Predictive Task Allocation

Saptarshi Bandyopadhyay

I. Motivation

Predictive task allocation is the problem of assigning tasks to robotic agents based on their state/capabilities and on the uncertain distribution of future tasks. Predictive task allocation determines which robot or group of robots will perform each task based on the robots' states (e.g., position, attitude, power-level) and capabilities (e.g., mobility, availability of appropriate sensor) and on the predicted distribution of future tasks (e.g. the predicted location of future areas of interest, or the likelihood that one of the robots will fail). Predictive task allocation can be used both for closely coupled coordination (e.g., deciding which robot takes what place in a formation) and for loosely coupled coordination (e.g., which robot traverses to a new spot to observe an interesting target).

The task allocation problem has long been of interest in the robotics community. Proposed decentralized solution techniques include auction algorithms, spatial partitioning algorithms, and Markov-chain based algorithms; centralized team-forming and temporal partitioning algorithms and centralized matching algorithms are also available. However, no algorithm is available to solve the predictive task allocation problem in a distributed manner for networks of heterogeneous agents (see Table).

Here we aim to design a near-optimal centralized algorithm for the predictive task allocation problem for the case where the probability distribution of the events that trigger new tasks is known a priori.

II. Problem Statement

A. Define variables

- We state this problem in discrete time. Let k represent the time index, and T is the total time. Therefore, $k \in \{1, \ldots, T\}$.
- The number of agents at time k is represented by N_k. Let i_k represent an agent index at time k, therefore $i_k \in \{1, \ldots, N_k\}$. These are heterogeneous agents with heterogeneous capabilities.
- The number of tasks at time k is represented by M_k. Let j_k represent a task index at time k, therefore $j_k \in \{1, \ldots, M_k\}$. Some of of these tasks are all-time tasks (e.g. patrolling) and some are event-driven tasks (e.g. tracking an intruder).
- Let x_{i_k,j_k} represent the indicator variable of agent i_k doing task j_k, i.e., $x_{i_k,j_k} = 1$ if agent i_k does task j_k at time k and $x_{i_k,j_k} = 0$ otherwise.
- Let $\mathcal{X}_{1:k-1}$ represent the list of all actions taken by the agents from time $1, \ldots, k-1$, i.e.,

$$\mathcal{X}_{1:k-1} = \{x_{i_\tau,j_\tau}, \forall \tau \in \{1, \ldots, k-1\}, \forall i_\tau \in \{1, \ldots, N_\tau\}, \forall j_\tau \in \{1, \ldots, M_\tau\}\} \tag{1}$$

- Let the matrix $C_{k,\mathcal{X}_{1:k-1}} \in \mathbb{R}^{N_k \times M_k}$ represent the cost matrix at time k. The element $C_{k,\mathcal{X}_{1:k-1}}[i_k, j_k]$ represents the value of doing task j_k by agent i_k at time k, which depends on the past actions $\mathcal{X}_{1:k-1}$. It can be composed of 2 terms:

$$\begin{aligned} C_{k,\mathcal{X}_{1:k-1}}[i_k, j_k] = &(\text{Intrinsic value of doing the task } j_k, \text{ which depends on } \mathcal{X}_{1:k-1}) \\ &- (\text{Effort incurred by agent } i_k \text{ to do the task } j_k, \text{ which depends on } \mathcal{X}_{1:k-1}) \end{aligned} \tag{2}$$

  Here "intrinsic value of doing the task j_k" captures the importance of the task, e.g., patrolling an area is significantly lower importance than checking out an intruder, but patrolling an area that hasn't been visited in some time is more valuable than patrolling a recently visited area. "Effort incurred by agent i_k to do the task j_k" captures if an agent can do the task (if not, then it is set to ∞) and how much effort does it need to expend, e.g., tasks which need the agent to travel large distances should cost more.

B. Additional comments about variables

We now state some important aspects on tasks and costs:

- If the same task j_k is present in time k and time $k+1$, and it was not done at time k, then "intrinsic value of doing the task j_k" $\leq$ "intrinsic value of doing the task j_{k+1}" (e.g., a region that hasn't been patrolled at time k will have more intrinsic value at time $k+1$)
- New tasks j_k could get created at time k with some known probability distribution. (e.g., intruder could enter the controlled space)
- Doing task j_k could trigger/create multiple task at time $k+1$. (e.g., after checking out an intruder to be friendly or not-friendly, different tasks are triggered)
- Some un-done task from time k could be destroyed at time $k+1$. (e.g., intruder leaving the controlled space)
- Agents could be created or destroyed at any time k (e.g., agent fails, or new agent is added)
- The cost function $C_{k,\mathcal{X}_{1:k-1}}[i_k, j_k]$ at time k depends on $\mathcal{X}_{1:k-1}$, i.e., the previous actions taken at times $1, \ldots, k-1$

