TECHNISCHE
UNIVERSITÄT
WIEN
Vienna | Austria

Institut für
Computertechnik
Institute of
Computer Technology

Self-Aware CPSs

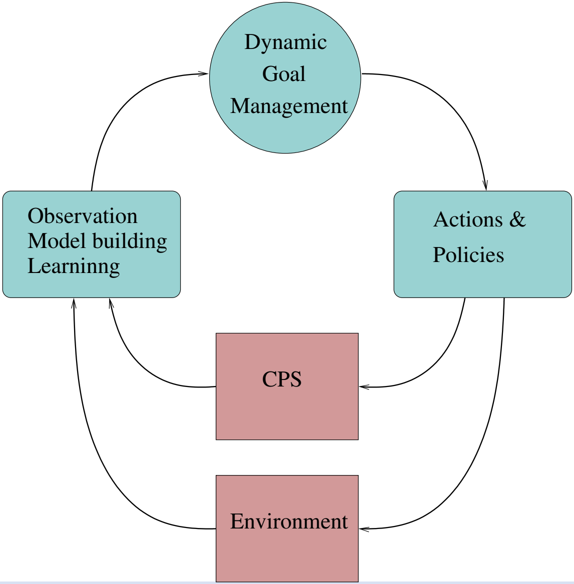
Axel Jantsch

TU Wien, Vienna, Austria

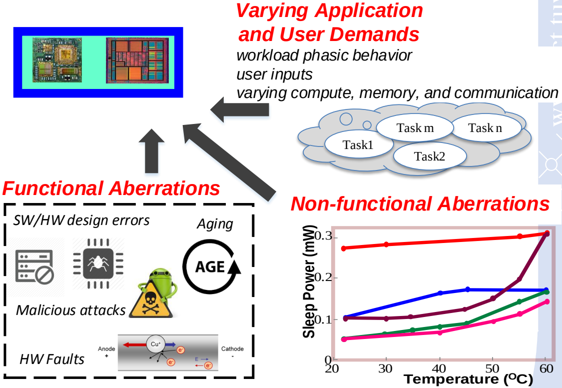
oCPS Fall School

October 2019

Self-Aware Control Loop

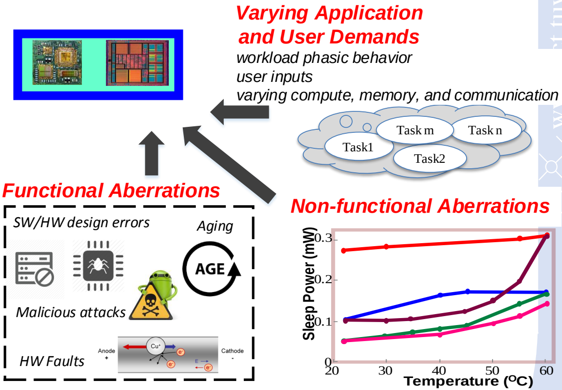


The Problem


Varying Application and User Demands
workload phasic behavior
user inputs
varying compute, memory, and communication
Task1
Task m
Task n
Task2
Functional Aberrations
SW/HW design errors
Aging
AGE
Malicious attacks
HW Faults
Anode
Cathode
Cu+
Non-functional Aberrations
Sleep Power (mW)
0.3
0.2
0.1
0
20
30
40
50
60
Temperature (°C)

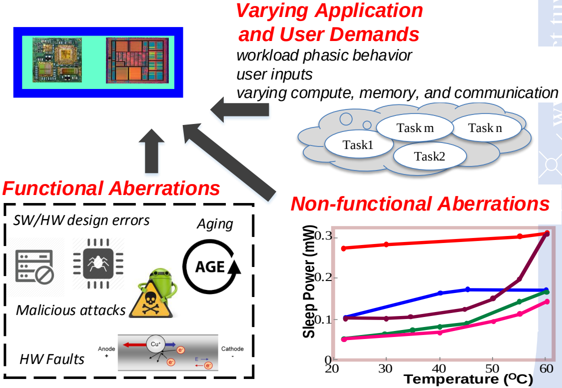
The Problem

- Large number of resources



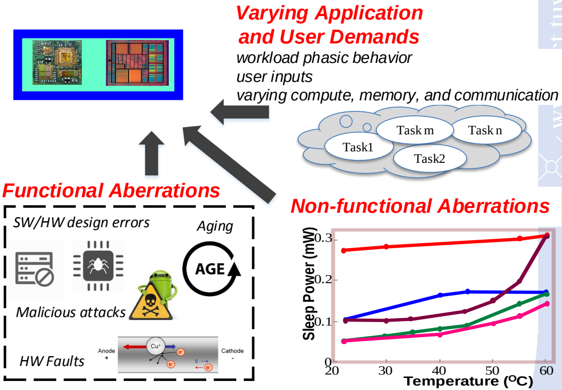
The Problem

- Large number of resources
- Many tight constraints



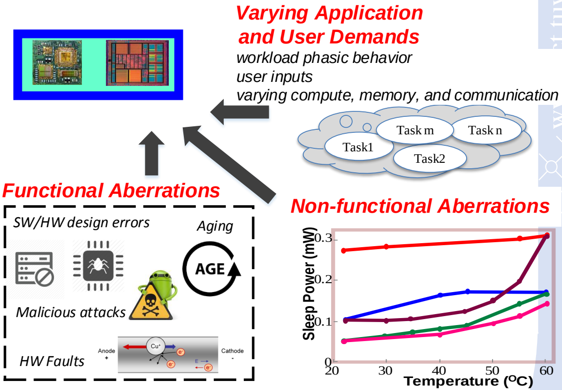
The Problem

- Large number of resources
- Many tight constraints
- Varying application demands, both within and between applications;

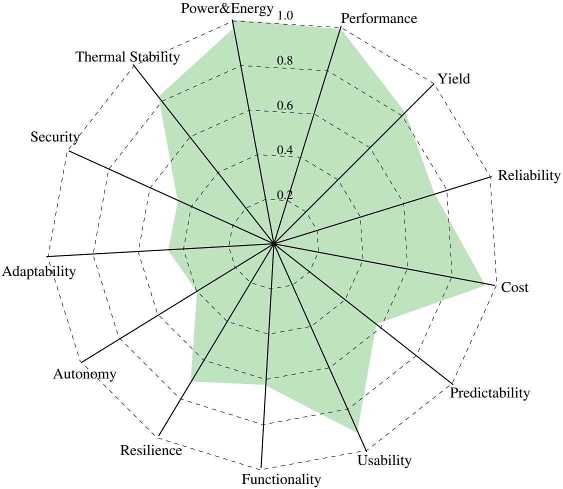


The Problem

- Large number of resources
- Many tight constraints
- Varying application demands, both within and between applications;
- Functional Aberrations:
  - Design errors or omissions;
  - Malicious attacks;
  - Aging;
  - Soft errors;
- Non-functional Aberrations:
  - Performance;
  - Power consumption;

