Constanta Maritime University, Faculty of Navigation and Naval Transport

A particle swarm-based procedure for task allocation in Cyber-Physical Systems

Constanta, August, 2022

Simona DINU Gabriel RAICU

ATOM –N 2022 Conference

Introduction

► **Task allocation in Cyber-Physical Systems:** from cyber perspective, real-time system research focuses on parallel application scheduling, where tasks must be scheduled in real-time, to obtain high performance in this heterogeneous computing environment. This is an optimization problem within a major research field in computer science and engineering.

► **Demands for scheduling parallel applications:** comparing to the traditional scheduling algorithm, where only one processor can execute a task at a time, parallel programming models are better suited to cope with new complex and challenging real-life environments.

► **Task allocation problem:** a specific machine-scheduling problem that characterizes real industrial frameworks with more complex functional demands: precedence-constrained tasks, individual processing times of operations on different machines and communication cost between tasks that are not assigned to the same machine. The specificity of this problem generates an increased complexity and needs higher computational effort compared to a classical task-scheduling problem.

Goals & Approaches

► **Major concerns:** complex real-time applications on embedded systems need to process large amounts of data and become more computational with the evolution of technology; but real-time computing process of massive data requires sufficient resources for computation and communication. Therefore, many of today's CPS models rely on multiple power-efficient multiprocessor platforms as a key element in performing calculations in CPS components. This raises challenges in organizing computations upon multiprocessors.

► **Proposed model:** the problem is modeled as a Mixed Integer-Linear Programming (MILP) formulation and is solved using a Particle Swarm Optimization approach - a heuristic population-based global optimization method that performs well in difficult multi-objective optimization problems arising in computer science and engineering.

Problem formulation

Generally, a parallel application running in a heterogeneous computing environment can be described by a directed acyclic graph G=(V,E) where nodes $v_i \in V$ represent the set of application tasks, which will be executed on m heterogeneous processors, $m \in M$, connected in a fully connected topology.

E is a set of edges, where each edge $e = (v_i, v_j) \in E$ represents a precedence constraint between nodes. A node is available to be executed when all its predecessors have been executed: task v_j may not be started until task v_i has finished.

All tasks must be performed, but because of heterogeneous processors, tasks running times (or execution costs) are dependent of the processor at which they are executed.

The execution time of task v_i if it is assigned to processor k is denoted as t_{ki}. Each edge $e = (v_i, v_j) \in E$ has a weight c_{ij} that represent the cost of communication for transferring data between task v_i (scheduled on processor k) and task v_j (scheduled on processor q).

One assume insignificant intraprocessor communication costs, so that if both tasks reside on the same processor, the cost of communication will be zero. In addition, $c_{ii}=0$ and $c_{ij}=c_{ji}$.

PD(i) is the set of direct predecessors of task v_i.

Decision variables:

$$d_{ik} = \begin{cases} 1, \text{ if task } v_i \text{ is allocated to processor } k. \\ 0, \text{ otherwise} \end{cases}, \quad i=1,2,\dots,|V| \text{ and } k=1,2,\dots,|M|$$

Objectives:

- **Minimize Obj_1:**

Minimize the overall cost: the total communication cost between tasks and the total execution cost of performing all tasks.

$$\min \left[\sum_{i=1}^{|V|-1} \sum_{j=i+1}^{|V|} c_{ij} \left(1 - \sum_{k=1}^{|M|} d_{ik} \cdot d_{ij} \right) + \sum_{i=1}^{|V|} \sum_{k=1}^{|M|} d_{ik} \cdot t_{ki} \right]$$

- **Minimize Obj_2:**

Maximize the system stability, i.e. minimize the amount of processors idle times:

$$\min \left[C_T - \sum_{i=1}^{|V|} \sum_{k=1}^{|M|} t_{ik} \cdot d_{ik} \right]$$

C_T = cycle time, i. e. the maximum time available at each processor

Note: minimizing the amount of processors idle times actually leads to another objective pursued , namely maximize the processor utilization factor:

$$\max \quad \frac{1}{|M| \cdot C_T} \cdot \sum_{i=1}^{|V|} \sum_{k=1}^{|M|} d_{ik} \cdot t_{ki}$$

Constraints:

1) Every task *i* is assigned to one and only one processor:

$$\sum_{k=1}^{|M|} d_{ik} = 1$$

2) The precedence constraints:

$$\sum_{k=1}^{|M|} k \cdot d_{ik} \le \sum_{k=1}^{|M|} k \cdot d_{jk}, \forall i \in PD(j)$$

3) Constraints imposed on decision variables:

$d_{ik} \in \{0,1\}$, $i = 1, \dots, |V|$, $k = 1, \dots, |M|$

Problem solution

► **PSO model:** swarm of interacting particles - candidate solutions.

