# Planning Placement Strategy using GraphPlan and POP

Project Report

November 30, 2025

## 1 Introduction

Placement preparation is often chaotic, with students struggling to effectively sequence and prioritize tasks. This project implements AI-driven planning algorithms that function as an intelligent scheduler, generating concrete, adaptive weekly roadmaps with transparent reasoning to optimize the student's strategy.

We utilize and compare two classical Artificial Intelligence planning approaches:

1. **GraphPlan:** Used to produce a valid linear sequence of actions required to reach a placement goal.

2. **Partial Order Planning (POP):** Used to create a flexible schedule that adheres to weekly workload constraints (parallelism).

Both methods ultimately answer the core question: *"What is the most efficient next step to achieve placement readiness?"*

## 2 Dataset and Knowledge Base

The planning algorithms rely on a domain-specific knowledge base derived from two primary sources.

### 2.1 LeetCode and DSA Metrics

We utilize performance data to quantify transitions between states like `DSA_LOW` and `DSA_MED`, estimating the specific time and effort required. This enables the planner to define precise action durations and state transitions while mitigating burnout risks.

### 2.2 Mock Interview Data

Mock interview data translates metrics into specific planning actions, such as establishing `MockInterview_full` as a prerequisite for reaching `CONF_HIGH`. This allows the planner

to construct robust heuristics based on effort estimation and confidence improvements.

# 3  System Inputs

To generate a personalized strategy, the algorithm requires a specific problem definition provided by the user. This includes three main components:

- **Initial State:** A set of facts describing the student's current proficiency.
  *Example:* {DSA_LOW, RESUME_LOW, ML_LOW, NOT_BURNOUT}

- **Goal State:** The target attributes the student wishes to achieve.
  *Example:* {DSA_HIGH, RESUME_HIGH, CONF_HIGH}

- **Constraints (Max Parallel Actions):** A real-world constraint defining how many major tasks can be handled per week. For example, a value of 1 implies the student can focus on only one major module per week.

Optionally, the system accepts a list of *Executed Actions* (e.g., Resume_Optimize :  done). This input facilitates "Plan Repair," allowing the AI to adjust the remaining schedule based on progress.

# 4  Algorithm Outputs

The two planners produce structured outputs in distinct formats suitable for different user needs.

## 4.1  GraphPlan Output (Linear Sequence)

GraphPlan generates a strict sequence of steps. It does not account for dates but ensures logical consistency.

1. Resume_Optimize_quick
2. DSA_Practice_light
3. Resume_Optimize_deep
4. DSA_Keep_practice
5. MockInterview_full

## 4.2  POP Output (Scheduled Roadmap)

The Partial Order Planner produces a Gantt-style schedule, assigning tasks to specific weeks based on effort and parallel constraints.

```
Week 0:   Resume_Optimize_quick (Duration: 1w)
Week 0-1: DSA_Practice_intense (Duration: 2w)
Week 1:   Resume_Optimize_deep (Duration: 1w)
Week 2:   MockInterview_full (Duration: 1w)
```

## 4.3 Reasoning Trace

To aid academic understanding, the system outputs a JSON log of the planning process, showing how the graph was built or how mutex constraints were resolved.

# 5 Methodology: How the Algorithms Work

## 5.1 GraphPlan

GraphPlan operates by constructing a "Planning Graph" that expands level by level. Level 0 represents the initial state. The algorithm projects all possible actions to create Level 1, then the resulting states for Level 2, and so on. Once the goal state appears in a level, GraphPlan performs a backward search to extract a valid plan.

**Strengths:** Extremely fast and guarantees finding a minimal set of steps.

**Weakness:** Lacks a concept of time duration or parallel workload limits.

## 5.2 Partial Order Planning (POP)

POP adopts a "least commitment" strategy. It begins with dummy `Start` and `Finish` actions and inserts real actions only to satisfy specific preconditions. Ordering constraints are added only when necessary to resolve conflicts (threats) between actions. Finally, a scheduler maps these valid sequences to actual weeks.

