A Design/Study of a Self-Aware Codelet Runtime System

Professor **Guang R. Gao**
Mentors:
Dr. Stephane Zuckerman, Aaron Landwehr, Joshua Suetterlein
Spring 2014

Laura Rozo
Chuanzhen Wu
Pouya Fotouhi

Agenda

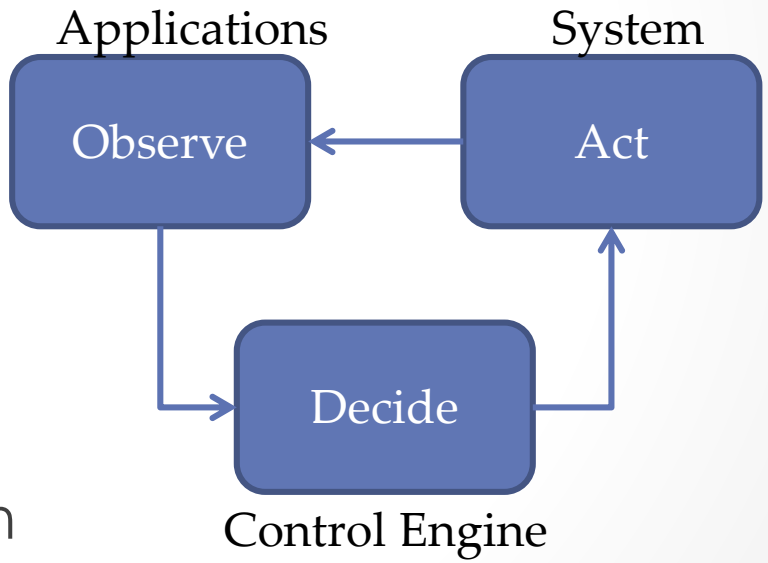
- Motivation
- Background
  - Control Theory
  - Codelet
  - Self-aware System for Exascale Architectures
- Problem Formulation
- Solution
- Related work
- Conclusion
- References

Motivation

- **To meet constrains defined by user:**
  - **Time, energy, reliability, etc.**
- **Increasing demand for higher performance and more complex functionality in exascale systems:**
  - **Increasing number of cores implies:**
    - **Increasing energy consumption.**
    - **More source of failures.**
  - **Shrinking device size: more sensitive to environmental issues**
- **To tackle these challenges, systems need to have the ability to:**
  - **Act proactively and adapt themself based on current conditions.**
  - **Apply the corresponding schemes that optimize desired goals (i.e. performance, reliability, energy consumption).**

Background (Control Theory)

- Includes:
  - Observation
  - Decision
  - Action
- The system observes the performance, use the observed result to modify the system resources allocation in order to reach the application's desired goal.



Katsuhiko Ogata, Modern Control Engineering

Background

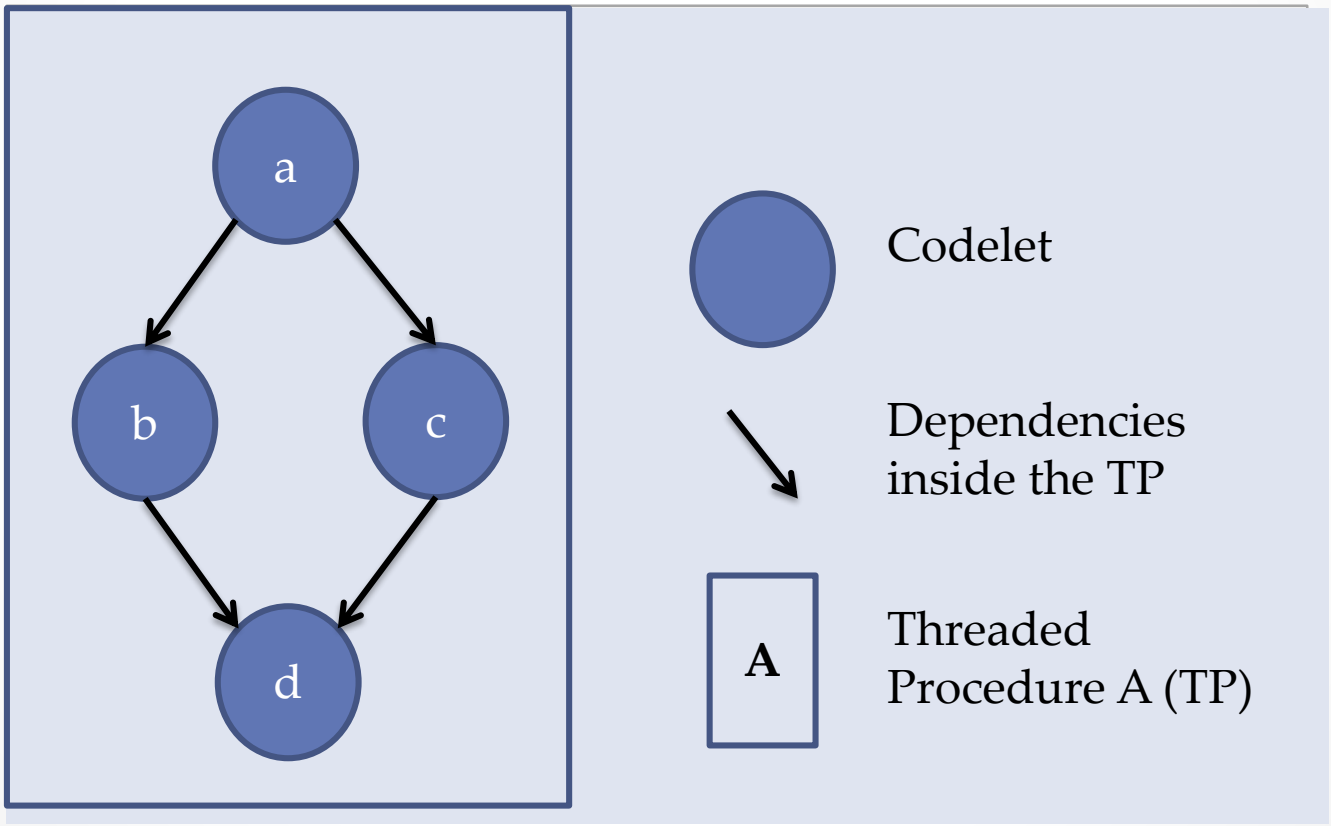
(Self-aware system for Exascale Architectures)