► **A particle:** "flies" in multiple directions guided by the following factors:

- **its own experience:** the local best position ; p_{best}
- **the experience of neighboring particles:** the global best position; g_{best}

► **Updates of particles:** are achieved according to :

$$v_i^{t+1} = \omega \cdot v_i^t + c_1 \cdot \varphi_1^t \cdot (pbest_i^t - x_i^t) + c_2 \cdot \varphi_2^t \cdot (gbest^t - x_i^t) \quad i=1,\dots, PS$$

$$x_i^{t+1} = x_i^t + v_i^{t+1}$$

PS = dimension of the swarm i.e. population size

v_i^t is the velocity of particle i at iteration t

x_i^t is the current position of particle i at iteration t

c_1 and c_2 are two positive values named acceleration coefficients

φ_1^t and φ_2^t are uniformly distributed random number in [0,1]

ω is the inertia factor

```
initialization:
$g_{max}$ ; PS ; $v_{max}$ ; $c_1$ , $c_2$ and $\omega$
for i=1 to PS
randomly initialize $x_i$ within the search range [$x_i^{min}$ ,$x_i^{max}$]; initialize $v_i = 0$;
repeat
while (convergence criterion is not satisfied)
    for i=1 to PS
        calculate fitness $f(x_i)$;
        update individual best position of each particle:
            if ($f(x_i^{t+1}) \ge f(pbest_i^t)$) then $pbest_i^{t+1} = pbest_i^t$
                                        else $pbest_i^{t+1} = x_i^{t+1}$
            endif
        update global best position: gbest = argmin{$f(pbest_i^t)$}
        update velocity of the particle; update position of the particle
    repeat
repeat
```

Inertia factor adjustment—fuzzy controller

two input variables:

- current value of inertia factor: ω^t
- normalized dev. of fitness values: Δ^t_{norm}

one output variable:

- inertia factor: ω

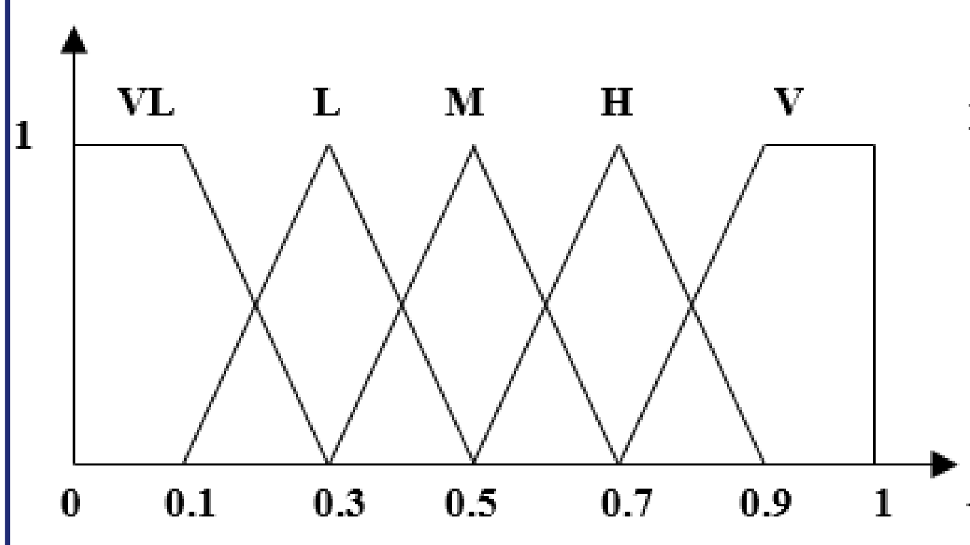

Figure 1. The membership functions for the inputs

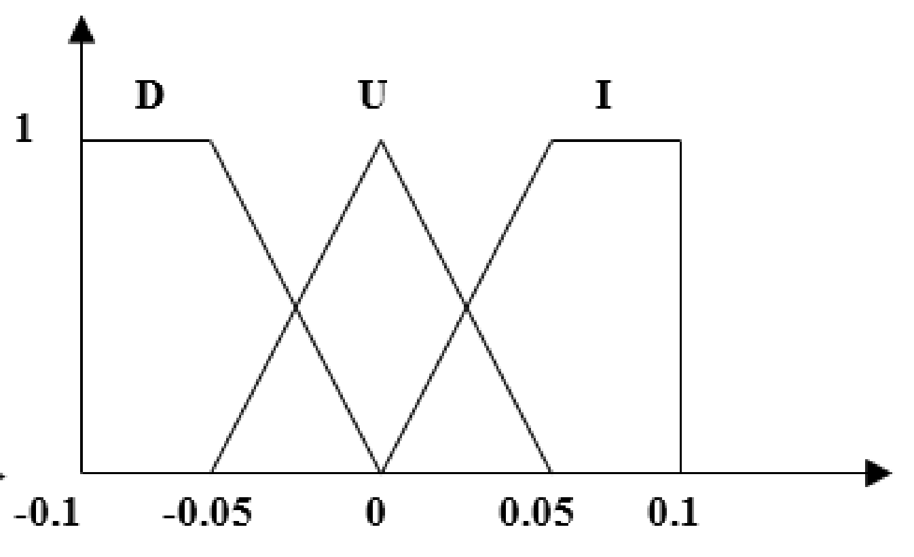

Figure 2. The membership function for the output

Table 1. The inference table

DEV^t_{norm} \ ω	VL	L	M	H	VH
VL	UM	UM	UM	D	D
L	UM	UM	D	D	D
M	I	UM	UM	UM	D
H	I	I	UM	UM	D
VH	I	I	UM	D	D

Encoding scheme

- for each task v_i, $i=1,2,\dots,|V|$ in the sequence mapped in that particle, the following information is recorded:
  - velocity v^i and its personal best position y^i;
  - position (priority) x^i;
  - task number, its processing time and the processor to which it is assigned.