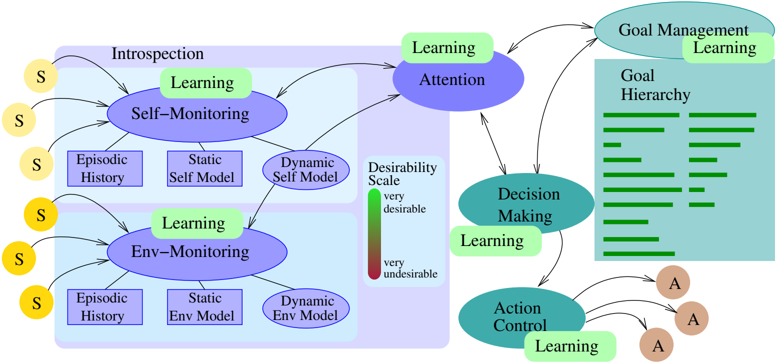


The SoC Radar



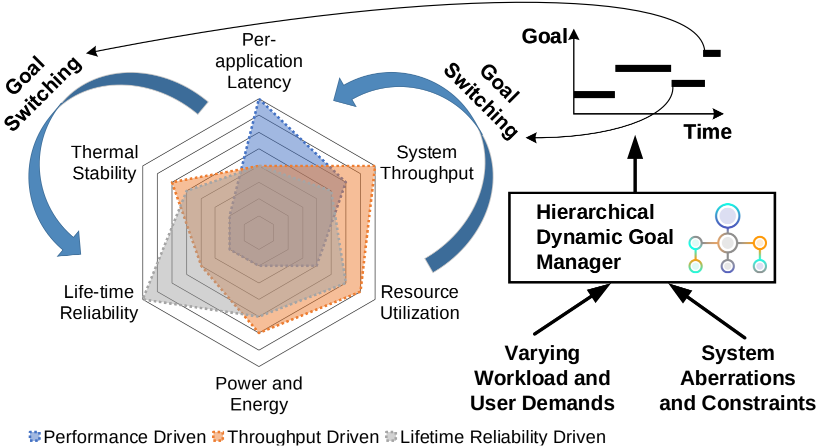
Santanu Sarma et al. “On-Chip Self-Awareness Using Cyberphysical-Systems-On-Chip (CPSoC)”. In: *Proceedings of the 12th International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. New Delhi, India, Oct. 2014

Self-Awareness Architecture

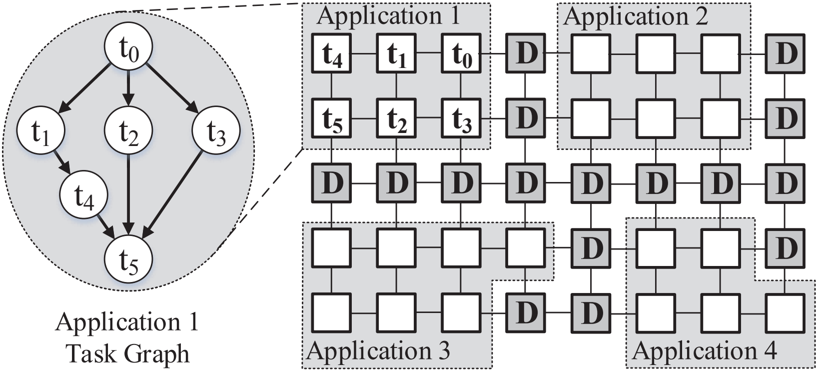


Goal Managment

Goals for Dynamic Task Mapping



Dynamic Task Mapping



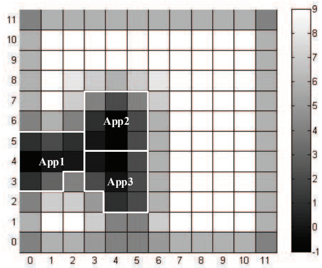
Example 1: Performance Driven Task Mapping

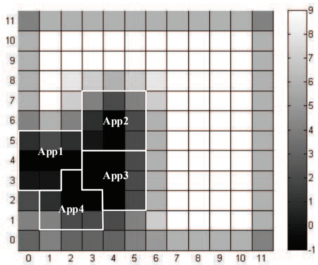
MapPro Objectives:

- Maximize performance for all applications;
- Minimize communication latency in the new application;
- Minimize fragmentation.

Mohammad-Hashem Haghbayan et al. "MapPro: Proactive Runtime Mapping for Dynamic Workloads by Quantifying Ripple Effect of Applications on Networks-on-Chip". In: *Proceedings of the International Symposium on Networks on Chip*. Vancouver, Canada, Sept. 2015

Example 1: Performance Driven Task Mapping

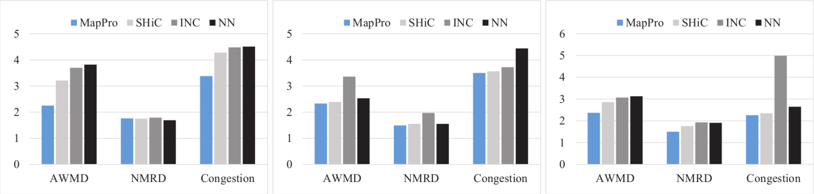




MapPro: Heuristic to minimize application internal communication delay and to minimize fragmentation.

1. First Node selection: Identifies a first node and a region for a new application;
2. Allocates specific cores around the first node;
3. Maps tasks to cores.

Example 1: Performance Driven Task Mapping



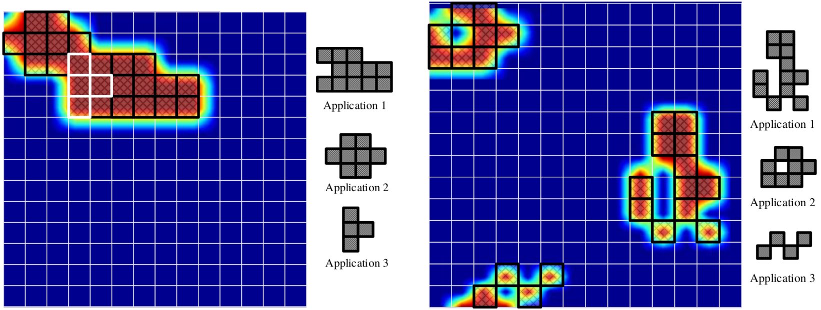
Experiments with 12x12 - 16x16 networks.

AWMD: Average Weighed Manhattan Distance: Measures the communication cost based on traffic volume.

NMRD: Normalized Mapped Region Dispersion is the normalized average of pairwise Manhattan distances of all communication nodes of a mapped application: measures the compactness of a region.

External Congestion: Number of contended packets belonging to different applications.

Example 2: Power- and Thermal Constrained Task Mapping

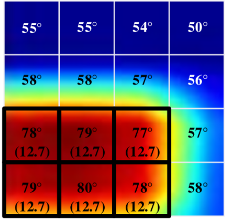


The patterning algorithm disperses mapped cores to maximize the Thermal Safe Power budget.

Anil Kanduri et al. “Dark Silicon Aware Runtime Mapping for Many-core Systems: A Patterning Approach”. In: *Proceedings of the International Conference on Computer Design (ICCD)*. New York City, USA, Oct. 2015, pp. 610–617

Example 2: Efficient Budgeting

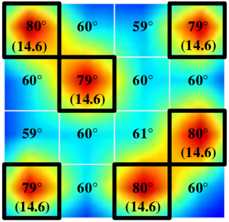
Tightly packed Cores



Neighbors accumulating temperature

Utilized Power Budget = **76.2 W**

Spreadout Cores



Neighbors dissipating temperature

Utilized Power Budget = **87.6 W**

- ✓ 15% Better Utilization
- ✓ Activate more cores
- ✓ Reduce temperatures
- ✓ Minimize Dark Silicon

Example 2: Power Budget Improvement

Percentage Power Budget Improvement for PAT over SC

Network Size	90% Dark		75% Dark		50% Dark	
	Avg.	Best	Avg.	Best	Avg.	Best
16x16	5.74	13.9	4.15	11.3	2.19	7.68
20x20	6.54	17.17	5.06	8.55	2.63	4.28

Percentage Power Budget Improvement for PAT over TSP-WC

Network Size	90% Dark		75% Dark		50% Dark	
	Avg.	Best	Avg.	Best	Avg.	Best
16x16	32.33	34.92	22.02	24.14	11.73	13.2
20x20	38.70	40.83	22.40	27.4	12.5	13.33

Example 2: Throughput Gain

Percentage Throughput gain for PAT over SC

Network Size	90% Dark		75% Dark		50% Dark	
	Avg.	Best	Avg.	Best	Avg.	Best
16x16	7.27	15.64	4.59	13.92	2.42	8.58
20x20	8.5	20.99	5.88	10.21	2.89	4.54

- ✓ Surplus Budget
- ➢ Added latency
- ➢ Per Application Latency
- ✓ Minimal congestion
- ✓ Per Chip Throughput