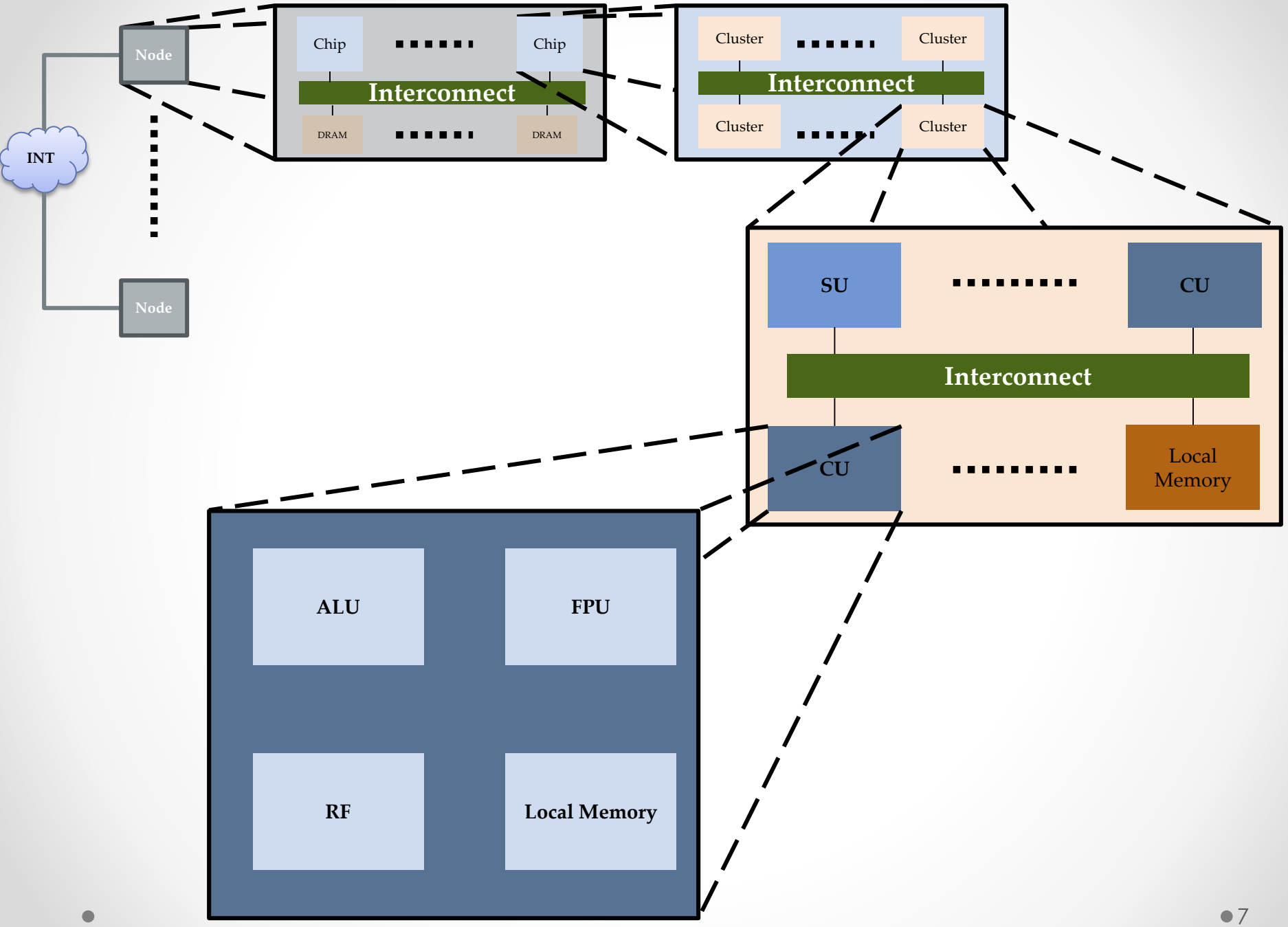
- Aaron et al. has proposed a self-aware system:
  - based on extreme scale architectures.
  - With one level of parallelism.
- Clusters contains blocks, including:
  - XEs (eXecution Engines).
  - CEs (Control Engines).
  - PMU (Performance Monitoring Unit).
  - Memory and Net.
- According to results from PMU, system will adapt for optimization.

Aaron Landwehr et al, ROME 2013, 2013.

Background (Codelet Model)

