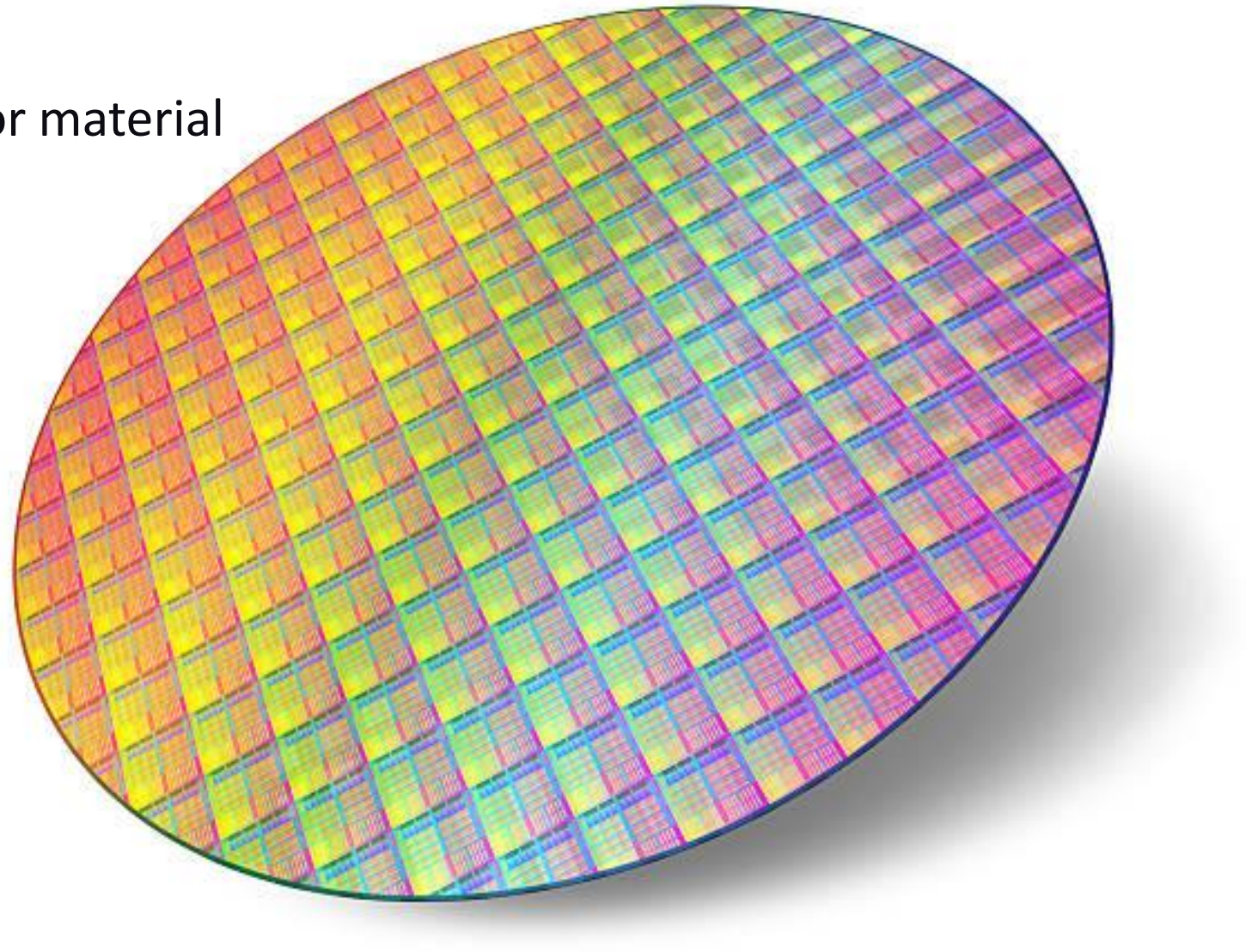
Wafer Processing Optimization

Software Engineering Workshop

December 2024

Wafer

Thin slices of semiconductor material



Optimization of Wafer Processing in Semiconductor Manufacturing

- In Semiconductor manufacturing, wafers go through a series of steps to build chips.
- Each step requires specific machines to process the wafers under controlled conditions.
- These conditions, or "process parameters", such as temperature and pressure, need to stay within certain limits for each step to ensure quality.
- However, as wafers are processed, the process parameters tend to degrade/fluctuate over time. Machines need to be put in ‘cooldown/idle mode’ to bring the parameters back to normal.

