Integer Programming Approach to Shift Generation

Summary of 2020 Summer Intern
Python Development

Approach

- Integer Programming (IP): Solve or optimize a constrained problem with integer variables
- Our Problem: maximize multi-base, multi-position shifts assigned to patterns
  - Assign as many shifts as possible (hopefully all) to patterns
  - Constraints:
    - Qualifications: base and/or position
    - Rules: e.g. no consecutive shifts in the pattern
    - Cost: e.g. smooth load over staff member (cost of overtime)
    - Preferences: e.g. standard patter 1-0-1-0-0-0-0-0

Python

- Python MIP tools
  - Supports/solves Mixed Integer Programming (IP is a subset of MIP)
    - Branch & Cut (branch & bound with cutting plane reduction)
- Variables
- Objective Function
- Constraints

Formulation

- Variables:
  - Bases (2), positions (2), clinicians/patterns (8), days(28) (also a week variable based on days)
  - Shifts (assumed 24 hrs, no sensitivity to start times)
  - Pseudo variable for max shift count
- Objective Function
  1. max(shifts assigned to patterns)
  2. + max(1-0-1 patterns)
  3. + min(max total shifts per pattern) *smoothing assignment over clinicians
     * Weighting each clause for relative importance: 1 for coverage, .5 for total shifts, .25 for 1-0-1 patterns
- Constraints
  1. 1 pattern assignment per shift per position per base per day (don't over assign)
  2. Assigned clinician is base qualified and position qualified
  3. No consecutive days assigned to a clinician/pattern
  4. At least 1 shift per clinician/pattern per week
  5. No more than x shifts per clinician/pattern per week (set to 2 in standard form)

Python Code

```
m = Model()

##VARIABLES
# binary for each day, clinician (S), position, base
x = [[[[m.add_var(var_type=BINARY) for d in D] for s in S] for p in P] for b in B]

#maximum of total days assigned per pattern
M = m.add_var(var_type=INTEGER)

##OBJECTIVE FUNCTION
m.objective = maximize(
                #maximize covered shifts
                1*(xsum(x[b][p][s][d] for b in range(B.shape[0]) for p in range(P.shape[0]) for s in range(S.shape[0]) for d in range(D.shape[0])))
                #minimize the max total shifts (shift evenly over staff)
                -0.5*M
               )

##CONSTRAINTS##
#one assignment per position per base per day
for d in range(D.shape[0]):
   for p in range(P.shape[0]) :
      for b in range(B.shape[0]):
         m += xsum(x[b][p][s][d] for s in range(S.shape[0])) <= 1, 'max one staff per position per day'
for s in range(staffToPattern):  #for each pattern

   for d in range(D.shape[0]-1):#no consecutive days
      m += xsum(x[b][p][s][d] for b in range(B.shape[0]) for p in range(P.shape[0]))+xsum(x[b][p][s][d+1] for b in range(B.shape[0]) for p in range(P.shape[0])) <= 1, 'No consecutive days'
   m += xsum(x[b][p][s][0] for b in range(B.shape[0]) for p in range(P.shape[0]))+xsum(x[b][p][s][D.shape[0]-1] for b in range(B.shape[0]) for p in range(P.shape[0])) <= 1, 'No consecutive
days--end of pattern'

   #to minimize/even out: count ttl shifts per staffer
   m += xsum(x[b][p][s][d] for d in range(D.shape[0]) for b in range(B.shape[0]) for p in range(P.shape[0])) <= M, 'count total shifts per staffer'

m.optimize()
```

Results & Discussion

- Notes on the formulation:
  - Clause 2 of objective function rewards 1-0-1 patterns but also rewards 1-0-1-0-1-0-1
  - Basic formulation is pattern generation. Including qualifications implies assignment
- Feasibility vs Optimality
  - Feasible: can we solve the problem?
    - Yes, if we don't over-constrain, Patterns are generated
  - Optimality
    - What are optimal patterns? Can we tailor the program to reward preferred patterns without impacting feasibility?
    - How do we tune the weights of the objective function clauses? Analyze multiple feasible solutions to tune objective function for optimality.
- How can the model be used?
  - Multi-base: for close bases, constraints to set patterns (e.g. 1-0-1-0-0-0-0-0) with open base assignment
  - Proposing Alternative patterns: base patterns surrounding the fixed patterns (e.g. 2 week traveller pattern or tues/thurs pattern)
  - Mix the 2 above—does this reduce the disruption of base patterns that results from traveller pattern?
  - Others?