List Based Scheduling

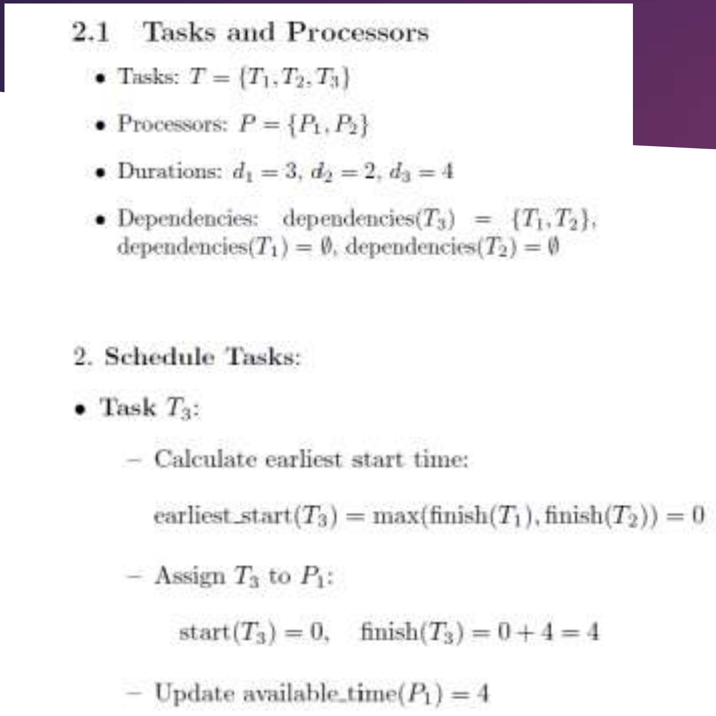
MASRUR JAMIL PROCHCHHOD

MOTIVATION

List scheduling

- Dynamic Prioritization
- Resource Allocation
- Dependency Management
- Increase throughput

Working Procedure



2.1 Tasks and Processors

- Tasks: $T = \{T_1, T_2, T_3\}$
- Processors: $P = \{P_1, P_2\}$
- Durations: $d_1 = 3$, $d_2 = 2$, $d_3 = 4$
- Dependencies: $\text{dependencies}(T_3) = \{T_1, T_2\}$, $\text{dependencies}(T_1) = \emptyset$, $\text{dependencies}(T_2) = \emptyset$

2. **Schedule Tasks:**

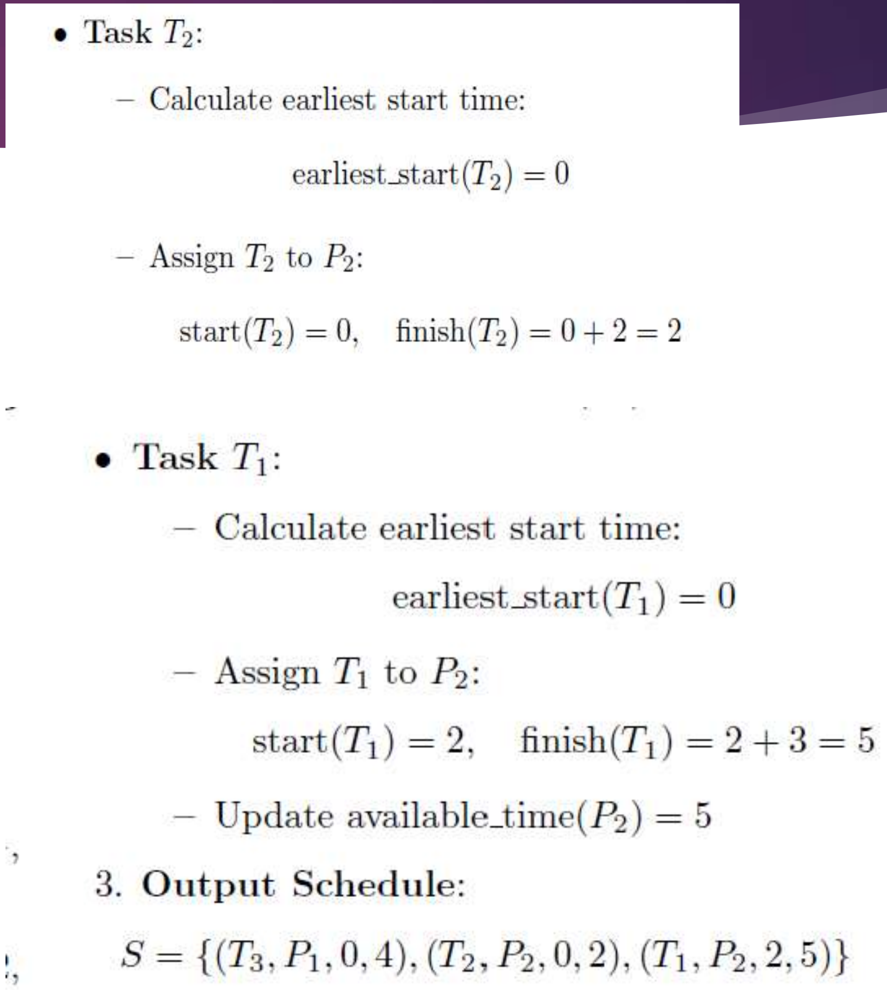
- Task T_3:
  - Calculate earliest start time:

$$\text{earliest\_start}(T_3) = \max(\text{finish}(T_1), \text{finish}(T_2)) = 0$$