- Our task is to create an optimized schedule plan that minimizes processing time for multiple wafers while keeping process parameters within acceptable limits by managing fluctuations and machine cooldowns.

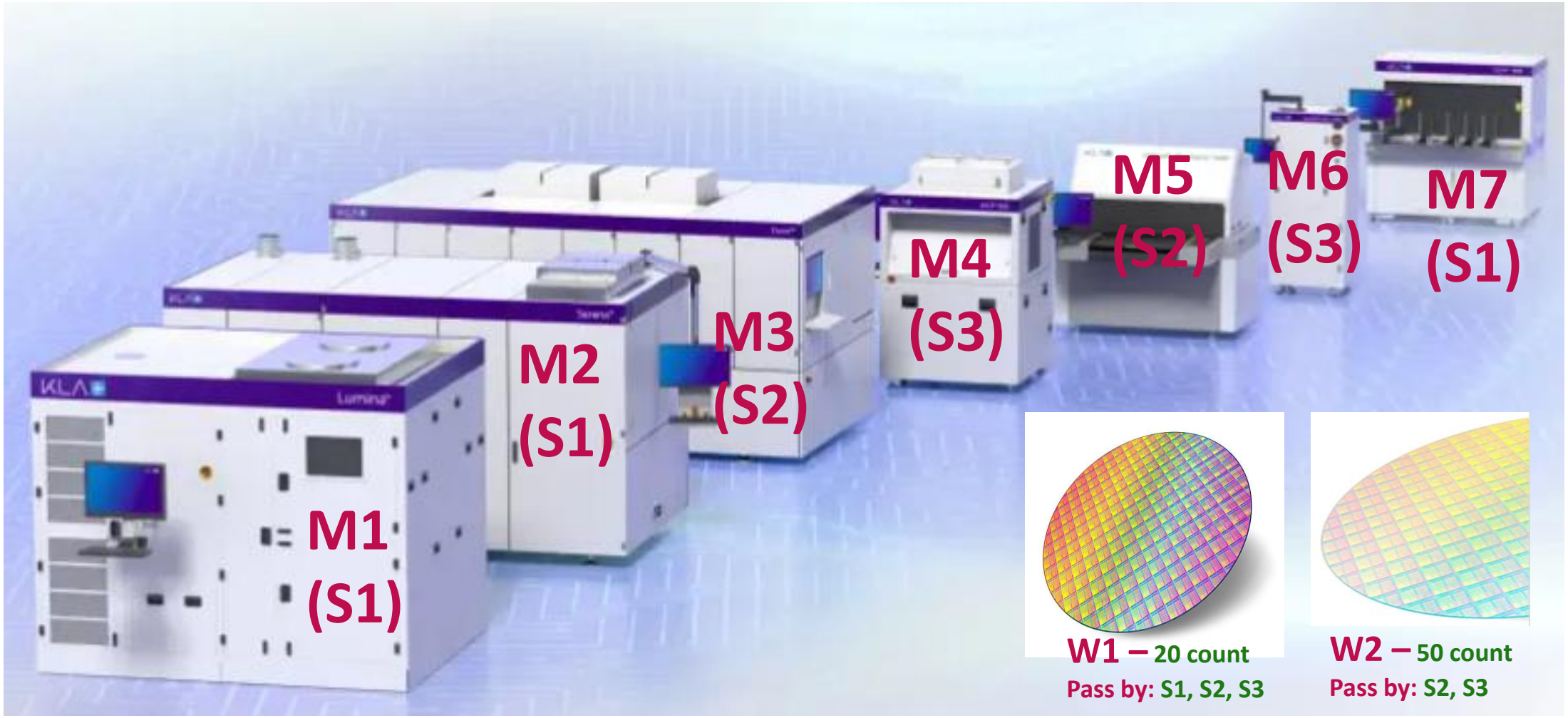
Objective

- **Schedule the wafers (of type W1, W2, etc.)** through **required steps (S1, S2, etc.),** minimizing total time taken to process all wafers.
- Ensure that **process parameters (p1, p2, etc.)** for every machine stay within the acceptable range during the steps.
  - Parameters start with initial values and fluctuate dynamically as machines process wafers.
- No defective wafer, that is, no wafer is processed with out-of-range parameter conditions (i.e., when the machine is in cooldown mode).