Example 3: Lifetime-Reliability-Driven Task Mapping

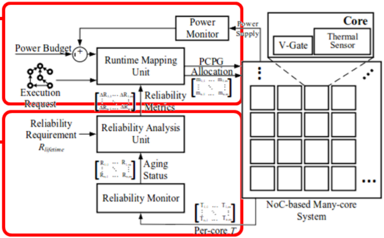
- To main limitations of many-cores:
    - Not enough power to turn on all cores (dark silicon)
    - Increased susceptibility of IC to aging and wear-out
- Goal: Introduce lifetime reliability awareness in the runtime resource management layer
    - Guarantee specified level of reliability
    - Satisfy the power budget
    - Optimize performance

M. H. Haghbayan et al. "A lifetime-aware runtime mapping approach for many-core systems in the dark silicon era". In: *Design, Automation Test in Europe Conference Exhibition (DATE)*. Mar. 2016, pp. 854–857

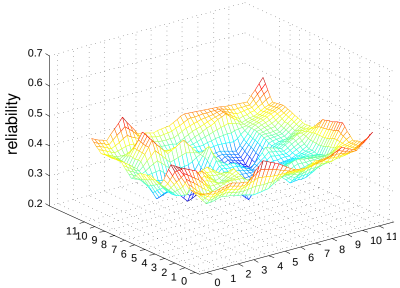
Example 3: Lifetime-Reliability-Driven Task Mapping

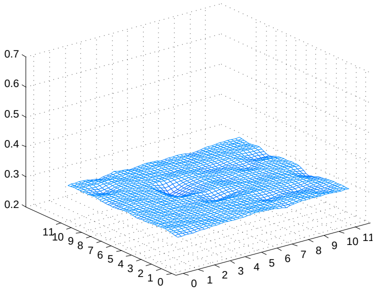
Proposed approach based on two feedback controllers

- Short-term controller
  - Application mapping
    - Select less aged cores
  - Power control
- Long-term controller
  - Reliability management
    - Compute current aging status
    - Disable highly stressed cores



Example 3: Lifetime-Reliability-Driven Task Mapping





MapPro:
lifetime=5.52 years

Reliability aware mapping:
lifetime=12 years

The plots show the reliability of cores at the end of the system's lifetime.
The end of the system's life is reached when the reliability of one core drops below 30%.

Challenges in Complex Many-Core SoCs

- A number and variety of objectives
    - Partially contradicting
    - At different time scales
- Objectives change over time
- The system state has to be known
- Application objectives have to be known

Goal Management Levels

1. Single objective; Design time;

Goal Management Levels

1. Single objective; Design time;
2. Multiple objectives; Design time;

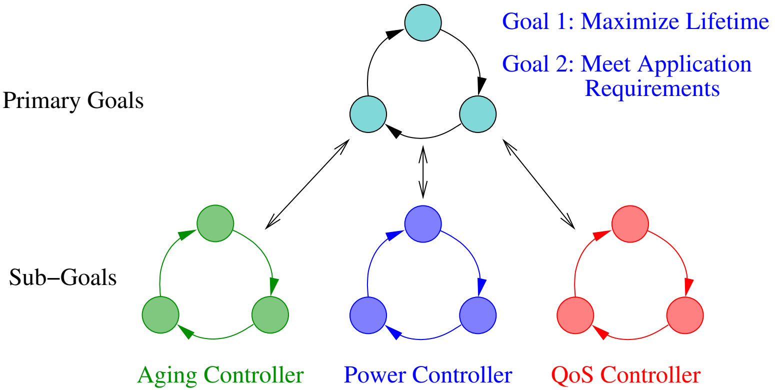
Goal Management Levels

1. Single objective; Design time;
2. Multiple objectives; Design time;
3. Multiple objectives; Run time;

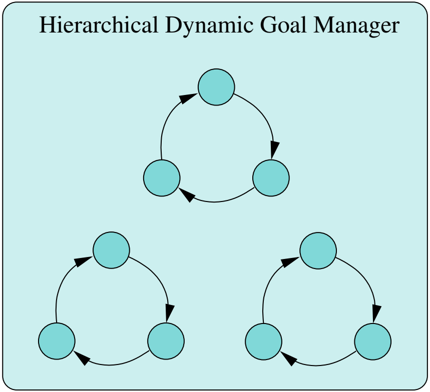
Goal Management Levels

1. Single objective; Design time;
2. Multiple objectives; Design time;
3. Multiple objectives; Run time;
4. Multiple, hierarchical objectives; Run time;