- **Data Flow Inspired Execution Model.**

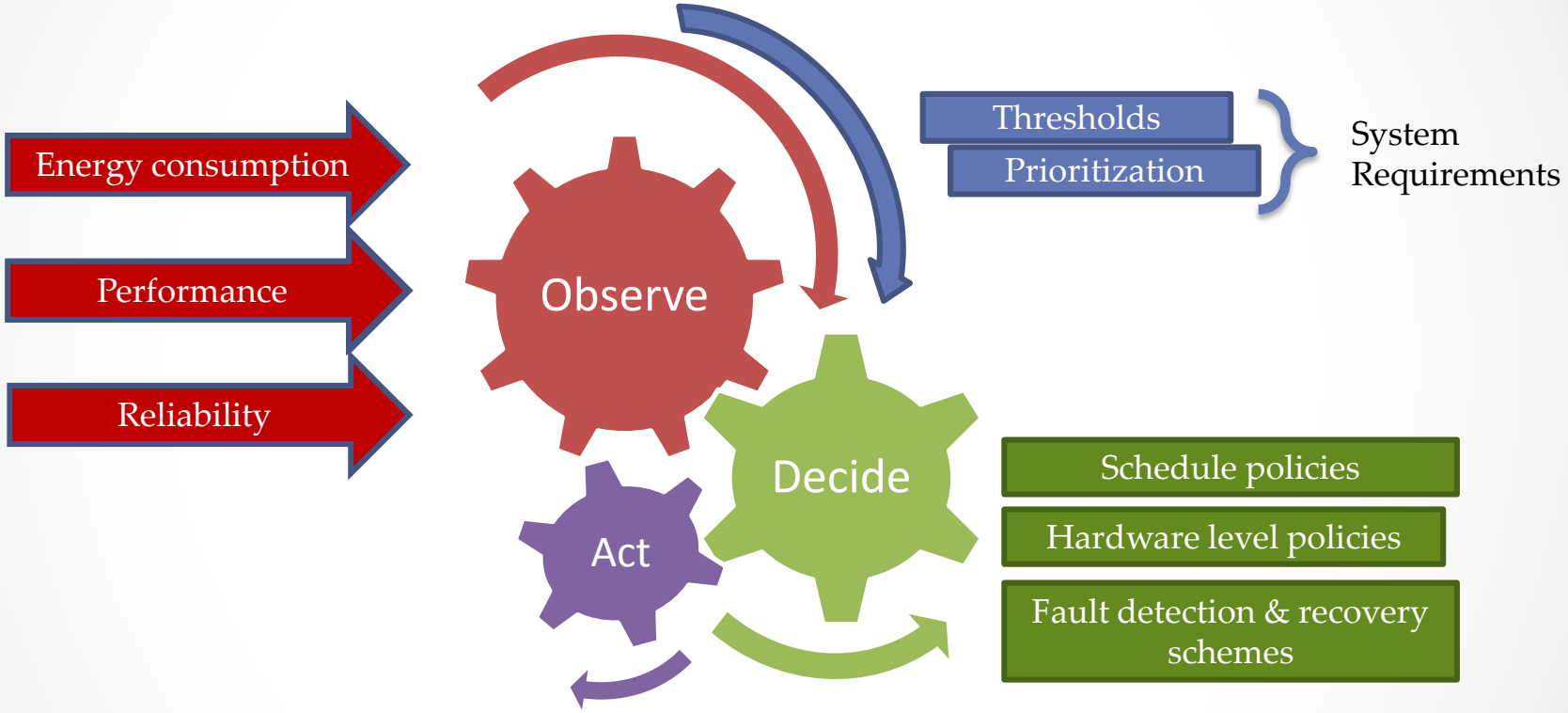


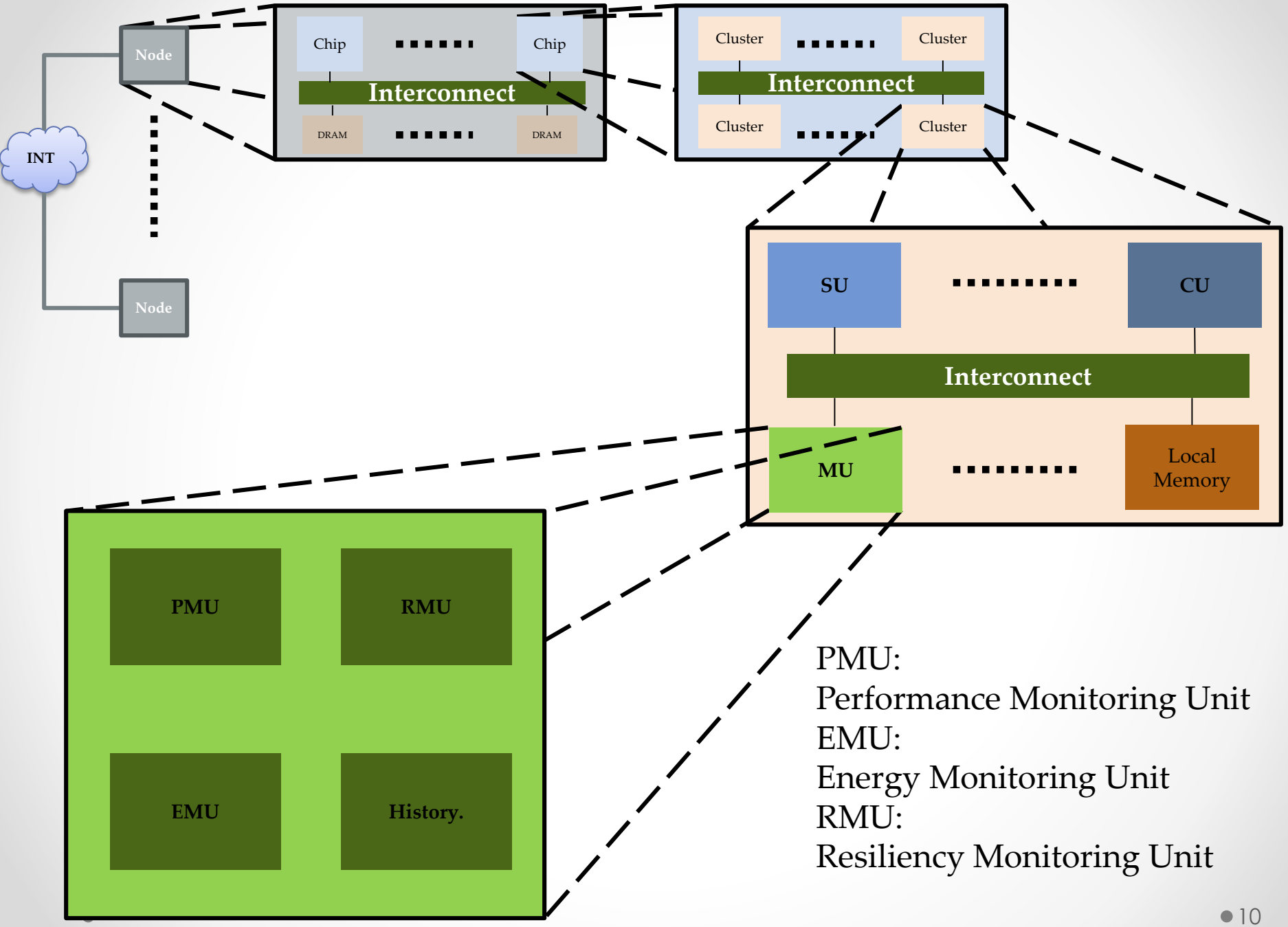

Node
INT
Node
Chip
Chip
Interconnect
DRAM
DRAM
Cluster
Cluster
Interconnect
Cluster
Cluster
SU
CU
Interconnect
CU
Local Memory
ALU
FPU
RF
Local Memory

Problem Formulation

- **Combine self aware cycle and codelet model and analyze the interaction between:**
  - **Performance**
  - **Energy Consumption**
  - **Resiliency**
- **Optimize "Resource Management" and "Scheduling Policies", considering the trade-off between:**
  - **Maximizing performance.**
  - **Minimizing energy consumption.**
  - **Enhancing reliability.**

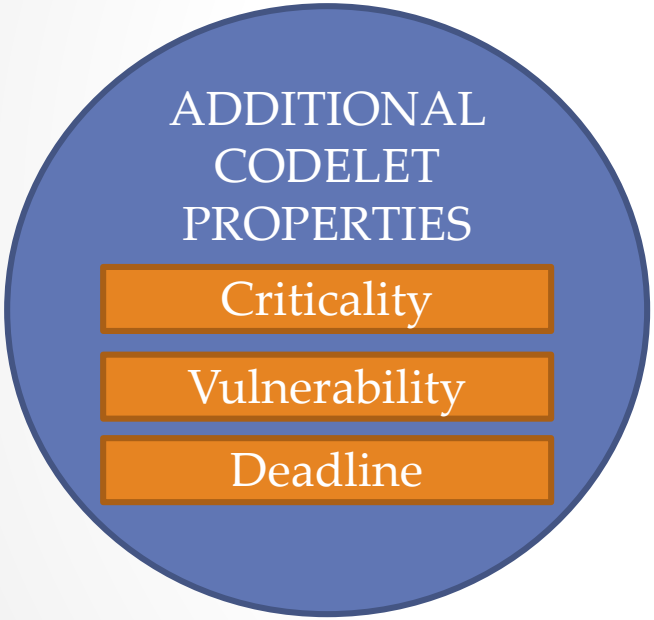
Solution


Energy consumption
Performance
Reliability
Observe
Decide
Act
Thresholds
Prioritization
System Requirements
Schedule policies
Hardware level policies
Fault detection & recovery schemes



