Planning algorithm for training cosmonauts in ISS

Alexander Lazarev[1,2,3,4], **Alexander Sologub**[2,4]

[1] *National Research University Higher School of Economics, Moscow;*
[2] *Lomonosov Moscow State University, Moscow;*
[3] *Institute of Physics and Technology State University, Moscow;*
[4] *Institute of control sciences, Moscow;*
jobmath@mail.ru; sologub10@gmail.com

We consider the problem of automated scheduling for Internation Space Station (ISS). During planning activities, an important step is the process of preparing cosmonaut to expedition. At this stage, we need to choose which works to teach each of the cosmonauts. Let flight mission comprises a plurality of operations $N = \{1, ..., n\}$ for the set of cosmonauts $M = \{1, ..., m\}$. In this problem, we believe that we know how much time is needed to prepare each of the cosmonaut $j \in M$ on each of the works $i \in N$ and this time equals $p(i, j, q)$, where $q \in \{0, .., , k\}$ is a quality of preparation. An important parameter in this problem is the time C when we must end training and should be ready to fly. Required to arrange work so that the preparation of each participant flight ended in about the same time.

The mathematical formulation

$$\text{minimize} \quad \delta$$

$$\text{subject to} \quad C - \sum_{q=1}^{k} \sum_{i=1}^{n} p(i, j, q)\, x(i, j, q) \le \delta, \quad j = 1, ..., m, \tag{1}$$

$$\sum_{j=1}^{m} \sum_{q=1}^{k} x(i, j, q) = 1, \qquad i = 1, ..., n, \tag{2}$$

$$x(i, j, q) = \{0, 1\}, \tag{3}$$

where $x(i, j, q)$ is a binary variable equalling 1 if work i is given to cosmonaut j and the training goes with quality q, and 0 otherwise.

This problem is $\mathcal{NP}$-hard in the strong sense, thus any dinamic programming approach would result in strictly exponential time bounds [1]. So in this work proposed to use a greedy algorithm with additions.

Lets introduce the additional function δ_i^j [2].

D e f i n i t i o n 1. Functions δ_i^j, $i = 1, ...n$, $j = 1, ...m$ describes the offset from the end of the overall preparation time at j cosmonaut, adding to

his training work i

$$\delta_i^j = \begin{array}{c} \delta_{i-1}^j - p(i,j,k), \text{ if we decide to prepare } j \text{ cosmonaut to } i \text{ work,} \\ \delta_{i-1}^j, \text{ if we don't set } j \text{ work.} \end{array}$$

For all $j = 1, ..., m$ we have the initial conditions $\delta_0^j = C$.

As an objective function for a fixed i choose

$$\text{minimize} \quad \max_j \delta_i^j - \min_j \delta_i^j, \quad j = 1, ..., m,$$

$$\text{subject to} \quad (1), (2), (3),$$

where minimum is selected for the job among all possible permutations. Going all the work we got the smallest variations in the δ for given permutation. The run time of this algorithm is $O(mn)$.

Lets take into account the additional condition $\delta_i^j \geq 0$. If at some stage i this condition is no longer met, the record number of this step in the memory as $b = i$, that is as the value of the present boundary conditions. For b step set to work preparing a lower quality, i.e. such, for which preparations are going faster. Next, the algorithm continues as usual [3].

If either step in what additional condition can not be met under any preparation time, then return to step $i = b - 1$ and put it with the worst quality of training and continue the algorithm. Thus we find the right solution either, or complete inability to scheduling under such initial parameters.

References

1. *David Pisinger, Hans Kellerer, Ulrich Pferschy* Knapsack problems // Springer. 2004
2. *A.A. Lazarev, E.R. Gafarov* Scheduling theory. Problems and algorithms, Lomonosov Moscow State University, Moscow (2011). (in Russian)
3. *Cormen, Thomas H. and Stein, Clifford and Rivest, Ronald L. and Leiserson, Charles E.* Introduction to Algorithms, McGraw-Hill Higher Education, 2001