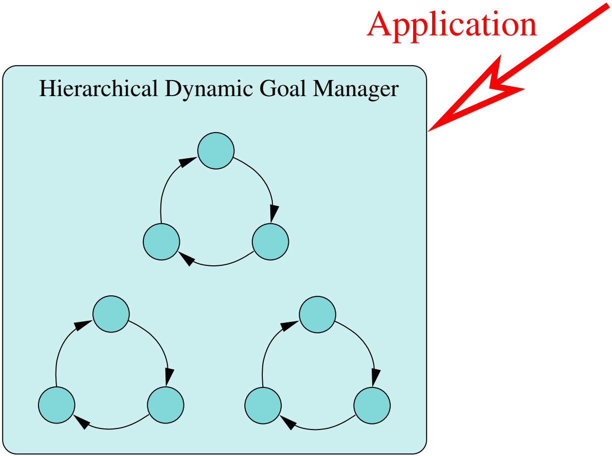
Hiararchical Goal Management



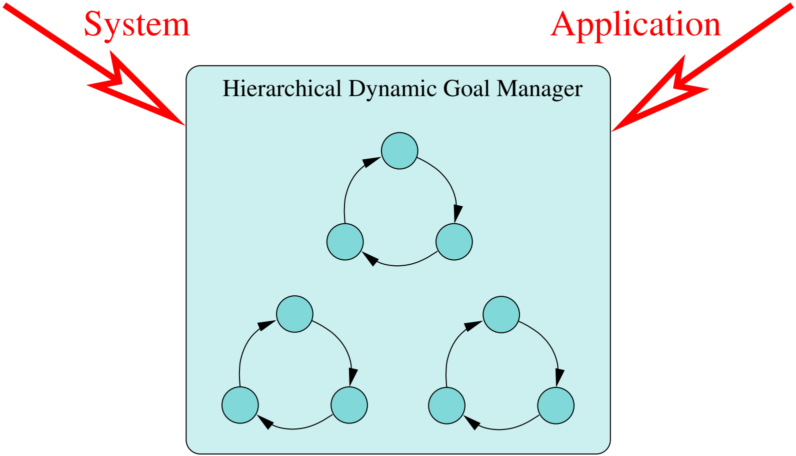
Goal Management Inputs



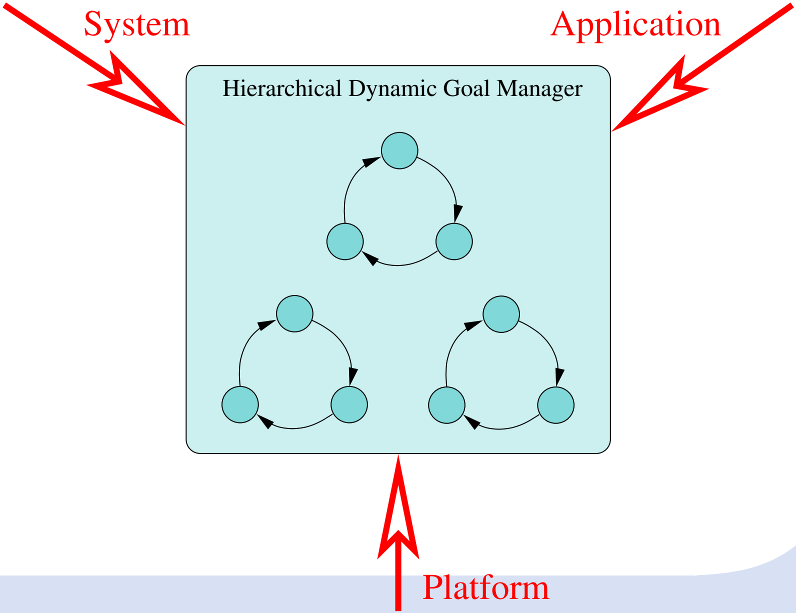
Goal Management Inputs



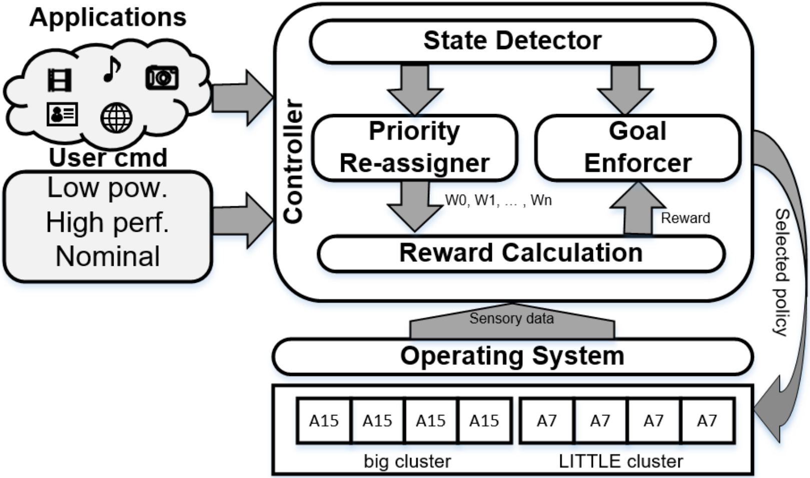
Goal Management Inputs



Goal Management Inputs



Goal Driven Autonomy



Elham Shamsa et al. "Goal-Driven Autonomy for Efficient On-chip Resource Management: Transforming Objectives to Goals". In: *Proceedings of the Design and Test Europe Conference (DATE)*. Florence, Italy, Mar. 2019

Terminology

Agent is an actor in the system, that pursues specific objectives. $\mathcal{B} = \{B_1, B_2, B_3\}$

Terminology

Agent is an actor in the system, that pursues specific objectives. $\mathcal{B} = \{B_1, B_2, B_3\}$

Application is an application running on the system. $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$.

Terminology

Agent is an actor in the system, that pursues specific objectives. $\mathcal{B} = \{B_1, B_2, B_3\}$

Application is an application running on the system. $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$.

Parameter: are entities measured and subject to control, like power consumption and application performance. E.g. $\mathcal{P}_{\text{pow}}(\text{core1})$,
$\mathcal{P}_{\text{pow}}(\text{PLATFORM})$,
$\mathcal{P}_{\text{perf}}(A_2)$.

Terminology

Objective function: either minimizes or maximizes a parameter or puts a constraint on a parameter. E.g.

$o_1 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \rightarrow \text{min},$

$o_2 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq C_1,$

$o_3 := \mathcal{P}_{\text{perf}}(A_1) \geq C_2.$

Terminology

Objective function: either minimizes or maximizes a parameter or puts a constraint on a parameter. E.g.

$o_1 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \to \min,$

$o_2 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq C_1,$

$o_3 := \mathcal{P}_{\text{perf}}(A_1) \geq C_2.$

Objective is a set of objective functions, e.g. $O = \{o_1, o_2\}$.

Terminology

Objective function: either minimizes or maximizes a parameter or puts a constraint on a parameter. E.g.

$o_1 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \rightarrow \min,$

$o_2 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq C_1,$

$o_3 := \mathcal{P}_{\text{perf}}(A_1) \geq C_2.$

Objective is a set of objective functions, e.g. $O = \{o_1, o_2\}$.

$O(B_1)$ ... objective of agent B_1

Terminology

Objective function: either minimizes or maximizes a parameter or puts a constraint on a parameter. E.g.

$o_1 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \rightarrow \min,$

$o_2 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq C_1,$

$o_3 := \mathcal{P}_{\text{perf}}(A_1) \geq C_2.$

Objective is a set of objective functions, e.g. $O = \{o_1, o_2\}$.

$O(B_1)$ ... objective of agent B_1

$O(A_1)$ ... objective of application A_1.

Terminology

Objective function: either minimizes or maximizes a parameter or puts a constraint on a parameter. E.g.

$o_1 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \rightarrow \min,$

$o_2 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq C_1,$

$o_3 := \mathcal{P}_{\text{perf}}(A_1) \geq C_2.$

Objective is a set of objective functions, e.g. $O = \{o_1, o_2\}$.

$O(B_1)$	... objective of agent B_1
$O(A_1)$	... objective of application A_1.
$O_{\mathcal{P}}(\mathcal{B}, \mathcal{A})$	... set of objective functions of agents $\mathcal{B}$ and applications $\mathcal{A}$ relevant for parameter $\mathcal{P}$.

Terminology

Objective function: either minimizes or maximizes a parameter or puts a constraint on a parameter. E.g.
$o_1 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \to \min$,
$o_2 := \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq C_1$,
$o_3 := \mathcal{P}_{\text{perf}}(A_1) \geq C_2$.

Objective is a set of objective functions, e.g. $O = \{o_1, o_2\}$.

$O(B_1)$ ... objective of agent B_1

$O(A_1)$ ... objective of application A_1.

$O_{\mathcal{P}}(\mathcal{B}, \mathcal{A})$ ... set of objective functions of agents $\mathcal{B}$ and applications $\mathcal{A}$ relevant for parameter $\mathcal{P}$.

$O_{\textbf{pow}}(\{B_2, B_4\}, \{A_1, A_3\})$ ... set of objective functions of agents B_2 and B_4 and applications A_1 and A_3 relevant to power.

Terminology

Hierarchy Level: is a number assigned to actors; the higher the level, the more important is the actor.

Terminology

Hierarchy Level: is a number assigned to actors; the higher the level, the more important is the actor.
E.g. $H(B_1) = 3$,

Terminology

Hierarchy Level: is a number assigned to actors; the higher the level, the more important is the actor.
E.g. $H(B_1) = 3$,
$H(\text{PLATFORM}) = 2$,

Terminology

Hierarchy Level: is a number assigned to actors; the higher the level, the more important is the actor.
E.g. $H(B_1) = 3$,
$H(\text{PLATFORM}) = 2$,
$H_P(\mathcal{B})$ is the highest hierarchy level of any agent in $\mathcal{B}$ which includes an objective function relevant for parameter P:

Terminology

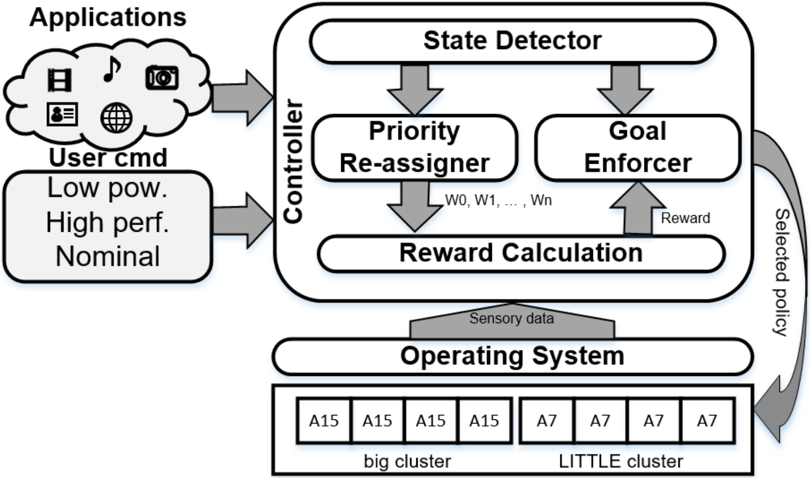
Hierarchy Level: is a number assigned to actors; the higher the level, the more important is the actor.
E.g. $H(B_1) = 3$,
$H(\textsc{Platform}) = 2$,
$H_P(\mathcal{B})$ is the highest hierarchy level of any agent in $\mathcal{B}$ which includes an objective function relevant for parameter P:

$$H_{\mathcal{P}}(\mathcal{B}) = \max_{B \in \mathcal{B}}(H(B)) \text{ for which } O_{\mathcal{P}}(\{B\}, \{\}) \neq \{\}$$

Terminology

Priority of a parameter is the highest hierarchy level of the involved agents: $\boldsymbol{P}_{\mathcal{P}} = H_{\mathcal{P}}$.