C. State Nonlinear Optimization Problem

Finally, we are ready to state the predictive task allocation problem:

$$\underset{x_{i_k,j_k}\,\forall i_k,\forall j_k,\forall k}{\text{maximize}} \sum_{k=1}^{T}\sum_{i_k=1}^{N_k}\sum_{j_k=1}^{M_k} C_{k,\mathcal{X}_{1:k-1}}\left[i_k,j_k\right] x_{i_k,j_k} \tag{3}$$

subject to

$$\sum_{i_k=1}^{N_k} x_{i_k,j_k} \leq 1 \qquad \forall j_k \in \{1,\ldots,M_k\}, \forall k \in \{1,\ldots,T\} \tag{4}$$

$$\sum_{j_k=1}^{M_k} x_{i_k,j_k} \leq 1 \qquad \forall i_k \in \{1,\ldots,N_k\}, \forall k \in \{1,\ldots,T\} \tag{5}$$

$$x_{i_k,j_k} \in \{0,1\} \qquad \forall i_k \in \{1,\ldots,N_k\}, \forall j_k \in \{1,\ldots,M_k\}, \forall k \in \{1,\ldots,T\} \tag{6}$$

- Eq. (5) represents that an agent can do only 1 task at a time step. (Is this need? Can an agent do all the tasks at that location? For example, both patrolling and checking out an intruder at the same location? In this case, we can collect all the tasks at the same location and call it a combined task. But capabilities of heterogeneous agents will come into play here when we do the combining.)
- Eq. (4) represents that a task can be done by only 1 agent at a time step.

III. Possible Simplifying Assumptions

A. Option 1: Assume $C_{k,\mathcal{X}_{1:k-1}} \rightarrow C_k$, i.e., remove the dependence on $\mathcal{X}_{1:k-1}$

This will reduce the problem to a MILP. In my opinion, this is a bad assumption because it removes the entire complexity of the problem we are trying to address:

- In the definition $C_{k,\mathcal{X}_{1:k-1}}\left[i_k,j_k\right]$ in Eq. (2), $\mathcal{X}_{1:k-1}$ ensures that the intrinsic value of doing the task increases if it hasn't been done in the previous time.
- $\mathcal{X}_{1:k-1}$ ensures that the effort incurred by agent i_k to do the task j_k depends on the current state of the agent. e.g., tasks which need the agent to travel large distances should cost more. This is necessary for the predictive part of the problem statement.

B. Option 2: Simplistic Assumption on Tasks

Let us assume that at the k^{th} time instant, M_k tasks are generated with some probability. The cost of doing a task depends on the intrinsic value of doing the task j_k (which does NOT depend on $\mathcal{X}_{1:k-1}$) and on the agents current location at the start of the k^{th} time instant.

All the agents try to do some the tasks at this time instant. The remaining undone tasks vanishes at the end of k^{th} time instant.

We feel this problem captures the core essence of the original problem. Here are the advantages of this assumption:

- We are ignoring deterministic tasks. We claim that they can be easily added back.
- We are ignoring recurrent tasks (like patrolling), whose intrinsic value depends on $\mathcal{X}_{1:k-1}$. Adding this back will be difficult, but not impossible.
- This problem can be solved using Approximate DP.

My main concern with Approximate DP is that we are losing the rigorousness of the DP. How do we show our solution is good?

IV. Task Assignment for Homogeneous Agents to Tasks at One Time Step

The state space is divided into $n_{\text{cell}} \in \mathbb{N}$ cells.

There are N agents at time step t_0.

The cost of an agent to move from cell i to cell j is given by $C[i, j]$, where $i, j \in \{1, \ldots, n_{\text{cell}}\}$. If an agent cannot transition from cell i to cell j, then $C[i, j] = \infty$.

There are M tasks at time step t_1.

The reward incurred by an agent in cell j is given by $R[j]$. If there is no task in cell j, then $R[j] = 0$.