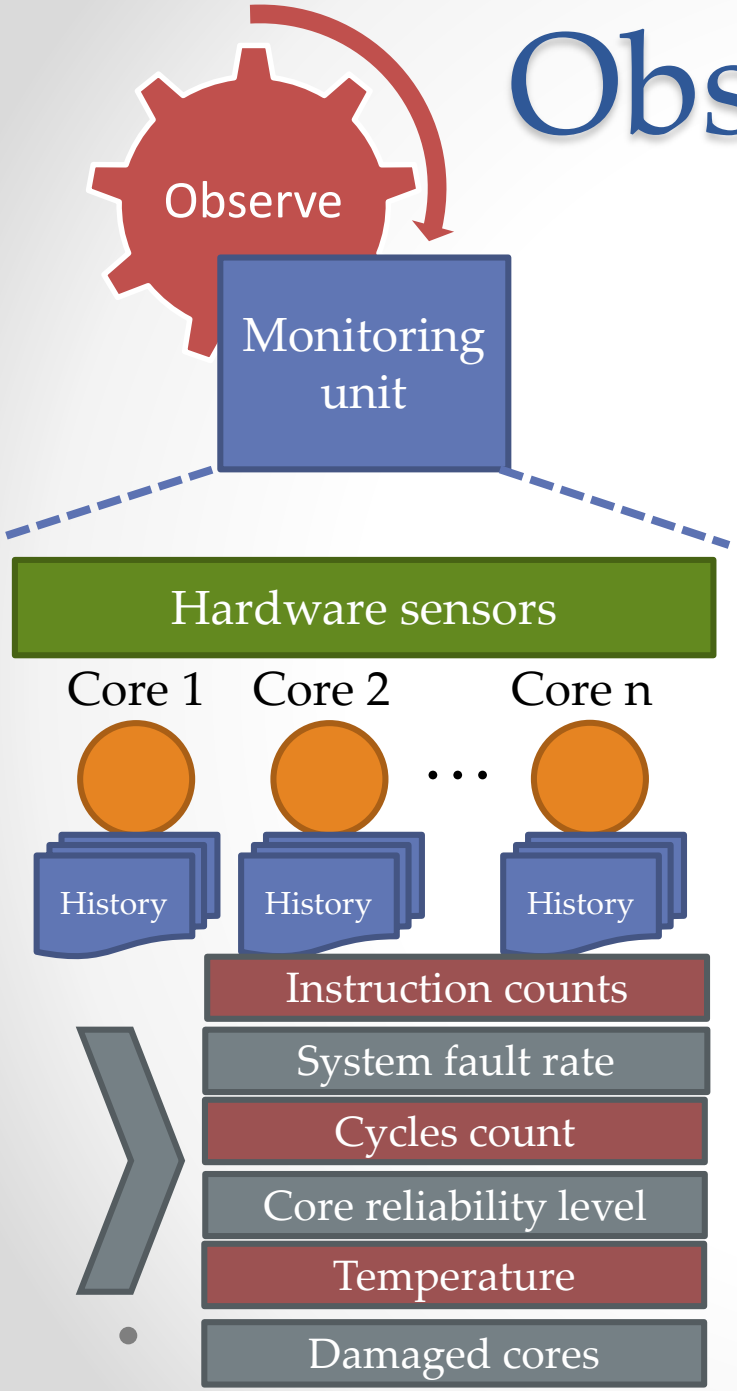
PMU:
Performance Monitoring Unit
EMU:
Energy Monitoring Unit
RMU:
Resiliency Monitoring Unit

Codelet's Meta-data



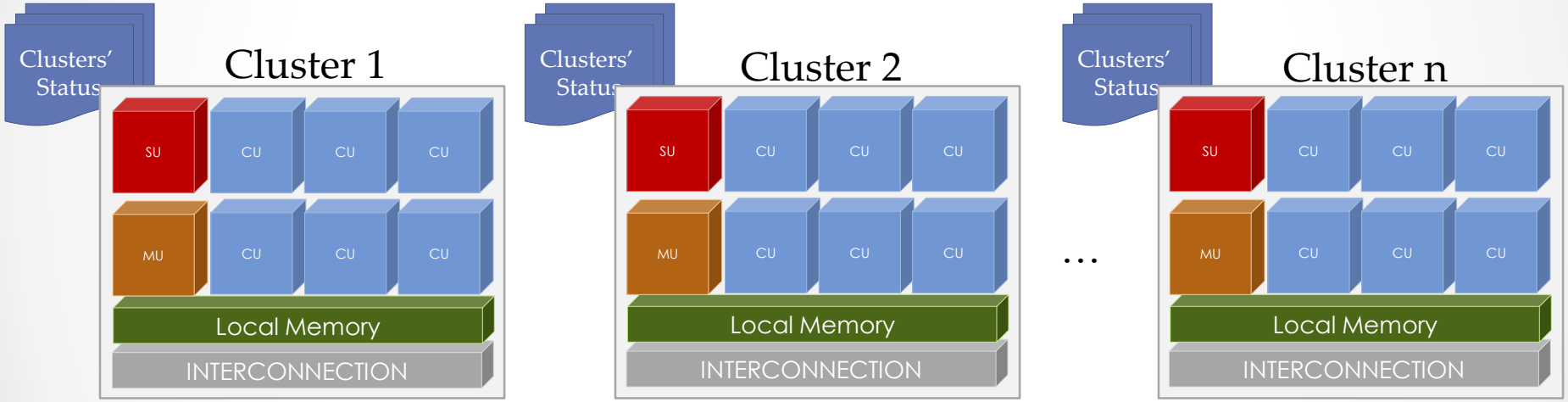
- In order to minimize computations at runtime, some useful information can be computed offline for each of the codelets.
  - Criticality for performance optimization.
  - Vulnerability for resiliency.
  - Time constrains based on system requirements.

