Modeling and Verification of Multi-Agent Cyber-Physical Systems in Verse: A Case Study

Sarthak Das [1] Rajarshi Ray [1]

[1]School of Mathematical and Computational Sciences,
Indian Association for the Cultivation of Science,
2A & 2B Raja SC Mullick Road, Kolkata-700032, India.

Abstract

Verification tools for cyber-physical systems (CPS) often require expertise in specialized formal modeling languages to define system dynamics and specifications. This poses barriers for wider adoption by users without backgrounds in formal methods. Verse is a recent open-source Python library that aims to make modeling and verification of multi-agent CPS more accessible. Through a case study in robotic swarm synchronization, we demonstrate Verse's capabilities and compare it against conventional CPS modeling approaches.

Our case study focuses on emergent clock synchronization in a swarm of n robots via pulse coupling of LED flashes. Such spontaneous synchronization has applications in distributed control of mobile robot swarms, but prior hybrid automaton models scale exponentially due to state space explosions. We propose an improved model that minimizes the number of locations and transitions while preserving semantics. Our model, though still subject to exponential blowup, scales significantly better than any of the existing models with just one location per agent. Our Verse implementation demonstrates the emergence of synchronization across robots.

In conclusion, Verse provides an accessible Python-based framework for modeling and verification without needing expertise in formal methods. Through extensions like our swarm model, Verse enables large-scale analysis of complex multi-agent CPS. By making such technology more accessible, libraries like Verse can greatly broaden adoption beyond just specialized researchers.

