ISC 2024, Arlington, VA

Property Guided Secure Configuration Space Search

You Li*[a], Kaiyu Hou*[b], **Yunqi He**[a], Yan Chen[a], and Hai Zhou[a]

[a]Northwestern University

[b]Alibaba Cloud**

*Equal Contribution

**Any opinions, recommendations, or findings are those of the authors and do not reflect the views of Alibaba Cloud.

Northwestern University

Alibaba Cloud

Outline

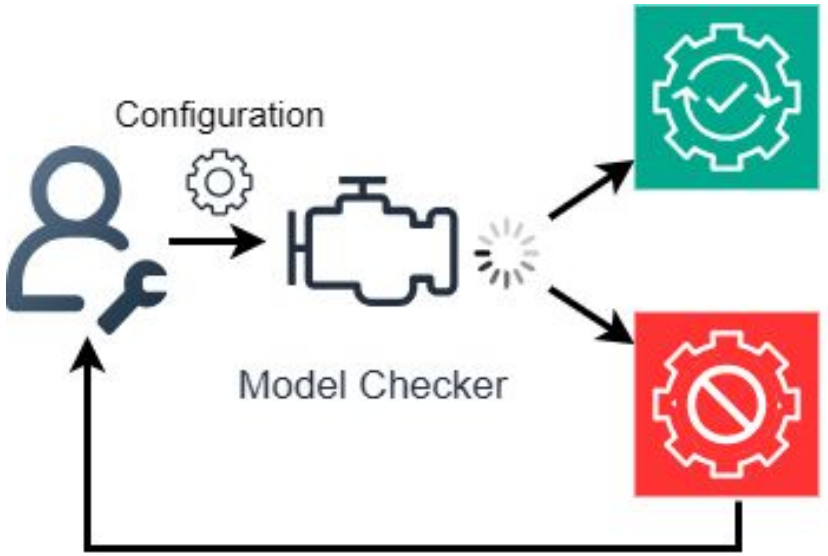
- **Motivations and problem definition**
  - The importance of configuration verification and drawbacks in current methods
  - Leverage “secure configuration space” to enable real-time verification
- **Technical contributions**
  - Convert configuration verification into a model checking problem
  - Reduce the decision problem into a search problem
  - Heuristics to improve search coverage
- **Evaluations** on testbenches and network protocols

Motivation: Configuration Verification

- **Network configurations** are constantly changing to:
  - support new services, accommodate new devices, enhance network security, etc.
- **Errors in configuration changes** can cause catastrophic network outages:
  - 56% of network incidents due to configuration changes in a leading cloud provider
- **Network configuration verification** checks before deployment whether a new configuration satisfies a set of desired correctness properties:
  - reachability, way-pointing, loop-freedom…

