of the additional target. This can be addressed by solving four optimization problems, one for each component of $\mathbf{x}^L, \mathbf{x}^U$. For instance, to compute the first component of $\mathbf{x}^L$, agents define the objective vector $c = [1, 0]^\top$ and they cooperatively solve the optimization problem

$$\begin{aligned} &\min_{\mathbf{x}} \; c^\top \mathbf{x} \\ &\text{subj. to } \mathbf{x} \in \bigcap_{i=1}^{N} X_i, \end{aligned} \tag{1.9}$$

which is in the common cost form (1.2). After an optimal solution $\mathbf{x}^\star$ is found, each agent computes the first component of $\mathbf{x}^L$ by using the first component of $\mathbf{x}^\star$, and similarly for the other coordinates.

1.3.5 Task allocation/assignment

Task allocation is a building block for decision making problems in which a certain number of agents must be assigned given tasks. The goal is to find the best matching of agents and tasks according to a given performance criterion. Here, we consider N agents and N tasks and we look for a one-to-one assignment. Define the variable $x_{i\kappa}$, which is 1 if agent i is assigned to task κ and 0 otherwise. Also, define the set E_A, which contains the tuple (i, κ) if agent i can be assigned to task κ. Finally, let $c_{i\kappa}$ be the cost occurring if agent i is assigned to task κ. In Figure 1.12, we show an illustrative example of the set-up.

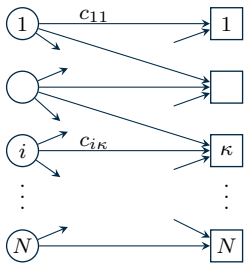

Figure 1.12: Graphical representation of the task assignment problem. Agents are represented by circles, while tasks are represented by squares. An arrow from agent i to task κ means that agent i can perform task κ (i.e., $(i, \kappa) \in E_A$) with corresponding cost equal to $c_{i\kappa}$.

Since the objective is to minimize the total cost, the task allocation problem can be formulated as an integer program. However, as pointed out in [14], integrality constraints

can be dropped to obtain the linear program

$$\begin{aligned} \min_{\mathbf{x}} \quad & \sum_{(i,\kappa)\in E_A} c_{i\kappa} x_{i\kappa} \\ \text{subj. to } & \mathbf{0} \leq \mathbf{x} \leq \mathbf{1}, \\ & \sum_{\{\kappa|(i,\kappa)\in E_A\}} x_{i\kappa} = 1 \quad \forall\, i \in \{1,\dots,N\}, \\ & \sum_{\{i|(i,\kappa)\in E_A\}} x_{i\kappa} = 1 \quad \forall\, \kappa \in \{1,\dots,N\}, \end{aligned} \tag{1.10}$$

where $\mathbf{x}$ is the variable stacking all $x_{i\kappa}$. If problem (1.10) is feasible, it can be shown that it admits an optimal solution such that $x_{i\kappa} \in \{0,1\}$ for all $(i,\kappa) \in E_A$ (see, e.g., [14]). Moreover, all the optimal assignments belong to the optimal solution set of problem (1.10).

Problem (1.10) can be cast to the constraint-coupled form (1.3). To see this, let us define K_i as the number of tasks that agent i can perform (i.e., $K_i = |\{\kappa \mid (i,\kappa) \in E_A\}|$). We assume that agent i deals with the variable $\mathbf{x}_i \in \mathbb{R}^{K_i}$, stacking the $x_{i\kappa}$ for all κ such that $(i,\kappa) \in E_A$. Then, the local sets X_i can be written as

$$X_i = \{\mathbf{x}_i \in \mathbb{R}^{K_i} \mid \mathbf{0} \leq \mathbf{x}_i \leq \mathbf{1} \text{ and } \mathbf{x}_i^\top \mathbf{1} = 1\}, \quad i \in \{1,\dots,N\}. \tag{1.11}$$

The coupling constraints can be written by defining, for all $i \in \{1,\dots,N\}$, the matrix $H_i \in \mathbb{R}^{N\times K_i}$, obtained by extracting from the $N \times N$ identity matrix the subset of columns corresponding to the tasks that agent i can perform. Problem (1.10) becomes

$$\begin{aligned} \min_{\mathbf{x}_1,\dots,\mathbf{x}_N} \quad & \sum_{i=1}^{N} c_i^\top \mathbf{x}_i \\ \text{subj. to } & \mathbf{x}_i \in X_i, \quad i \in \{1,\dots,N\} \\ & \sum_{i=1}^{N} H_i \mathbf{x}_i = \mathbf{1}, \end{aligned}$$

where each c_i stacks the costs $c_{i\kappa}$, for all κ such that $(i,\kappa) \in E_A$. Notice that problem (1.10) can be also tackled by resorting to its dual, which can be solved by using distributed optimization algorithms for common-cost problems.

In a distributed context, the goal for the agents is to find an optimal solution $\mathbf{x}^\star$, but each agent i is only interested in computing its portion $\mathbf{x}_i^\star$ of optimal solution, which contains only one entry $x_{i\kappa} = 1$, corresponding to the task κ that agent i is eventually assigned.