \wedge \mathrm{BR}(x')$

$P' = P \vee BR$

If $\neg(P' \Longrightarrow P)$ is UNSAT **then return SAFE**

// Fixpoint check

Else P = P' **repeat**

Every inductive invariant that is proved is conjuncted to the next invariant for strengthening

When is termination guaranteed?

- Depends on the format of the safety property
- If it is of the form $\exists i \in [N]:P(i)$ and Pre computation is also of the form $\exists i \in [N]:Q(i)$
- And, there is only discrete interaction amongst A_i then we can reach a fixpoint
- For SATS, several properties bound the number of aircrafts that can actually be present in the system

Verification Methodology

- Model Checker Modulo Theories (MCMT)
  - Performs satisfiability checks of formulas using the satisfiability modulo theories (SMT) solver Yices
  - Supports real parameters (needed for timed dynamics)
- Provide a list of properties
  - Main property is separation assurance
  - Python script calls MCMT to prove a property from this list
  - If the property is established, script assumes this property in subsequent calls to MCMT, then tries another property in the list

Properties and Runtimes

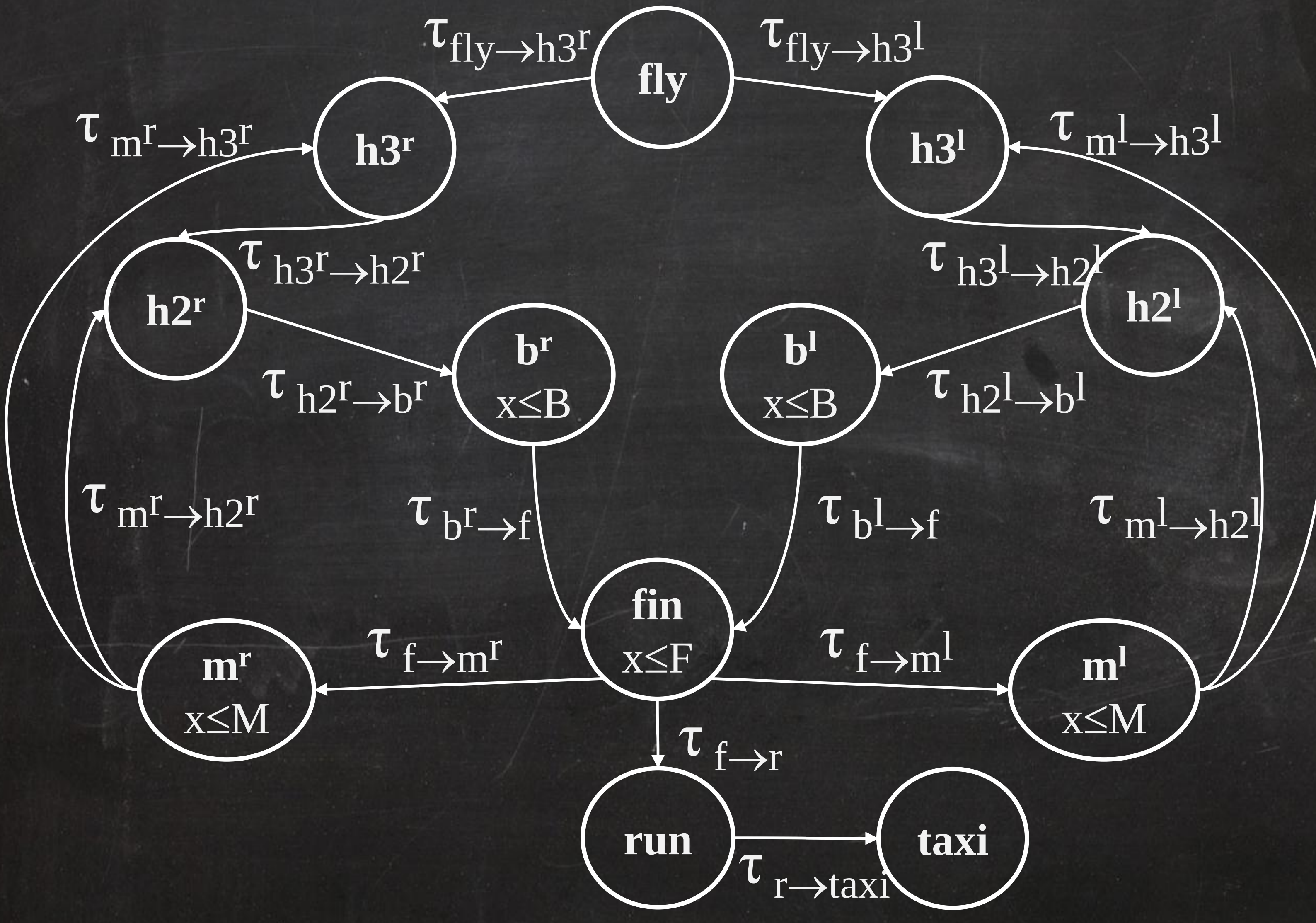
Property	Runtime (s)	Memory (MB)
No more than four aircraft in system	25.95	10
No two aircrafts violate separation	283.08	32
No more than two aircraft on the left (right)	24.50	5
At most one aircraft in each holding zone	0.81	4
No more than two aircraft on a missed approach on the left (right)	491.61	274