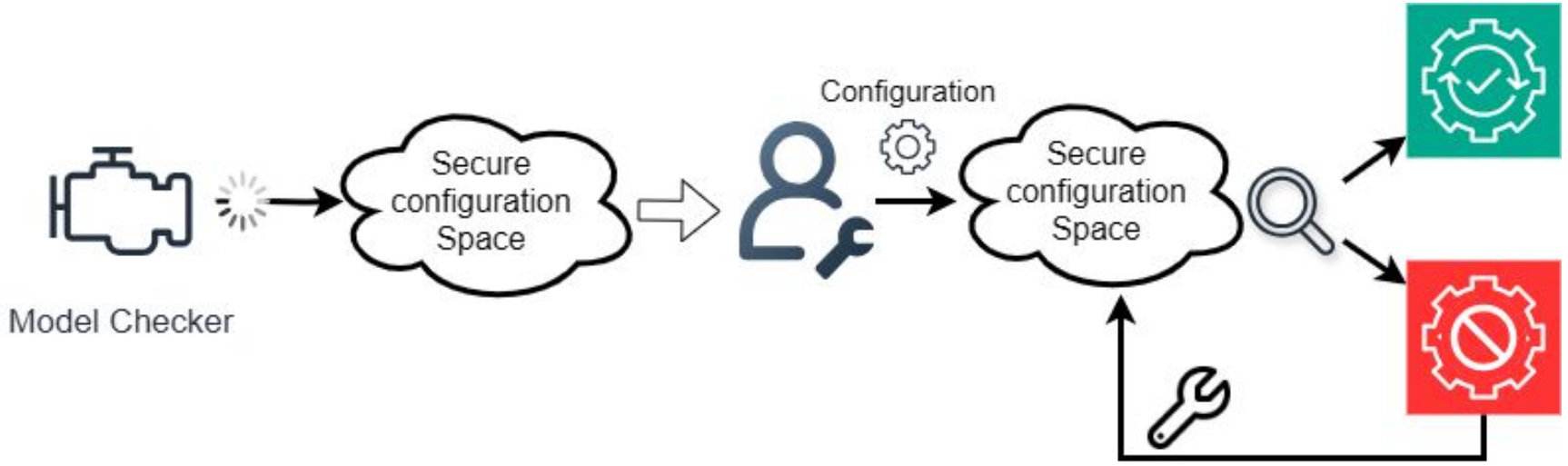
Drawbacks of Existing Methods

- Network verification is **time consuming:**
  - Verification out of time (24h) for Purdue topology (2159 nodes & 3,607 links)
- **Real-time network verification** should be agile:
  - Check the correctness of configurations as they are deployed to real networks.



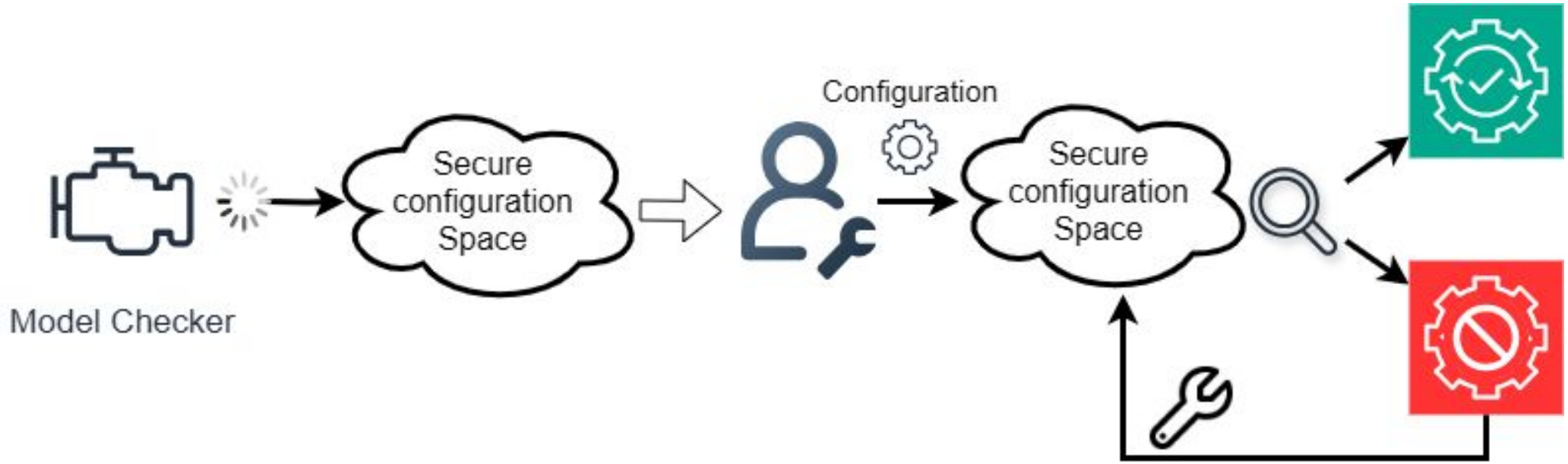
Secure Configuration Space

- We aim to find a **secure configuration space**: any configurations within the space are guaranteed to be secure.
- It reduces the original *online decision problem* into *i) an offline search problem* and *ii) an online SAT query*.
- Additionally, it can suggest a *correct alternative configuration* that is as close as possible to the previous insecure configuration.

