# Power-Aware Container Placement Mechanism for CaaS Systems

Abdulelah Alwabel

Prince Sattam Bin Abdulaziz University

a.alwabel@psau.edu.sa

جامعة الأمير سطام بن عبدالعزيز
PRINCE SATTAM BIN ABDULAZIZ UNIVERSITY

IARIA

# The Author

- MSc in Internet Technologies with Security, Bristol University, UK.
- Ph.D. in Computer Science, Southampton University, UK.
- Assistant Professor, Department of Computer Science PSA University, KSA.
- Research interests:
  - Cloud computing: reliability, optimization and security
  - MEC & Fog computing,: optimization
  - IoT
- Lead consultation firm in Cloud and digital transformation.

# Agenda

- Background
- Problem Statement
- PCP Mechanism
- Experiments & Findings
- Conclusion & Future Work

# Background

- Container Technology is widely employed in Cloud systems as a Container-as-a-Service (CaaS) because containers are lightweight and portable, which drives their wide adoption in cloud systems.
- Container Placement (CP) is the mechanism that allocates containers to PMs within a CaaS system.
- Growing workload size and complexity make placement more challenging.
- Poor CP mechanism can increase:
  - Power consumption.
  - Operational cost
  - Resource under utilization
  - Performance degradation

# Problem Statement

- The challenge is to place containers on physical machines to:
  - Reduce power consumption
  - Maintain acceptable performance.
- Existing approaches often improve one objective at the expense of another.
- Need for a better balance between energy efficiency and performance

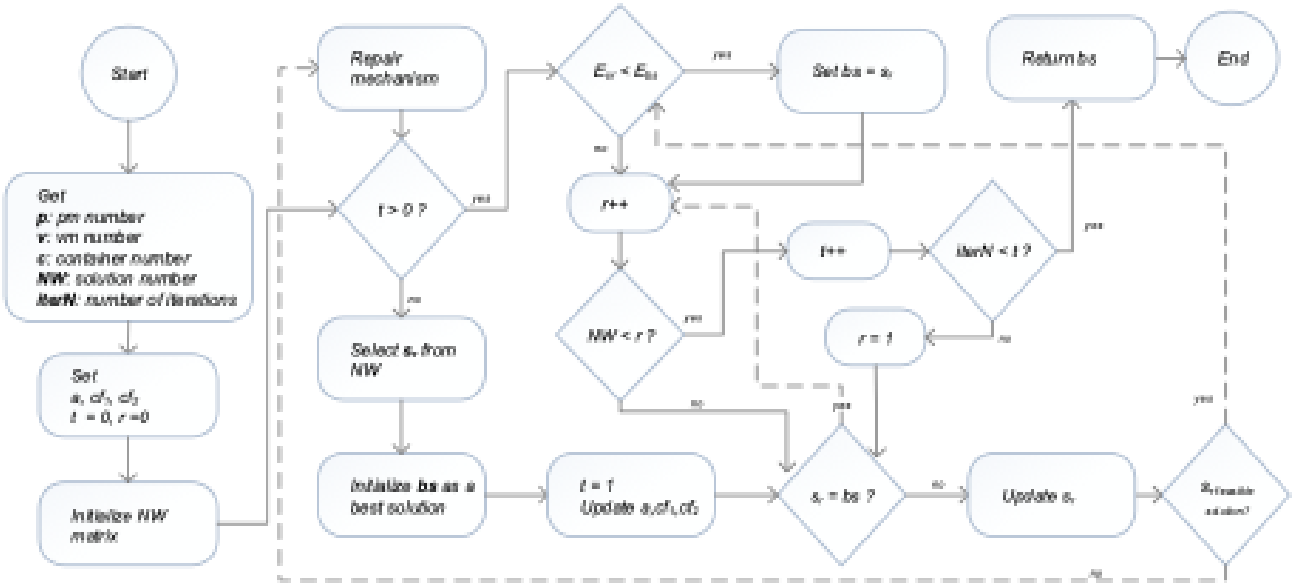
# PCP Mechanism

- Power-aware Container Placement (PCP) mechanism is developed to:
  - Reduce power consumption in CaaS systems.
  - Maintain an acceptable level of performance.
- PCP employs:
  - Whale Optimization Algorithm (WOA) to reduce power consumption
  - Apply an additional best-fit optimization phase

# WOA Algorithm

- It is a nature-inspired metaheuristic optimization algorithm proposed by Mirjalili and Lewis in 2016.

- WOA is popular because it is:
  - simple to implement
  - flexible
  - effective for many optimization problems




Source: Alwabel, A. (2023). A Novel Container Placement Mechanism Based on Whale Optimization Algorithm for CaaS Clouds. Electronics, 12(15), 3369. https://doi.org/10.3390/electronics12153369

# PCP Mechanism: Initial Placement

**Algorithm 1** PCP Mechanism

**input:** $c_l, p_l$
**initialize** $s_l$ as a list of $n_s$ random solutions
**for** $i$ to $itrN$ **do**
    $b \leftarrow findBestSolution(s_l)$
    **foreach** $s$ in $s_l$ **do** //update the current solution $s$
        Let $s_r$ be a random solution
        $cf_1 \leftarrow 2 \times a \times rand$
        $cf_2 \leftarrow 2 \times rand$
        Let $prob$ be a random number from 0 to 1
        **if** $prob < 0.5$ **then**
            $dir \leftarrow |cf_2 \times s_r - s|$
            **if** $|cf_1| < 1$ **then**
                $s \leftarrow b - cf_1 \times dir$
            **else**
                $s \leftarrow s_r - cf_1 \times dir$
            **end if**
        **else**
            $dir \leftarrow (|b - s|)$
            Let $z$ be a random number from $-1$ to 1
            $s \leftarrow dir \times e^z \times cos(2\pi z) \times s$
        **end if**
    **end for**
**end for**
$b \leftarrow optimizeSolution(b)$
**return:** $b$