An Outline of Verse

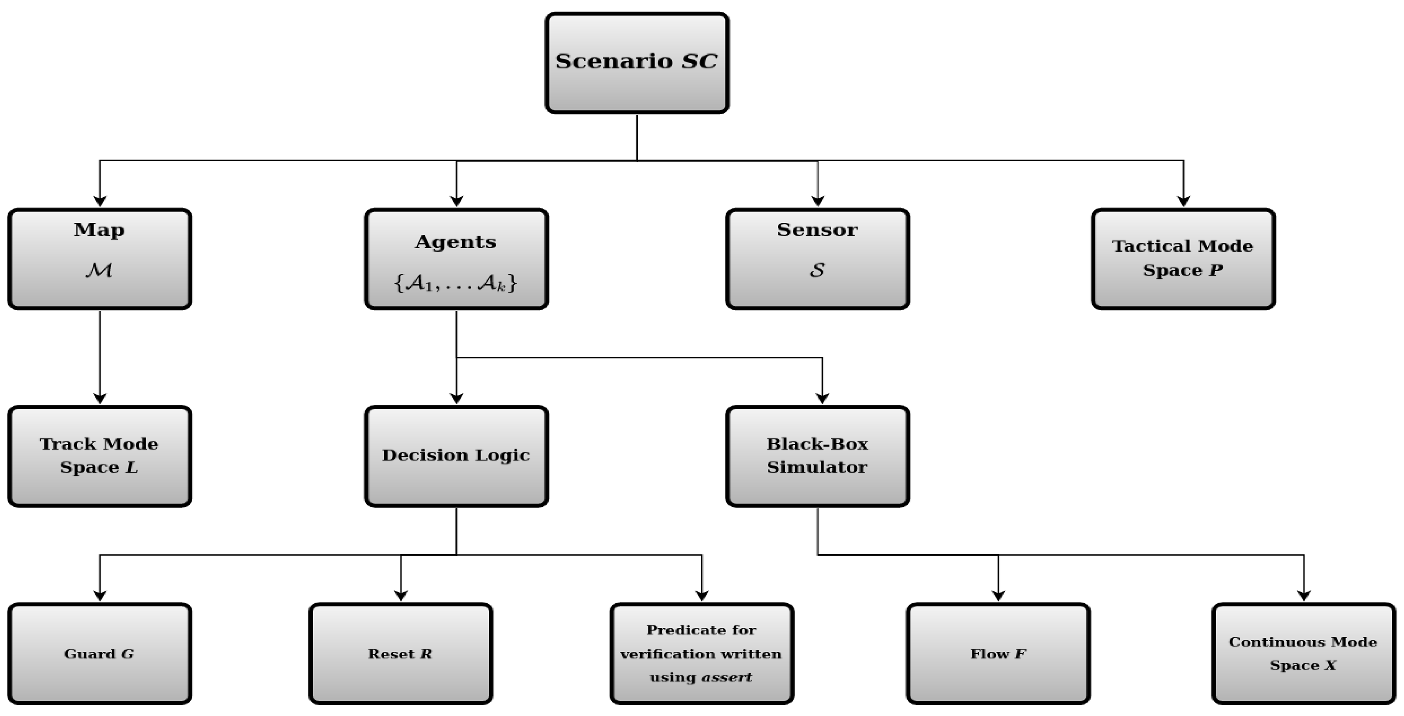


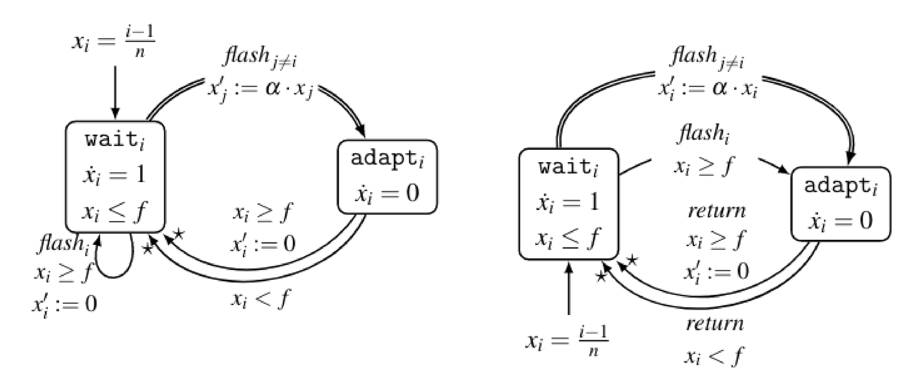
Figure 1. Scenario object $SC = (\mathcal{M}, \{\mathcal{A}_1, \ldots, \mathcal{A}_k\}, \mathcal{S}, P)$ in Verse.

Existing Models

For $i \in \{1, \ldots, n\}$, the ith robot is characterized by a clock x_i with continuous dynamics $\dot{x}_i = 1$, $0 \leq x_i \leq f$ for some firing threshold $f \in \mathbb{R}_{>0}$. When $x_i = f$, robot i flashes its central LED, and x_i is discretely reset to zero. Robots interact by pulse coupling: when robot i flashes, the clocks of all other robots are reset as:

$$x_i = f \implies x_j := \begin{cases} \alpha x_j & \text{if } \alpha x_j < f \\ 0 & \text{if } \alpha x_j \geq f \end{cases}$$

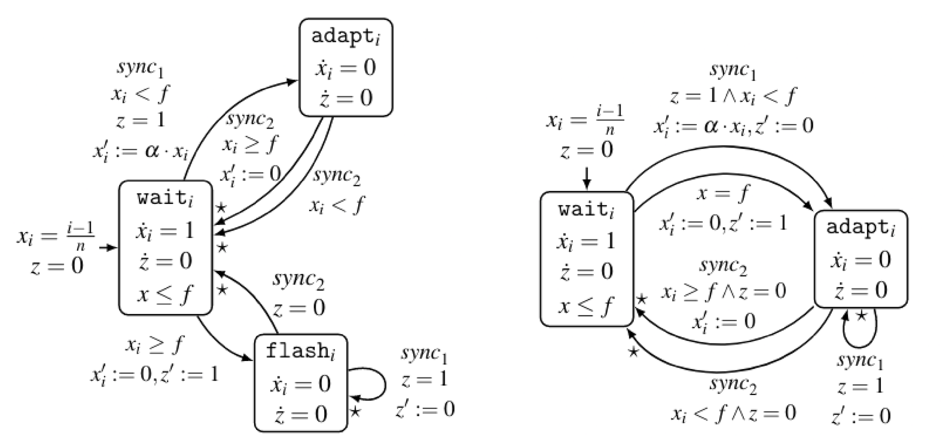
$\forall j \in \{1, \ldots, n\} \setminus \{i\}$ given $\exists \alpha \in \mathbb{R}_{>1}$. Robots are modeled by hybrid automata H_i, and the swarm by $H = H_1 \parallel \ldots \parallel H_n$.



(a) First instance of model using label synchronization (lsync I).

(b) Refined instance using label synchronization with synchronized return to wait (lsync II).