Problem Details

- **Each wafer type (W1, W2, etc.)** has a different **processing time** for each step.
- **Each step** has different **ranges** for each process parameter. For example, p1 might need to stay between [150-200] units, while p2 needs to be within [5-10] units. If any of the parameters go out of range (< min or > max), the wafer processed is considered as defective.
- **Machines (M1, M2, etc.)** are assigned to each step. One machine is for one step only. There could be multiple machines assigned for one step.
- **Each machine** starts with given values of process parameters (**p1, p2 etc.**). For each machine, after processing certain number of wafers **(n)**, the process parameters increase or decrease by a specified units.
- Each machine has different **cooldown time**. After machines are kept idle for 'cooldown time' period, their parameters reset to initial values.

Workflow


M1
(S1)
M2
(S1)
M3
(S2)
M4
(S3)
M5
(S2)
M6
(S3)
M7
(S1)
W1 – 20 count
Pass by: S1, S2, S3
W2 – 50 count
Pass by: S2, S3

Input Format

- Steps and Parameter Requirements:
  - Each step (e.g., S1, S2) has specific acceptable ranges for each parameter (p1, p2, etc.).
  - Each step may also depend on the completion of previous step(s). For example, if S3 depends on S1 and S2, then a wafer must complete S1 and S2 before starting S3.
  - If there is no dependency of steps, they can be executed in any order.
- Machines:
  - Machines (e.g., M1, M2) are assigned to various steps and have different cooldown times. For instance, M1 might need 5 minutes of cooldown time.
  - Each machine has its initial parameter levels, which fluctuate as wafers are processed.
- Parameter Fluctuation:
  - For each machine, after processing ‘n’ wafers, the parameters fluctuate (increase or decrease) by a specified number of units.
  - After the cooldown period, the parameters reset to their initial values.
- Wafers:
  - Each wafer type (W1, W2, etc.) has specific processing times for each step.
  - The input will also specify the number of wafers for each type that need to be processed.

Example: Input


```
{
    "steps": [
                {"id": "S1", “parameters”: {"p1": [150, 250], "p2": [5, 10]}, "dependency": null},
                {"id": "S2", “parameters”: {"p1": [200, 250], "p2": [7, 12]}, "dependency": [“S1”]} ],
    "machines": [
                { "machine_id": “M1", "step_id": "S1", "cooldown_time": 5, "initial_parameters":
                        {"p1": 160, "p2": 7}, "fluctuation": {"p1": 5, "p2": 1}, "n": 2},
                { "machine_id": “M2", "step_id": "S1", "cooldown_time": 3, "initial_parameters":
                        {"p1": 215, "p2": 6}, "fluctuation": {"p1": 4, "p2": 2}, "n": 3},
                { "machine_id": “M3", "step_id": "S2", "cooldown_time": 3, "initial_parameters":
                        {"p1": 210, "p2": 9}, "fluctuation": {"p1": 6, "p2": 1}, "n": 1},
    "wafers": [
                {"type": "W1", "processing_times": {"S1": 10, "S2": 15}, "quantity": 3},
                {"type": "W2", "processing_times": {"S1": 12, "S2": 10}, "quantity": 2} ]
}
```


There are 2 steps, S1 and S2.

Step S1:

- Parameters for S1 must stay within p1: [150, 200] and p2: [5, 10].
- Two machines (M1, M2) are assigned for S1.
- Machine M1 starts with p1 = 160 and p2 = 7. After processing 2 wafers (n=2), p1 increases by 5 and p2 increases by 1. Cooldown of 5 minutes resets these values to their initial state.
- Machine M2 starts with p1 = 215 and p2 = 6. After processing 3 wafers (n=3), p1 increases by 4 and p2 increases by 2.