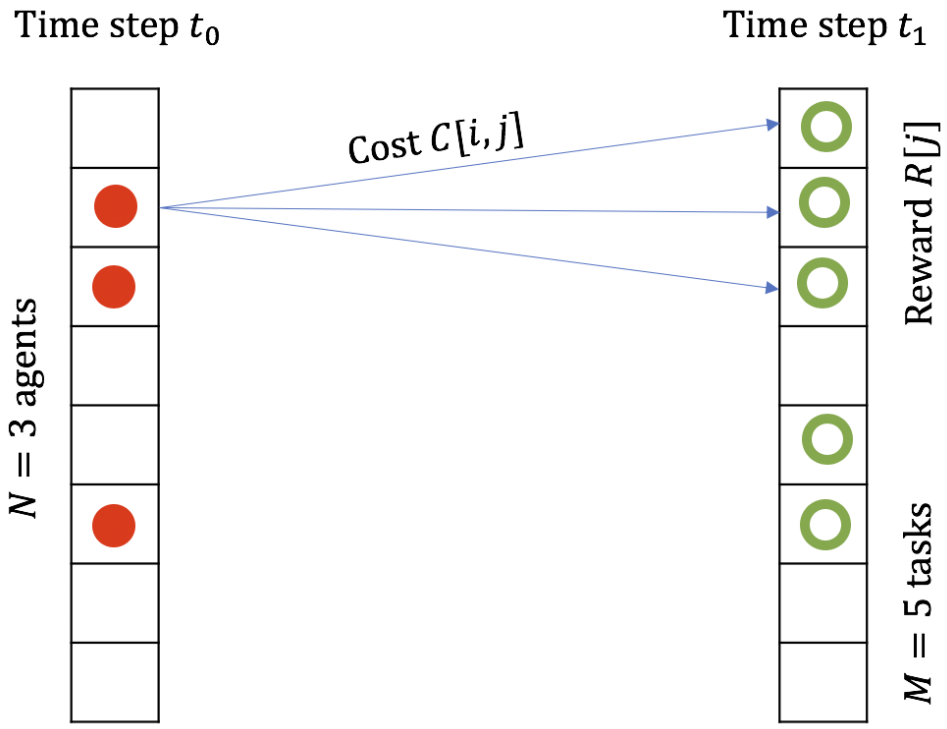

Figure 1. Simple Assignment Problem

Let $x[i, j]$ represent the indicator variable that an agent moves from cell i to cell j.

Let $A[i]$ represent the indicator variable that an agent is present in cell i at time step t_0.

Our objective is the maximize the (reward - cost) gained by the agents.

The assignment problem is given by the following ILP:

$$\underset{x[i,j],\, \forall i,j \in \{1,\ldots,n_{\text{cell}}\}}{\text{maximize}} \left(\sum_{j \in \{1,\ldots,n_{\text{cell}}\}} \left(R[j] \sum_{i \in \{1,\ldots,n_{\text{cell}}\}} x[i,j] \right) \right) - \left(\sum_{i \in \{1,\ldots,n_{\text{cell}}\}} \sum_{j \in \{1,\ldots,n_{\text{cell}}\}} C[i,j]\, x[i,j] \right) \tag{7}$$

subject to

$$x[i,j] \in \{0,1\} \qquad \forall i, j \in \{1, \ldots, n_{\text{cell}}\} \tag{8}$$

$$\sum_{i \in \{1,\ldots,n_{\text{cell}}\}} x[i,j] \leq 1 \qquad \forall j \in \{1, \ldots, n_{\text{cell}}\} \tag{9}$$

$$\sum_{j \in \{1,\ldots,n_{\text{cell}}\}} x[i,j] = A[i] \qquad \forall i \in \{1, \ldots, n_{\text{cell}}\} \tag{10}$$

Eq. (9) ensures that at most 1 agent can be in any cell.

Eq. (10) ensures that initial conditions are satisfied.

We could also state this problem as a LP:

$$\underset{x[i,j],\,\forall i,j\in\{1,\ldots,n_{\text{cell}}\}}{\text{maximize}} \left(\sum_{j\in\{1,\ldots,n_{\text{cell}}\}} \left(R[j] \sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x[i,j] \right) \right) - \left(\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} \sum_{j\in\{1,\ldots,n_{\text{cell}}\}} C[i,j]\, x[i,j] \right) \tag{11}$$

subject to

$$x[i,j] \geq 0 \qquad \forall i,j \in \{1,\ldots,n_{\text{cell}}\} \tag{12}$$

$$x[i,j] \leq 1 \qquad \forall i,j \in \{1,\ldots,n_{\text{cell}}\} \tag{13}$$

$$\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x[i,j] \leq 1 \qquad \forall j \in \{1,\ldots,n_{\text{cell}}\} \tag{14}$$

$$\sum_{j\in\{1,\ldots,n_{\text{cell}}\}} x[i,j] = A[i] \qquad \forall i \in \{1,\ldots,n_{\text{cell}}\} \tag{15}$$

Eq. (12) and (13) convert the integer constraint in Eq. (8) into linear constraints.

