Machine Learning Models to Enhance the Science of Cognitive Autonomy

Ganapathy Mani, Bharat Bhargava, Pelin Angin, Miguel Villarreal-Vasquez, Denis Ulybyshev, Jason Kobes*

THE VALUE OF PERFORMANCE.

NORTHROP GRUMMAN

CS & CERIAS, Purdue University
***Northrop Grumman Corporation**

AIKE 2018

Intelligent Autonomous Systems

- Autonomous Systems should be
  - Able to perform complex tasks without or with limited ongoing connection to humans.
  - Cognitive enough to act without a human's judgment lapses or execution inadequacies.

- Intelligent Autonomous Systems (IAS) are characterized as highly **Cognitive**, effective in **Knowledge Discovery**, **Reflexive**, and **Trusted**.

- The focus of this research will be on the smart cyber systems.

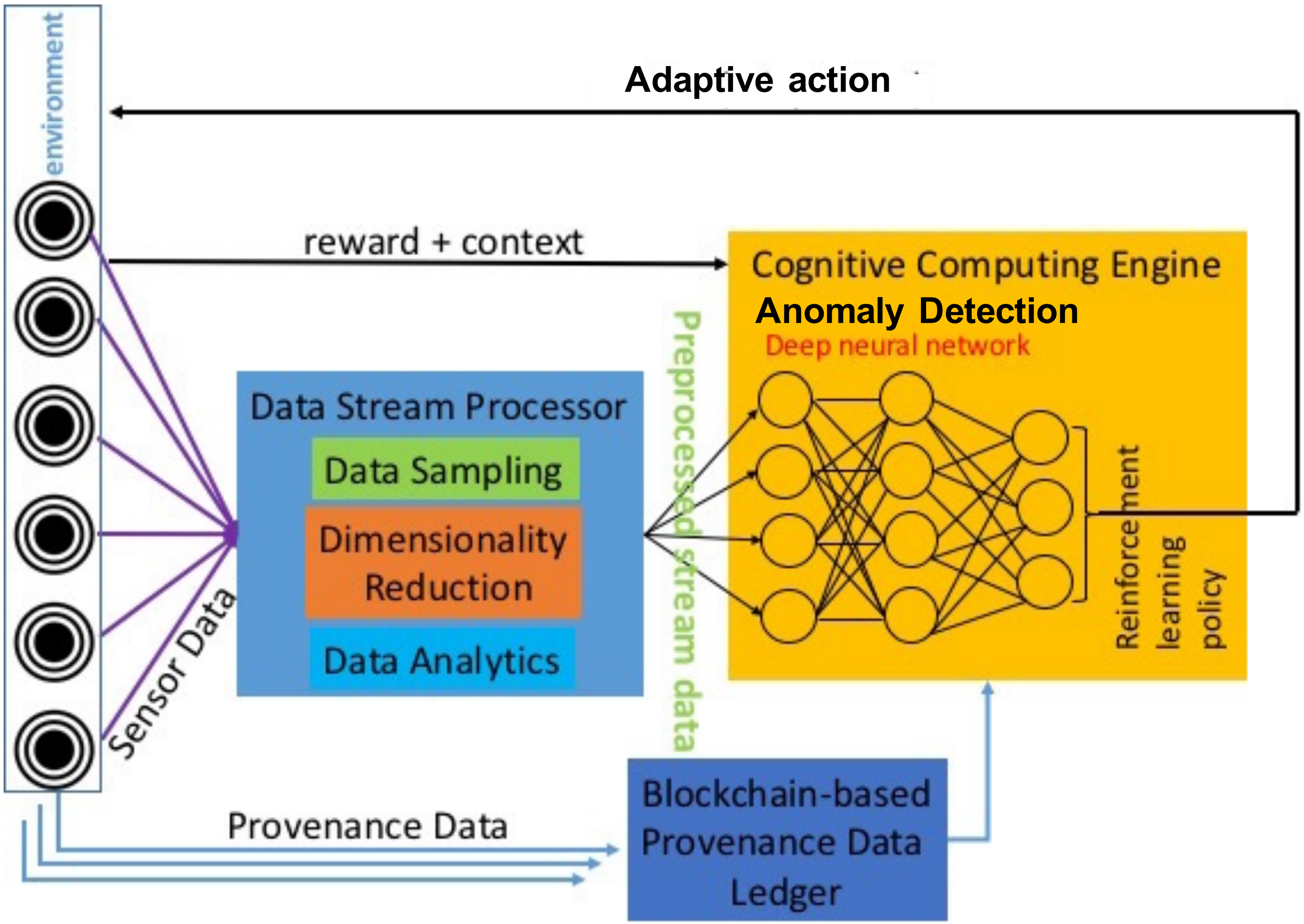
Motivation – A Holistic Approach

- Autonomous systems should learn at the network level as well as about their environment and context.

- Autonomous systems should be trained to work with
  - Meta-data, limited data, incomplete data, and unknown (new) data
  - Dynamic, unpredictable, and adversarial environment

- In this presentation, we will present theoretical framework and our implementation details.

Comprehensive IAS Architecture



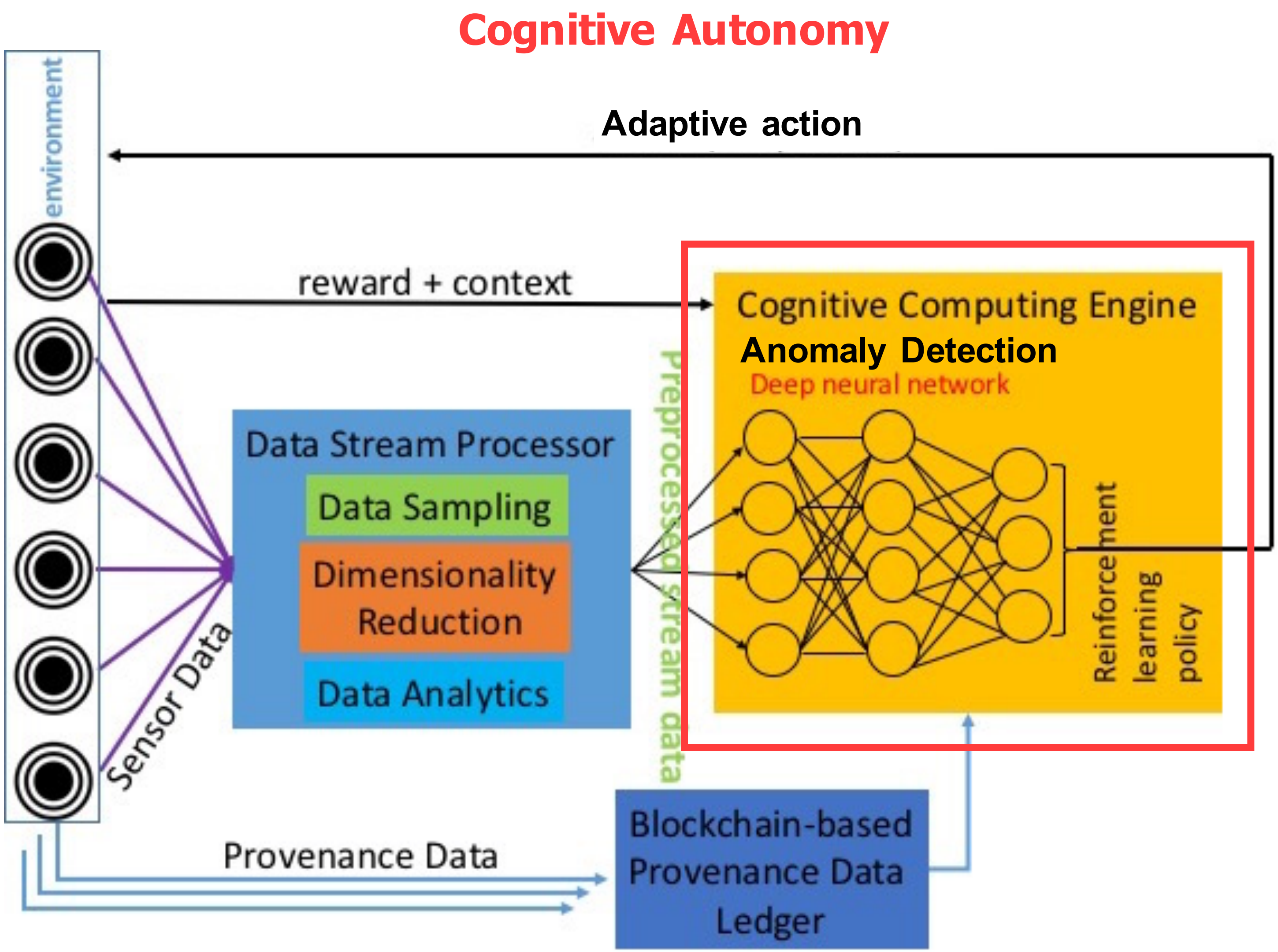
Implementation of Components of IAS

- **Cognitive Autonomy** & **Knowledge Discovery:**
  - Monitors and records system's activities (Data provenance and sequence of system calls)
  - Conducts privacy-preserving aggregated analytics on provenance data.
  - Utilizes Deep learning based anomaly detection by analyzing sequence of system calls.
- **Reflexivity:**
  - Adaptive actions are performed through graceful degradations without disrupting the ongoing critical processes by incremental learning.
- **Trust:**
  - Uses blockchain to store provenance data for trust.