Step S2:

- It has step dependency on S1.
- Parameters for S2 must stay within p1: [200, 250] and p2: [7, 12].
- Machine M3 starts with p1 = 210 and p2 = 9. After 1 wafer (n=1), p1 increases by 6 and p2 increases by 1.

Wafers of **type W1** need to go through steps S1 and S2. Count = 3

Wafers of **type W2** need to go through steps S1 and S2. Count = 2.

Output Format

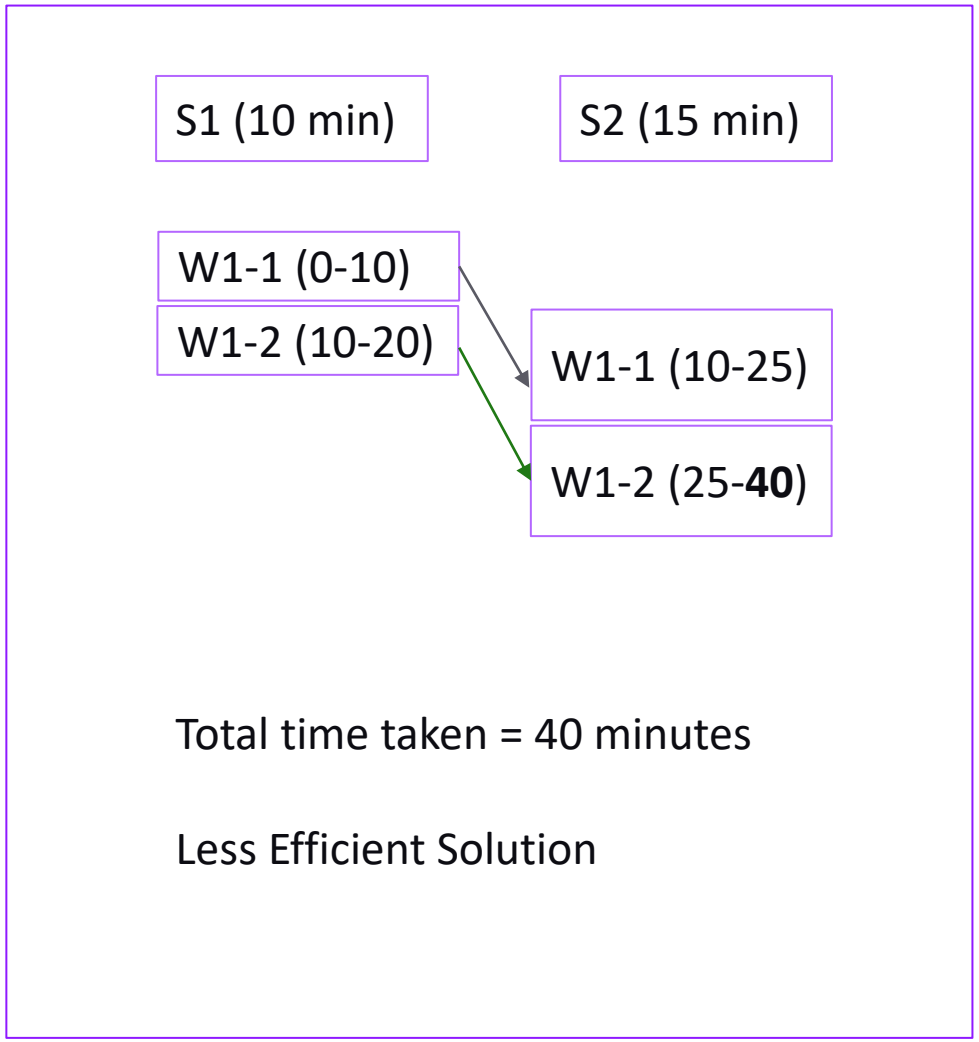
- The output should provide the **schedule plan of wafers** including **wafer ID**, **step number**, **machine used, start time** and **end time** for each wafer processing task.
- **Wafer ID: "**Wafer Type – Wafer number of that type".
  - W1-1, W1-2 for wafers of type W1
  - W2-1, W2-2 for wafers of type W2 etc.
- **Start and End Times:** Provide the time range when each wafer is processed at a specific step using a specific machine.
  - [0, 10]. – Here the step started at 0^{th} minute and finished by 10^{th} minutes. The next step can start from 10^{th} minute [10, …]
- Example Output