Observe cycle

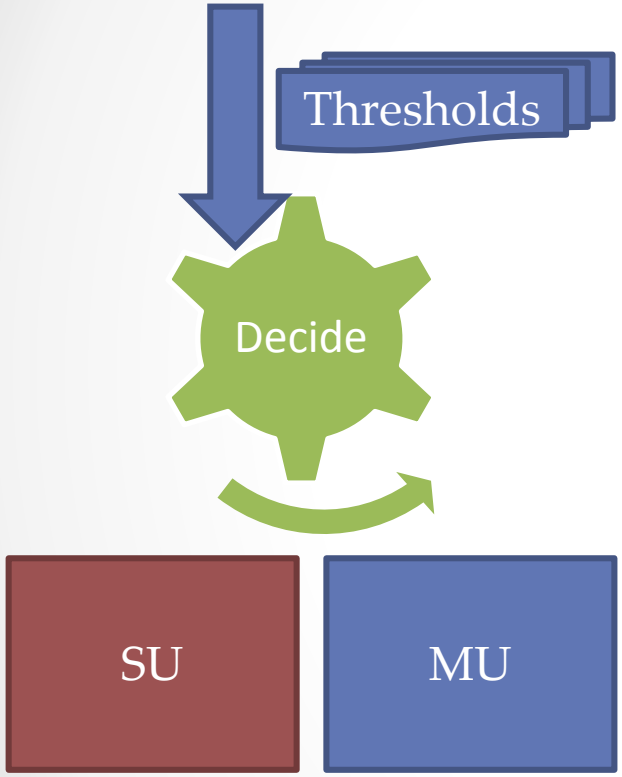


- Monitoring unit collects data from hardware sensors and from cores and records a history of them.
- Energy consumption
  - Temperature
  - Instruction counts
- Performance
  - Instruction counts
  - Cycles counts
  - Bandwidth usage
- Reliability
  - Temperature
  - System Fault rate
  - Core diagnosis
    - Unhealthy and healthy cores
    - Reliability level for each core

MU and SU keep updating the status of each cluster. Status info includes currently executing TPs, energy consumption, fault rate etc.

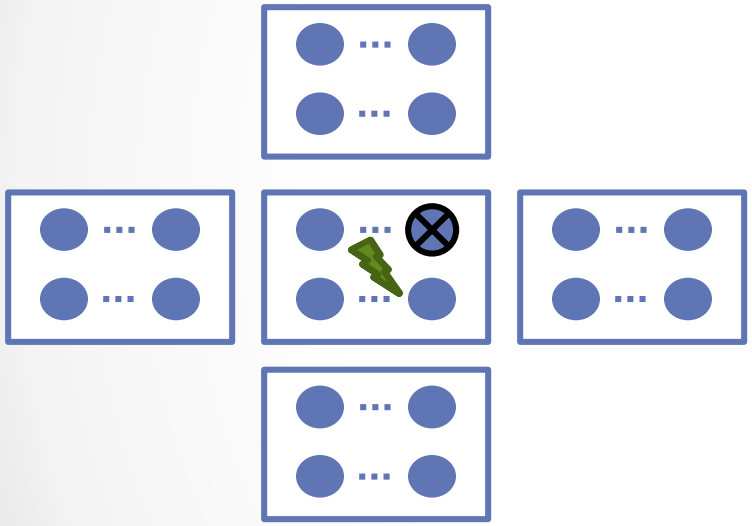


Decide



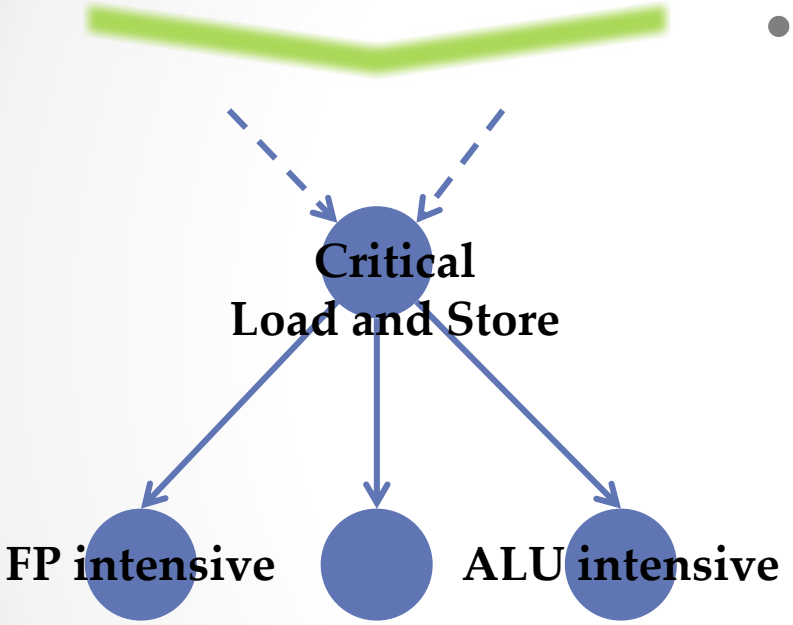
- Based on collected data and thresholds define which schemes to apply for each goal.
- For this step, two elements are going to be involved:
  - Synchronization Unit
  - Monitoring unit

Decide



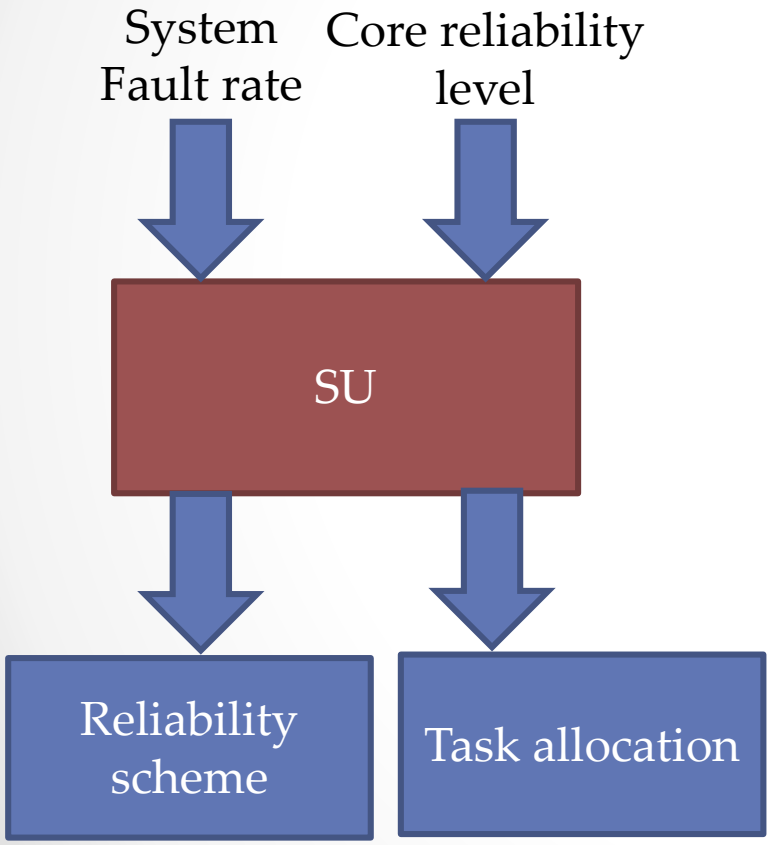
- Energy consumption
  - Energy budgets for each cluster are defined: Initialization
  - Based on number of unhealthy cores energy budgets should be updated
  - Based on current energy consumption of the whole cluster, decide the appropriate energy optimization schemes
    - Clock gate
    - Power gate
    - Dynamic Voltage and Frequency Scaling (DVFS)