**Strengths:** Realistic handling of parallel tasks and easier plan repair during execution.

**Weakness:** Higher computational complexity compared to GraphPlan.

# 6 Worked Example

Consider a scenario where a student needs to upgrade both their DSA skills and Resume quality.

**Input:**

- Initial: `DSA_LOW, RESUME_LOW, NOT_BURNOUT`

- Goal: `DSA_HIGH, RESUME_HIGH`

- Constraint: Max 1 parallel action.

**GraphPlan Execution:** The graph expands from Level 0. In Level 1, actions like `Resume_Optimize_quick` are applied, leading to intermediate states (`RESUME_MED`). Eventually, the graph satisfies the high-level goals. The resulting linear plan might look like: *Optimize Quick → Practice Light → Optimize Deep → Mock Interview.*

**POP Execution with Scheduling:** POP recognizes that `DSA_Practice_intense` takes two weeks.

1. `Resume_Optimize_quick` is assigned to **Week 0**.

2. `DSA_Practice_intense` is scheduled across **Weeks 0–1**.

3. `Resume_Optimize_deep` is pushed to **Week 1** to respect the prerequisite chain.

4. `MockInterview_full` lands in **Week 2**.

# 7 Test Case

| Parameter | Value |
|---|---|
| Initial State | DSA_LOW, RESUME_LOW, ML_LOW, NOT_BURNOUT |
| Goals | DSA_HIGH, RESUME_HIGH, CONF_HIGH |
| Max Parallel | 1 |

Table 1: Test Case 1 Parameters

## Terminal Input (example)

```
Planner example (interactive).
Press Enter to use defaults or provide comma-separated values.

Initial state [DSA_LOW, ML_LOW, RESUME_LOW, NOT_BURNOUT]:
DSA_LOW,ML_HIGH,RESUME_LOW,BURN_OUT

Goals [DSA_HIGH, RESUME_HIGH, CONF_HIGH]:

Chosen inputs printed by script:
Initial state: BURN_OUT, DSA_LOW, ML_HIGH, RESUME_LOW
Goals: CONF_HIGH, DSA_HIGH, RESUME_HIGH
```

## Expected Outputs

**GraphPlan (example outputs):**

```
Graph construction levels (states per level):
Level 0: ['BURN_OUT', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW']
Level 1: [...]
Level 2: [...]
Level 3: [...]
Level 4: [...]
```

```
GraphPlan final ordered sequence:
1. DSA_Practice_light
2. Resume_Optimize_quick
3. Resume_Optimize_deep
4. DSA_Review
5. DSA_Keep_practice
6. MockInterview_full

GraphPlan step-by-step state progression:
[start] State: ['BURN_OUT', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW']
Step 1: Apply 'DSA_Practice_light' -> removes ['DSA_LOW'];
        adds ['DSA_MED', 'NOT_BURNOUT']
... Final state includes CONF_HIGH, DSA_HIGH, RESUME_HIGH
```