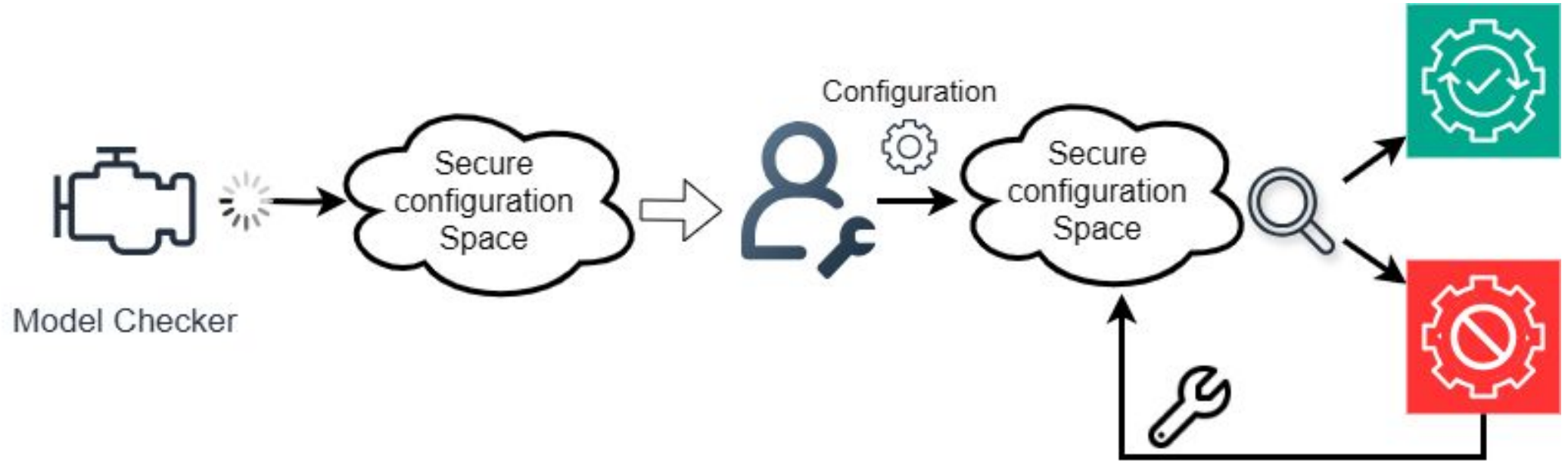


Secure Configuration Space

- We aim to find a **secure configuration space**: any configurations within the space are guaranteed to be secure.

