Optimal Task Assignment by Learning Members' Skills (MURI)

Omid Askari

November 4, 2015

1 Problem Setup

Consider we have a group of people and there is a sequence of tasks. Suppose each person has a set of skills and each task requires some of these tasks to a extent.

$people = \{p_1, p_2, ..., p_n\}, \quad tasks = \{t_1, t_2, ..., t_T\}$

Each person $p = (s_1, s_2, ..., s_m)$ and task $t = (r_1, r_2, ..., r_m)$ has m skills set and requirements. We suppose that skill values are discrete and fall in $[0, b]$ and b is a configuration parameter which directly affects problem's complexity. The larger b is, bigger the boundary of learning space would be.

2 Problem Definition

Suppose we know task requirements; however, people's skills are unknown. We need to learn the values of people's skills through applying payoff function. We also consider that for handling each task, merely one person is needed. Additionally, payoff function f for person p and task t is defined as follows

$$f(p,t) = \begin{cases} 1, & s_1 \geq r_1 \wedge s_2 \geq r_2 \wedge ... \wedge s_m \geq r_m \\ 0, & otherwise \end{cases} \tag{1}$$

In fact, the goal is handling more tasks successfully in the long run.

3 Proposed Method

To maximize the performance, we have to learn people's skills. If we know everybody's expertises, we assign best people to their matching task to a great extent To learn, we propose a simple rule-based learning approach. If agent handles successfully the task, it conveys:

$s_1 \geq r_1 \wedge s_2 \geq r_2 \wedge ... \wedge s_m \geq r_m$

and he or she fails it means:
$s_1 < r_1 \vee s_2 < r_2 \vee ... \vee s_m < r_m$

In this method, we can learn, and hence, limit the boundaries for each person's skills with assigning more tasks to him or her.

3.1 Rule-based learning example

For a simple example, we consider person p with 2 unknown skills, $p = (x, y)$. Then, we assign task $t_1(2, 5)$ which requires known values for each skills as follows,

$p(x, y), t_1(2, 5) =>$ success, therefore : $x \geq 2 \wedge y \geq 5$

Then we assign another task,

$p(x, y), t_1(4, 1) =>$ failure, therefore : $x \geq 2 \wedge y \geq 5 \wedge (x < 4 \vee y < 1)$

Thus with a simple logic we can have,

$2 \leq x < 4 \wedge y \geq 5$.

Now we have a smaller boundary for x and y and having this knowledge could pave the way for deciding more efficient in task assignment to person p.

3.2 Method Steps

One goal in this study could be finding a lower bound for the number of task assignment to a single person with m different skills that vary in $[0, b]$ to ensure that more than 90% of his or her skill values are uncovered so far. Even if we consider that pay off function is stochastic, this will still be useful to have a tight boundary for each person's skill.

First we start with no knowledge about people's skills and initialize skill values with 0s. To learn actual values there are two important phases in each round. In every round, people either explore or exploit. There is a method to choose one of these phases; however, first we define them descriptively in the following.

3.2.1 Exploration

In exploration, we choose one person who is completely unknown and has all 0s in his or her skills. If there is no one left unknown in people, then we will choose that person has the largest boundary distance (who is more unknown). To do this, we define a boundary $[x, y]$ such that $x \geq 0$ and $y \leq b$ for each skill in every person. Boundary distance is defined as

$$dist = \sqrt{\sum_{i=1}^{m} (x_i - y_i)^2} \tag{2}$$

If there are more than one person having the largest distance, then we choose one of them by random.

