Mathematical Formulation and Implementation of the Book Scanning Optimization

Google Hash Code 2020 - CP-SAT Solver

Problem Overview

Given a set of books and libraries, each with specific constraints, we aim to maximize the total score of books scanned in a limited number of days.

Sets and Parameters

- $B = \{1, 2, \ldots, N_B\}$: Set of books
- $L = \{1, 2, \ldots, N_L\}$: Set of libraries
- D: Total number of days
- s_l: Signup duration of library l
- c_l: Number of books library l can ship per day
- $\text{books}_l \subseteq B$: Books in library l
- v_b: Score of book b

Decision Variables

- $y_l \in \{0, 1\}$: Whether library l is selected
- s_l^{start}: Start day of signup for library l
- $z_{l,b} \in \{0, 1\}$: Whether book b is scanned by library l

Implementation

```
y = {l: model.NewBoolVar(f"y[{l}]") for l in L}
start = {l: model.NewIntVar(0, D - 1, f"s[{l}]") for l in L}
z = {}
for l in L:
    for b in libraries[l]["books"]:
        z[(l, b)] = model.NewBoolVar(f"z[{l},{b}]")
        model.Add(z[(l, b)] <= y[l])
```

Objective Function

$$\max \sum_{l \in L} \sum_{b \in \text{books}_l} v_b \cdot z_{l,b}$$

Implementation

```
model.Maximize(sum(book_scores[b] * z[(l, b)] for (l, b) in z))
```

Constraints

C1. Library Book Assignment Only If Selected

$$z_{l,b} \leq y_l \quad \forall l, b \in \text{books}_l$$

```
model.Add(z[(l, b)] <= y[l])
```

C2. Unique Book Scanning

$$\sum_{l:b \in \text{books}_l} z_{l,b} \leq 1 \quad \forall b \in B$$

```
for b in B:
    zlist = [z[(l, b)] for l in L if (l, b) in z]
    if zlist:
        model.Add(sum(zlist) <= 1)
```

C3. Shipping Capacity

Let $d_l^{\text{ship}} = D - s_l - s_l^{\text{start}}$:

$$\sum_{b \in \text{books}_l} z_{l,b} \leq c_l \cdot d_l^{\text{ship}} + M \cdot (1 - y_l)$$

```
for l in L:
    max_days = D - libraries[l]["signup"] - start[l]
    cap = libraries[l]["ship"]
    big_m = len(libraries[l]["books"])
    model.Add(sum(z[(l, b)] for b in libraries[l]["books"]) <= cap * max_days + big_m * (1 -
        y[l]))
```

C4. No Overlapping Signups

$$\text{interval}(l) \cap \text{interval}(k) = \emptyset$$

```
interval = {
    l: model.NewOptionalIntervalVar(start[l], libraries[l]["signup"], start[l] + libraries[l
        ]["signup"], y[l], f"int[{l}]")
    for l in L
}
model.AddNoOverlap(interval.values())
```

Warm Start Heuristic (Greedy)

Use a greedy heuristic to provide hints to the solver.

```
g_order, g_books = greedy_schedule(D, libraries)
acc = 0
for l in g_order:
    model.AddHint(y[l], 1)
    model.AddHint(start[l], acc)
    acc += libraries[l]["signup"]
    for b in g_books[l]:
        if (l, b) in z:
            model.AddHint(z[(l, b)], 1)
```

Solving and Output

```
solver = cp_model.CpSolver()
solver.parameters.max_time_in_seconds = time_limit_s
solver.parameters.num_search_workers = workers
status = solver.SolveWithSolutionCallback(model, cb)

if status not in (cp_model.OPTIMAL, cp_model.FEASIBLE):
    raise RuntimeError(solver.StatusName(status))
```