Decide



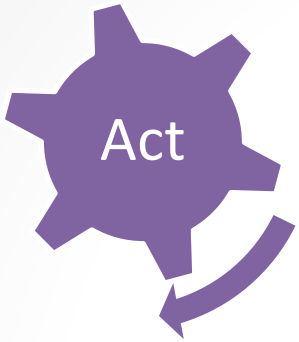
- Performance
  - Meet deadlines by using frequency switching methods.
  - Based on the meta-data of each codelet and history of each core :
    - Choose the right Cluster/Core for scheduling the codelet on.
    - DVFS accordingly.

Decide



- Resiliency
  - Two schemes: codelet level and threaded procedure level.
  - Based on the system fault rate decide which scheme is better.
    - High fault rate -> codelet level
    - Low fault rate -> threaded procedure
  - Task allocation based on core diagnosis

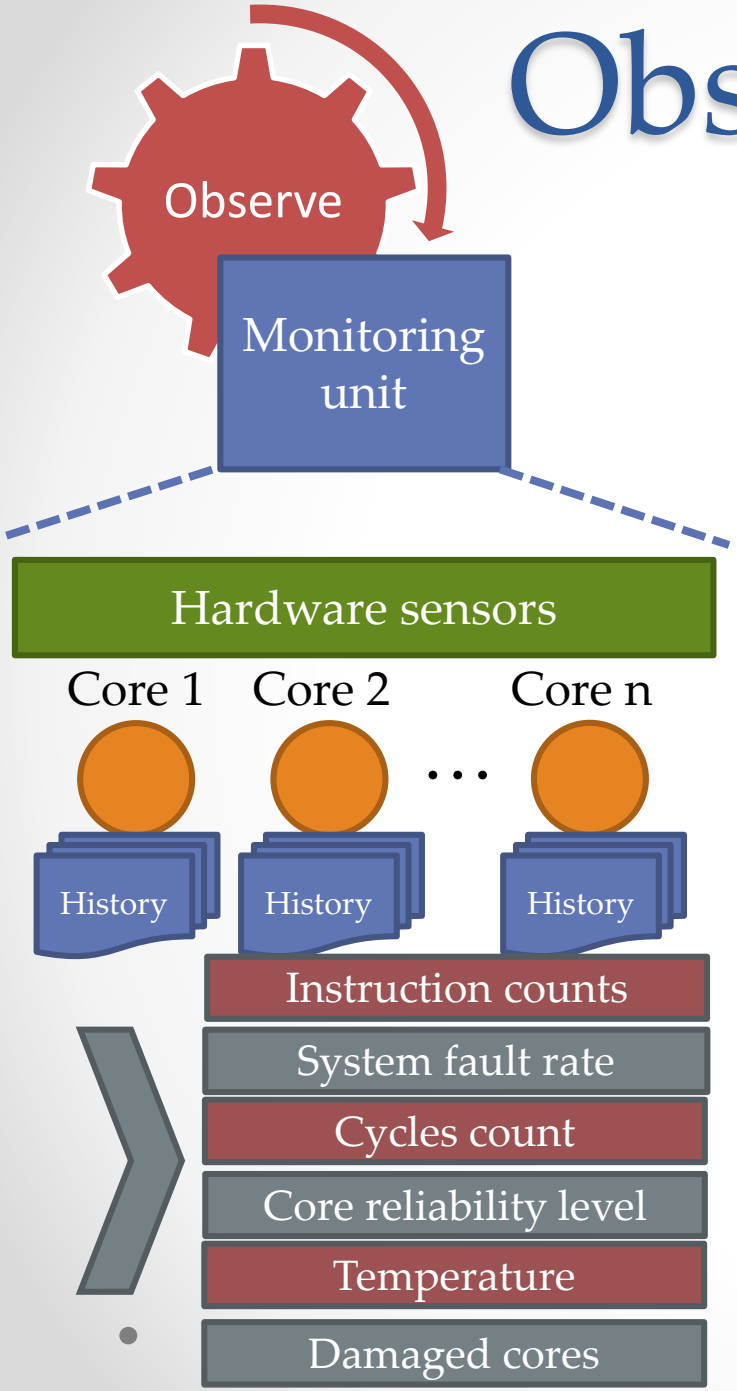
Act



- Apply schemes selected in previous step
  - Energy Consumption
  - Performance
  - Reliability

SU	MU
Fault detection and recovery scheme	Re-distribution of energy budgets
Deadlines and criticality assurance	Frequency switching
	Energy thresholds assurance

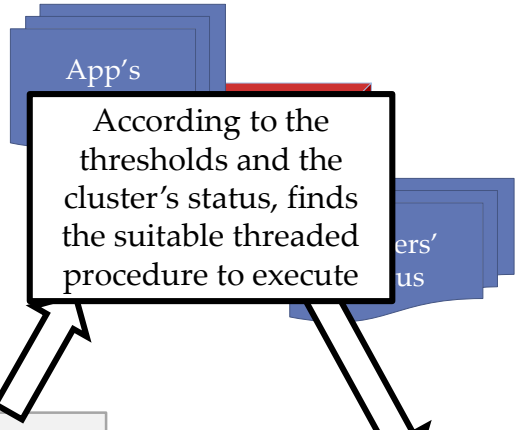
Observe cycle



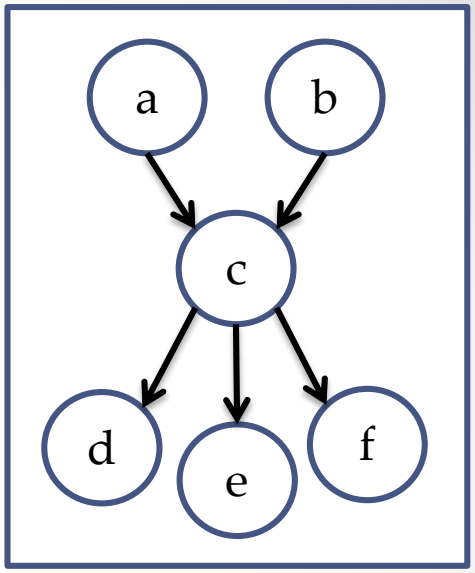
- Monitoring unit collects data from hardware sensors and from cores and records a history of them.
- Energy consumption
  - Temperature
  - Instruction counts
- Performance
  - Instruction counts
  - Cycles counts
  - Bandwidth usage
- Reliability
  - Temperature
  - System Fault rate
  - Core diagnosis
    - Unhealthy and healthy cores
    - Reliability level for each core

When a cluster is idle, SU will choose a threaded procedures from the pool.