SoC Example



Applications
User cmd
Low pow.
High perf.
Nominal
Controller
State Detector
Priority Re-assigner
Goal Enforcer
W0, W1, ... , Wn
Reward
Reward Calculation
Selected policy
Sensory data
Operating System
A15
A15
A15
A15
A7
A7
A7
A7
big cluster
LITTLE cluster

SoC Example

Agents: $\mathcal{B} = \{\mathsf{USER}, \mathsf{PLATFORM}, \mathsf{APPLICATION}\}$

SoC Example

Agents: $\mathcal{B} = \{\text{USER}, \text{PLATFORM}, \text{APPLICATION}\}$

Applications: There ar n applications active:

$$\mathcal{A} = \{A_1, A_2, \ldots, A_n\}.$$

SoC Example

Agents: $\mathcal{B} = \{\text{USER}, \text{PLATFORM}, \text{APPLICATION}\}$

Applications: There ar n applications active:

$$\mathcal{A} = \{A_1, A_2, \ldots, A_n\}.$$

Parameters: $\mathcal{P}_{\text{pow}}, \mathcal{P}_{\text{perf}}$

SoC Example

Agents: $\mathcal{B} = \{\text{USER}, \text{PLATFORM}, \text{APPLICATION}\}$

Applications: There ar n applications active:

$$\mathcal{A} = \{A_1, A_2, \ldots, A_n\}.$$

Parameters: $\mathcal{P}_{\text{pow}}, \mathcal{P}_{\text{perf}}$

Platform objectives:

$$O(\text{PLATFORM}) = \{\mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq \text{TDP}\}$$

SoC Example

Agents: $\mathcal{B} = \{\text{USER}, \text{PLATFORM}, \text{APPLICATION}\}$

Applications: There ar n applications active:

$$\mathcal{A} = \{A_1, A_2, \ldots, A_n\}.$$

Parameters: $\mathcal{P}_{\text{pow}}, \mathcal{P}_{\text{perf}}$

Platform objectives:

$$O(\text{PLATFORM}) = \{\mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq \text{TDP}\}$$

User Objectives:

$$O(\text{USER}) = \begin{cases} \{\mathcal{P}_{\text{perf}}(\text{PLATFORM}) \rightarrow \max\} \\ \quad \text{if user command = "High Performance"} \\ \{\mathcal{P}_{\text{pow}}(\text{PLATFORM}) \rightarrow \min\} \\ \quad \text{if user command = "Low Power"} \end{cases}$$

SoC Example

Agents: $\mathcal{B} = \{\text{USER}, \text{PLATFORM}, \text{APPLICATION}\}$

Applications: There ar n applications active:

$$\mathcal{A} = \{A_1, A_2, \ldots, A_n\}.$$

Parameters: $\mathcal{P}_{\text{pow}}, \mathcal{P}_{\text{perf}}$

Platform objectives:

$$O(\text{PLATFORM}) = \{\mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq \text{TDP}\}$$

User Objectives:

$$O(\text{USER}) = \begin{cases} \{\mathcal{P}_{\text{perf}}(\text{PLATFORM}) \to \max\} \\ \quad \text{if user command = "High Performance"} \\ \{\mathcal{P}_{\text{pow}}(\text{PLATFORM}) \to \min\} \\ \quad \text{if user command = "Low Power"} \end{cases}$$

Application Objectives: A minimum ($C_A^{\min}$) and a maximum ($C_A^{\max}$) performance constraint is given for each application A:

$$O_A = \{\mathcal{P}_{\text{perf}}(A) \leq C_A^{\max}, \mathcal{P}_{\text{perf}}(A) \geq C_A^{\min}\}$$

State Detection

State vector:

- **Power**: Violation: TDP$< p$
  Potential Violation: $0.8\,\text{TDP} \leq p \leq \text{TDP}$
  No Violation: $p \leq 0.8\,\text{TDP}$

- **User Command**: High Performance
  Low Power

- **Performance per application**: [Min run time,
  Max run time]

Goal Hierarchy

$$
\begin{aligned}
H(\textsc{platform}) &= \begin{cases} 5 & \text{if } \mathcal{P}_{\text{pow,cur}} > 0.9\text{TDP} \\ 2.5 & \text{if } 0.9\text{TDP} > \mathcal{P}_{\text{pow,cur}} > 0.8\,\text{TDP} \\ 1 & \text{if } \mathcal{P}_{\text{pow,cur}} < 0.8\text{TDP} \end{cases} \\
H(\textsc{user}) &= 2 \\
H(\textsc{application}) &= (1 + \frac{n_{viol}}{n})
\end{aligned}
$$

Priority Assignment

- Primary goals: thermal safety
- Secondary goals: User experience
- Tertiary goals: Application requirements

Priority

$$\boldsymbol{P}_{\mathcal{P}_{\text{pow}}} = H_{\mathcal{P}_{\text{pow}}}$$

Priority

$$\boldsymbol{P}_{\mathcal{P}_{\text{pow}}} = H_{\mathcal{P}_{\text{pow}}} = \max(H(\textsf{USER}), H(\textsf{PLATFORM}))$$

Priority

$$\begin{aligned} \boldsymbol{P}_{\mathcal{P}_{\text{pow}}} &= H_{\mathcal{P}_{\text{pow}}} = \max(H(\text{USER}), H(\text{PLATFORM})) \\ \boldsymbol{P}_{\mathcal{P}_{\text{perf}}}(A) &= H_{\mathcal{P}_{\text{perf}}} \end{aligned}$$

Priority

$$
\begin{aligned}
\boldsymbol{P}_{\mathcal{P}_{\text{pow}}} &= H_{\mathcal{P}_{\text{pow}}} = \max(H(\text{USER}), H(\text{PLATFORM}) \\
\boldsymbol{P}_{\mathcal{P}_{\text{perf}}}(A) &= H_{\mathcal{P}_{\text{perf}}} \\
&= \begin{cases} \max(H(\text{USER}), H(\text{APPLICATION})) \\ \qquad\qquad \text{(if User Command = ""High Performance")} \\ \\ H(\text{APPLICATION})) \\ \qquad\qquad \text{(if User Command = "Low Power")} \end{cases}
\end{aligned}
$$

Goal Enforcement

- Selects action that most likely will satisfy the highest priority goal;
- Action = Resource allocation policy;
- Initial action is randomly selected;
- Actions are assessed in a reinforcement learning loop;
- Reinforcement learning is based on a reward function.

Rewards

- A function is assigned to every objective function.
- A $[\mathrm{min}, \mathrm{max}]$ interval is assumed for each reward function.
- The reward function is normalized to $[0, 1] \rightarrow [0.1]$.
- For minimizing and maximizing a linear reward function is used.
- For bounds a variant of the generalized logistic function is used:
  - $R(x) = \frac{1}{(1+e^{B(x-A)})^C}$
  - For lower bound constraints: $A = 0.1, B = -10, C = 1$.
  - For upper bound constraints: $A = 0.9, B = 10, C = 1$.

Reward Functions

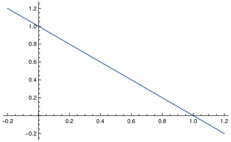
Objective function Reward function

y' is y normalized to $[0, 1]$.

Reward Functions

Objective function | Reward function

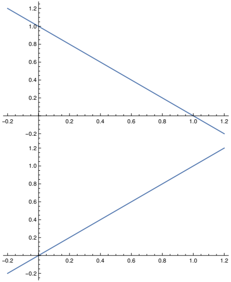
$y = f(x) \to \min$ | $R_{\min}(y') = -y'$



y' is y normalized to $[0, 1]$.

