Planning algorithm

Ladislav Petera

January 5, 2018

1 Problem space

The number of possible permutations of sub tasks within the sprints is

$$n = (n_{team} \cdot n_{sprint})^{\sum_{i=0}^{n_{task}} w_i}$$

where

- n - number of possible solutions
- n_{team} - number of teams
- n_{sprint} - number of sprints
- n_{task} - number of planned tasks
- w_i - work of given task

In our example we will get $n = (5 \cdot 11)^{280} = 1.6 \cdot 10^{687}$. For comparison - the estimated number of atoms in the known universe is around 10^{80}

Let's consider a more realistic example:

- 5.000 story points (50 projects 100 SP each)
- 16 sprints (a year)
- 6 teams

This will result in problem space $n = (6 \cdot 16)^{5000} = 2.2 \cdot 10^{9911}$. Looks like we are going to have to optimize a bit in order to be able to get some usable solution in reasonable time (smaller than few billion years).

2 Planning constraints

Hard constraints The hard constraints limit what is physically possible. The construction heuristics will initialize our solution within these constraints.

- team must have required skill
- task dependency must not be violated
- sprint velocity must not be exceeded

- max task velocity per spring must not be exceeded
- task cannot start before first possible sprint
- one task can only be done by one team

Soft constraints The soft constraints create pressure towards the desirable output.

- minimize costs of delay
- sprints should have high utilization

3 Construction heuristic

The construction heuristics phase quickly initializes our solution. We will pre sort the planning entities and then use the FIRST_FIT_DECREASING algorithm. This algorithm cycles through all the planning entities starting with the most difficult ones, initializing one entity at a time. It takes the best available value and continues with the next entity.

A meaningful sort of the entities matching the constraints highly improves the chances of this phase to produce a meaningful result:

- blocking tasks before blocked tasks
- tasks of earlier projects before tasks of later projects
- projects with higher delay costs before projects with lower delay costs

Splitting up tasks is smaller subtasks will make it easier to find a relatively good solution based solely on construction heuristic. However it will lead to problems during the meta heuristic.

4 Metaheuristic

Every metaheuristic will be facing the problem, that a move of an sub-task seldom leads to a score improvement because it is the last sub-task in a task which really matters. This will result in discarding moves which lead towards global minimum. Combination of this fact and the huge problem space leads to an almost impossible optimization task for the metaheuristic.

4.1 Move selector

Finding a good method for selecting the next move is crucial.

Brain storming:

- priorize sub tasks of already selected task for move selection
- priorize blockers of moved tasks for move selection

4.2 Score function improvement

Alternatively we need to make the sub tasks have an effect on the score.

Brain storming:

- currently we calculate score sometimes based on the tasks, sometimes based on subtasks. Changing all to sub tasks might have a positive effect.