Conclusions & Ongoing Work

- SATS: A Benchmark for distributed cyber-physical system
- Our modeling framework: Networks of Hybrid Automata with discrete (atomic) interactions
- Verification: Parameterized backward reachability
  - Derived bounds aid termination
- Challenges: Termination & Liveness in parameterized systems
- Ongoing work:
  - Small Model Results [Forte/FMOODs paper to appear]
  - Z3-based tool implementation
  - Application to mobile peer-to-peer applications

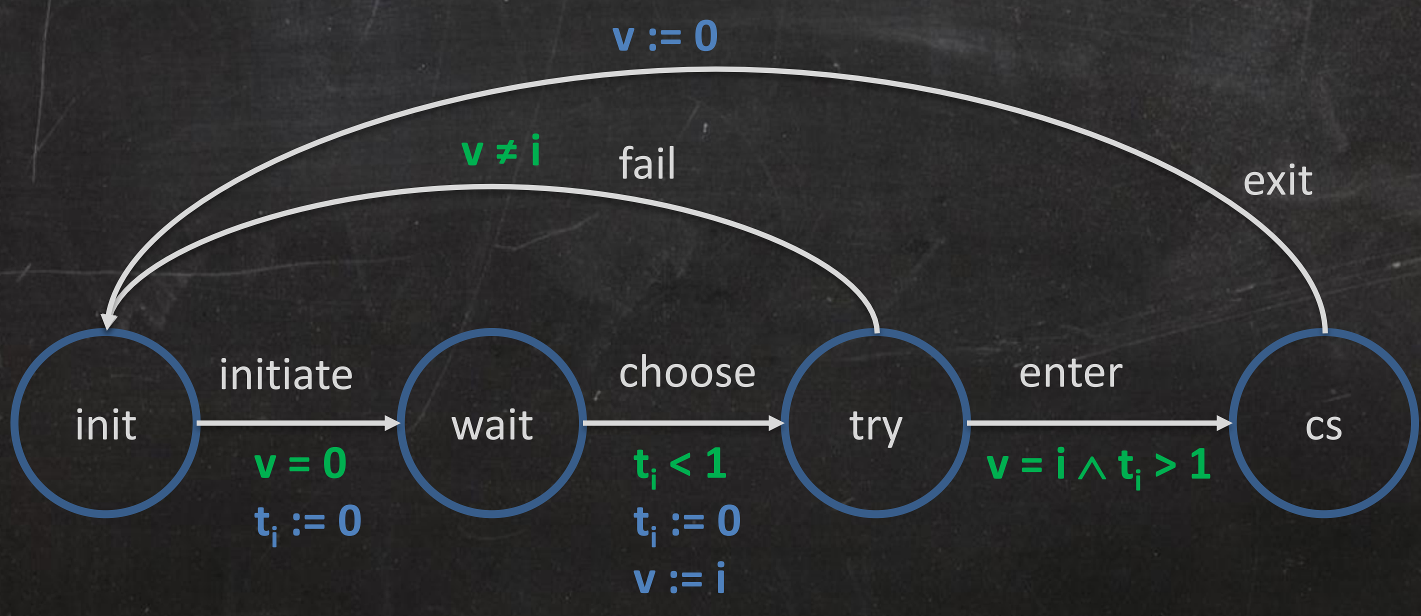
Questions?