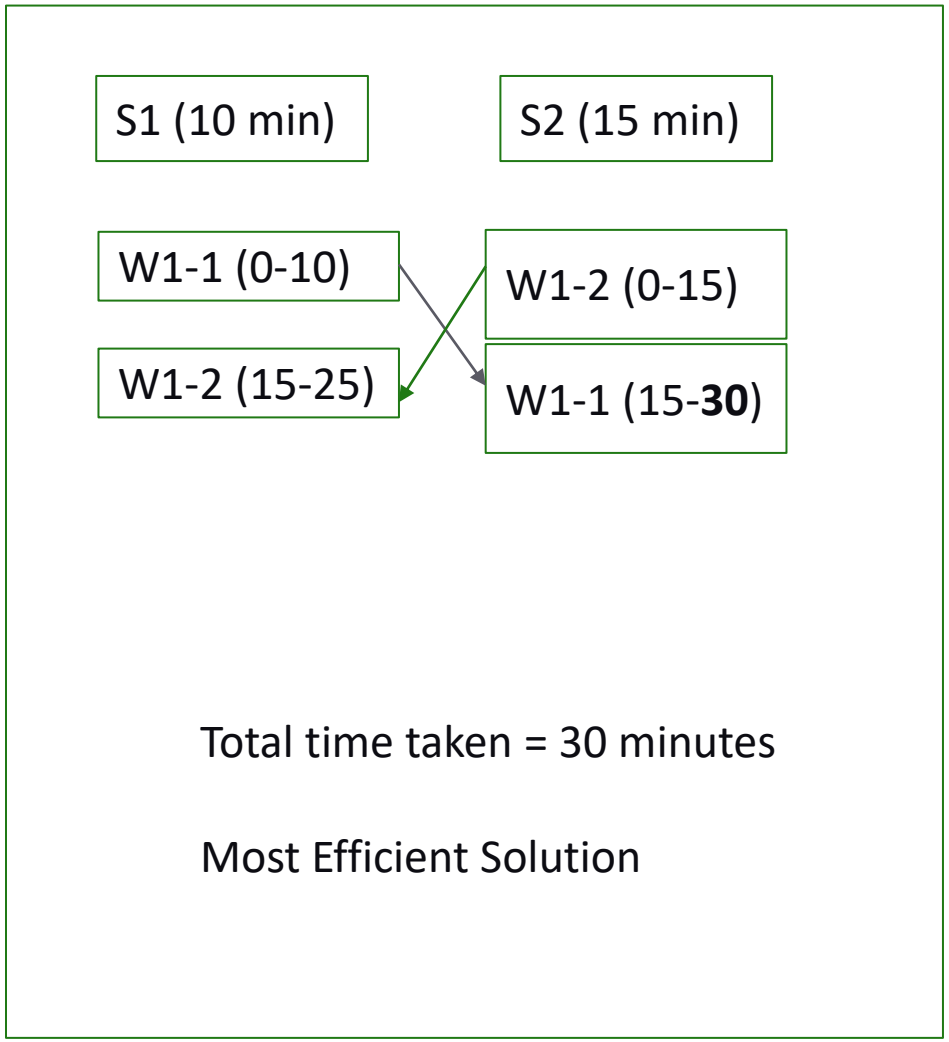
```
{ "schedule" : [
        {"wafer_id": "W1-1", "step": "S1", "machine": "M1", "start_time": 0, "end_time": 10},
        {"wafer_id": "W1-1", "step": "S2", "machine": "M3", "start_time": 10, "end_time": 25},
        {"wafer_id": "W1-2", "step": "S1", "machine": "M2", "start_time": 0, "end_time": 10},
        {"wafer_id": "W2-1", "step": "S1", "machine": "M1", "start_time": 10, "end_time": 22} ]
}
```