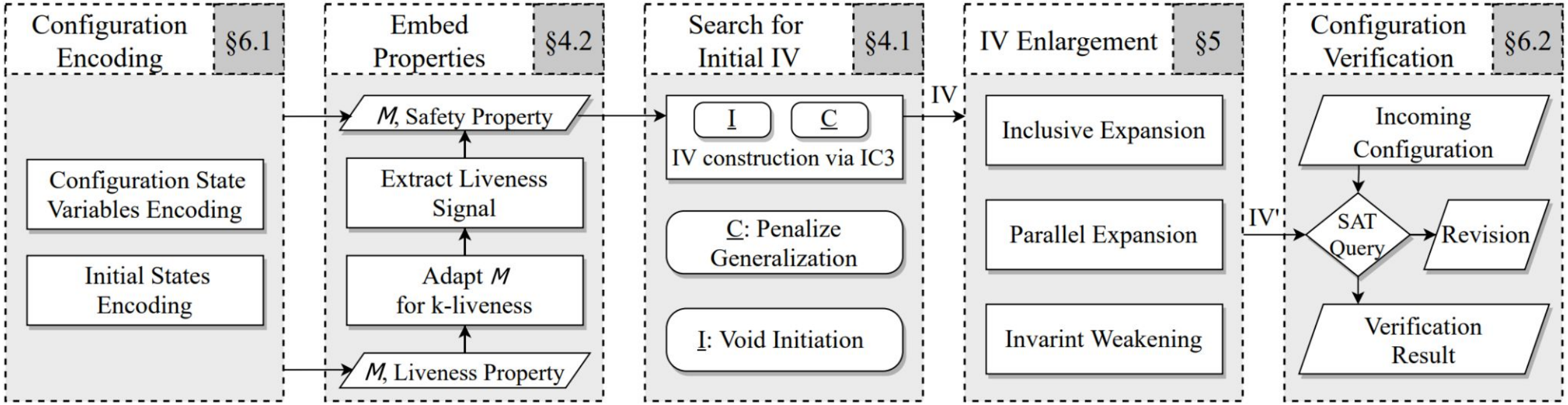


Advantages of Secure Configuration Space



- It reduces the original *online decision problem* into *i) an offline search problem* and *ii) an online SAT query*.
- Additionally, it can suggest a *correct alternative configuration* that is as close as possible to the previous insecure configuration.