Simulation results

Test problem: a test instance consisting in 7 processors and 45 tasks, with 60 task-precedence constraints, and a cycle time C_T = 94 time units.

The corresponding precedence graph was transformed to a binary precedence matrix $M=(m_{i,j})_{45x45}$ to describe the task-precedence constraints:

$$m_{ij} = \begin{cases} 1, \text{ if } j \text{ is a predecessor of } i.. \\ 0, \text{ otherwise} \end{cases}, \quad i=1,2,\dots,45$$

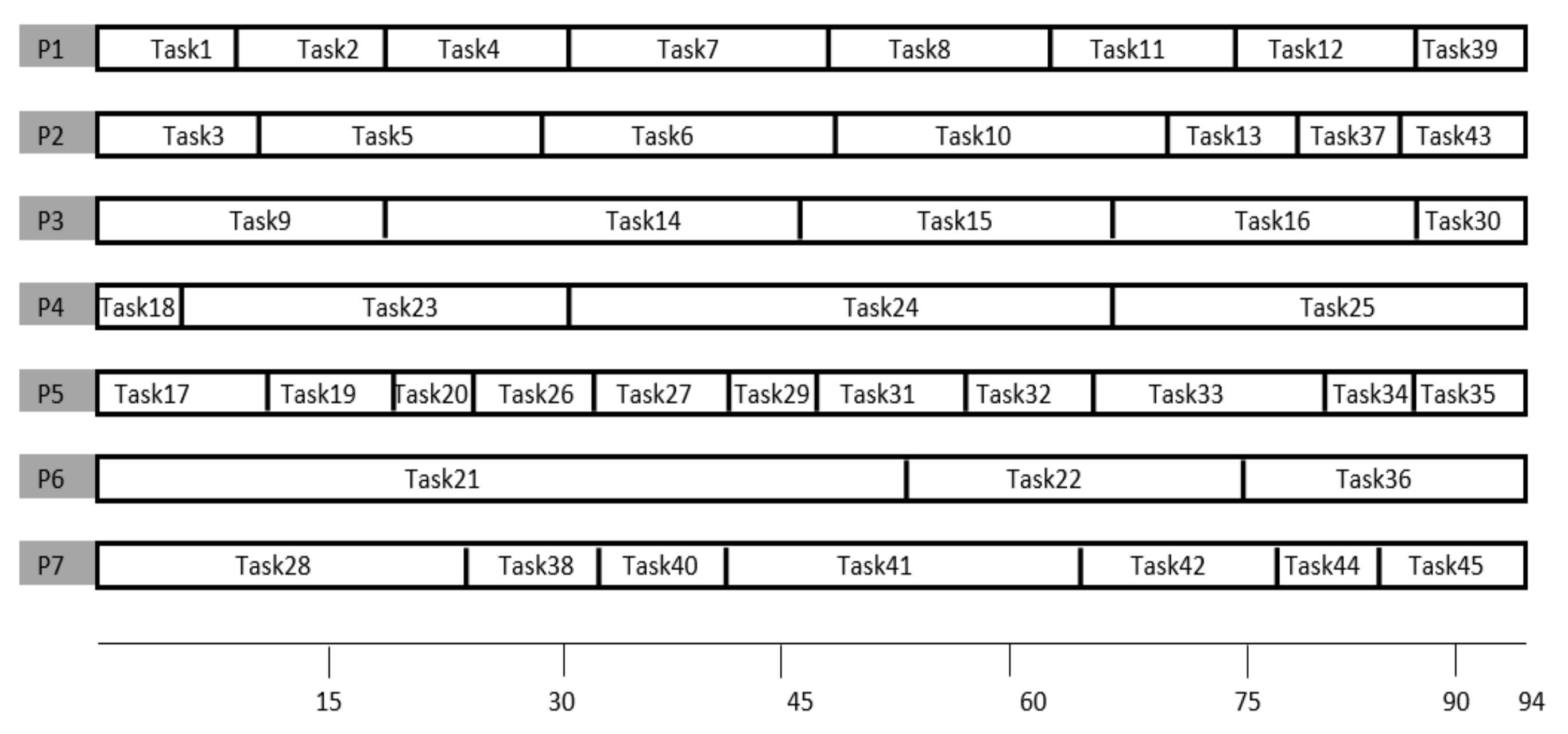

Figure 3: Example of an optimization result of task allocation

Conclusions and future work

► The computational experiments on all data sets showed good performance of the proposed algorithm.

► The algorithm is easy to design and implement and is tractable for practical problems with a large number of tasks and outsourcing providers.

► The research is still in progress and we will further investigate the relationship between the structure of the acyclic directed graphs and the overall performance of the algorithm.