3.2.2 Exploitation

In exploitation, we choose one of known people (they have skills values greater than 0) who has the largest skills for the require ones and it means it is the most promising one for handling the task successfully (however in this method we are somehow wasting resources and there is a more intelligent idea which is presented in the 4 section). The selection method for task $t = (r_1, r_2, ..., r_m)$ and if each person such as $p = (s_1, s_2, ..., s_m)$ is

$$\arg\max_p o(t, p) \tag{3}$$

$$o(t,p) = \begin{cases} \sqrt{\sum_{i=1}^{m}(s_i - r_i)^2}, & \text{if } s_1 \geq r_1 \wedge s_2 \geq r_2 \wedge ... \wedge s_m \geq r_m \\ -1, & otherwise \end{cases}$$

3.2.3 Exploration and exploitation trade-off

One of the most substantial part of this setting should be a policy to manage exploration and exploitation. First, we start with a constant probability such as c (which means $0 < c < 1$) for exploration $P_{exploration} = c$ and $P_{exploitation} = 1 - c$ and analyze how this parameter could affect the performance in the long run. To do this, we run the whole setting with different values for c. Second, there are various existing methods in Multi Armed Bandits literature [1, 2, 3] to manage between exploration and exploitation; however, for the sake of simplicity, we will apply the following method which is completely logical,

$$P_{exploration} = \frac{\mathrm{c}}{t} \tag{4}$$

where c is a constant and t is the time-step variable. In the beginning exploration power is very high; however, since t increases over time; as a consequence, the exploitation power gradually increases and exploration decays. This concept is inspired since in the beginning we do not have enough knowledge and is better to explore more; nonetheless, as time passes, we know more and hence it is better to exploit more and refine the available solutions.

4 Extensions

1. Find a group of people instead of just one person to handle a task.
2. We suppose there is a prior knowledge regarding each person; however, this prior probability gradually change through time with a Bayesian learning.
3. The payoff could be continuous instead of binary. Also it cannot be euclidean distance for all, thus:

$$f(p,t) = \sqrt{\sum_{i=1}^{m} u_i}; \quad u_i = \begin{cases} (r_i - s_i)^2, & \text{if } r_i \geq s_i \\ 0, & otherwise \end{cases} \tag{5}$$

4. The payoff function could be stochastic as follows:

$$f(p,t) = \begin{cases} 1, & \text{if } s_1 + \mathcal{N}(1,\sigma^2) \geq r_1 \wedge s_2 + \mathcal{N}(1,\sigma^2) \geq r_2 \wedge ... \wedge s_m + \mathcal{N}(1,\sigma^2) \geq r_m \\ 0, & otherwise \end{cases} \tag{6}$$

5. We can use some other multi armed bandits algorithms (contextual MAB) and for instance Soft Max to facilitate exploration and exploitation trade-off [2, 3].

6. If we consider that tasks are decomposable then we can have a group of people handling a task instead of solely one person. Therefore, to solve this problem, we will use solutions for multiple knapsack problem [4].

7. In exploitation phase, it is more logical choosing whom is the most appropriate for the task and not who has the largest skills since we are somehow wasting resources and we are using the best person for a easy task. The selection method could be changed as

$$\arg\min_p o(t,p) \ st \ o(t,p) > 0 \tag{7}$$

$$o(t,p) = \begin{cases} \sqrt{\sum_{i=1}^{m}(s_i - r_i)^2}, & \text{if } s_1 \geq r_1 \wedge s_2 \geq r_2 \wedge ... \wedge s_m \geq r_m \\ -1, & otherwise \end{cases}$$

8. We also can suppose that people after handling a task, grow some knowledge and experience. Consequently, their skills are gradually increased. For instance after being successful in a task, skill i will be updated as

$$s_i \leftarrow s_i + \alpha.r_i \tag{8}$$

And if he or she fails then

$$s_i \leftarrow s_i + \beta.r_i \tag{9}$$

Where α and β both are configurable parameters such that $0 < \alpha < 1$, $0 < \beta < 1$, $\alpha > \beta$. Hence, if one task requires one specific skill more than other, person will increase that skill more comparing other ones.

References

[1] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[2] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

[3] Volodymyr Kuleshov and Doina Precup. Algorithms for multi-armed bandit problems. *arXiv preprint arXiv:1402.6028*, 2014.

[4] Chandra Chekuri and Sanjeev Khanna. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM Journal on Computing*, 35(3):713–728, 2005.