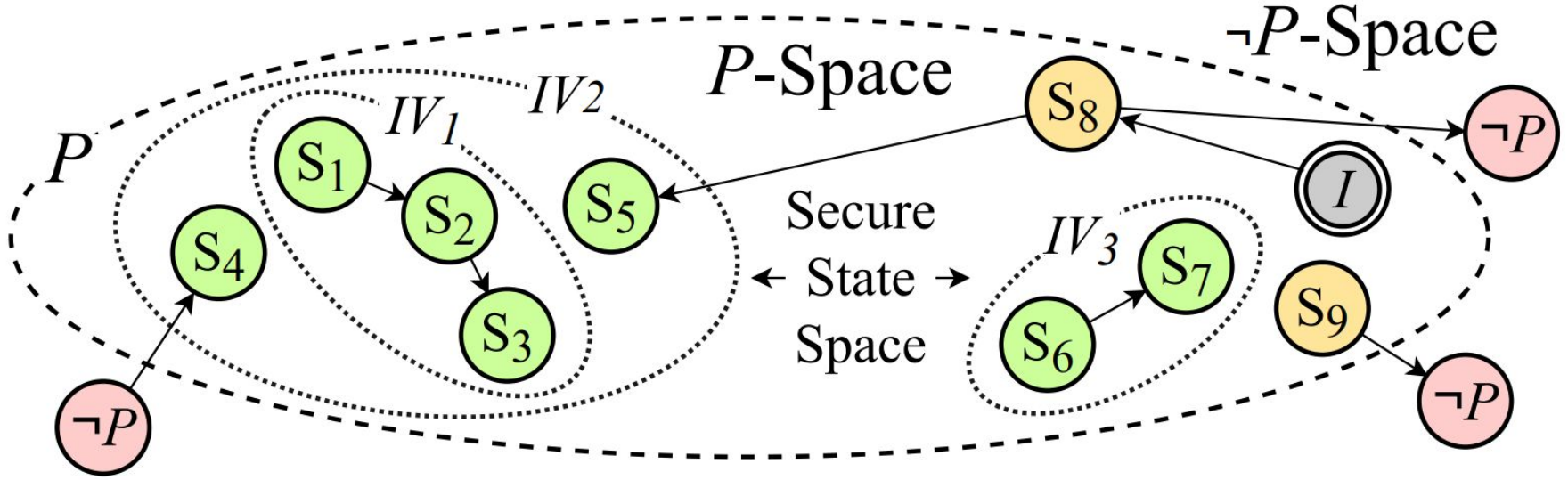
Technical Contribution: Framework



Encode Configuration into State Space

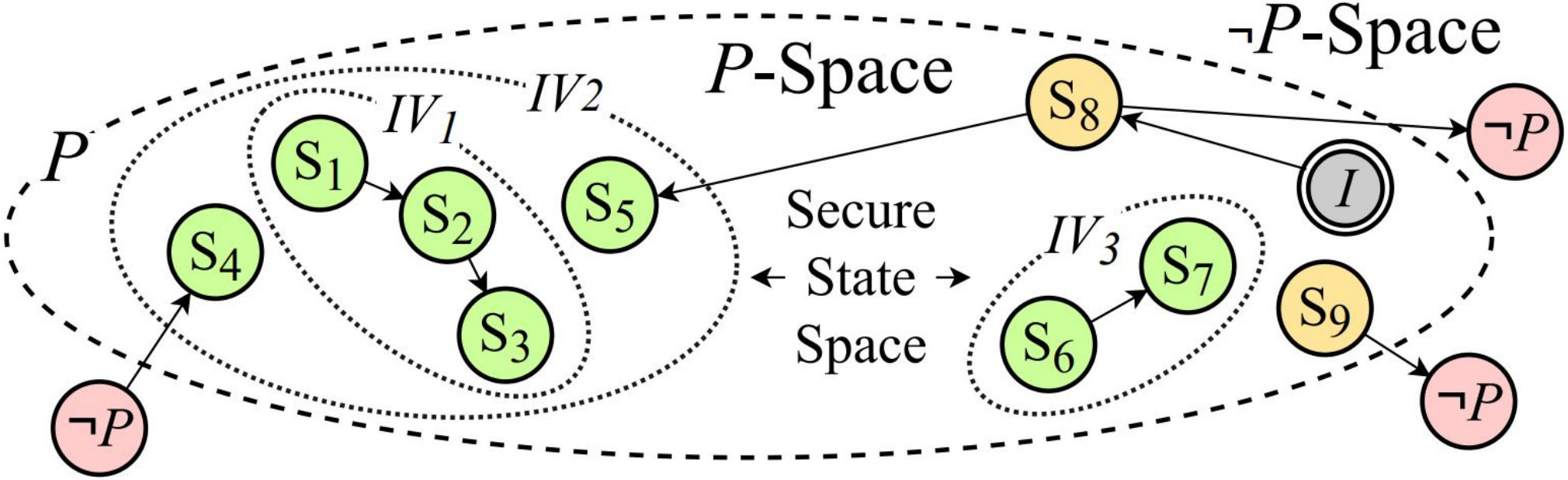
- We add *configuration state variables* alongside the normal state variables to implicitly encode configurations: $\overline{x} = \overline{x}_{conf} + \overline{x}_s$
  - The *configuration state variables* $\overline{x}_{conf}$ are unconstrained at initialization, and their values remain unchanged during the execution.
  - In contrast, the *normal state variables* $\overline{x}_s$ are constrained to the reset state at initialization, and their values can evolve as usual during the execution.

