Mathematical Formulation for the 7-Day Task Scheduler

Based on Python PuLP Model

March 29, 2025

1 Problem Definition

The goal is to schedule a set of tasks over a 7-day horizon, divided into discrete 15-minute time slots. The schedule should respect task deadlines, avoid overlapping tasks, avoid pre-defined blocked time intervals (commitments), adhere to user-defined daily scheduling windows and task-specific time-of-day preferences, while maximizing available leisure time and minimizing a measure of "stress" associated with scheduled tasks (based on priority and difficulty).

2 Parameters and Sets

2.1 Sets

- $I = \{1, \ldots, n\}$: Set of tasks to be scheduled.
- $T = \{0, \ldots, S-1\}$: Set of discrete time slots over the 7-day horizon. $S = \text{TOTAL\_SLOTS} = 392$.
- $T_{day} = \{0, \ldots, \text{SLOTS\_PER\_DAY} - 1\}$: Set of slot indices within a single day (0 to 55).
- $D = \{0, \ldots, \text{TOTAL\_DAYS} - 1\}$: Set of days (0 to 6).

2.2 Task Parameters (for each task $i \in I$)

- p_i: Priority of task i (integer, e.g., 1-5).
- d_i: Difficulty of task i (integer, e.g., 1-5).
- dur_i: Duration of task i in number of time slots.
- dl_i: Deadline slot index for task i. The task must be completed (i.e., its last slot must end) at or before slot dl_i.
- $Pref_i \subseteq T$: Set of allowed starting slots for task i based on user preference (e.g., morning, afternoon, evening slots).

2.3 Time Slot Parameters (for each slot $s \in T$)

- $commit_s$: Duration (in minutes, here 15) of fixed commitment/blocked time in slot s. $commit_s = 15$ if slot s is blocked, $commit_s = 0$ otherwise.

2.4 Scheduling Window Parameters

- h_{start}: Start hour of the daily scheduling window (e.g., 8 for 8:00).
- h_{end}: End hour of the daily scheduling window (e.g., 17 for 17:00).
- $s^{start}_{offset} = (h_{start} - 8) \times 4$: Starting slot index offset within any day.
- $s^{end}_{offset} = (h_{end} - 8) \times 4$: Ending slot index offset (exclusive) within any day.

2.5 Objective Function Parameters

- α: Weight for maximizing total leisure time (typically 1.0).
- β: Weight for minimizing total stress (typically 1.0).

3 Decision Variables

- $X_{i,s} \in \{0,1\}$: Binary variable. $X_{i,s} = 1$ if task i starts at time slot s, and 0 otherwise. $(\forall i \in I, s \in T)$
- $Y_s \in \{0,1\}$: Binary variable. $Y_s = 1$ if time slot s is occupied by any task, and 0 otherwise. $(\forall s \in T)$
- $L_s \in \mathbb{R}_{\geq 0}$: Continuous variable representing the amount of leisure time (in minutes) available in time slot s. $(\forall s \in T)$

4 Objective Function

The objective is to maximize the total leisure time minus the total "stress", where stress for a task is its priority multiplied by its difficulty.

$$\text{Maximize} \quad Z = \alpha \sum_{s \in T} L_s - \beta \sum_{i \in I} \sum_{s \in T} X_{i,s}(p_i \cdot d_i)$$

Note: The stress term $\sum_{s \in T} X_{i,s}(p_i \cdot d_i)$ simplifies to just $p_i \cdot d_i$ because Constraint (1) ensures each task i starts exactly once. The objective could equivalently be written as:

$$\text{Maximize} \quad Z = \alpha \sum_{s \in T} L_s - \beta \sum_{i \in I} (p_i \cdot d_i) \left(\sum_{s \in T} X_{i,s} \right)$$

5 Constraints

5.1 (1) Task Assignment

Each task must be assigned to start in exactly one time slot.