The solution of this LP is the same as the above ILP. This can be proved using Total Unimodularity.[1]

V. Task Assignment for Homogeneous Agents to Tasks in Multiple Time Steps

The state space is divided into $n_{\text{cell}} \in \mathbb{N}$ cells.

There are N agents at time step t_0.

Let t_F be the final time step for this lookahead policy. Hence the time steps are $\{t_0, t_1, \ldots, t_F\}$.

The cost of an agent to move from cell i in time step $t_{(k-1)}$ to cell j in time step t_k is given by $C_k[i,j]$, where $i,j \in \{1,\ldots,n_{\text{cell}}\}$ and $k \in \{1,\ldots,F\}$. If an agent cannot transition from cell i in time step $t_{(k-1)}$ to cell j in time step t_k, then $C_k[i,j] = \infty$.

There are M_k tasks at time step t_k, where $k \in \{1,\ldots,F\}$.

The reward incurred by an agent in cell j in time step t_k is given by $R_k[j]$. If there is no task in cell j in time step t_k, then $R_k[j] = 0$.

The reward incurred by an agent in cell j in time step t_F is given by $R_F[j]$, where $R_F[j]$ is an approximation of the future reward that the agent might get from that location.

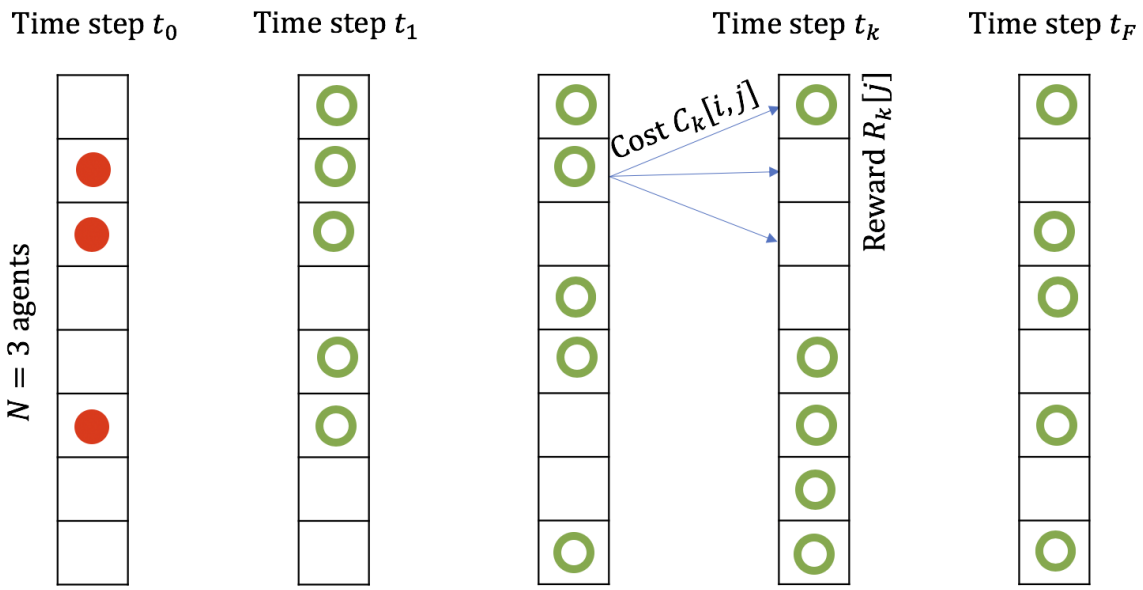

Figure 2. Multiple-Step Assignment Problem

Let $x_k[i,j]$ represent the indicator variable that an agent moves from cell i in time step $t_{(k-1)}$ to cell j in time step t_k.

[1]Totally unimodular matrices give a quick way to verify that a linear program is integral. https://en.wikipedia.org/wiki/Unimodular_matrix#Total_unimodularity

