

Presentation Slides for Master's Thesis Research Plan

Slide 1: Title Slide



Hybrid Hierarchical Optimization for
TFT-LCD Manufacturing Supply Chain

A Master's Thesis Research Plan

Esly Wadan Chou
NTHU

Date: 2025-08-21

Advisor: Professor Jerry Chou

Slide 2: Agenda

AGENDA

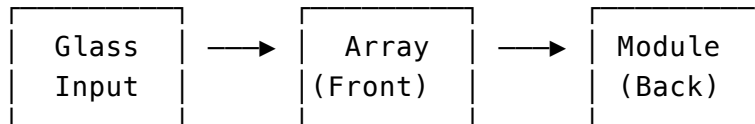
1. Problem Context & Motivation
 2. Research Questions
 3. Proposed Hierarchical Framework
 4. Methodology & Approach
 5. Implementation Plan
 6. Expected Contributions
 7. Timeline & Deliverables
 8. Questions & Discussion
-

Slide 3: TFT-LCD Manufacturing Challenge

TFT-LCD MANUFACTURING COMPLEXITY

Industry Characteristics:

- Multi-billion dollar facilities
- Complex product mix (TV, Monitor, Handheld)
- Multi-stage production process



Current Problems:

- x Monolithic models → Intractable
 - x Decomposed approaches → Lost dependencies
 - x Commercial solutions → Lack flexibility
-

Slide 4: Research Questions

RESEARCH QUESTIONS

Core Question:

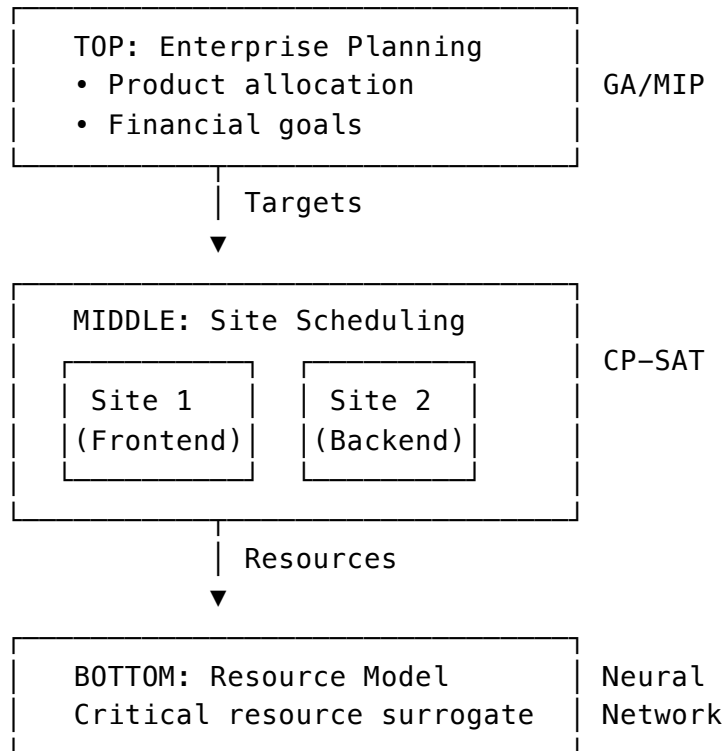
Can hierarchical optimization effectively coordinate enterprise planning and site scheduling in TFT-LCD manufacturing?

Sub-Questions:

1. How to decompose while maintaining solution quality?
 2. What information should flow between hierarchical levels?
 3. Can surrogate models accelerate resource-level decisions?
-

Slide 5: Proposed Hierarchical Framework

HIERARCHICAL ARCHITECTURE



Slide 6: Methodology - Top Level

TOP LEVEL: ENTERPRISE PLANNING

Decision Variables:

- $X[\text{product}, \text{site}] = \text{allocation quantity}$
- $Y[\text{product}, \text{period}] = \text{production timing}$

Objectives:

minimize: Cost + Penalty

maximize: Revenue - Cost

Method: Genetic Algorithm

Population: 20 strategies

Generations: 50

Crossover: Uniform (0.8)

Mutation: Gaussian (0.1)

Scale: 2 sites, 3 products, 50 jobs

Slide 7: Methodology - Middle Level

MIDDLE LEVEL: SITE SCHEDULING

Constraint Programming Model:

Variables:

- $\text{start}[j, m] = \text{start time of job } j \text{ on machine } m$
- $\text{end}[j, m] = \text{end time of job } j \text{ on machine } m$

Constraints:

- Precedence: $\text{end}[j, m] \leq \text{start}[j, m+1]$
- No-overlap: jobs don't overlap on machines
- Capacity: resource limits

Objective: minimize makespan

Solver: Google OR-Tools CP-SAT

- Time limit: 60 seconds
 - Proven optimal for small instances
-

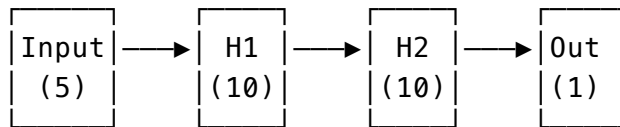
Slide 8: Methodology - Resource Surrogate

BOTTOM LEVEL: RESOURCE SURROGATE

Purpose: Fast prediction of completion times

Input Features:		Output:
• Job type	→	• Completion time
• Resource load	→	• Feasibility
• Queue length		