Example: Input (Milestone 0)


```
{
  "steps": [
    {
      "id": "S1",  "parameters": { "P1": [ 100, 200 ] }, "dependency": null
    },
    {
      "id": "S2",  "parameters": { "P1": [ 100, 200 ] }, "dependency": null
    }
  ],
  "machines": [
    {
      "machine_id": "M1",   "step_id": "S1",    "cooldown_time": 5, "initial_parameters": { "P1": 100 },
      "fluctuation": { "P1": 5 },   "n": 20
    },
    {
      "machine_id": "M2",   "step_id": "S2",    "cooldown_time": 5, "initial_parameters": { "P1": 100 },
      "fluctuation": { "P1": 5 },   "n": 20
    }
  ],
  "wafers": [
    {
      "type": "W1", "processing_times": { "S1": 10, "S2":  15 },    "quantity": 2
    }
  ]
}
```

Working of the problem:





Example: Output (Milestone 0)

Solution 1

Less Efficient

Total Time = 40 minutes


```
{
  "schedule": [
    {
      "wafer_id": "W1-1",   "step": "S1",   "machine": "M1",   "start_time": 0,   "end_time": 10
    },
    {
      "wafer_id": "W1-2",   "step": "S1",   "machine": "M1",   "start_time": 10,   "end_time": 20
    },
    {
      "wafer_id": "W1-1",   "step": "S2",   "machine": "M2",   "start_time": 10,   "end_time": 25
    },
    {
      "wafer_id": "W1-2",   "step": "S2",   "machine": "M2",   "start_time": 25,   "end_time": 40
    }
  ]
}
```


Solution 2

Most Efficient

Total Time = 30 minutes

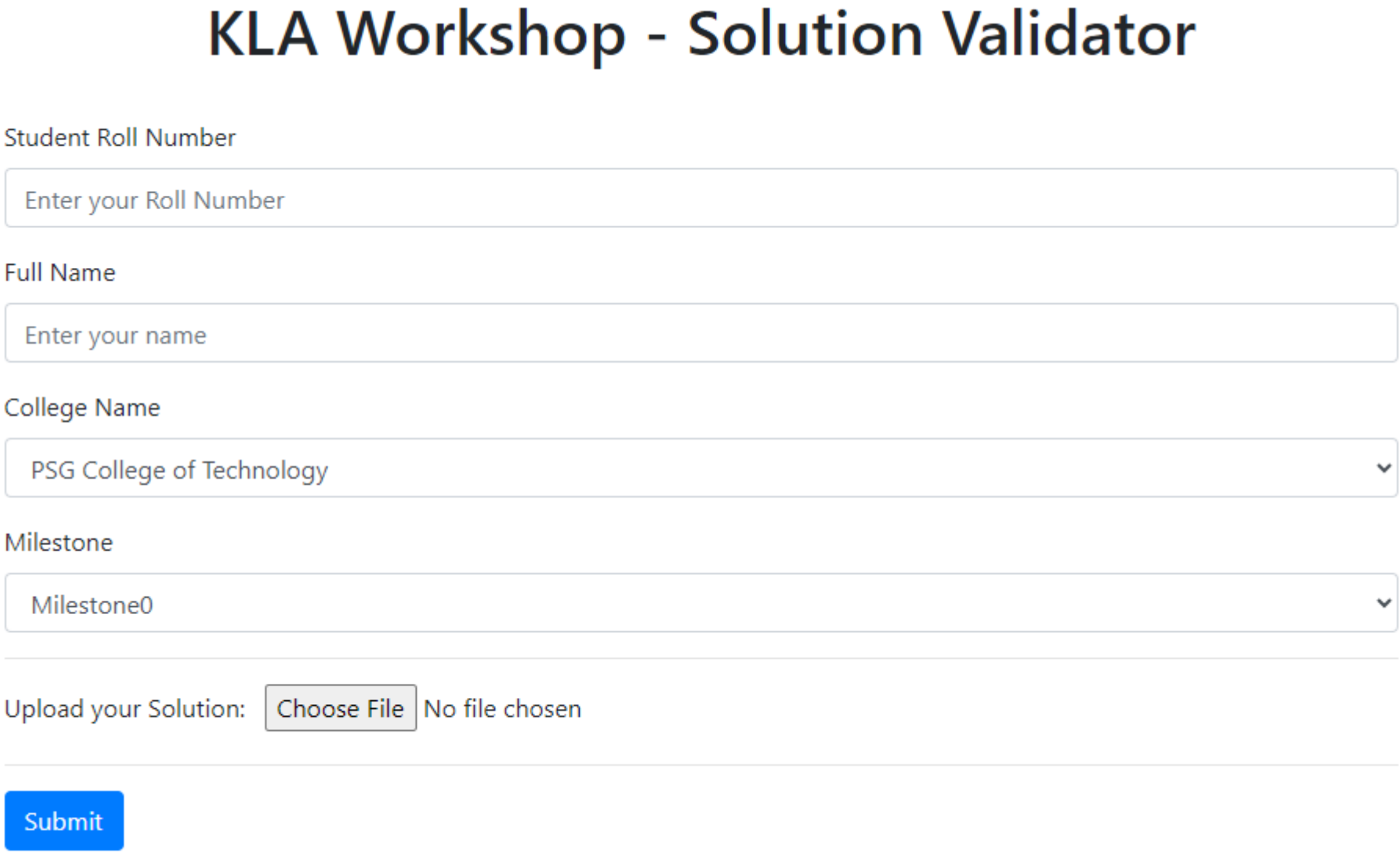

```
{
  "schedule": [
    {
      "wafer_id": "W1-1",   "step": "S1",   "machine": "M1",   "start_time": 0,   "end_time": 10
    },
    {
      "wafer_id": "W1-2",   "step": "S2",   "machine": "M2",   "start_time": 0,   "end_time": 15
    },
    {
      "wafer_id": "W1-2",   "step": "S1",   "machine": "M1",   "start_time": 15,   "end_time": 25
    },
    {
      "wafer_id": "W1-1",   "step": "S2",   "machine": "M2",   "start_time": 15,   "end_time": 30
    }
  ]
}
```