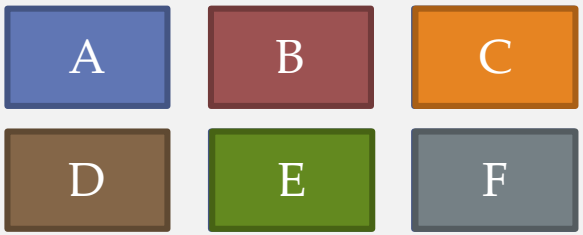
Then, enabled codelets will be allocated to different cores.



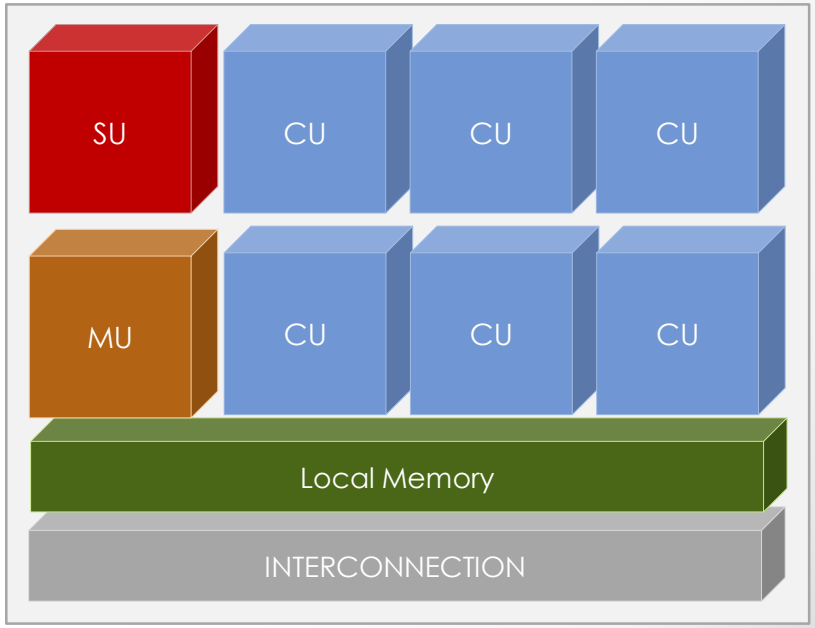
CDG



Pool of Threaded procedures



Cluster



Proactive:

Based on codelets meta-data (offline computations):

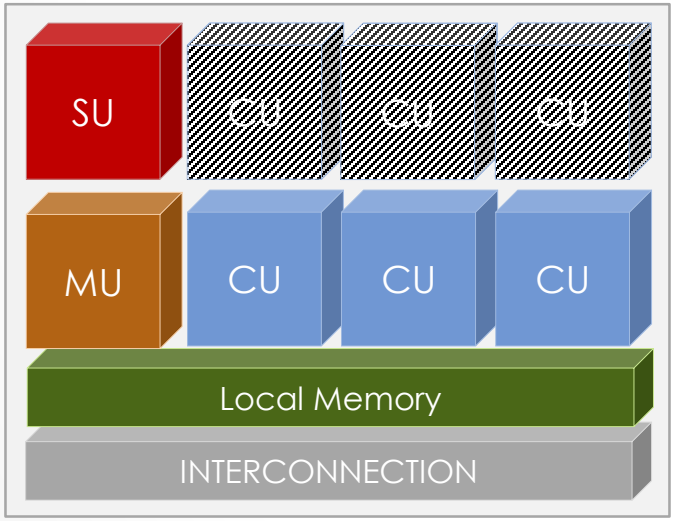
- Number of Dependents
- Criticality
- Vulnerability
- Intensity and ...

- And historical information:

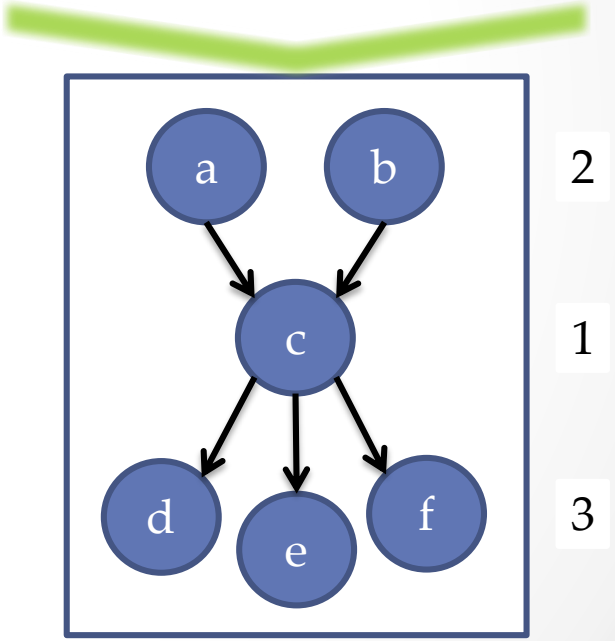
  The system will update the resource management and scheduling policies.