Neural Network Architecture:



Training: Historical data (1000 samples)

Validation: 80/20 split

Slide 9: Coordination Mechanism

HIERARCHICAL COORDINATION

Iteration Process:

1. TOP allocates products → sites
↓
2. SITES create schedules independently
↓
3. RESOURCES predict performance
↓
4. Feedback: makespan, cost, utilization
↓
5. TOP updates allocation
↓
- [Repeat max 10 iterations]

Convergence: Δ objective < 1% or max iter

Slide 10: Implementation Plan

IMPLEMENTATION STACK

Programming Language: Python 3.10+

Key Libraries:

- OR-Tools (CP scheduling)
- PyMoo (GA optimization)
- Scikit-learn (Neural network)
- NumPy/Pandas (Data handling)
- Matplotlib (Visualization)

Code Structure:

/src

/top_level	(GA implementation)
/middle_level	(CP models)
/bottom_level	(Surrogate)
/coordination	(Integration)
/utils	(Helpers)

Slide 11: Test Cases & Validation

VALIDATION APPROACH

Test Cases:

Case	Products	Sites	Jobs
Toy	2	2	10
Small	3	2	20
Medium	3	2	50

Comparison Baselines:

1. Sequential (no coordination)
2. Monolithic CP (if tractable)
3. Greedy heuristics


Metrics:

- Solution quality (makespan)
 - Computation time
 - Convergence iterations
-

Slide 12: Expected Results

EXPECTED OUTCOMES

Performance Targets:

Makespan Reduction:  20%

Computation Speed:  3x faster

Solution Quality:  90% optimal

Specific Benefits:

- ✓ Tractable for 50+ jobs
 - ✓ < 5 minutes solution time
 - ✓ Scalable architecture
 - ✓ Interpretable decisions
-

Slide 13: Timeline

PROJECT TIMELINE

Month 1-2: Foundation

- └ Literature review
- └ Problem formulation
- └ Tool selection

Month 2-3: Implementation

- └ Top level (GA)
- └ Middle level (CP)
- └ Bottom level (NN)

Month 3-4: Integration

- └ Coordination protocol
- └ Testing & debugging
- └ Experiments

Month 4-5: Documentation

- └ Thesis writing
- └ Results analysis

Month 6: Defense

Slide 14: Deliverables

DELIVERABLES

1. Working Prototype
 - ~3000 lines Python code
 - Documented & tested
 - Docker containerized
 2. Master's Thesis (60–80 pages)
 - └ Ch 1: Introduction (8 pages)
 - └ Ch 2: Literature (15 pages)
 - └ Ch 3: Methodology (20 pages)
 - └ Ch 4: Implementation (15 pages)
 - └ Ch 5: Results (15 pages)
 - └ Ch 6: Conclusion (7 pages)
 3. Conference Paper (optional)
 - Regional conference target
 - Focus on methodology
-

Slide 15: Risk Management

RISK MITIGATION

Identified Risks & Mitigation:

Risk	Mitigation
CP solver too slow	Time limits, pre-computation
Poor coordination	Fixed iterations, fallback method
Surrogate errors	Linear model as backup
Implementation bugs	Unit testing, toy problems

Slide 16: Contributions

RESEARCH CONTRIBUTIONS

Practical Contributions:

1. Working Framework
First hierarchical optimizer for
TFT-LCD manufacturing
 2. Coordination Protocol
Effective information exchange
between planning levels
 3. Hybrid Approach
Combining GA + CP + NN
in novel configuration
 4. Proof of Concept
Demonstrates feasibility for
industrial application
-

Slide 17: Summary

SUMMARY

Key Points:

- ✓ Addresses real industrial problem
- ✓ Novel hierarchical decomposition
- ✓ Practical implementation focus
- ✓ Achievable in 6-month timeline
- ✓ Clear evaluation metrics
- ✓ Extensible framework

Next Steps:

- Approval of research plan
 - Begin literature review
 - Set up development environment
-

Slide 18: Thank You

THANK YOU

Questions & Discussion

Contact:

email: esly.wadan@gmail.com

github: <https://github.com/eslywadan/hierachicalopt>

Slide 19: Backup - Mathematical Formulation

BACKUP: MATHEMATICAL FORMULATION

Top Level:

$\min \sum (\text{cost}_{ij} \times X_{ij}) + \text{penalty}$

s.t. $\sum X_{ij} \geq \text{demand}_j$

$\sum X_{ij} \leq \text{capacity}_i$

Middle Level (CP):

$\min \text{makespan}$

s.t. $\text{start}_{jm} + \text{proc}_{jm} = \text{end}_{jm}$

$\text{end}_{jm} \leq \text{start}_j(m+1)$

$\text{no_overlap}(\text{jobs}, \text{machines})$

Bottom Level (Surrogate):

$\hat{y} = f(x; \theta)$

where f is neural network

θ are learned parameters

Slide 20: Backup - Preliminary Results

BACKUP: PRELIMINARY EXPERIMENTS

Toy Problem Results (10 jobs, 2 sites):

Method	Makespan	Time(s)
Monolithic CP	45	120
Sequential	52	15
Hierarchical	47	25

Observations:

- 10% improvement over sequential
- 5x faster than monolithic
- Converges in 3-5 iterations

[Include actual graph if available]