Validation

- Output Accuracy:
  - All wafers are processed in the correct sequence (step dependencies are followed)
  - Correct combination of Steps and Machines are used
  - Machines are not double-booked
  - Same wafer is not present on two machines simultaneously
  - Each step is taking correct amount of time for each wafer type.
- Process Parameter Compliance:
  - Parameters are always in-range
  - Cool-down is followed when parameters go out-of-range; no wafer should be processed during this cool-down time
- Efficiency:
  - Lesser the time (to process all given wafers), more efficient the solution
  - Aim: Minimize time required to process all given wafers.

Solution Validator

https://klavalidator2024.azurewebsites.net



Validation screens

Valid

Student Roll Number: Test123

Student Name: Chinmaya

Student College: PSG College of Technology

Milestone Level: Milestone0

Result Summary

Overall Status
Valid

Total time taken: 30 minutes

Error(s):

Validate more testcases

Invalid

Student Roll Number: Test123

Student Name: Chinmaya

Student College: PSG College of Technology

Milestone Level: Milestone0

Result Summary

Overall Status
Invalid

Total time taken: NA

Error(s):
Invalid wafer ID: W1-3 in schedule.

Invalid machine M1 for step S2 in wafer W1-2.

Processing time error: Step S2 for wafer W1-2 expected 15, got 10.

Invalid machine M2 for step S1 in wafer W1-1.

Processing time error: Step S1 for wafer W1-1 expected 10, got 15.

. . .

Validate more testcases

Milestones

	Input Files
Milestone0	Milestone0.json
Milestone1	Milestone1.json
Milestone2a	Milestone2a.json
Milestone2b	Milestone2b.json
Milestone3a	Milestone3a.json
Milestone3b	Milestone3b.json
Milestone3c	Milestone3c.json
Milestone4a	Milestone4a.json
Milestone4b	Milestone4b.json
Milestone4c	Milestone4c.json
Milestone5a	Milestone5a.json
Milestone5b	Milestone5b.json

Git Instruction: After each milestone completion and verification with mentors, upload to your personal git repo and share with **“UnHack-2024”** git ID

All the best!