Implementation Plan for LNS heuristik

Claes Arvidson, Emelie Karlsson

April 5, 2016

First version

Requirements

- Implement a 5 week schedule
- Library on Wheels not considered
- Softer values such as PL, free day or HB preferences not considered
- Meetings not considered
- Objective function:
  - Minimize violated worker demand. The total number of tasks is constant, as well as the weekly number of tasks. However, days and shifts can violate demand constraints (relaxed hard constraint)
  - Maximize number of stand in librarians
  - Maximize number of stand in assistants
- Hard constraints:
  - 1 task/day for a worker
  - worker must be available to perform task
  - worker must be qualified for task (lib/ass)
  - the total number of tasks to be performed is constant

Implementation: task based approach

- Initial solution. Firstly, distribute weekends to obtain a feasible weekend schedule. Secondly, distribute tasks according to some rule (for example by finding the critical task hours). Initial solution not necessarily feasible with respect to the worker demand at each shift.
- Destroy and repair:

- Weekend destroy: Destroy a subset of all weekends. Subsequently destroy the week follwing the new weekends to create a feasible schedule.
- Weekend repair: Repair by first placing new weekends, then, after second destroy, place tasks from week rest at deficit days.
- Task destroy: Identify the days with the most **surplus** of workers. Also being an unused stand in generates some cost. Destroy these days.
- Task repair: Repair by redistributing over the destroyed days and the days with **deficit** of workers. Create stand ins by placing tasks for workers who are not stand in avail.

- Possible classes:
  - Library. Contains:
    - Demand schedule (int [w][d][s])
    - Total_cost
    - Cost function for weekend destroy
    - Cost function summing workers in a week day (int[d])
    - Cost function summing number of unused stand ins in a week day (int[d])
  - Worker. Contains:
    - Availability (int [[d][s], [d][s], ...])
    - Stand in avail (int [w][d])
    - Identity of worker (ID, name, position, preferences etc.)
    - Current schedule variables (int tasks [w][d][s],int num_tasks[w][d], int num_PL[w][d], int weekend...)
    - Cost function calculating worker satisfaction with schedule (int [w], int [d], implemented later...)

Implementation: week block based approach

Implementation: week block based approach

- Initial solution. Generate a schedule by assigning a feasible solution for each worker. The resulting schedule will most likely violate the demand constraint that is relaxed into the objective function.

- Classes in consideration:
  - *Library* (Information about current demand, total demand, workers assigned at a day d and objective functions such as number of stand-ins are stored here)
  - *Worker* (contains all personal information about demands, qualifications, assigned blocks, blocks available for assignment etc.)

  - *Block* (a week long block containing an ID, type of block (weekend, weekrest, normal), number of tasks, tasks with information of day, shift and tasktype etc)
  - $Pair\_of\_blocks$ (used later when implementing for 10 week period, containts information of $shift\_differ$ and which two blocks that forms the pair)

- Destroy:
  - Destroy week 1 for all workers (10 % destroy) - need more than one week to move weekends (and weekrest).
  - Destroy five consecutive weeks for around a fifth of the workers
  - Identify and order the most problematic weeks for all workers and destroy in the order. Add some randomization as well to prevent getting stuck on a plateau.

- Repair:
  - Generate a new schedule with regard to number of tasks destroyed. The repair function searches through the pool of blocks available for each person and creates a new solution using a heuristic.
  - The heuristic takes the demand and availability for a person into account. If a worker is available for a whole day - minimize the number of tasks given that day in order to generate a stand-in.

Second version

- Implement a 10 week schedule
- Library on Wheels considered
- Meetings considered
- Objective function:
  - Minimize unfulfilled worker demand (relaxed hard constraint)
  - Maximize number of stand in librarians
  - Maximize number of stand in assistants
  - Maximize similar weeks for workers