  - Assign T_3 to P_1:

$$\text{start}(T_3) = 0, \quad \text{finish}(T_3) = 0 + 4 = 4$$

  - Update $\text{available\_time}(P_1) = 4$



- Task T_2:
  - Calculate earliest start time:

$$\text{earliest\_start}(T_2) = 0$$

  - Assign T_2 to P_2:

$$\text{start}(T_2) = 0, \quad \text{finish}(T_2) = 0 + 2 = 2$$

- Task T_1:
  - Calculate earliest start time:

$$\text{earliest\_start}(T_1) = 0$$

  - Assign T_1 to P_2:

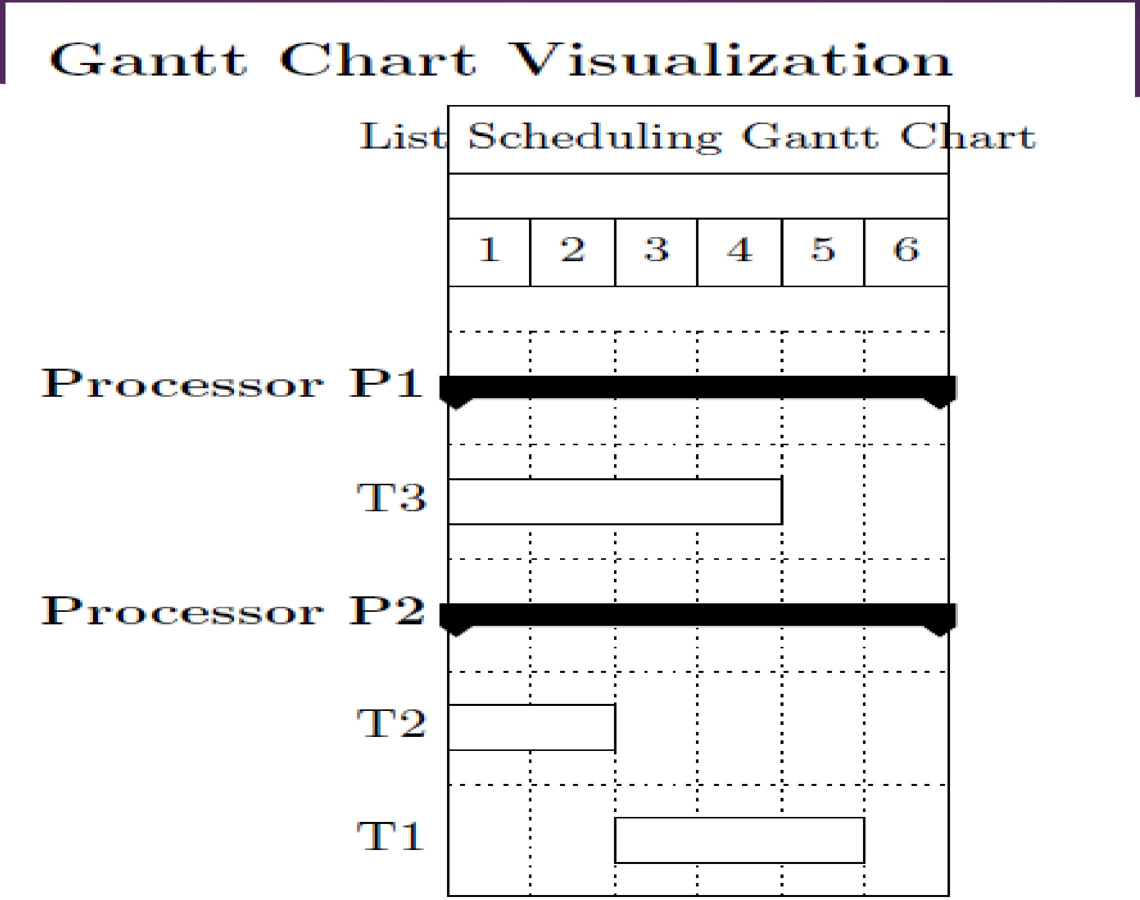
$$\text{start}(T_1) = 2, \quad \text{finish}(T_1) = 2 + 3 = 5$$