Figure 2. Existing models proposed in [2] using label synchronization.



(a) First model using shared variables (shd I).

(b) Reduced instance using shared variables (shd II).

Figure 3. Existing models proposed in [2] using shared variable z for synchronization.

Our Proposed Model

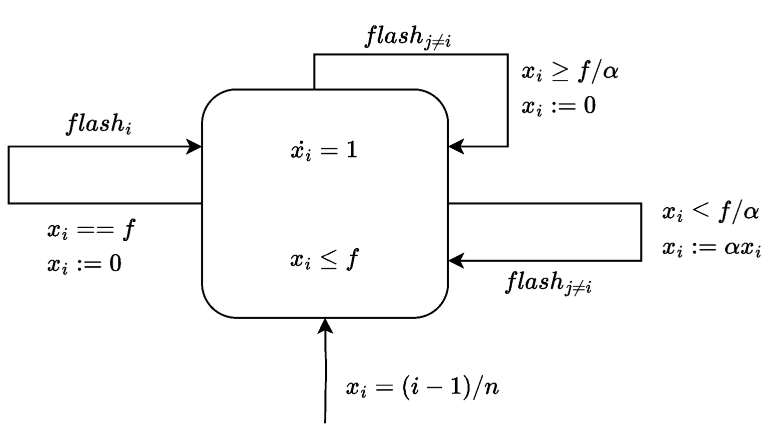


Figure 4. Our proposed model, Isync III, is essentially a minimization of Isync I.

Model	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$
Isync I	2	3	7	15	31	63	127	255
Isync II	2	3	4	5	6	7	8	9
shd I	3	7	15	31	63	127	255	511
shd II	3	4	8	16	32	64	128	256
Isync III	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

Table 1. The number of locations in the parallel composition $H = H_1 \parallel \ldots \parallel H_n$ of the individual agents for varying n, in different models of clock synchronization.

Model	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$
Isync I	3	6	33	164	755	3310	14077	58728
Isync II	3	6	15	36	85	198	455	1032
shd I	6	18	54	162	486	1458	4374	13122
shd II	5	13	35	97	275	793	2315	6817
Isync III	**1**	**4**	**12**	**32**	**80**	**192**	**448**	**1024**

Table 2. The number of transitions in the parallel composition $H = H_1 \parallel \ldots \parallel H_n$ of the individual agents for varying n, in different models of clock synchronization.

Modeling in Verse

We implemented Isync III in Verse. Each automaton in the swarm was modelled as a `RobotAgent`, with a single state variable x, representing its clock. `RobotAgent` only has one tactical mode `Wait` with dynamics $\dot{x} = 1$ (and no track mode as the scenario has no map). By taking advantage of the Verse framework, the modeling was achieved exclusively in a subset of Python. The decision logic of each `RobotAgent` is as follows:

```
def decisionLogic(ego:State, others:List[State]):
    output = copy.deepcopy(ego)
    if ego.x > flash:
        output.x = 0
    if any(other.x >= flash for other in others):
        output.x = alpha * ego.x
        if ego.x >= flash / alpha:
            output.x = 0
    assert not all(other.x == ego.x
    for other in others), "Sync"
    return output
```

Our implementation is available as a demo in the official repository of Verse [2].

We also implemented our model in **SpaceEx** and verified that mutual synchronization is achieved. In Verse, we initialize n RobotAgents by implementing a simple `for` loop, then we can add any number of agents by simply initializing n. On the other hand, in SpaceEx, each automaton has to be manually added using its model editor tool.

```
swarm = Scenario()
for i in range(n):
    myagent = RobotAgent(f"robot_{i+1}",
    file_name="./demo/robot_swarm/Isync.py")
    swarm.add_agent(myagent)
```