Cognitive Autonomy

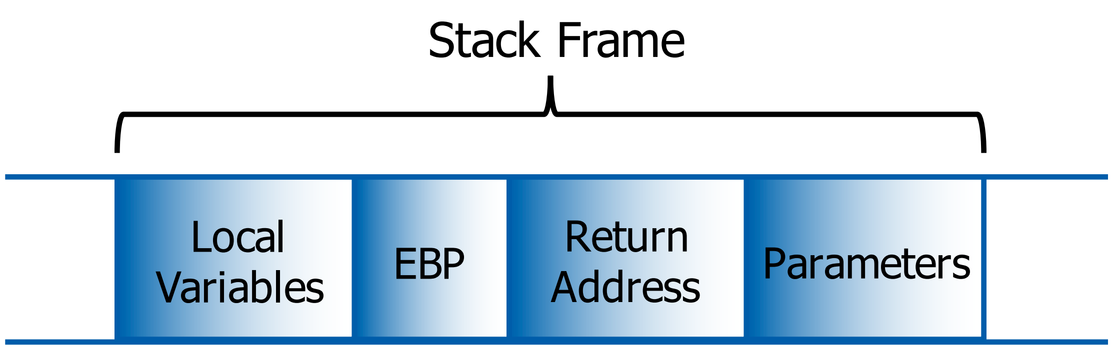
A Deep Learning Based Anomaly Detection Solution

Comprehensive Architecture of IAS



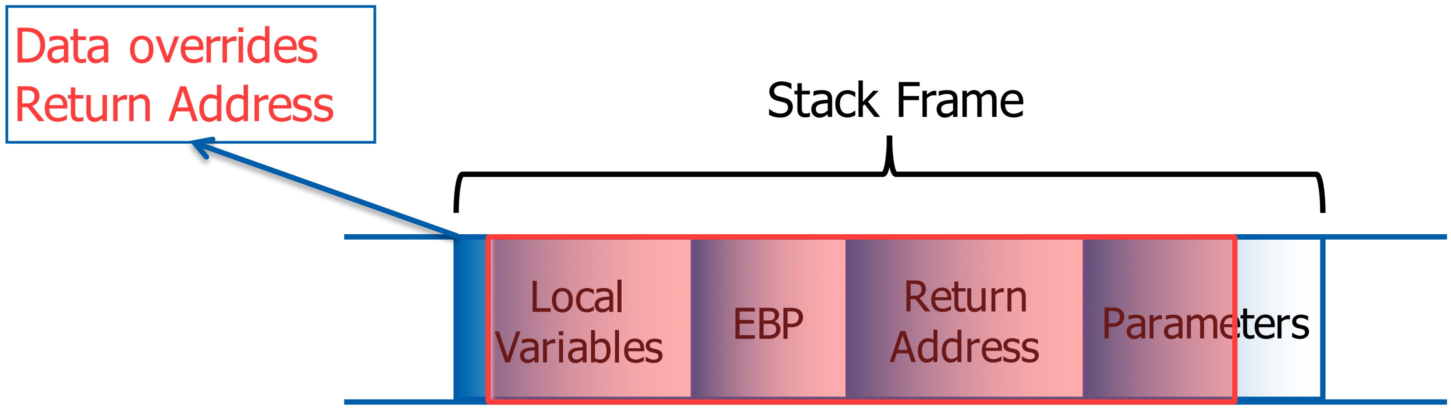
Problem Statement

- Programs store Return Addresses (control flow) along with data in the stack.
- Control-hijacking attacks execute arbitrary code on the target IAS program by hijacking its control flow.
- A Deep Learning based anomaly detection technique has been developed to protect IAS programs against these attacks.



Problem Statement

- Programs store Return Addresses (control flow) along with data in the stack.
- Control-hijacking attacks execute arbitrary code on the target IAS program by hijacking its control flow.
- A Deep Learning based anomaly detection technique has been developed to protect IAS programs against these attacks.

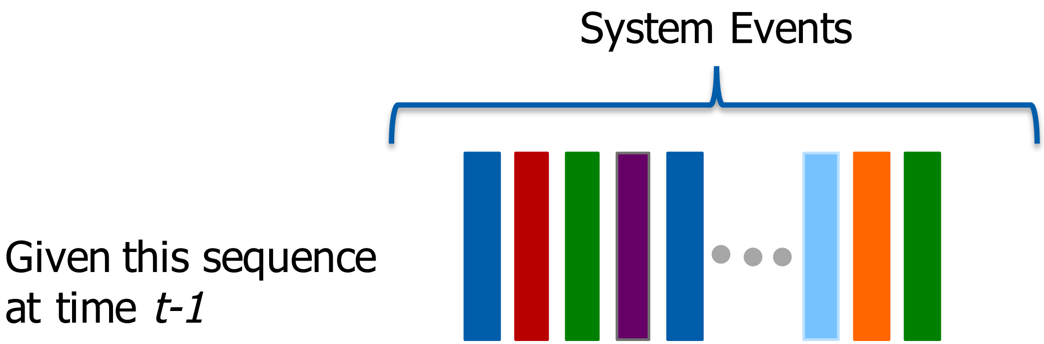


Research Approach

- An event e_i *is* defined as a function call (**system or library call**) in the execution trace of a program.
- Use Deep Learning to answer **the binary classification problem** of given a sequence of function calls (or system events) $e_1 e_2 e_3 \ldots e_k$ **whether or not the sequence should occur**?

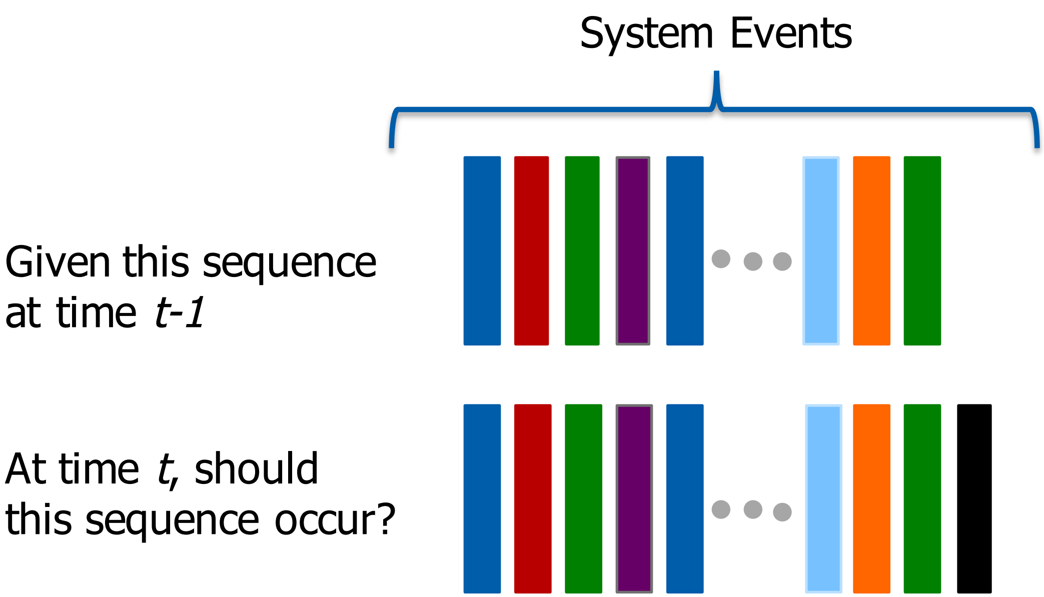
Research Approach

- An event e_i *is* defined as a function call (**system or library call**) in the execution trace of a program.
- Use Deep Learning to answer **the binary classification problem** of given a sequence of function calls (or system events) $e_1e_2e_3\ldots e_k$ **whether or not the sequence should occur**?



Research Approach

- An event e_i *is* defined as a function call (**system or library call**) in the execution trace of a program.
- Use Deep Learning to answer **the binary classification problem** of given a sequence of function calls (or system events) $e_1e_2e_3 \ldots e_k$ **whether or not the sequence should occur**?



Types of attacks and mitigation

Attacks:

- **Code injection:** Malicious instruction sequences are executed using injected codes in the data portion of the stack. Examples: Buffer overflow and buffer specified injection.
- **Code reuse:** Malicious instruction sequences are executed without injecting external code. Examples: Return-oriented programming and memory disclosure.

Mitigation:

- Control Flow Integrity (CFI) is required.
- Deep Learning is used to guarantee Control Flow Integrity (CFI) as the model detects non-conforming sequences of execution traces in run time.

Deep Learning Based Anomaly Detection

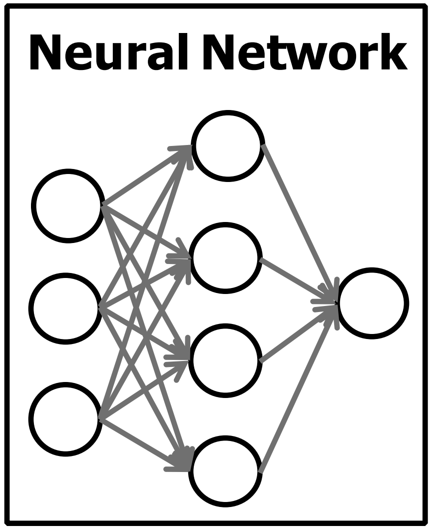
- For a given program, a code coverage is conducted to obtain all the possible execution traces.
- An event e_i *is defined* defined as a function call (**system or library call**) in the execution trace of a program.
- Each possible system event (function calls) is uniquely identified as they will form the vocabulary of system events.
- The Deep Learning model (neural network) is trained with the obtained sequences of events.
- The model is based on Recurrent Neural Networks: Long-Short Term Memory (**LSTM**) and Gated Recurrent Units (**GRU**.)

Deep Learning Based Anomaly Detection

- After training, given a sequence of events as input, the neural network produces as output an array of probabilities, one for each of the possible events in the system.
- At any time *t* each possible event (system call or library call) in the system is assigned a probability estimated with respect to the sequences of events **observed until** time *t-1*.
- At classification time *t*, the decision is made with respect to a pre-defined threshold of the top-*k* most likely events.