Maximum Width: 3
Criticality: Codelet c

Cluster



3 CUs are power-gated and 3 CUs keep working.



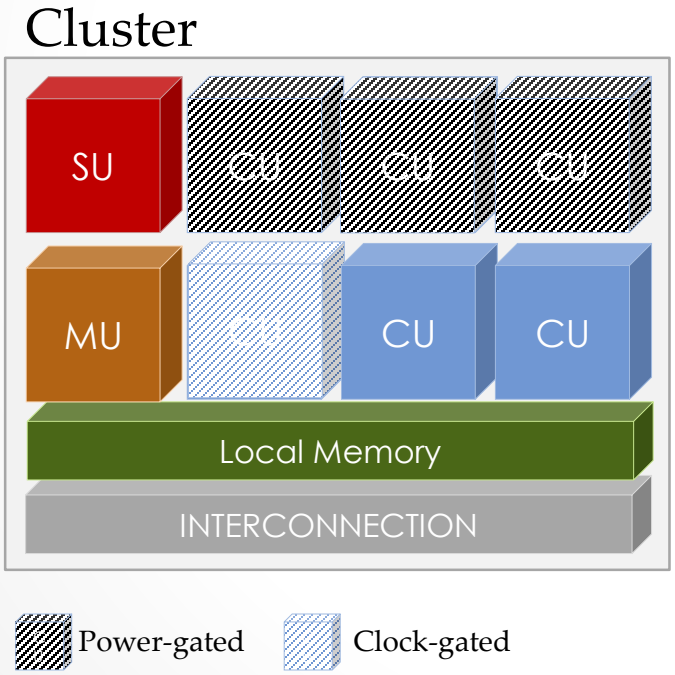
Power-gated Clock-gated

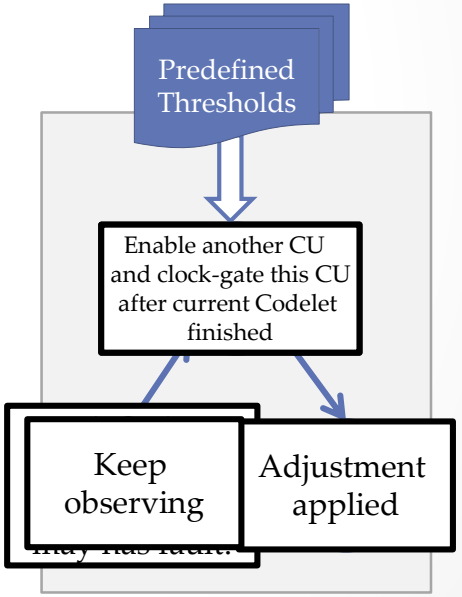
Self-aware:

SU and MU will tune the work according to the hardware or OS events:

- Temperature rises
- Incorrect behaviors and …

Example:



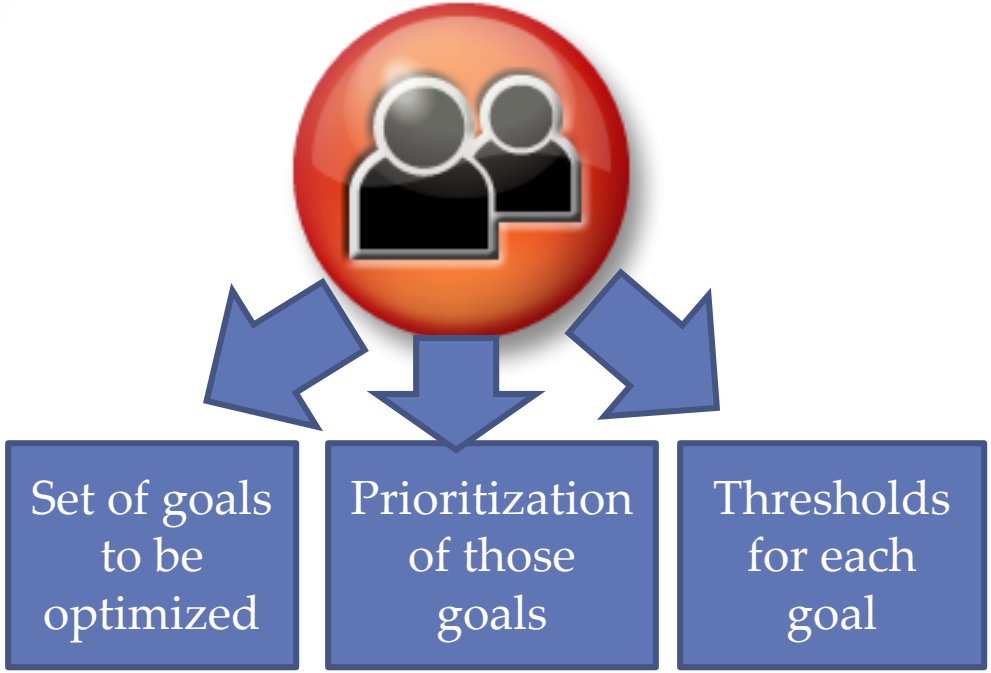


Related work (SEEC)

- Uses ODA loop to meet performance and energy consumption goals.
  - Heartbeat, A method of observing performance.
- Uses Machine Learning techniques.
- Cons:
  - One prioritization for optimization.
  - Not designed for extreme scale machines.

Henry Hoffmann et al, MIT-CSAIL-TR-2011-046, 2011
Martina Maggio et al, FeBID 2011

Conclusion



- Additional information need to be specified. Based on this information the system will be able to prioritize some goals over other ones.

Thank you!

References

- Shekhar Borkar , Thousand Core Chips—A Technology Perspective, Proceedings of the 44th Annual Design Automation Conference, San Diego, California, DAC '07, 2007
- Henry Hoffmann, Martina Maggio, Marco D. Santambrogio, Alberto Leva, and Anant Agarwal, SEEC: A General and Extensible Framework for Self-Aware Computing, MIT CSAIL Technical Report, MIT-CSAIL-TR-2011-046, November 2011.
- Henry Hoffmann, Jim Holt, George Kurian, Eric Lau, Martina Maggio, Jason E. Miller, Sabrina M. Neuman, Mahmut Sinangil, Yildiz Sinangil, Anant Agarwal, Anantha P. Chandrakasan, Srinivas Devadas, Self-aware Computing in the Angstrom Processor, Proceedings of the 49th Design Automation Conference (DAC), June 2012.
- Tom St. John, Benoit Meister, Andres Marquez, Joseph B. Manzano, Guang R. Gao, and Xiaoming Li, ASAFESSS: A Scheduler-driven Adaptive Framework for Extreme Scale Software Stacks, In Proceedings of the 4th International Workshop on Adaptive Self-Tuning Computing Systems (ADAPT'14), Vienna, Austria. January 20-22, 2014.
- Aaron Landwehr, Stephane Zuckerman, and Guang R. Gao, Toward a Self-aware System for Exascale Architectures, the 1st Workshop on Runtime and Operating Systems for the Many-core Era (ROME 2013), Aachen, Germany. August 26th, 2013.
- Martina Maggio, Henry Hoffmann, Anant Agarwal, and Alberto Leva, Control-theoretical CPU allocation: Design and Implementation with Feedback Control, The 6th International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks (FeBID 2011).
- Stéphane Zuckerman, Joshua Suetterlein, Rob Knauerhase and Guang R. Gao, Using a "Codelet" Program Execution Model for Exascale Machines, the 1st International Workshop on Adaptive Self-Tuning Computing Systems for the Exaflop Era.

Interaction between different goals

Goal\ Impact	Performance	Energy	Reliability
Performance		The faster tasks are executed, higher energy levels are reached.	Additional tasks can be executed faster.
Energy	The less energy the slower tasks are executed.		Additional computations are delayed.
Reliability	Additional tasks that will cause the application to run slower.	More tasks to be executed, higher energy consumption.	

Hence, the question is how to deal with this interaction?