**POP (example outputs):**

```
POP scheduled plan (sorted by EST):
Resume_Optimize_quick: start_week=0, duration=1.0, effort=8.0
DSA_Practice_intense:  start_week=0, duration=1.0, effort=25.0
Resume_Optimize_deep:  start_week=1.0, duration=1.0, effort=20.0
MockInterview_full:    start_week=2.0, duration=1.0, effort=5.0
\begin{figure}
    \centering
    \includegraphics[width=0.5\linewidth]{image.png}
    \caption{Enter Caption}
    \label{fig:placeholder}
\end{figure}
POP step-by-step state progression:
Week 0: Action 'Resume_Optimize_quick' starts
\begin{figure}
    \centering
    \includegraphics[width=0.5\linewidth]{image2.png}
    \caption{Enter Caption}
    \label{fig:placeholder}
\end{figure}
Week 0: Action 'DSA_Practice_intense' starts
Week 1.0: Action 'Resume_Optimize_deep' starts
Week 2.0: Action 'MockInterview_full' starts
```

```
        benefit: {'sum_prob': 0.95, 'adds': {'CONF_HIGH': 0.95}}
        risk: {'burnout_risk_est': 0.05}

Done.
(base) charlie@charlie-HP-Pavilion-Laptop-15-eh1xxx:~/AI-Project-Group-1/Planning Placement Strategy using GraphPlan and POP$ python3 run_example.py
Loaded estimates.json (data-driven heuristics).
=====================================================================================
Planner example (interactive).
Press Enter to use defaults or provide comma-separated values.
Initial state [DSA_LOW, ML_LOW, RESUME_LOW, NOT_BURNOUT]: DSA_LOW,ML_HIGH,RESUME_LOW,BURN_OUT
Goals [DSA_HIGH, RESUME_HIGH, CONF_HIGH]:
=====================================================================================
Chosen inputs:
Initial state: BURN_OUT, DSA_LOW, ML_HIGH, RESUME_LOW
Goals: CONF_HIGH, DSA_HIGH, RESUME_HIGH
=====================================================================================

=== Running GraphPlan ===

Graph construction levels (states per level):
  Level 0: 4 state facts -> ['BURN_OUT', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW']
  Level 1: 10 state facts -> ['BURNOUT', 'BURN_OUT', 'CONF_MED', 'DSA_HIGH', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESEARCH_DONE', 'RESUME_LOW', 'RESUME_MED']
  Level 2: 9 state facts -> ['BURN_OUT', 'CONF_MED', 'DSA_HIGH', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESEARCH_DONE', 'RESUME_HIGH', 'RESUME_MED']
  Level 3: 10 state facts -> ['BURN_OUT', 'CONF_HIGH', 'CONF_MED', 'DSA_HIGH', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESEARCH_DONE', 'RESUME_HIGH', 'RESUME_MED']
  Level 4: 8 state facts -> ['BURN_OUT', 'CONF_HIGH', 'CONF_MED', 'DSA_HIGH', 'DSA_MED', 'ML_HIGH', 'RESEARCH_DONE', 'RESUME_HIGH']
  Level 5: 8 state facts -> ['BURN_OUT', 'CONF_HIGH', 'CONF_MED', 'DSA_HIGH', 'DSA_MED', 'ML_HIGH', 'RESEARCH_DONE', 'RESUME_HIGH']

Actions available per level (graph):
  Level 0: 9 actions -> ['Noop_pos(BURN_OUT)', 'Noop_pos(DSA_LOW)', 'Resume_Optimize_quick', 'Noop_pos(RESUME_LOW)', 'Noop_pos(ML_HIGH)', 'Company_Research', 'DSA_Pr
  Level 1: 18 actions -> ['Noop_pos(NOT_BURNOUT)', 'DSA_Keep_practice', 'Noop_pos(BURNOUT)', 'DSA_Review', 'Noop_pos(CONF_MED)', 'Noop_pos(RESUME_LOW)', 'Noop_pos(DS
Resume_Optimize_deep', 'MockInterview_easy', 'Noop_pos(RESUME_MED)', 'Noop_pos(RESEARCH_DONE)', 'Company_Research', 'Noop_pos(BURN_OUT)', 'Rest', 'Noop_pos(DSA_HIGH)
  Level 2: 16 actions -> ['Noop_pos(CONF_MED)', 'Noop_pos(RESUME_HIGH)', 'Noop_pos(DSA_MED)', 'Noop_pos(ML_HIGH)', 'DSA_Keep_practice', 'DSA_Review', 'Resume_Optimiz
Noop_pos(RESEARCH_DONE)', 'MockInterview_full', 'Noop_pos(BURN_OUT)', 'Company_Research', 'Noop_pos(DSA_HIGH)', 'Noop_pos(NOT_BURNOUT)', 'Encouragement']