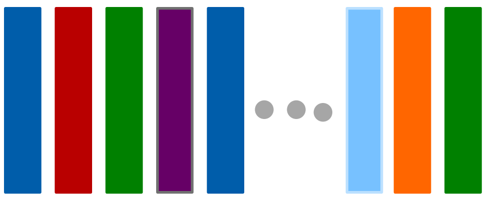
Deep Learning Based Anomaly Detection

Set of all system events

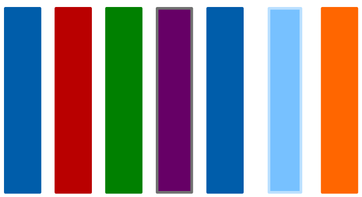


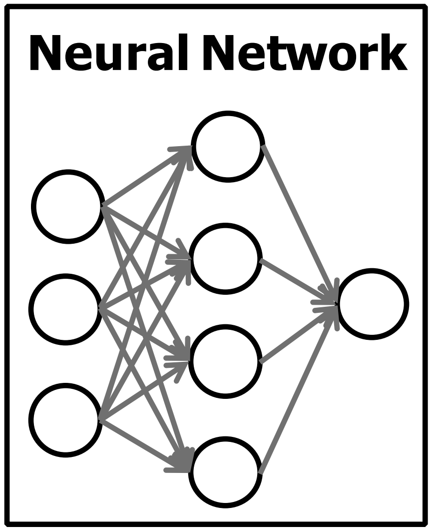
Deep Learning Based Anomaly Detection

Sequence of system events
at time *t-1*

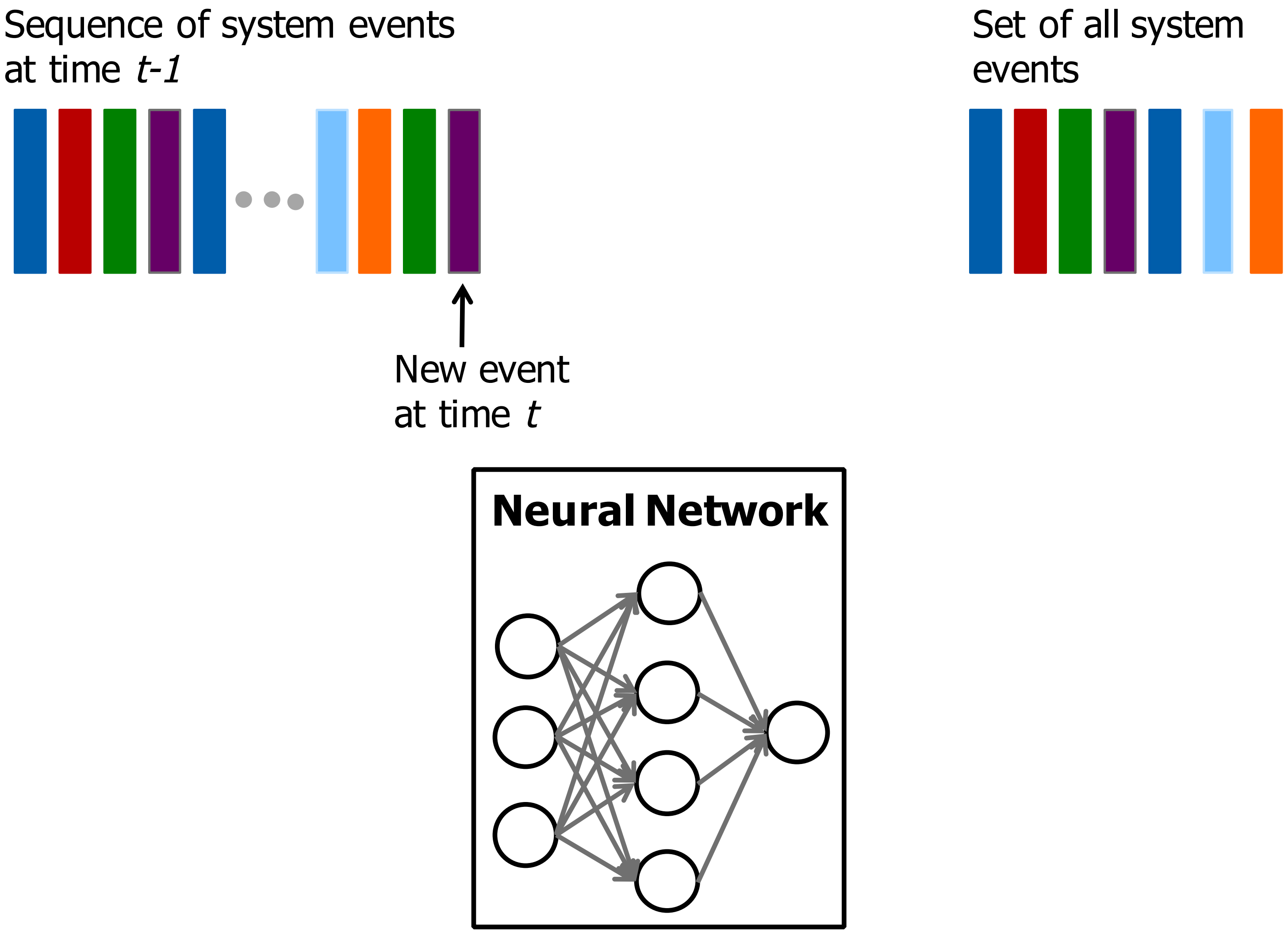


Set of all system
events

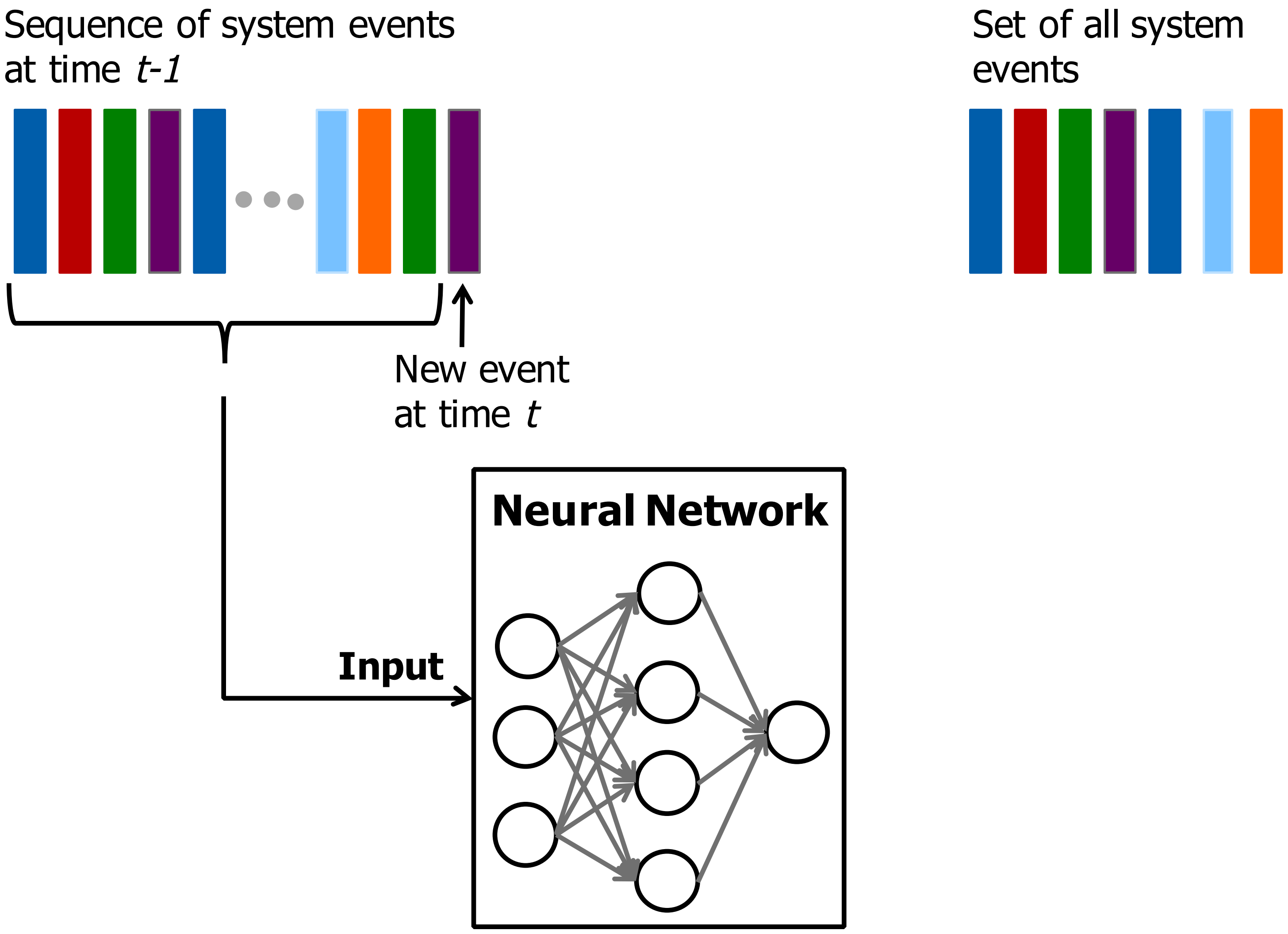




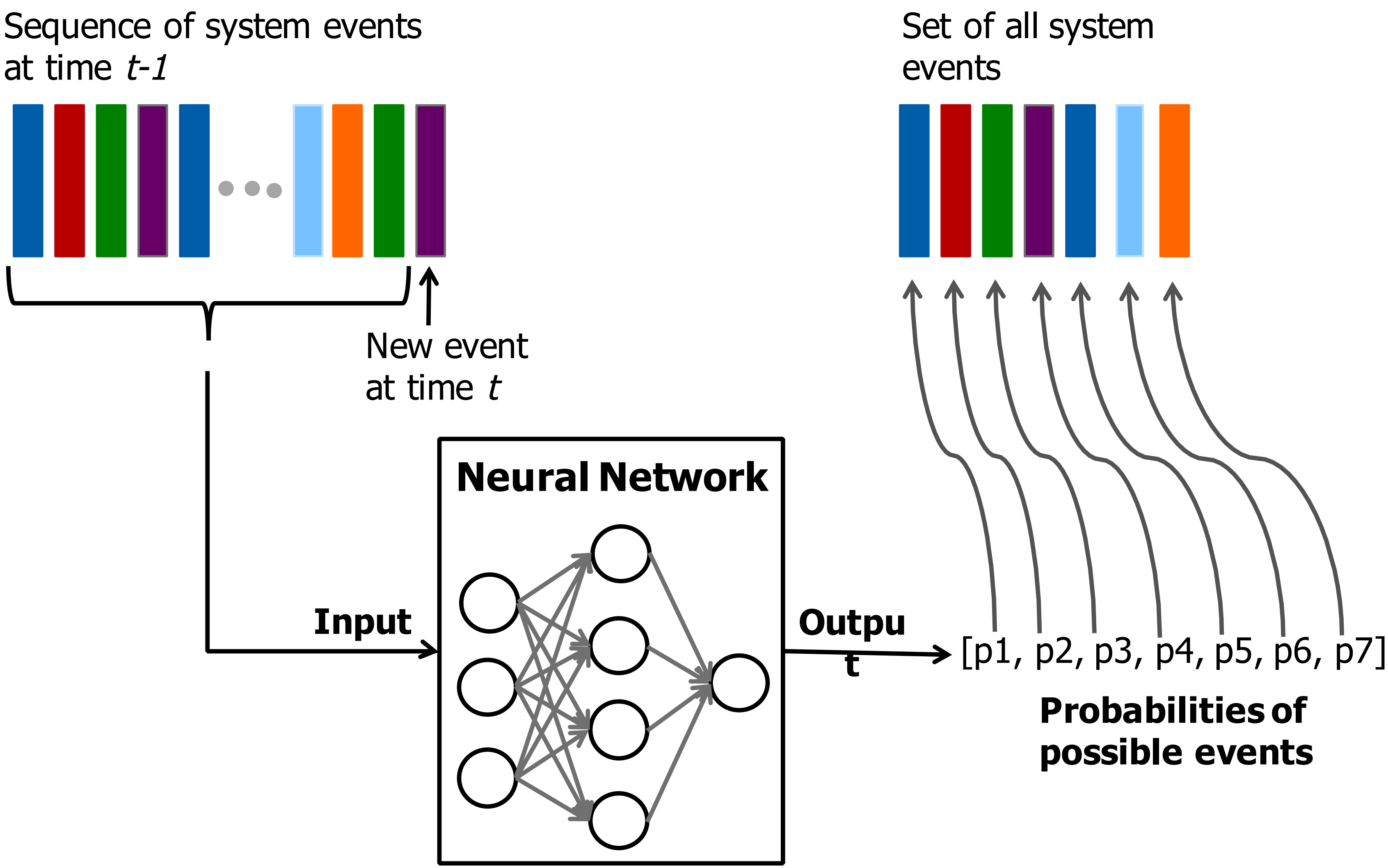
Deep Learning Based Anomaly Detection



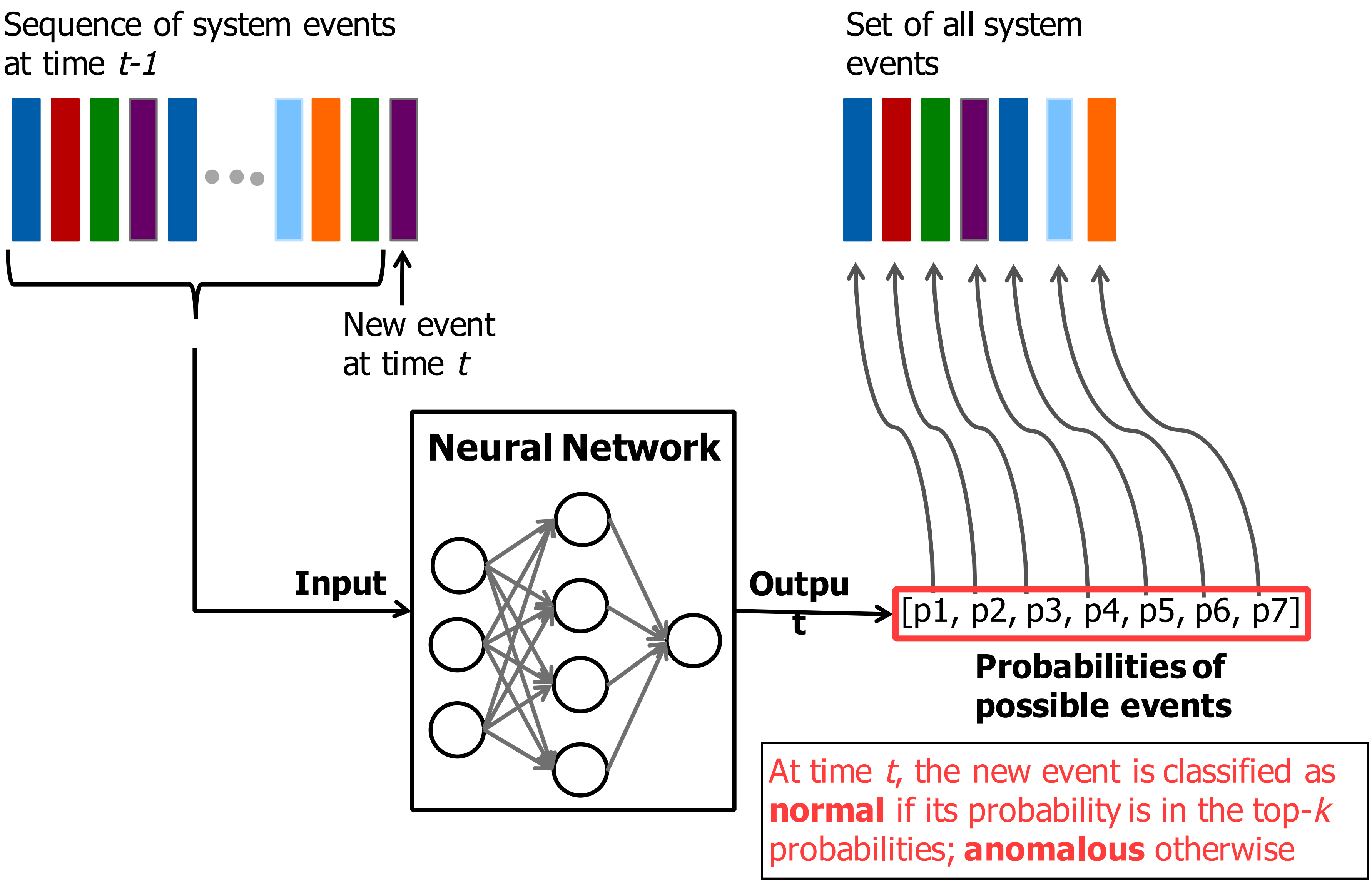
Deep Learning Based Anomaly Detection



Deep Learning Based Anomaly Detection



Deep Learning Based Anomaly Detection



Deep Learning Based Anomaly Detection

Input: Sequence of events in the system
Output: normal or anomalous

- **Step 1:** Define a finite set E of events $e_1, e_2, ..., e_N$ in the system. Events occur in a time-series fashion.
- **Step 2:** At time $t-1$, given an observed series of events $\{e_i{}^1, e_i{}^2, ..., e_i{}^{t-1}\}$ (with $i = 1, 2, ...orN$) find the set K of the top k events to occur in time t.
- **Step 3:** At time t, the sequence $\{e_i{}^1, e_i{}^2, ..., e_i{}^{t-1}, e_i{}^t\}$ is non-anomalous if $e_i{}^t \in K$, otherwise anomalous.