Reward Functions

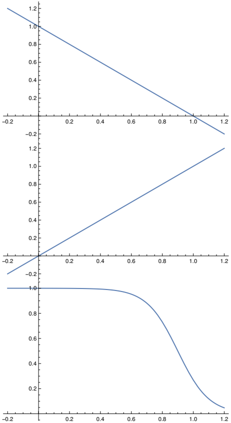
Objective function	Reward function
$y = f(x) \to \min$	$R_{\min}(y') = -y'$
$y = f(x) \to \max$	$R_{\max}(y') = y'$



y' is y normalized to $[0, 1]$.

Reward Functions

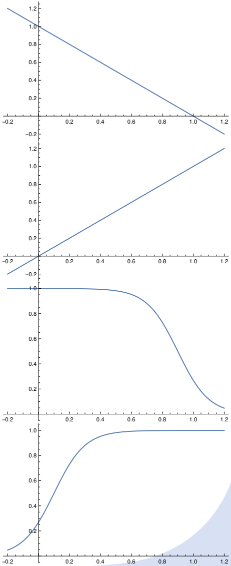
Objective function	Reward function
$y = f(x) \to \min$	$R_{\min}(y') = -y'$
$y = f(x) \to \max$	$R_{\max}(y') = y'$
$y = f(x) \leq C_{\max}$	$R_{\mathrm{ub}}(y') = \frac{1}{(1+e^{10(y'-0.9)})^1}$



y' is y normalized to $[0, 1]$.

Reward Functions

Objective function	Reward function
$y = f(x) \rightarrow \min$	$R_{\text{min}}(y') = -y'$
$y = f(x) \rightarrow \max$	$R_{\text{max}}(y') = y'$
$y = f(x) \leq C_{\text{max}}$	$R_{\text{ub}}(y') = \frac{1}{(1+e^{10(y'-0.9)})^1}$
$y = f(x) \geq C_{\text{min}}$	$R_{\text{lb}}(y') = \frac{1}{(1+e^{-10(y'-0.1)})^1}$



y' is y normalized to $[0, 1]$.

Reward Calculation

$$Reward = W_0 R_0 + W_1 R_1 + W_2 R_2 + ... + W_n R_n$$

With the objective functions for power and performance:

$$R = W_{\mathcal{P}_{\text{pow}}} \cdot R_{\mathcal{P}_{\text{pow}}} + \sum_{A \in \mathcal{A}} W_{\mathcal{P}_{\text{perf}}}(A) \cdot R_{\mathcal{P}_{\text{perf}}}(A)$$

Reward Calculation for Power

Objective Functions for power with user command “Low Power”:

$$
\begin{aligned}
O_{\mathcal{P}_{\text{pow}}}(\mathcal{B},\mathcal{A}) &= O(\text{PLATFORM}) \cup O(\text{USER}) \\
&= \{\mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq \text{TDP}, \\
&\quad \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \rightarrow \min\}
\end{aligned}
$$

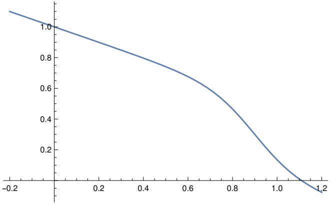
Reward Calculation for Power

Objective Functions for power with user command “Low Power”:

$$
\begin{aligned}
O_{\mathcal{P}_{\text{pow}}}(\mathcal{B},\mathcal{A}) &= O(\text{PLATFORM}) \cup O(\text{USER}) \\
&= \{\mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq \text{TDP}, \\
&\quad \mathcal{P}_{\text{pow}}(\text{PLATFORM}) \rightarrow \min\}
\end{aligned}
$$

Reward function:

$$
\begin{aligned}
R_{\mathcal{P}_{\text{pow}}} &= \frac{1}{2}(R_{\min}(y') + R_{\text{ub}}(y')) \\
&= \frac{1}{2}(-y + \frac{1}{1+e^{10((y'-0.9)}})
\end{aligned}
$$



where y' is the normalized $\mathcal{P}_{\text{pow,cur}}$.

Reward Calculation for Power

Objective Functions for power with user command “High Performance”:

$$\begin{aligned} O_{\mathcal{P}_{\text{pow}}}(\mathcal{B}, \mathcal{A}) &= O(\text{PLATFORM}) \cup O(\text{USER}) \\ &= \{\mathcal{P}_{\text{pow}}(\text{PLATFORM}) \leq \text{TDP}\} \end{aligned}$$

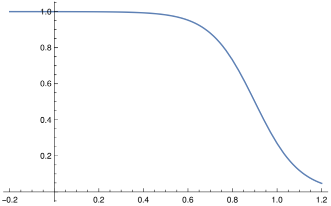
Reward Calculation for Power

Objective Functions for power with user command “High Performance”:

$$
\begin{aligned}
O_{\mathcal{P}_{\text{pow}}}(\mathcal{B},\mathcal{A}) &= O(\textsc{platform}) \cup O(\textsc{user}) \\
&= \{\mathcal{P}_{\text{pow}}(\textsc{platform}) \leq \text{TDP}\}
\end{aligned}
$$

Reward function:

$$
\begin{aligned}
R_{\mathcal{P}_{\text{pow}}} &= R_{\text{ub}}(y')) \\
&= \frac{1}{1 + e^{10((y'-0.9)}}
\end{aligned}
$$



where y' is the normalized $\mathcal{P}_{\text{pow,cur}}$.

Reward Calculation for Performance

Objective Functions for performance with user command “Low Power”:

$$
\begin{aligned}
O_{\mathcal{P}_{\mathrm{perf}}}(\mathcal{B},\{A\}) &= O_{\mathcal{P}_{\mathrm{perf}}}(\mathrm{USER}) \cup O_{\mathcal{P}_{\mathrm{perf}}}(\{A\}) \\
&= \{\mathcal{P}_{\mathrm{perf}}(A) \leq C_A^{\mathrm{max}}, \mathcal{P}_{\mathrm{perf}}(A) \geq C_A^{\mathrm{min}}\}
\end{aligned}
$$

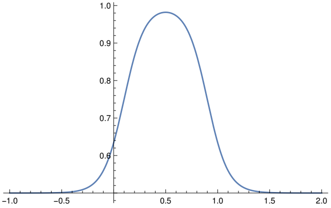
Reward Calculation for Performance

Objective Functions for performance with user command “Low Power”:

$$
\begin{aligned}
O_{\mathcal{P}_{\text{perf}}}(\mathcal{B}, \{A\}) &= O_{\mathcal{P}_{\text{perf}}}(\text{USER}) \cup O_{\mathcal{P}_{\text{perf}}}(\{A\}) \\
&= \{\mathcal{P}_{\text{perf}}(A) \leq C_A^{\text{max}}, \mathcal{P}_{\text{perf}}(A) \geq C_A^{\text{min}}\}
\end{aligned}
$$

Reward function for A:

$$
\begin{aligned}
R_{\mathcal{P}_{\text{perf}}}(A) &= \frac{1}{2}(R_{\text{lb}}(y') + R_{\text{ub}}(y')) \\
&= \frac{1}{2}(\frac{1}{1 + e^{10((y'-0.9)}} \\
&\quad + \frac{1}{1 + e^{-10((y'-0.1)}})
\end{aligned}
$$



where y' is the normalized $\mathcal{P}_{\text{perf,cur}}(A)$.