Enlargement of Secure Configuration Space



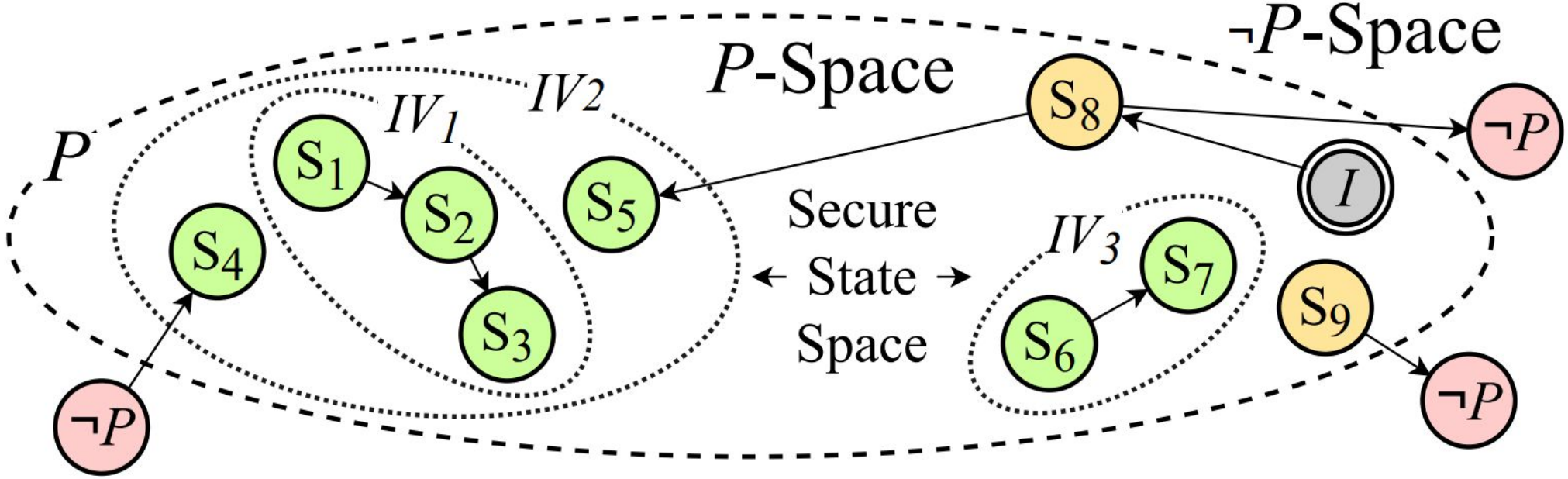
- **Inclusive Expansion**: Suppose *IV1* is the initial secure state space, we set *IV1* as the initiation constraint and expands the secure state space to *IV2*
- **Parallel Expansion:** we set *S6* as the initiation constraint and start another search, thus yields a larger secure state space *IV2* ∪ *IV3*

Enlargement of Secure Configuration Space



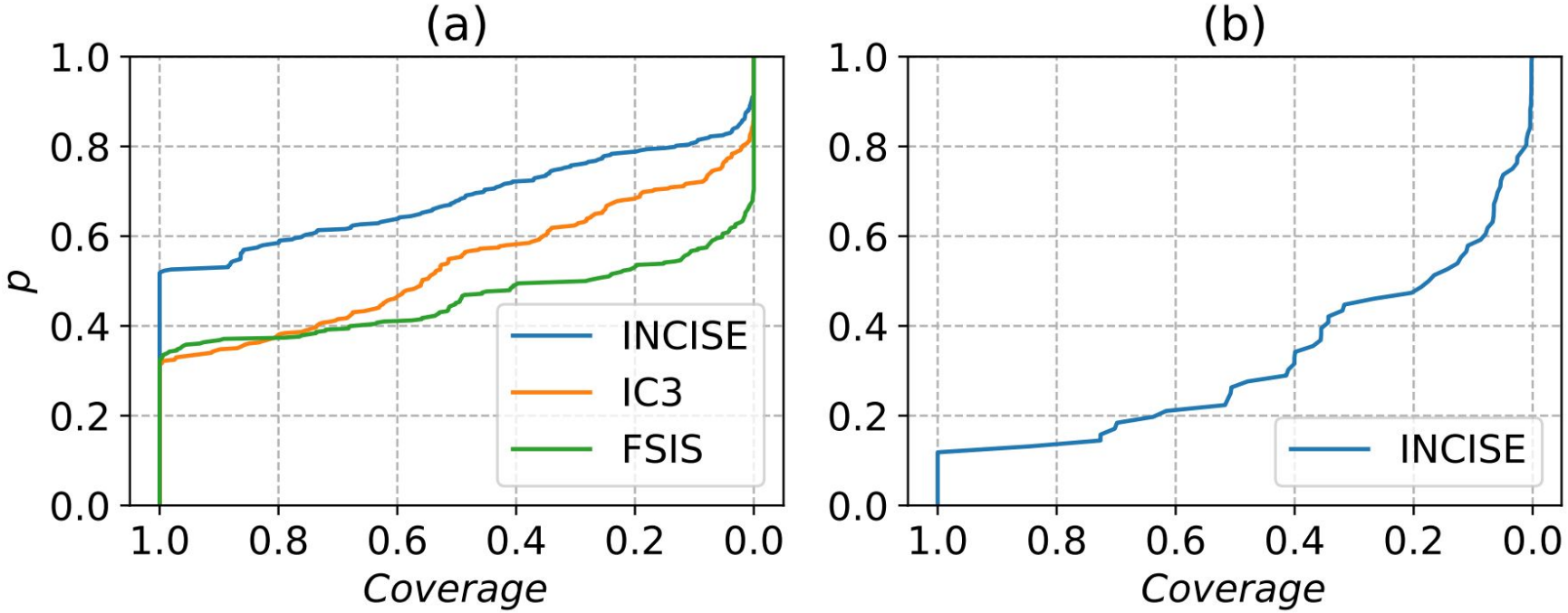
- **Inclusive Expansion**: Suppose *IV1* is the initial secure state space, we set *IV1* as the initiation constraint and expands the secure state space to *IV2*