Algorithm 1: Anomaly detection algorithm

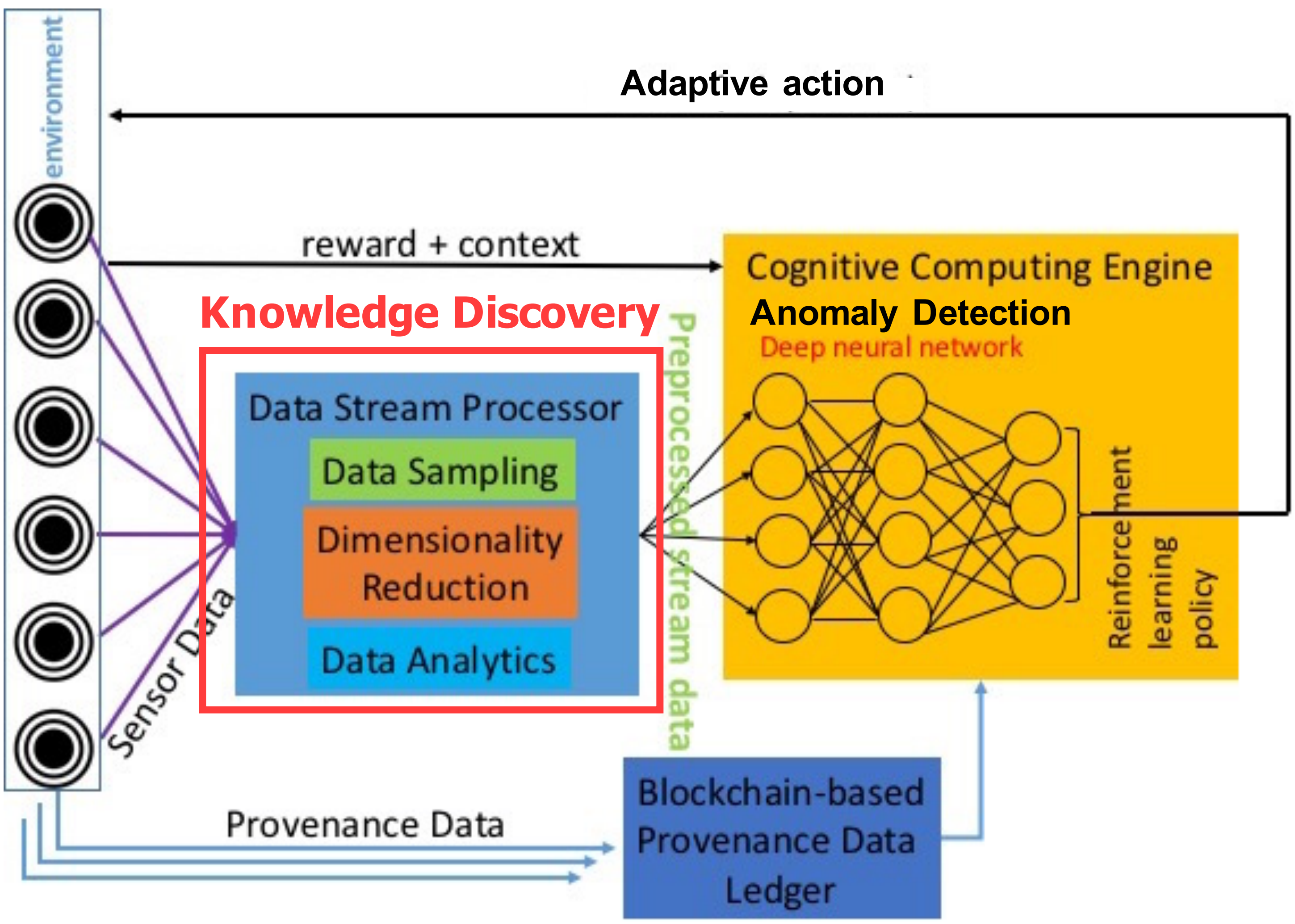
Other Deep Learning Related Projects

- **User and Entity Behavior Analytics (UEBA)**:
  - Process of obtaining the baseline of user activity and behavior to detect potential intrusions and protect from insider threats.
  - Traffic patterns of users would represent the sequences to learn.
- **Network Intrusion Detection Systems (NIDS)**:
  - The application of the DL approach is straightforward.
  - Network packets would represent the set of events to monitor in the system.

Knowledge Discovery

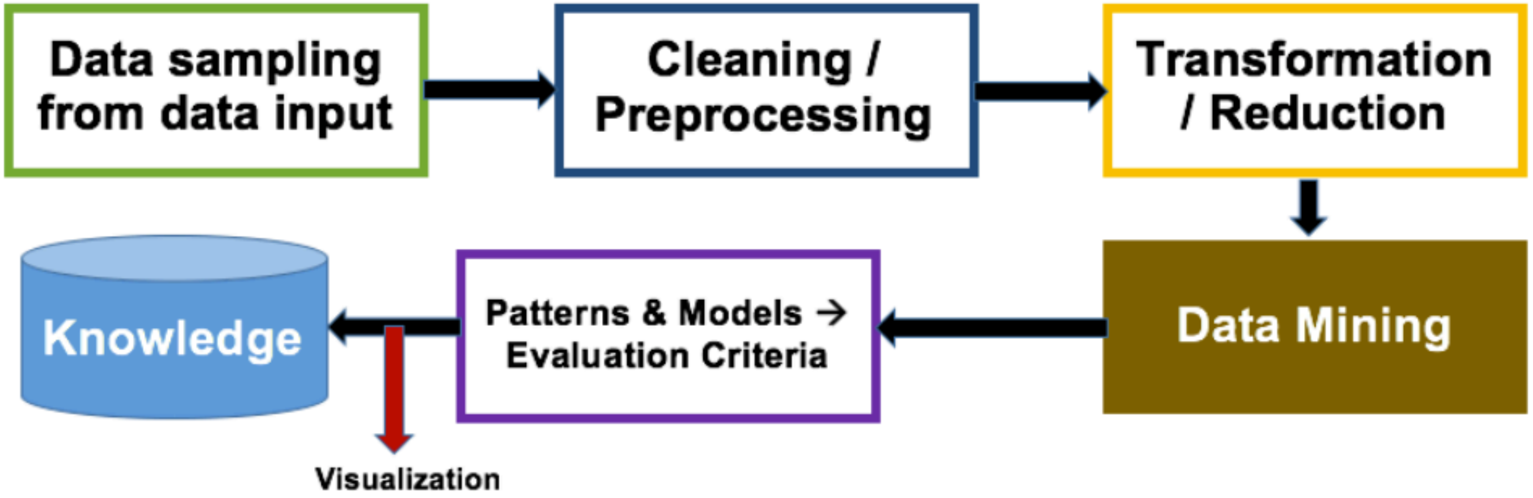
Solutions Based on Pattern Recognition

Comprehensive Architecture of IAS



Knowledge Discovery in IAS

- Knowledge discovery constitutes data transformation for processing, dimensionality reduction, and feature selection, which leads to pattern recognition and visualization.



Knowledge Discovery By Light-weight ML Algorithms

- Compared to deep learning methodologies, pattern recognition through feature extraction is one of the cost effective methodologies.

- Based on the best feature selection approach, light-weight machine learning algorithms such as Support Vector Machine (SVM), k-means, Random Forests, and K-Nearest Neighbors (KNN) can be very efficient.

- Features can be selected through Filter methods (scoring each feature), Wrapper methods (set of features as a search problem), or embedded methods (learning features on-the-fly).

Knowledge Discovery – Inference Models

- Hidden Markov Models (HMM) can be used to infer the probability of observed sequences, probability of latent variables, and statistical significance.

- Models such such as these cannot handle large sequences of data but for limited data, HMMs are better preforming than deep learning methodologies.

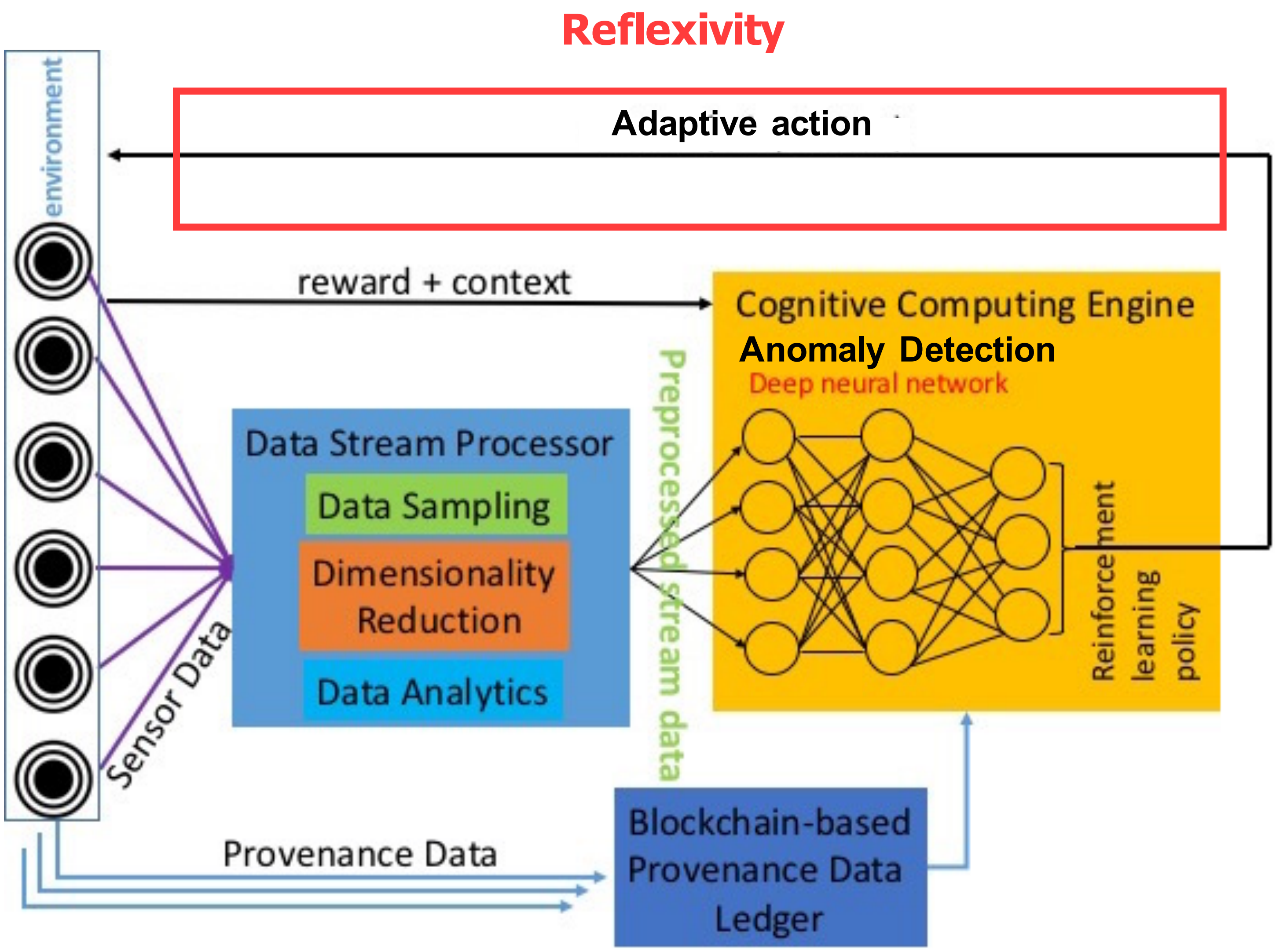
- Similarly, Bayesian inference functions as the probability update function as the new data (or context) comes to light.