  - Update $\text{available\_time}(P_2) = 5$

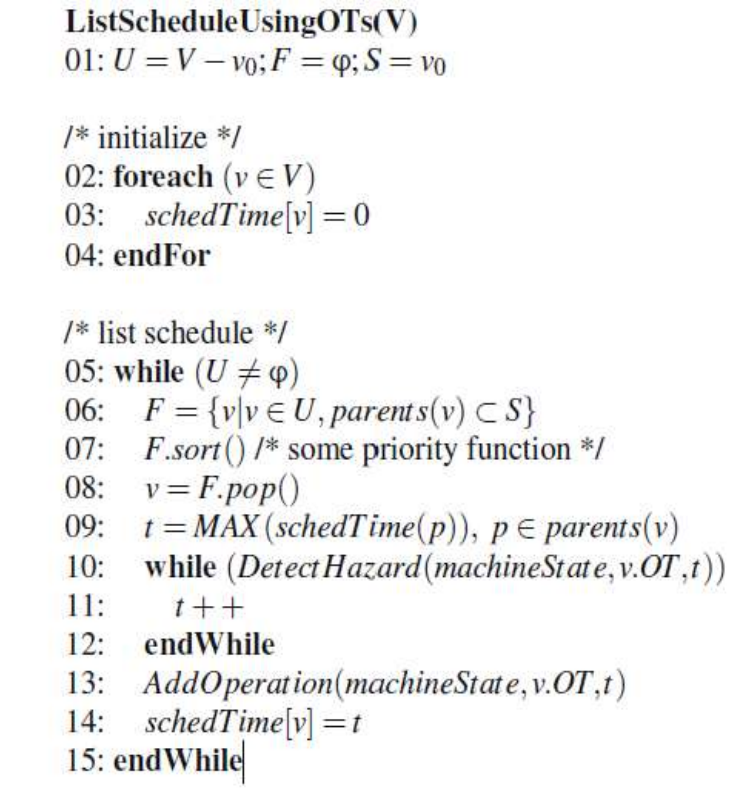
3. **Output Schedule:**

$$S = \{(T_3, P_1, 0, 4), (T_2, P_2, 0, 2), (T_1, P_2, 2, 5)\}$$

Gantt chart


Gantt Chart Visualization
List Scheduling Gantt Chart
1
2
3
4
5
6
Processor P1
T3
Processor P2
T2
T1

List Scheduling using OT

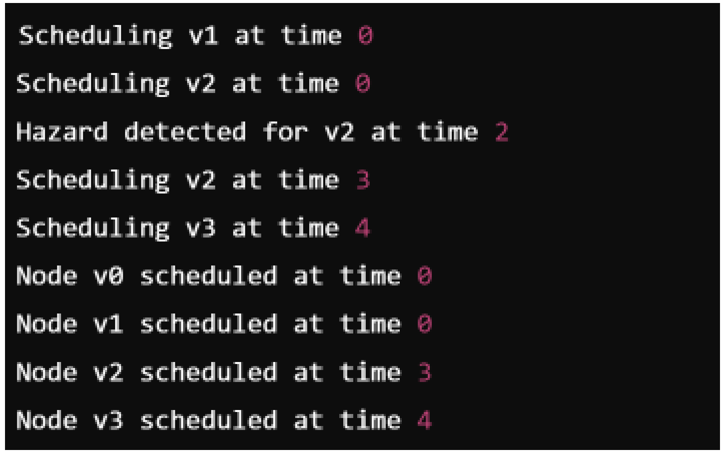


```
ListScheduleUsingOTs(V)
$U = V - v_0; F = \varphi; S = v_0$

/* initialize */
foreach $(v \in V)$
    $schedTime[v] = 0$
endFor

/* list schedule */
while $(U \neq \varphi)$
    $F = \{v | v \in U, parents(v) \subset S\}$
    $F.sort()$ /* some priority function */
    $v = F.pop()$
    $t = MAX(schedTime(p)),\ p \in parents(v)$
    while $(DetectHazard(machineState, v.OT, t))$
        $t{+}{+}$
    endWhile
    $AddOperation(machineState, v.OT, t)$
    $schedTime[v] = t$
endWhile
```

Fig:01 List Scheduling using OT [1]



```
Scheduling v1 at time 0
Scheduling v2 at time 0
Hazard detected for v2 at time 2
Scheduling v2 at time 3
Scheduling v3 at time 4
Node v0 scheduled at time 0
Node v1 scheduled at time 0
Node v2 scheduled at time 3
Node v3 scheduled at time 4
```

Fig:02 Result after Implementation

Parallel computing innovations

- ❑ Advance Load Balancing
- ❑ Optimized Task Scheduling
- ❑ Scalability Enhancements

Distributed Computing innovations

- ❑ Resource Management
- ❑ Enhance Error Tolerance
- ❑ Energy Efficient Scheduling

Emperical Assessments

- Experimental Setup
- Data Collection
- Performance Evaluation
- Comparative Analysis

Practical Applications

- Real Time Systems
- Cloud Computing
- Parallel Systems

Pros and cons

Advantages

- Simplicity
- Efficiency
- Flexibility
- Real Time Applications
- Dependency Management

Disadvantages

- Overhead
- Resource Conficts
- Scalability Limitations
- Adaptability

Conclusion

- Effective Task Management
- Versatility
- Future Prospects
- Overall Impact

Thank You

References

1. J¨urgen Teich. Handbook of Hardware/Software Codesign. Springer, 2016. Code taken from page 822
2. Dror G Feitelson. Experimental analysis of the root causes of overheads in parallel job schedulers. ACM Transactions on Parallel Computing,5(1):1–26, 2018.
3. Michael J Flynn. Some computer organizations and their effectiveness. IEEE Transactions on Computers, C-21(9):948–960, 1972