$$\sum_{s \in T} X_{i,s} = 1 \quad \forall i \in I$$

5.2 (2) Deadline Constraint

A task i cannot start at slot s if it would finish after its deadline dl_i. The task occupies slots $s, s+1, \ldots, s + dur_i - 1$. The last occupied slot must be $\leq dl_i$.

$$X_{i,s} = 0 \quad \forall i \in I, s \in T \text{ such that } s + dur_i - 1 > dl_i$$

5.3 (3) No Overlap Constraint

At most one task can occupy any given time slot t. A task i starting at st occupies slot t if $st \leq t < st + dur_i$.

$$\sum_{i \in I} \sum_{\substack{st \in T \\ st \leq t < st + dur_i}} X_{i,st} \leq 1 \quad \forall t \in T$$

5.4 (4) Daily Scheduling Window Constraint

Tasks must start and end within the specified daily time window (h_{start} to h_{end}).

- Tasks cannot start before the window begins: Let $s_{day} = s \pmod{\text{SLOTS\_PER\_DAY}}$.

$$X_{i,s} = 0 \quad \forall i \in I, s \in T \text{ such that } s_{day} < s_{offset}^{start}$$

- Tasks cannot start so late that they end after the window closes:

$$X_{i,s} = 0 \quad \forall i \in I, s \in T \text{ such that } s_{day} \geq s_{offset}^{end} - dur_i + 1$$

(This ensures that the last slot $s + dur_i - 1$ has an index modulo SLOTS_PER_DAY less than s_{offset}^{end}).

5.5 (5) Time Preference Constraint

A task i can only start in a slot s that belongs to its preferred set $Pref_i$.

$$X_{i,s} = 0 \quad \forall i \in I, s \in T \text{ such that } s \notin Pref_i$$

5.6 (6) Blocked Time / Commitment Constraint

A task i cannot start at slot st if any of the slots it would occupy $(st, \ldots, st + dur_i - 1)$ are blocked ($commit_t = 15$).

$$X_{i,st} = 0 \quad \forall i \in I, st \in T, \text{ if there exists } t \in \{st, \ldots, st + dur_i - 1\} \text{ such that } commit_t = 15$$

Alternatively, formulated slot by slot: For any blocked slot t (where $commit_t = 15$), no task i can be active during that slot.

$$\sum_{i \in I} \sum_{\substack{st \in T \\ st \leq t < st + dur_i}} X_{i,st} = 0 \quad \forall t \in T \text{ such that } commit_t = 15$$

(Note: The Python code implements this by directly setting $X_{i,st} = 0$ if the task starting at st overlaps with *any* blocked slot t.)

5.7 (7) Leisure Calculation Constraints

These constraints link the task assignments ($X_{i,s}$) to the slot occupancy (Y_s) and the leisure time (L_s).

- Link X to Y: If any task occupies slot s, Y_s must be 1. Due to the non-overlap constraint (3), the sum is always ≤ 1.

$$\sum_{i \in I} \sum_{\substack{st \in T \\ st \leq s < st + dur_i}} X_{i,st} \leq Y_s \quad \forall s \in T$$

- Link Y and $commit$ to L: Leisure time L_s in a slot s is at most the total time in the slot (15 minutes) minus any committed time, and is only non-zero if the slot is not occupied by a task ($Y_s = 0$).

$$L_s \leq (15 - commit_s) \cdot (1 - Y_s) \quad \forall s \in T$$

  Since $L_s \geq 0$ is also enforced, L_s will be exactly $(15 - commit_s)$ if $Y_s = 0$ and $commit_s = 0$, exactly 0 if $Y_s = 1$, and exactly 0 if $commit_s = 15$. The objective function maximizing L_s ensures L_s takes its maximum possible value allowed by this constraint.

5.8 (8) Variable Types

Ensure variables adhere to their defined types.

$$X_{i,s} \in \{0,1\} \quad \forall i \in I, s \in T$$
$$Y_s \in \{0,1\} \quad \forall s \in T$$
$$L_s \geq 0 \quad \forall s \in T$$