- In our reflexivity module, we used Bayesian inference model to update the probabilities.

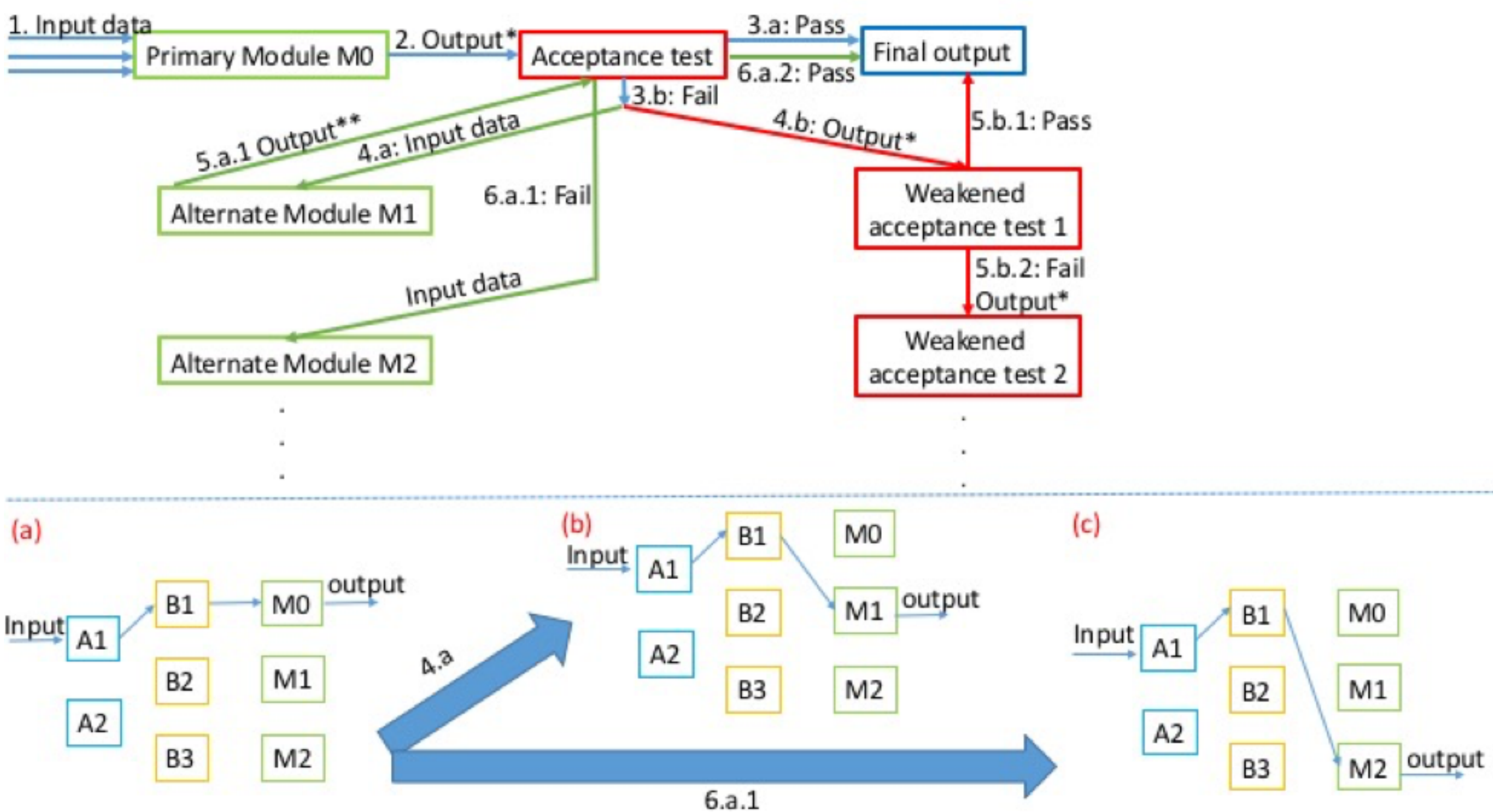
Reflexivity

A Solution Based on Graceful Degradation

Comprehensive Architecture of IAS



Generic Model of Dynamic Adaptation



Problem Statement

Given a smart cyber system operating in a distributed computing environment, it should be able to:

1. Replace anomalous/underperforming modules

2. Swiftly adapt to changes in context

3. Achieve continuous availability even under attacks and failures.

Graceful Degradations: Combinatorial Replica Replacement Scheme

- Combinatorial Structure is a subset satisfying certain conditions.

- Each block contains systems and their replicas that are mathematically distributed.

- The systems and their replicas in the distributed blocks are strategically connected to receive updates from primary modules.

- Resources are mathematically balanced, enabling scalable designs for the systems.

(7, 7, 3, 3, 1)-configuration

- 7 systems $\{S_1, S_2, S_3, S_4, S_5, S_6, S_7\}$
- 7 Distributed Autonomous Blocks (DABs) each with 3-system subset

$$DAB_1 = \{S_1, S_5, S_7\}, DAB_2 = \{S_1, S_2, S_6\},$$
$$DAB_3 = \{S_2, S_3, S_7\}, DAB_4 = \{S_1, S_3, S_4\},$$
$$DAB_5 = \{S_2, S_4, S_5\}, DAB_6 = \{S_3, S_5, S_6\},$$
$$DAB_7 = \{S_4, S_6, S_7\}.$$

(7, 7, 3, 3, 1)-configuration

- 7 systems $\{S_1, S_2, S_3, S_4, S_5, S_6, S_7\}$
- 7 Distributed Autonomous Blocks (DABs) each with 3-system subset
- Each system appears in 3 DABs (Say, S_6)

$$DAB_1 = \{S_1, S_5, S_7\}, \boldsymbol{DAB_2} = \{S_1, S_2, \boldsymbol{S_6}\},$$
$$DAB_3 = \{S_2, S_3, S_7\}, DAB_4 = \{S_1, S_3, S_4\},$$
$$DAB_5 = \{S_2, S_4, S_5\}, \boldsymbol{DAB_6} = \{S_3, S_5, \boldsymbol{S_6}\},$$
$$\boldsymbol{DAB_7} = \{S_4, \boldsymbol{S_6}, S_7\}.$$

(7, 7, 3, 3, 1)-configuration

- 7 systems $\{S_1, S_2, S_3, S_4, S_5, S_6, S_7\}$
- 7 Distributed Autonomous Blocks (DABs) each with 3-system subset
- Each system appears in 3 DABs
- Each pair of systems appear in 1 DAB (Say, S_1 and S_5)

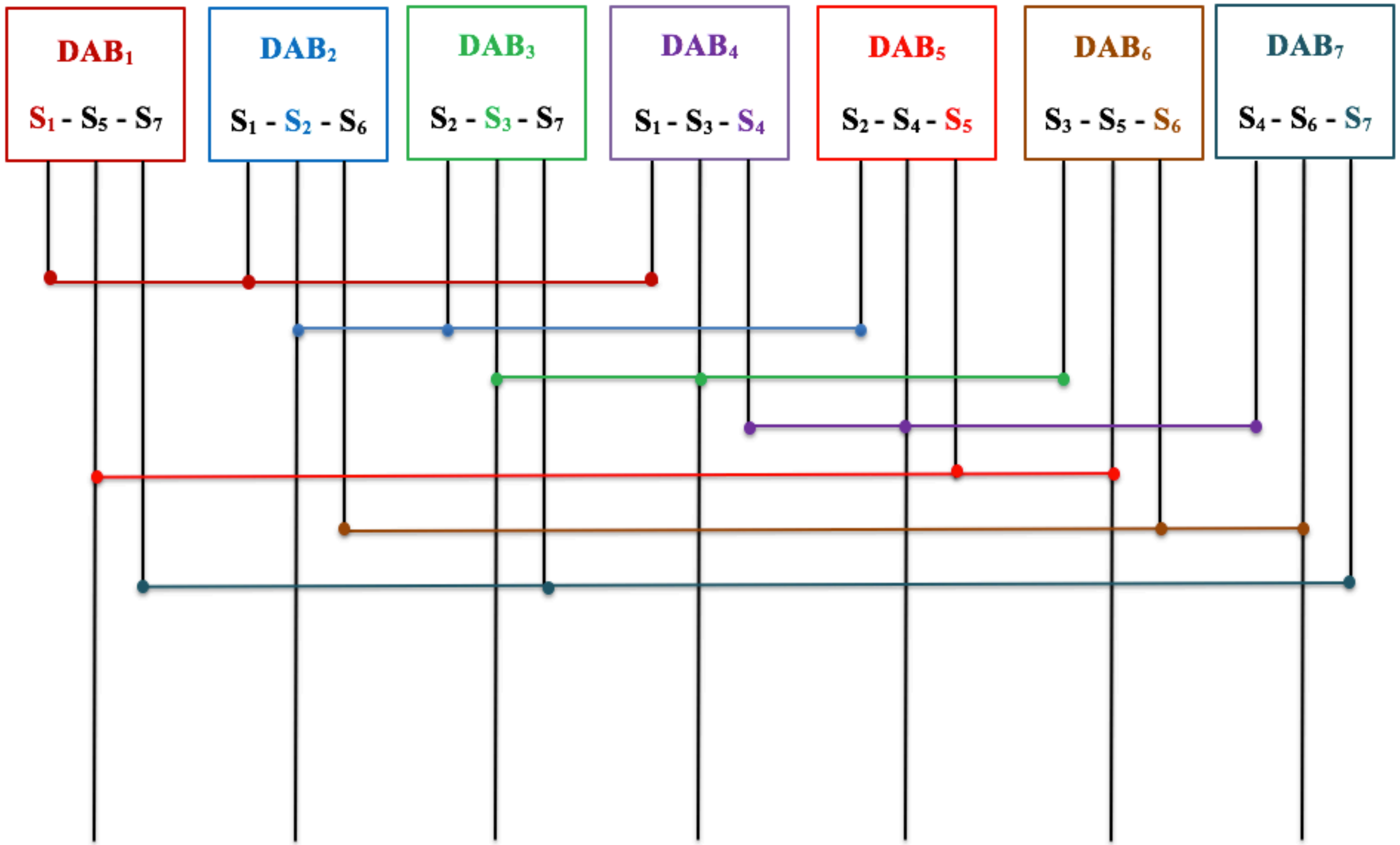
$$\boldsymbol{DAB_1} = \{\boldsymbol{S_1}, \boldsymbol{S_5}, S_7\},\ DAB_2 = \{S_1, S_2, S_6\},$$
$$DAB_3 = \{S_2, S_3, S_7\},\ DAB_4 = \{S_1, S_3, S_4\},$$
$$DAB_5 = \{S_2, S_4, S_5\},\ DAB_6 = \{S_3, S_5, S_6\},$$
$$DAB_7 = \{S_4, S_6, S_7\}.$$