$\mathcal{A}_i$: Aircraft Hybrid Automaton



Example: Fischer's Protocol

- Timed mutual exclusion protocol
  - 4 states: initial, waiting, trying, critical section
  - 1 real-valued clock per process
  - 1 globally shared (atomic) variable, v, ranging over process ids
  - Process ids: {1, …, n}
  - Safety property: at most one process is in critical section



Termination Example

- Parameterized finite state machines

$$
\begin{aligned}
\phi_B &\triangleq \exists z. q_z = b_0 \text{ and} \\
\phi_I &\triangleq \forall z. q_z = b_2,
\end{aligned}
$$

$$
\begin{aligned}
\tau_{b_1 \to b_0}(q, q') &\triangleq \exists z. (q_z = b_1 \wedge q' = \lambda j. (\text{if } j = z \text{ then } b_0 \text{ else } q_j)), \\
\tau_{b_1 \to b_1}(q, q') &\triangleq \exists z. (q_z = b_1 \wedge q' = \lambda j. (\text{if } j = z \text{ then } b_1 \text{ else } q_j)), \\
\tau_{b_2 \to b_2}(q, q') &\triangleq \exists z. (q_z = b_2 \wedge q' = \lambda j. (\text{if } j = z \text{ then } b_2 \text{ else } q_j)), \\
\tau(q, q') &\triangleq \tau_{b_1 \to b_0}(q, q') \vee \tau_{b_1 \to b_1}(q, q') \vee \tau_{b_2 \to b_2}(q, q').
\end{aligned}
$$

Termination Example (cont)

k	ϕ_k	ρ_k
0	$\exists z.q_z = b_0$	$\exists z.q_z = b_0$
1	$Pre(\exists z.q_z = b_0) \equiv \exists q' \tau(q, q') \wedge \phi_0(q') \equiv$ $\exists q' \, (\exists z_1.q[z_1] = b_1 \wedge q' = \lambda j.$ $(\text{if } j = z_1 \Rightarrow b_0 \text{ else } q_j) \wedge \exists z_2.q[z_2]' = b_0) \equiv$ $\exists z.q_z = b_1$	$\exists z.q_z = b_0 \vee \exists z.q_z = b_1$
2	$Pre(\exists z.q_z = b_1) \equiv \exists q' \tau(q, q') \wedge \phi_1(q') \equiv$ $\exists q' \, (\exists z_1.q[z_1] = b_1 \wedge q' = \lambda j.$ $(\text{if } j = z_1 \Rightarrow b_1 \text{ else } q_j) \wedge \exists z_2.q[z_2]' = b_1) \equiv$ $\exists z.q_z = b_1$	$\exists z.q_z = b_0 \vee \exists z.q_z = b_1$ $\vee \exists z.q_z = b_1 \equiv$ $\exists z.q_z = b_0 \vee \exists z.q_z = b_1$

k	$\rho_k \wedge \phi_I$	$\neg(\rho_k \implies \rho_{k-1})$
0	$\exists z.q_z = b_0 \wedge \forall z.q_z = b_2$	undefined
1	$(\exists z.q_z = b_0 \vee \exists z.q_z = b_1) \wedge \forall z.q_z = b_2 \equiv$ unsatisfiable	$(\exists z.q_z = b_0 \wedge \forall z.q_z \neq b_0)$ $\vee(\exists z.q_z = b_1 \wedge \forall z.q_z \neq b_0) \equiv$ satisfiable
2	$(\exists z.q_z = b_0 \vee \exists z.q_z = b_1) \wedge \forall z.q_z = b_2 \equiv$ unsatisfiable	$\neg(\exists z.q_z = b_0 \vee \exists z.q_z = b_1$ $\implies \exists z.q_z = b_0 \vee \exists z.q_z = b_1) \equiv$ unsatisfiable