Reward Calculation for Performance

Objective Functions for performance with user command "High Performance":

$$
\begin{aligned}
O_{\mathcal{P}_{\text{perf}}}(\mathcal{B}, \{A\}) &= O_{\mathcal{P}_{\text{perf}}}(\mathsf{USER}) \cup O_{\mathcal{P}_{\text{perf}}}(\{A\}) \\
&= \{\mathcal{P}_{\text{perf}}(A) \rightarrow \max, \\
&\quad \mathcal{P}_{\text{perf}}(A) \leq C_A^{\max}, \mathcal{P}_{\text{perf}}(A) \geq C_A^{\min}\}
\end{aligned}
$$

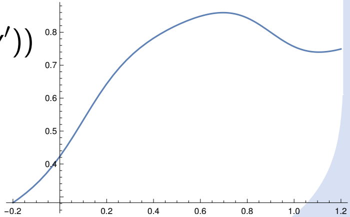
Reward Calculation for Performance

Objective Functions for performance with user command “High Performance”:

$$
\begin{aligned}
O_{\mathcal{P}_{\text{perf}}}(\mathcal{B}, \{A\}) &= O_{\mathcal{P}_{\text{perf}}}(\text{USER}) \cup O_{\mathcal{P}_{\text{perf}}}(\{A\}) \\
&= \{\mathcal{P}_{\text{perf}}(A) \rightarrow \max, \\
&\quad \mathcal{P}_{\text{perf}}(A) \leq C_A^{\max}, \mathcal{P}_{\text{perf}}(A) \geq C_A^{\min}\}
\end{aligned}
$$

Reward function for A:

$$
\begin{aligned}
R_{\mathcal{P}_{\text{perf}}}(A) &= \frac{1}{3}(R_{\max}(y') + R_{\text{lb}}(y') + R_{\text{ub}}(y')) \\
&= \frac{1}{3}(y' + \frac{1}{1 + e^{10((y'-0.9)}} \\
&\quad + \frac{1}{1 + e^{-10((y'-0.1)}})
\end{aligned}
$$



where y' is the normalized $\mathcal{P}_{\text{perf,cur}}(A)$.

Reward Calculation

$$\begin{aligned} R &= W_0 \times R_0 + W_1 \times R_1 + W_2 \times R_2 + \ldots + W_n \times R_n \\ &= W_{\mathcal{P}_{\text{pow}}} \cdot R_{\mathcal{P}_{\text{pow}}} + \sum_{A \in \mathcal{A}} W_{\mathcal{P}_{\text{perf}}}(A) \cdot R_{\mathcal{P}_{\text{perf}}}(A) \end{aligned}$$

Reward Calculation

$$
\begin{aligned}
R &= W_0 \times R_0 + W_1 \times R_1 + W_2 \times R_2 + \ldots + W_n \times R_n \\
&= W_{\mathcal{P}_{\text{pow}}} \cdot R_{\mathcal{P}_{\text{pow}}} + \sum_{A \in \mathcal{A}} W_{\mathcal{P}_{\text{perf}}}(A) \cdot R_{\mathcal{P}_{\text{perf}}}(A)
\end{aligned}
$$

For the weights we use priorities:

$$
\begin{aligned}
\boldsymbol{P}_{\mathcal{P}_{\text{pow}}} &= H_{\mathcal{P}_{\text{pow}}} = \max(H(\text{USER}), H(\text{PLATFORM})) \\
\boldsymbol{P}_{\mathcal{P}_{\text{perf}}}(A) &= H_{\mathcal{P}_{\text{perf}}}
\end{aligned}
$$

Reward Calculation

$$
\begin{aligned}
R &= W_0 \times R_0 + W_1 \times R_1 + W_2 \times R_2 + \ldots + W_n \times R_n \\
&= W_{\mathcal{P}_{\text{pow}}} \cdot R_{\mathcal{P}_{\text{pow}}} + \sum_{A \in \mathcal{A}} W_{\mathcal{P}_{\text{perf}}}(A) \cdot R_{\mathcal{P}_{\text{perf}}}(A)
\end{aligned}
$$

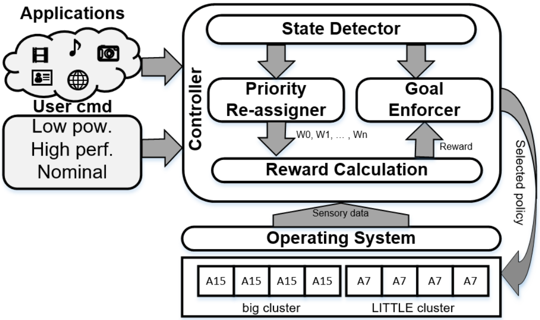
For the weights we use priorities:

$$
\begin{aligned}
\boldsymbol{P}_{\mathcal{P}_{\text{pow}}} &= H_{\mathcal{P}_{\text{pow}}} = \max(H(\text{USER}), H(\text{PLATFORM})) \\
\boldsymbol{P}_{\mathcal{P}_{\text{perf}}}(A) &= H_{\mathcal{P}_{\text{perf}}}
\end{aligned}
$$

Thus:

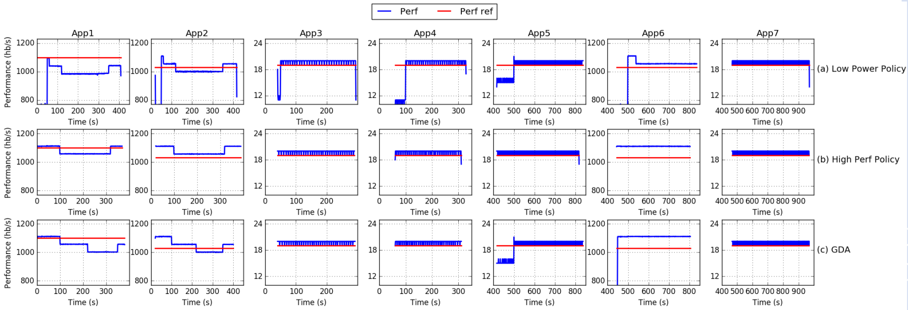
$$
R = \boldsymbol{P}_{\mathcal{P}_{\text{pow}}} \cdot R_{\mathcal{P}_{\text{pow}}} + \sum_{A \in \mathcal{A}} \boldsymbol{P}_{\mathcal{P}_{\text{perf}}}(A) \cdot R_{\mathcal{P}_{\text{perf}}}(A)
$$

Q-Learning



- Actions are task migration, cluster DVFS;
- Rewards are updated;
- Actions with highest rewards are executed;
- Initially, actions are selected randomly.

Experiments

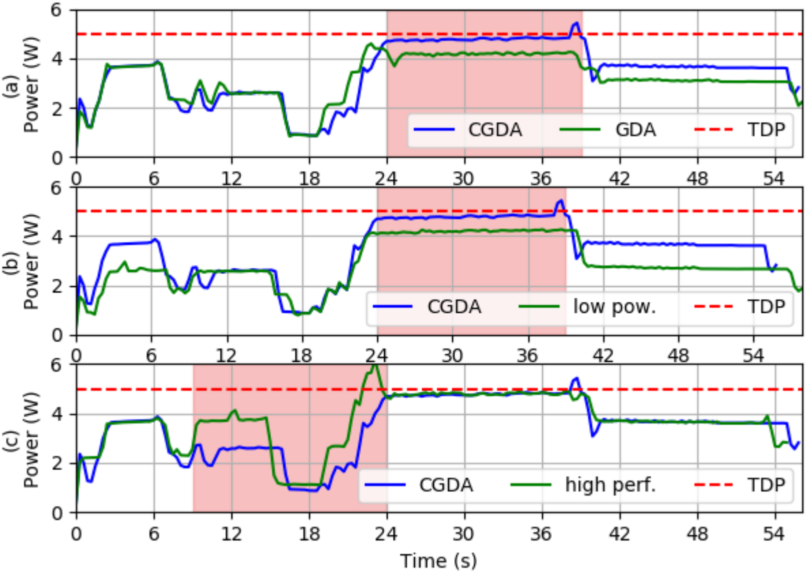


Experiments with a set of microkernel benchmarks;
Hardkernel Odroid XU3 board,
with two clusters (4 big (A15) and 4 little (A7) CPU cores);
Performance in heartbeats/sec.