# PCP Mechanism: Optimization Mechanism

**Algorithm 2** Optimization Mechanism

```
1:  input: b
2:  containerList ← b.removeContinaerList()
3:  pmList ← b.removePmList()
4:  sort containerList by CPU requirement ascending
5:  foreach container in containerList do
6:      foreach pm in pmList do
7:          if pm.isBestFit() then
8:              pm.host(container)
9:              pmList.add(pm)
10:         end if
11:     end for
12: end for
13: b.setPMList(pmList)
14: return: b
```

# Experiment Configuration

- The PCP was compared against EDCP mechanism
- Table 1 shows the experiment settings

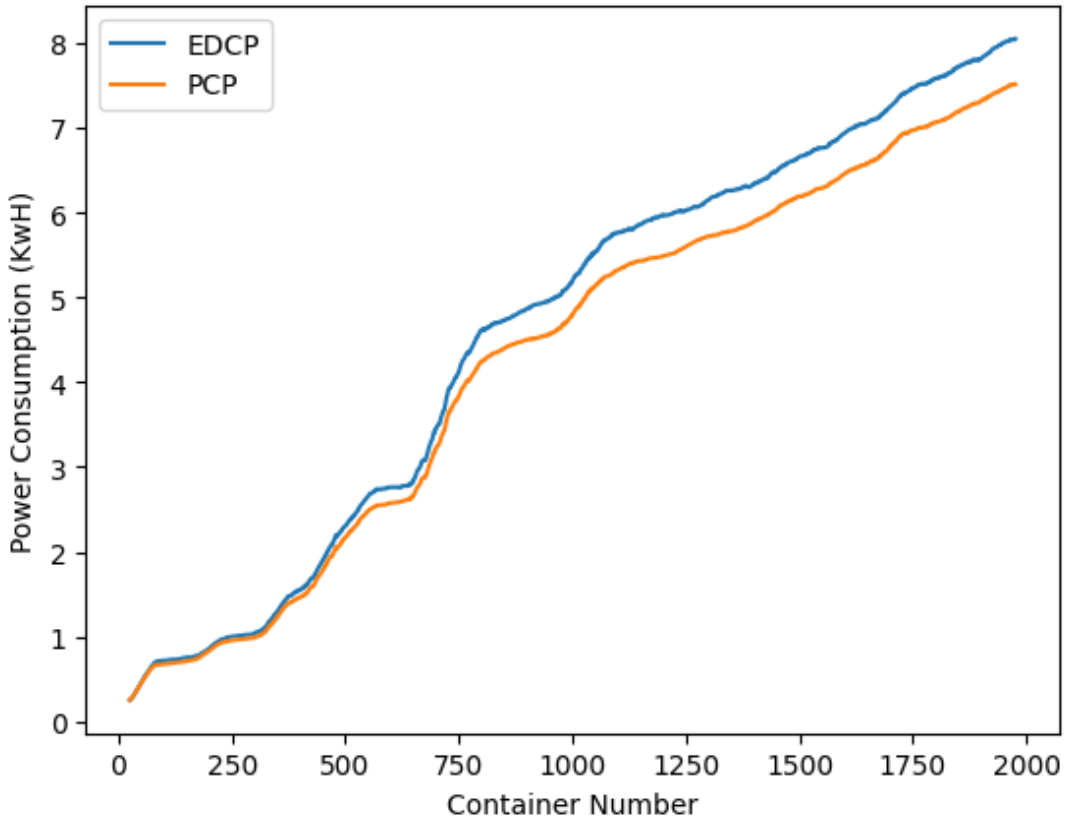
TABLE I. EXPERIMENT CONFIGURATIONS

| Parameter | Values |
|---|---|
| Container Specifications: | |
| Number | 2000 |
| CPU Requirements (MIPS) | 256, 512, 1024, 2048, 4096 |
| Memory Requirements (MB) | 128, 256, 512, 1024, 2048 |
| PM Specifications: | |
| No of PMs | 1000 |
| No of Cores | 6, 8, 20, 32, 56, 64 |
| CPU (GHz) | 2.2, 2.7, 2.9, 3.2 |
| Memory (GB) | 16, 128, 192, 512, 768 |
| $pm_{idle}$ (watt) | 15.6, 21.7, 48.6, 53.2, 127 |
| $pm_m$ (watt) | 58.9, 82.8, 269, 291, 377, 410 |
| Simulation Settings: | |
| $n_s$ | 5 |
| $itrN$ | 10 |
| $ou$ | 80% |

# Summary of Findings

- PCP reduces power consumption
  - PCP outperformed EDCP in power reduction
  - The gap grows as the number of containers increases
  - Overall reduction: about 7.4%

- PCP reduces search time by about 5% on average
  - Average runtime: EDCP ≈ 0.27 s PCP ≈ 0.26 s



# Conclusion & Future Work

- Conclusion
  - PCP is an effective energy-aware placement mechanism for CaaS
- Future Directions
  - Dynamic runtime management including container migrations
  - Modeling of PM heterogeneity
  - Guarantees SLA objectives