  Level 3: 17 actions -> ['Noop_pos(RESUME_MED)', 'Noop_pos(RESEARCH_DONE)', 'Noop_pos(BURN_OUT)', 'Noop_pos(DSA_HIGH)', 'Noop_pos(CONF_HIGH)', 'Noop_pos(NOT_BURNOUT
(DSA_MED)', 'DSA_Review', 'Noop_pos(RESUME_HIGH)', 'Noop_pos(ML_HIGH)', 'Resume_Optimize_deep', 'MockInterview_easy', 'MockInterview_full', 'Company_Research', 'Enc
  Level 4: 13 actions -> ['Noop_pos(ML_HIGH)', 'DSA_Keep_practice', 'DSA_Review', 'Noop_pos(RESEARCH_DONE)', 'Noop_pos(BURN_OUT)', 'Noop_pos(DSA_HIGH)', 'Noop_pos(CO
Company_Research', 'Noop_pos(DSA_MED)', 'Noop_pos(RESUME_HIGH)', 'Encouragement']

GraphPlan final ordered sequence:
  1. DSA_Practice_light: duration=2.1614794520547944, effort=21.614794520547946, adds=['DSA_MED', 'NOT_BURNOUT'], dels=['DSA_LOW']
  2. Resume_Optimize_quick: duration=1.0, effort=8.0, adds=['RESUME_MED'], dels=['RESUME_LOW']
  3. Resume_Optimize_deep: duration=1.0, effort=20.0, adds=['RESUME_HIGH'], dels=['RESUME_MED']
  4. DSA_Review: duration=1.0, effort=6.0, adds=['DSA_MED'], dels=[]
```

```
GraphPlan final ordered sequence:
  1. DSA_Practice_light: duration=2.1614794520547944, effort=21.614794520547946, adds=['DSA_MED', 'NOT_BURNOUT'], dels=['DSA_LOW']
  2. Resume_Optimize_quick: duration=1.0, effort=8.0, adds=['RESUME_MED'], dels=['RESUME_LOW']
  3. Resume_Optimize_deep: duration=1.0, effort=20.0, adds=['RESUME_HIGH'], dels=['RESUME_MED']
  4. DSA_Review: duration=1.0, effort=6.0, adds=['DSA_MED'], dels=[]
  5. DSA_Keep_practice: duration=1.0, effort=12.0, adds=['DSA_HIGH'], dels=['DSA_MED']
  6. MockInterview_full: duration=1.0, effort=5.0, adds=['CONF_HIGH'], dels=['BURNOUT']

GraphPlan step-by-step state progression:
  [start] State (4): ['BURN_OUT', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW']
  Step 1: Apply 'DSA_Practice_light' -> removes: ['DSA_LOW'] ; adds: ['DSA_MED', 'NOT_BURNOUT']
    before: ['BURN_OUT', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW']
    after : ['BURN_OUT', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_LOW']
  Step 2: Apply 'Resume_Optimize_quick' -> removes: ['RESUME_LOW'] ; adds: ['RESUME_MED']
    before: ['BURN_OUT', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_LOW']
    after : ['BURN_OUT', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_MED']
  Step 3: Apply 'Resume_Optimize_deep' -> removes: ['RESUME_MED'] ; adds: ['RESUME_HIGH']
    before: ['BURN_OUT', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_MED']
    after : ['BURN_OUT', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_HIGH']
  Step 4: Apply 'DSA_Review' -> removes: [] ; adds: ['DSA_MED']
    before: ['BURN_OUT', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_HIGH']
    after : ['BURN_OUT', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_HIGH']
  Step 5: Apply 'DSA_Keep_practice' -> removes: ['DSA_MED'] ; adds: ['DSA_HIGH']
    before: ['BURN_OUT', 'DSA_MED', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_HIGH']
    after : ['BURN_OUT', 'DSA_HIGH', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_HIGH']
  Step 6: Apply 'MockInterview_full' -> removes: ['BURNOUT'] ; adds: ['CONF_HIGH']
    before: ['BURN_OUT', 'DSA_HIGH', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_HIGH']
    after : ['BURN_OUT', 'CONF_HIGH', 'DSA_HIGH', 'ML_HIGH', 'NOT_BURNOUT', 'RESUME_HIGH']
=====================================================================================

=== Running POP (with scheduling) ===