(7, 7, 3, 3, 1)-configuration

- 7 systems $\{S_1, S_2, S_3, S_4, S_5, S_6, S_7\}$ **M**
- 7 Distributed Autonomous Blocks (DABs) **A**
- each with 3-system subset **C**
- Each system appears in 3 DABs **R**
- Each pair of systems appear in 1 DAB **O**

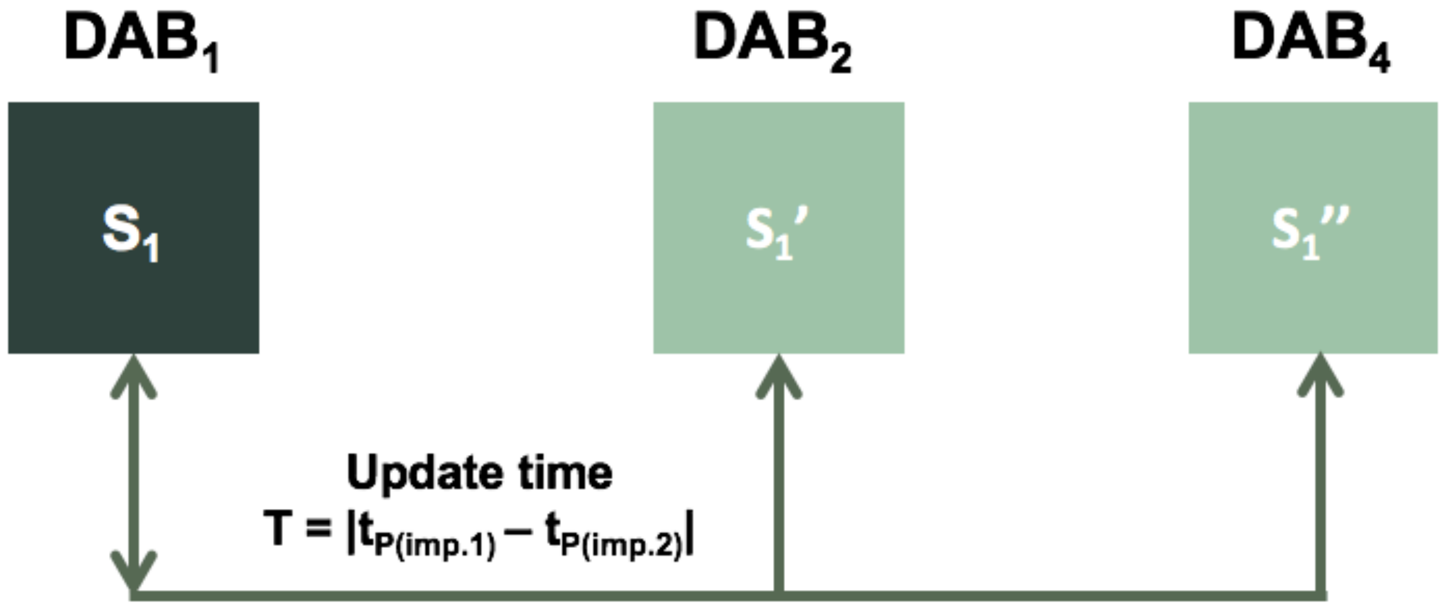
The configuration (M, A, C, R, O) = (7, 7, 3, 3, 1)

(7, 7, 3, 3, 1)-configuration
DAB: Distributed Autonomous Block



(7, 7, 3, 3, 1)-configuration

- Each primary module periodically updates its replicas in corresponding distributed block connected by communication links (CC).



- Update the interval dynamically through learning models with Bayesian learning by continuously updating the prior.

(7, 7, 3, 3, 1)-configuration

- Update time is defined as

  P_I(importance (I) | operational context (C)) = $\frac{P(C|I)P(I)}{P(C)}$

  Update interval T = | $t^1_{P(I)} - t^2_{P(I)}$ |

- Operational Context can be set dynamically and importance is a binary classifier (important /not important)

- When any system in any primary module's DAB acts in anomalous fashion, that system can be
  - Replaced with one of the replicas that can be selected in round robin fashion.
  - Anomalous module will be set for self-healing or repair by external source

(7, 7, 3, 3, 1)-configuration

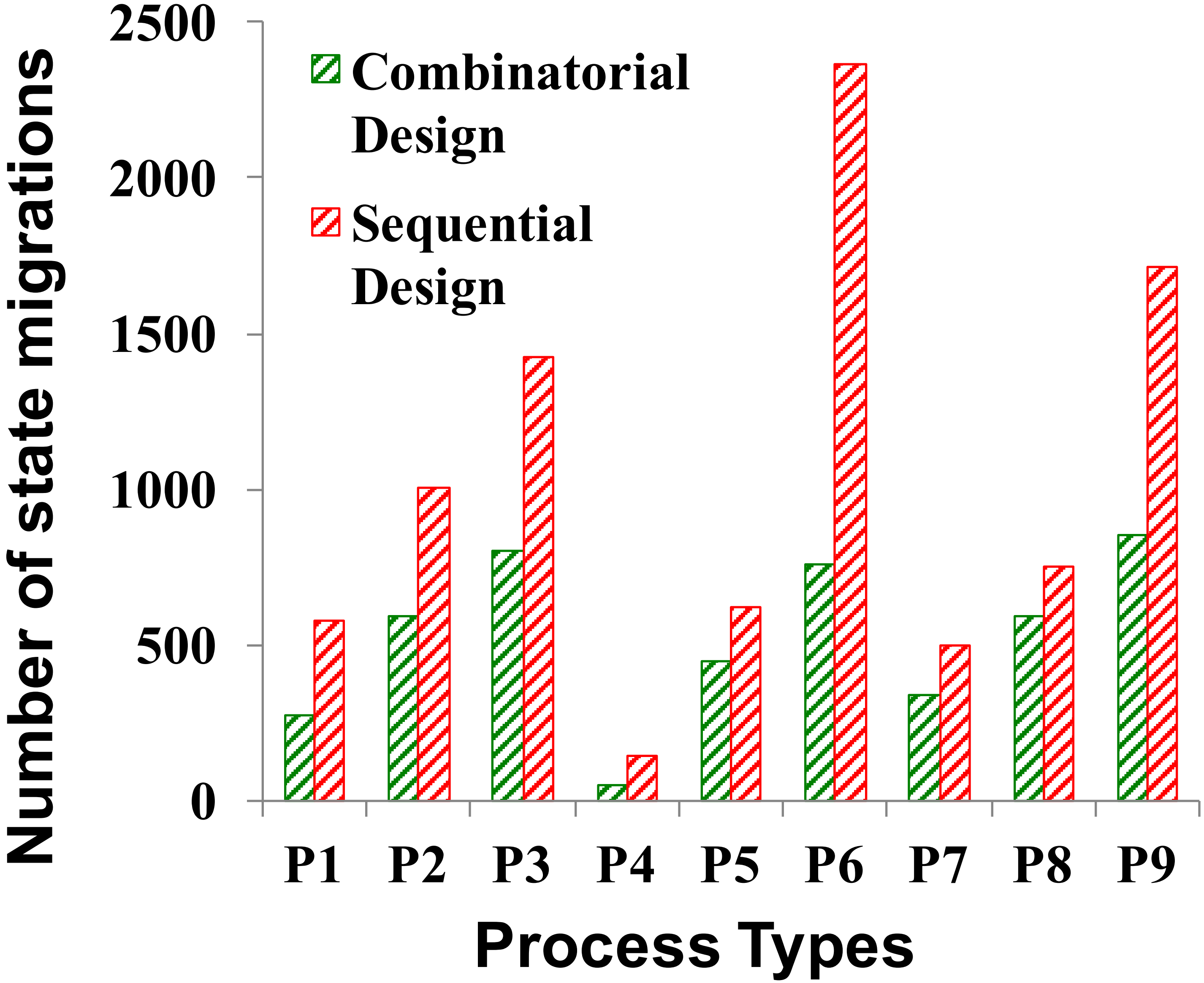
- The prototype is built with FAYE framework[1] with Node.js.
- It is a server-client framework where servers act as primary modules and clients as replicated system.
- Replica updates are done through a combinatorial design simulator[2].
- Combinatorial simulator is loaded with finite processes to compare the updates and processing time compared to a regular or sequential processing.