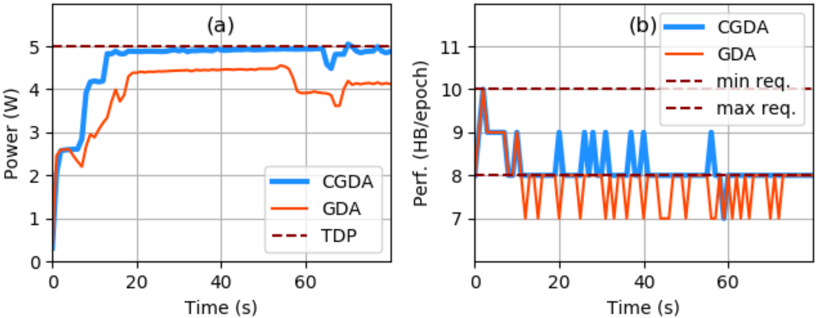
Comparison

Tech.	Obj	Cmd	Pwr viol.	Perf. viol.	Avg. pwr (W)
LP	Power	X	0%	27%	2.86
HP	Perf.	X	3%	0%	3.7
GDA	Dynamic	✓	0%	14%	3.1
CGDA	Dynamic	✓	1%	2%	3.4

Experiments - Power Evaluation

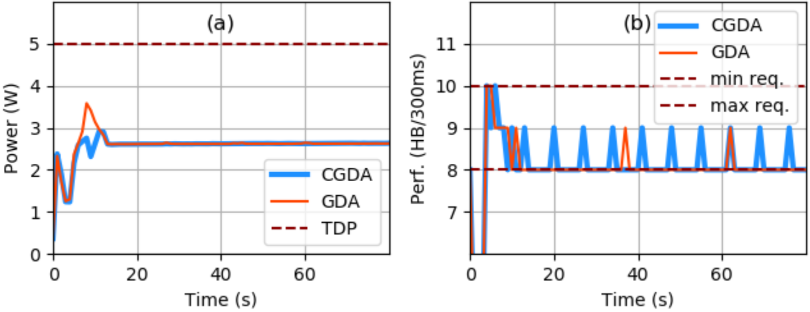


Experiments - Load Evaluation I



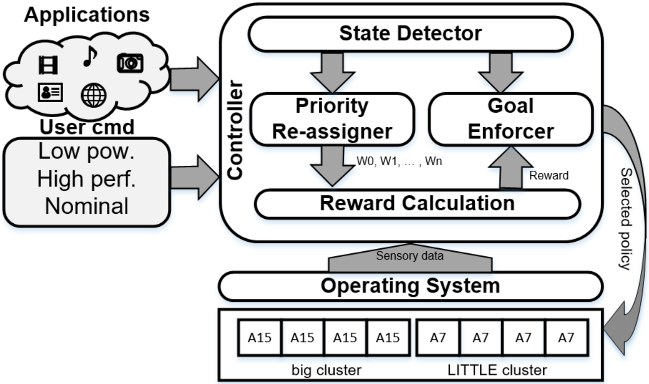
High Load Scenario

Experiments - Load Evaluation II



Low Load Scenario

Goal Driven Autonomy



Elham Shamsa et al. "Goal-Driven Autonomy for Efficient On-chip Resource Management: Transforming Objectives to Goals". In: *Proceedings of the Design and Test Europe Conference (DATE)*. Florence, Italy, Mar. 2019

Axel Jantsch et al. "Hierarchical Dynamic Goal Management for IoT Systems". In: *Proceedings of the IEEE International Symposium on Quality Electronic Design (ISQED 2018)*. USA, Mar. 2018

Amir M. Rahmani, Axel Jantsch, and Nikil Dutt. "HDGM: Hierarchical Dynamic Goal Management for Many-Core Resource Allocation". In: *IEEE Embedded Systems letters* 10.3 (Sept. 2018)

Self-Aware CPS

Summary

- Observation, Model building, Learning
- Goal management
  - Framework for managing various different goals and objectives;
  - Goals can dynamically change;
  - Actions are improved during operation based on reinforcement learning.

Questions ?

References I

Robin Arbaud, Dávid Juhász, and Axel Jantsch. "Management of Resources for Mixed-Critical Systems on Multi-Core Platforms with explicit consideration of Communication". In: *Proceedings of the Euromicro Conference on Digital System Design (DSD)*. invited tutorial. Sept. 2018.

Nikil Dutt, Axel Jantsch, and Santanu Sarma. "Towards Smart Embedded Systems: A Self-Aware System-on-Chip Perspective". In: *ACM Transactions on Embedded Computing Systems, Special Issue on Innovative Design Methods for Smart Embedded Systems* 15.2 (Feb. 2016). invited, pp. 22–27.

Nikil Dutt, Amir M. Rahmani, and Axel Jantsch. "Empowering Autonomy through Self-awareness in MPSoCs". In: *Proceedings of the IEEE NEWCAS Conference*. Strasbourg, France, June 2017.

Mohammad-Hashem Haghbayan et al. "MapPro: Proactive Runtime Mapping for Dynamic Workloads by Quantifying Ripple Effect of Applications on Networks-on-Chip". In: *Proceedings of the International Symposium on Networks on Chip*. Vancouver, Canada, Sept. 2015.

References II

M. H. Haghbayan et al. "A lifetime-aware runtime mapping approach for many-core systems in the dark silicon era". In: *Design, Automation Test in Europe Conference Exhibition (DATE)*. Mar. 2016, pp. 854–857.

Axel Jantsch et al. "Hierarchical Dynamic Goal Management for IoT Systems". In: *Proceedings of the IEEE International Symposium on Quality Electronic Design (ISQED 2018)*. USA, Mar. 2018.

Axel Jantsch, Nikil Dutt, and Amir M. Rahmani. "Self-Awareness in Systems on Chip – A Survey". In: *IEEE Design Test* 34.6 (Dec. 2017), pp. 1–19.

Axel Jantsch and Kalle Tammemäe. "A Framework of Awareness for Artificial Subjects". In: *Proceedings of the 2014 International Conference on Hardware/Software Codesign and System Synthesis*. CODES '14. New Delhi, India: ACM, 2014, 20:1–20:3.

Anil Kanduri et al. "Dark Silicon Aware Runtime Mapping for Many-core Systems: A Patterning Approach". In: *Proceedings of the International Conference on Computer Design (ICCD)*. New York City, USA, Oct. 2015, pp. 610–617.

S. Kounev et al., eds. *Self-Aware Computing Systems*. Springer, 2017.

References III

Hedyeh A. Kholerdi, Nima TaheriNejad, and Axel Jantsch. “Enhancement of Classification of Small Data Sets Using Self-awareness - An Iris Flower Case-Study”. In: *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*. Florence, Italy, May 2018.

Peter R. Lewis et al. “Architectural Aspects of Self-aware and Self-expressive Computing Systems”. In: *IEEE Computer* (Aug. 2015).

Peter R. Lewis et al., eds. *Self-Aware Computing Systems: An Engineering Approach*. Springer, 2016.

Kasra Moazzemi et al. “Trends in On-Chip Dynamic Resource Management”. In: *Proceedings of the Euromicro Conference on Digital System Design (DSD)*. invited. Prague, Czech Republic, Sept. 2018.

T. R. Mück et al. “Design Methodology for Responsive and Robust MIMO Control of Heterogeneous Multicores”. In: *IEEE Transactions on Multi-Scale Computing Systems* PP.99 (2018), pp. 1–1.

References IV

Amir M. Rahmani et al. “SPECTR - Formal Supervisory Control and Coordination for Many-core Systems Resource Management”. In: *Proceedings of the 23rd ACM International Conference on Architectural Support for Programming Languages and Operating Systems*. Williamsburg, VA, USA, Mar. 2018.

Amir M. Rahmani, Axel Jantsch, and Nikil Dutt. “HDGM: Hierarchical Dynamic Goal Management for Many-Core Resource Allocation”. In: *IEEE Embedded Systems letters* 10.3 (Sept. 2018).

Santanu Sarma et al. “On-Chip Self-Awareness Using Cyberphysical-Systems-On-Chip (CPSoC)”. In: *Proceedings of the 12th International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. New Delhi, India, Oct. 2014.

Elham Shamsa et al. “Goal Formulation: Abstracting Dynamic Objectives for Efficient On-chip Resource Allocation”. In: *IEEE Nordic Circuits and Systems Conference (NorCAS)*. Tallinn, Estonia, Oct. 2018.

Elham Shamsa et al. “Goal-Driven Autonomy for Efficient On-chip Resource Management: Transforming Objectives to Goals”. In: *Proceedings of the Design and Test Europe Conference (DATE)*. Florence, Italy, Mar. 2019.