POP scheduled plan (sorted by EST):
  - Resume_Optimize_quick: start_week=0, duration_weeks=1.0, effort_hours=8.0
      why: Improves resume quality to pass recruiter screen.
      benefit: {'sum_prob': 0.7, 'adds': {'RESUME_MED': 0.7}}
      risk: {'burnout_risk_est': 0.05}
      adds: ['RESUME_MED']
  - DSA_Practice_intense: start_week=0, duration_weeks=1.0, effort_hours=25.0
      why: Improves algorithmic problem-solving skill (DSA).
      benefit: {'sum_prob': 1.15, 'adds': {'DSA_HIGH': 0.85, 'BURNOUT': 0.3}}
      risk: {'burnout_risk_est': 0.6}
```

```
========================================================================
=== Running POP (with scheduling) ===

POP scheduled plan (sorted by EST):
  - Resume_Optimize_quick: start_week=0, duration_weeks=1.0, effort_hours=8.0
     why: Improves resume quality to pass recruiter screen.
     benefit: {'sum_prob': 0.7, 'adds': {'RESUME_MED': 0.7}}
     risk: {'burnout_risk_est': 0.05}
     adds: ['RESUME_MED']
  - DSA_Practice_intense: start_week=0, duration_weeks=1.0, effort_hours=25.0
     why: Improves algorithmic problem-solving skill (DSA).
     benefit: {'sum_prob': 1.15, 'adds': {'DSA_HIGH': 0.85, 'BURNOUT': 0.3}}
     risk: {'burnout_risk_est': 0.6}
     adds: ['DSA_HIGH', 'BURNOUT']
  - Resume_Optimize_deep: start_week=1.0, duration_weeks=1.0, effort_hours=20.0
     why: Improves resume quality to pass recruiter screen.
     benefit: {'sum_prob': 0.9, 'adds': {'RESUME_HIGH': 0.9}}
     risk: {'burnout_risk_est': 0.05}
     adds: ['RESUME_HIGH']
  - MockInterview_full: start_week=2.0, duration_weeks=1.0, effort_hours=5.0
     why: Simulates interview to increase confidence and reveal weaknesses.
     benefit: {'sum_prob': 0.95, 'adds': {'CONF_HIGH': 0.95}}
     risk: {'burnout_risk_est': 0.05}
     adds: ['CONF_HIGH']

POP step-by-step state progression (ordered by start week):
  [start] State (4): ['BURN_OUT', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW']

  Week 0: Action 'Resume_Optimize_quick' starts (duration 1.0 weeks)
     before: ['BURN_OUT', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW']
     after : ['BURN_OUT', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW', 'RESUME_MED']

  Week 0: Action 'DSA_Practice_intense' starts (duration 1.0 weeks)
     before: ['BURN_OUT', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW', 'RESUME_MED']
     after : ['BURNOUT', 'BURN_OUT', 'DSA_HIGH', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW', 'RESUME_MED']

  Week 1.0: Action 'Resume_Optimize_deep' starts (duration 1.0 weeks)
     before: ['BURNOUT', 'BURN_OUT', 'DSA_HIGH', 'DSA_LOW', 'ML_HIGH', 'RESUME_LOW', 'RESUME_MED']
     after : ['BURNOUT', 'BURN_OUT', 'DSA_HIGH', 'DSA_LOW', 'ML_HIGH', 'RESUME_HIGH', 'RESUME_LOW', 'RESUME_MED']

  Week 2.0: Action 'MockInterview_full' starts (duration 1.0 weeks)
     before: ['BURNOUT', 'BURN_OUT', 'DSA_HIGH', 'DSA_LOW', 'ML_HIGH', 'RESUME_HIGH', 'RESUME_LOW', 'RESUME_MED']
 master*      0  0
```

# 8    Conclusion

While both algorithms effectively solve the planning problem, they serve different purposes
in the context of placement preparation. GraphPlan is excellent for determining *what*
actions are logically necessary. However, Partial Order Planning (POP) demonstrates
superior utility for real-world application. Its ability to model multi-week tasks, handle
execution failures, and respect human workload limits makes it the more adaptive and
strategic choice for guiding students through their placement journey.