[1]https://faye.jcoglan.com/node.html [2]https://goo.gl/pgVHdk

Measurements for Various Process Completions

Process Type	Process Name	**Speed Up Due to Combinatorial Replica Scheme** (Compared to regular sequential design)
P1	FIBSEARCH	1.3
P2	DOUBLE MULT	1.4
P3	FIBB	1.5
P4	SEARCH	1.8
P5	COPY	1.8
P6	SCALAR	2
P7	SUM	2.1
P8	PRINT	3
P9	MOVEMENT	3.1

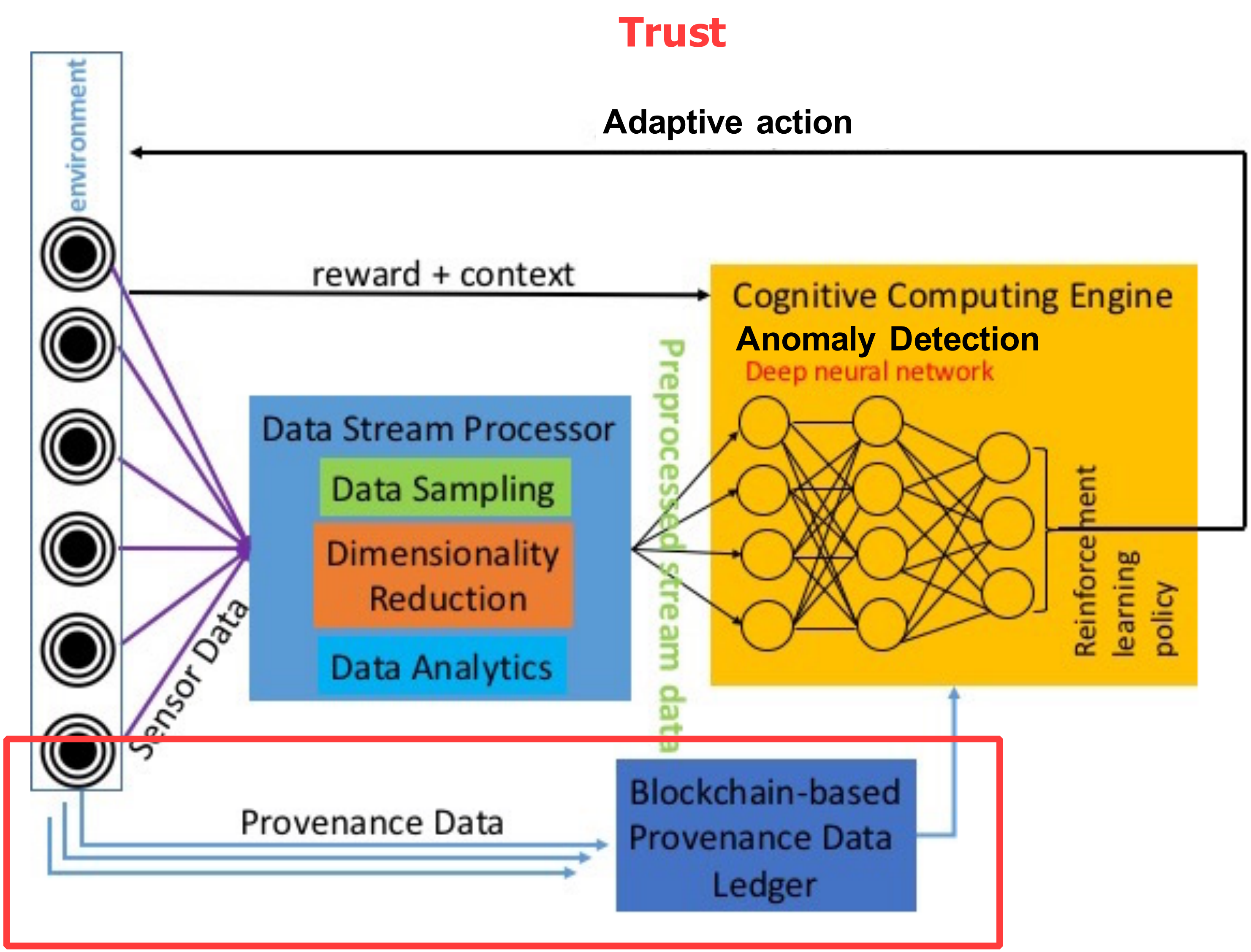
Measurements for Various Process Completions



Trust

A Solution Based on Blockchain

Comprehensive Architecture of IAS



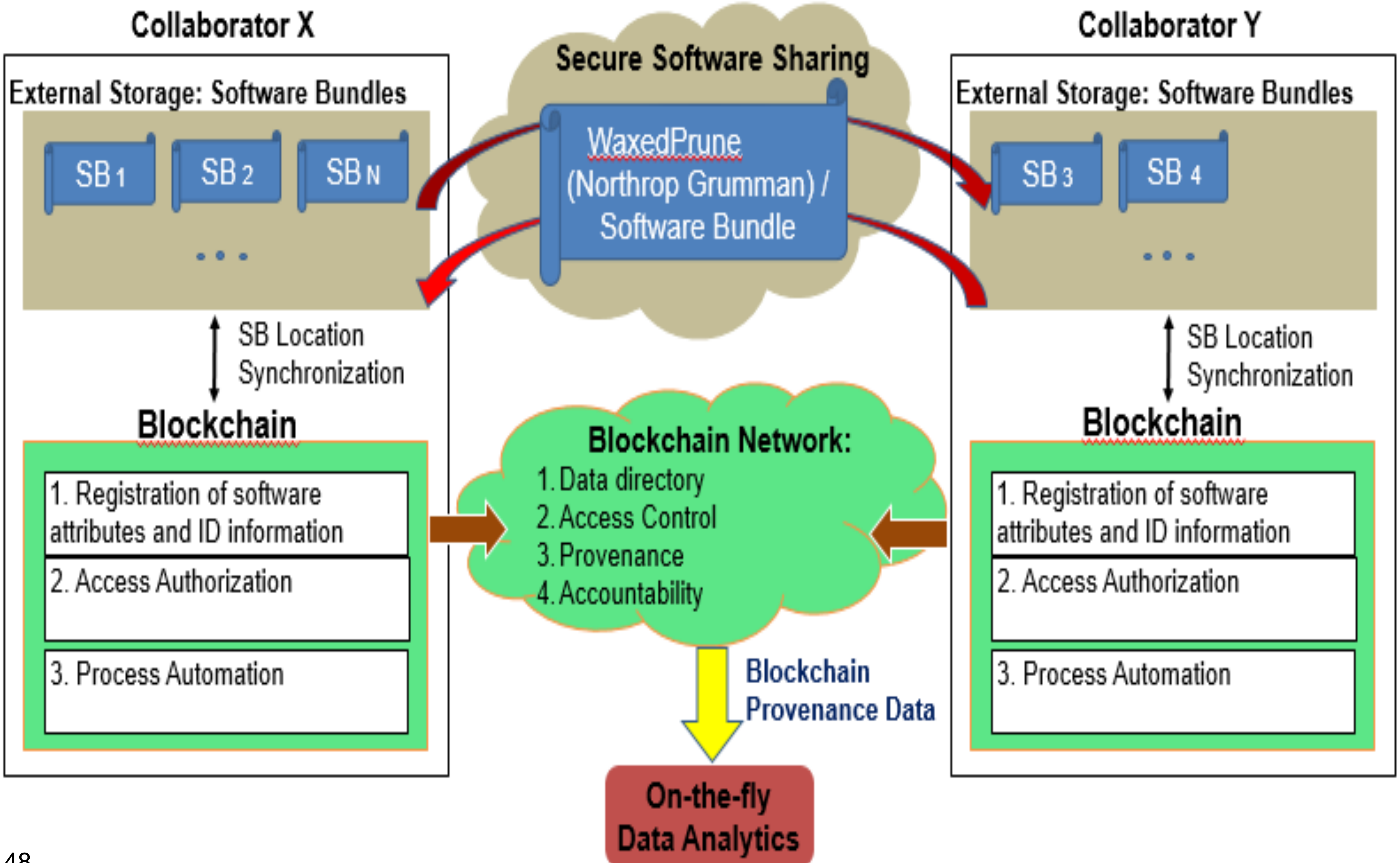
Problem Statement

- Provide trust (integrity, confidentiality, verifiability) to provenance data in IAS
  - Interactions between services are logged
  - Log records can not be corrupted
- Provide trust for network participants in IAS
  - Ensure data confidentiality
  - Ensure data integrity
- Provide privacy-preserving data exchange in IAS

Blockchain Technology Deployment

- Fine-grained role-based and attribute-based access control with data leakage detection capabilities is provided by integration with 'WAXEDPRUNE'
- Performance improvements:
  - Depth-robust graphs to store blockchain for faster transaction verification: no need to verify all the links in the chain

Blockhub: blockchain-platform for IAS



Future Work

- Develop cyber attribution techniques with machine learning to enhance the forensics and malware detection.

- Optimize the reflexivity property's replacement policy with distributed voting and Hidden Markov Model to determine update interval.

- Failure recovery for blockchain framework with mobile environments.

References:

1. Mani, Ganapathy, Bharat Bhargava, and Basavesh Shivakumar. "Incremental Learning Through Graceful Degradations in Autonomous Systems." In *2018 IEEE International Conference on Cognitive Computing (ICCC)*, pp. 25-32. IEEE, 2018.

2. Ulybyshev, Denis, Miguel Villarreal-Vasquez, Bharat Bhargava, Ganapathy Mani, Steve Seaberg, Paul Conoval, Robert Pike, and Jason Kobes. "(WIP) Blockhub: Blockchain-Based Software Development System for Untrusted Environments." In *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pp. 582-585. IEEE, 2018.

3. Ranchal, Rohit, Denis Ulybyshev, Pelin Angin, and Bharat Bhargava. "PD3: policy-based distributed data dissemination." In *Proceedings of the 16th Annual Information Security Symposium*, p. 13. CERIAS-Purdue University, 2015.

Thank you!!!