Results and Discussion

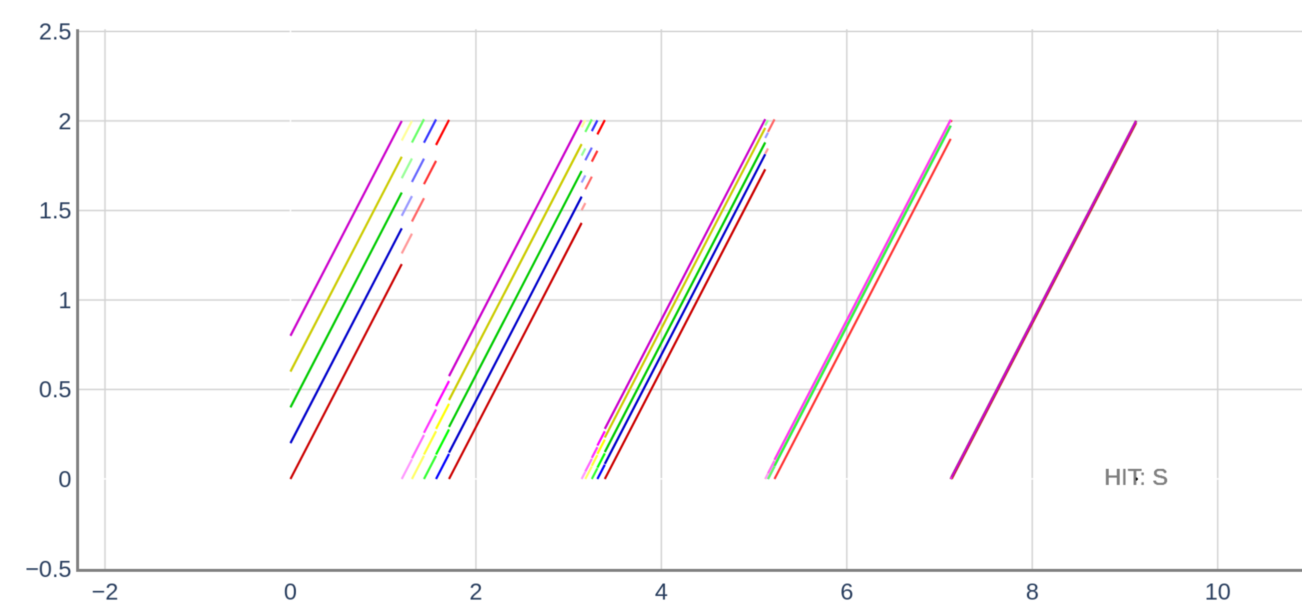


Figure 5. Simulation of Isync III with $n = 5$, $f = 2$, and $\alpha = 1.05$, using the Verse library. The time horizon is 10, the time step is 0.01. The system is deterministic with a fixed initial point, hence no need for running the verification routine. The clock values x_i are plotted against time t. With each flash (denoted by the discrete jumps in the trajectories), the clock values of the robots come closer and eventually synchronize at $t = 9.12$.

References

[1] Yangge Li, Haoqing Zhu, Katherine Braught, Keyi Shen, and Sayan Mitra. Verse: A python library for reasoning about multi-agent hybrid system scenarios. In *Computer Aided Verification: 35th International Conference, CAV 2023, Paris, France, July 17–22, 2023, Proceedings, Part I*, page 351–364, Berlin, Heidelberg, 2023. Springer-Verlag.

[2] Stefan Schupp, Francesco Leofante, Leander Behr, Erika Ábrahám, and Armando Taccella. Robot swarms as hybrid systems: Modelling and verification. *Electronic Proceedings in Theoretical Computer Science*, 361:61–77, July 2022.

Acknowledgements

The authors acknowledge Prof. Sayan Mitra and Yangge Li from the University of Illinois at Urbana Champaign for their invaluable input. The authors would also like to thank Mir Md. Sajid Sarwar and Atanu Kundu, both Senior Research Fellow at the Indian Association for the Cultivation of Science for their support and feedback.

Appendix

Mathematical Proof for the Number of Transitions in the Parallel Composition of Isync III

Consider one of the n robots, say robot i in the swarm to take a $flash_i$ transition, which is a synchronizing transition. The structure of the automata are such that the guards of the two $flash_{j \neq i}$ transitions are complementary ($x \geq f/\alpha$ and $x < f/\alpha$) for all i, j.

Therefore, when robot i takes its $flash_i$ transition, each of the remaining $n-1$ robots has to take exactly one of the two $flash_i$ transitions defined in their automata, resulting in a total of 2^{n-1} possible combinations. Each combination corresponds to a transition in the parallel composition with the synchronization label $flash_i$.

Since there are n such possible $flash_i$ labels, the number of transitions in the parallel composition of Isync III satisfies no. of trans = $n.2^{n-1}$.

ug18sd@iacs.res.in, rajarshi.ray@iacs.res.in