Enlargement of Secure Configuration Space



- **Parallel Expansion:** we set *S6* as the initiation constraint and start another search, thus yields a larger secure state space *IV2* ∪ *IV3*

Evaluation: Safety Properties



(a) Safe benchmarks: our algorithm (INCISE) achieves better coverage, e.g., larger safe configuration space than others on safe benchmarks.

(b) Unsafe benchmarks: INCISE still works on unsafe benchmarks, while the other algorithms cannot find a valid invariant.

Case Study: Cellular Emergency Protocols

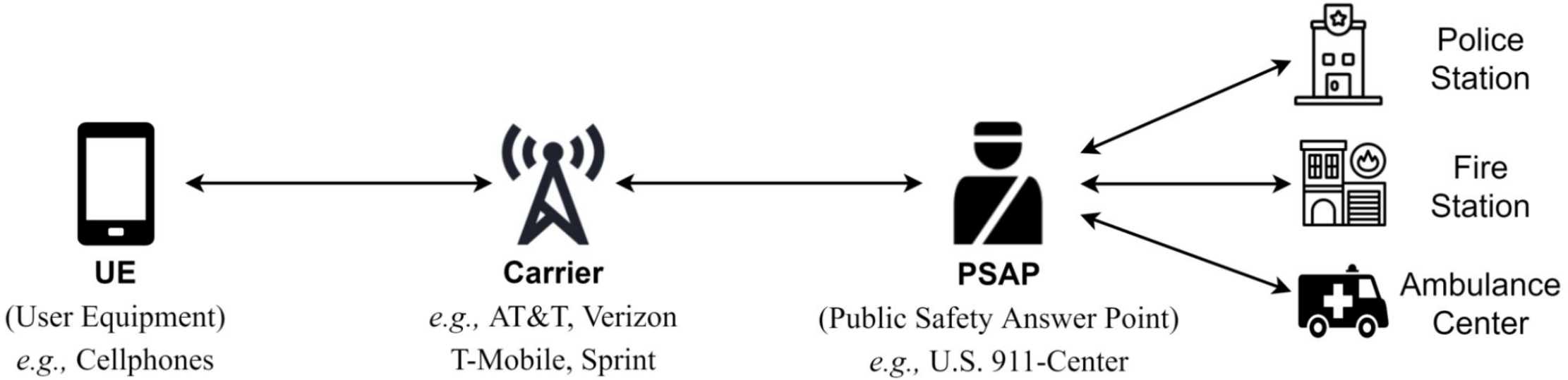


Table 2. Secure configuration space search results for the cellular emergency call system.

Total #Secure	Algorithm	Time(s)	Found #Secure	Coverage
518	INCISE (Basic)	0.036	367	70.8%±3.9%
	w/Inclusive Expansion	0.069	518	100%±0%
	w/Parallel Expansion	0.068	518	100%±0%

Thank you!

You Li*[a], Kaiyu Hou*[b], **Yunqi He**[a], Yan Chen[a], and Hai Zhou[a]

[a]Northwestern University

[b]Alibaba Cloud**