Let $A[i]$ represent the indicator variable that an agent is present in cell i at time step t_0.
Our objective is the maximize the (reward - cost) gained by the agents over multiple time steps.
The assignment problem is given by the following ILP:

$$\underset{\substack{x_k[i,j],\, \forall i,j \in \{1,\ldots,n_{\text{cell}}\},\\ \forall k \in \{1,\ldots,F\}}}{\text{maximize}} \quad \sum_{k\in\{1,\ldots F\}} \left(\left(\sum_{j\in\{1,\ldots,n_{\text{cell}}\}} \left(R_k[j] \sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k[i,j] \right) \right) - \left(\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} \sum_{j\in\{1,\ldots,n_{\text{cell}}\}} C_k[i,j]\, x_k[i,j] \right) \right) \tag{16}$$

subject to

$$x_k[i,j] \in \{0,1\} \qquad \forall i,j \in \{1,\ldots,n_{\text{cell}}\},\, \forall k \in \{1,\ldots,F\} \tag{17}$$

$$\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k[i,j] \leq 1 \qquad \forall j \in \{1,\ldots,n_{\text{cell}}\},\, \forall k \in \{1,\ldots,F\} \tag{18}$$

$$\sum_{j\in\{1,\ldots,n_{\text{cell}}\}} x_1[i,j] = A[i] \qquad \forall i \in \{1,\ldots,n_{\text{cell}}\} \tag{19}$$

$$\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k[i,j] = \sum_{i\in n_{\text{cell}}} x_{k+1}[i,j] \qquad \forall j \in \{1,\ldots,n_{\text{cell}}\},\, \forall k \in \{1,\ldots,F-1\} \tag{20}$$

Eq. (18) ensures that at most 1 agent can be in any cell at any time step.
Eq. (20) ensures that equal number of agents are entering and exiting a cell. This is used to conserve the number of agents.

We could also state this problem as a LP:

$$\underset{\substack{x_k[i,j],\, \forall i,j \in \{1,\ldots,n_{\text{cell}}\},\\ \forall k \in \{1,\ldots,F\}}}{\text{maximize}} \quad \sum_{k\in\{1,\ldots F\}} \left(\left(\sum_{j\in\{1,\ldots,n_{\text{cell}}\}} \left(R_k[j] \sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k[i,j] \right) \right) - \left(\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} \sum_{j\in\{1,\ldots,n_{\text{cell}}\}} C_k[i,j]\, x_k[i,j] \right) \right) \tag{21}$$

subject to

$$x_k[i,j] \geq 0 \qquad \forall i,j \in \{1,\ldots,n_{\text{cell}}\},\, \forall k \in \{1,\ldots,F\} \tag{22}$$

$$x_k[i,j] \leq 1 \qquad \forall i,j \in \{1,\ldots,n_{\text{cell}}\},\, \forall k \in \{1,\ldots,F\} \tag{23}$$

$$\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k[i,j] \leq 1 \qquad \forall j \in \{1,\ldots,n_{\text{cell}}\},\, \forall k \in \{1,\ldots,F\} \tag{24}$$

$$\sum_{j\in\{1,\ldots,n_{\text{cell}}\}} x_1[i,j] = A[i] \qquad \forall i \in \{1,\ldots,n_{\text{cell}}\} \tag{25}$$

$$\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k[i,j] = \sum_{i\in n_{\text{cell}}} x_{k+1}[i,j] \qquad \forall j \in \{1,\ldots,n_{\text{cell}}\},\, \forall k \in \{1,\ldots,F-1\} \tag{26}$$

Eq. (22) and (23) convert the integer constraint in Eq. (17) into linear constraints.
We think the solution of this LP is the same as the above ILP.
Note that this formulation already accepts predictive tasks and deterministic tasks.

VI. Task Assignment for Heterogeneous Agents to Tasks in Multiple Time Steps

The state space is divided into $n_{\text{cell}} \in \mathbb{N}$ cells.
There are N heterogeneous agents at time step t_0.
The agents are partitioned into Q types, based on their capabilities.
Let t_F be the final time step for this lookahead policy. Hence the time steps are $\{t_0, t_1, \ldots, t_F\}$.
The cost of an agent of type q to move from cell i in time step $t_{(k-1)}$ to cell j in time step t_k is given by $C_k^q[i,j]$, where $i,j \in \{1,\ldots,n_{\text{cell}}\}$ and $k \in \{1,\ldots,F\}$ and $q \in \{1,\ldots,Q\}$. If an agent of type q cannot transition from cell i in time step $t_{(k-1)}$ to cell j in time step t_k, then $C_k^q[i,j] = \infty$.

There are M_k tasks at time step t_k, where $k \in \{1, \ldots, F\}$.

The reward incurred by an agent of type q in cell j in time step t_k is given by $R_k^q[j]$. If there is no task in cell j in time step t_k, then $R_k^q[j] = 0$.

The reward incurred by an agent of type q in cell j in time step t_F is given by $R_F^q[j]$, where $R_F^q[j]$ is an approximation of the future reward that the agent of type q might get from that location.

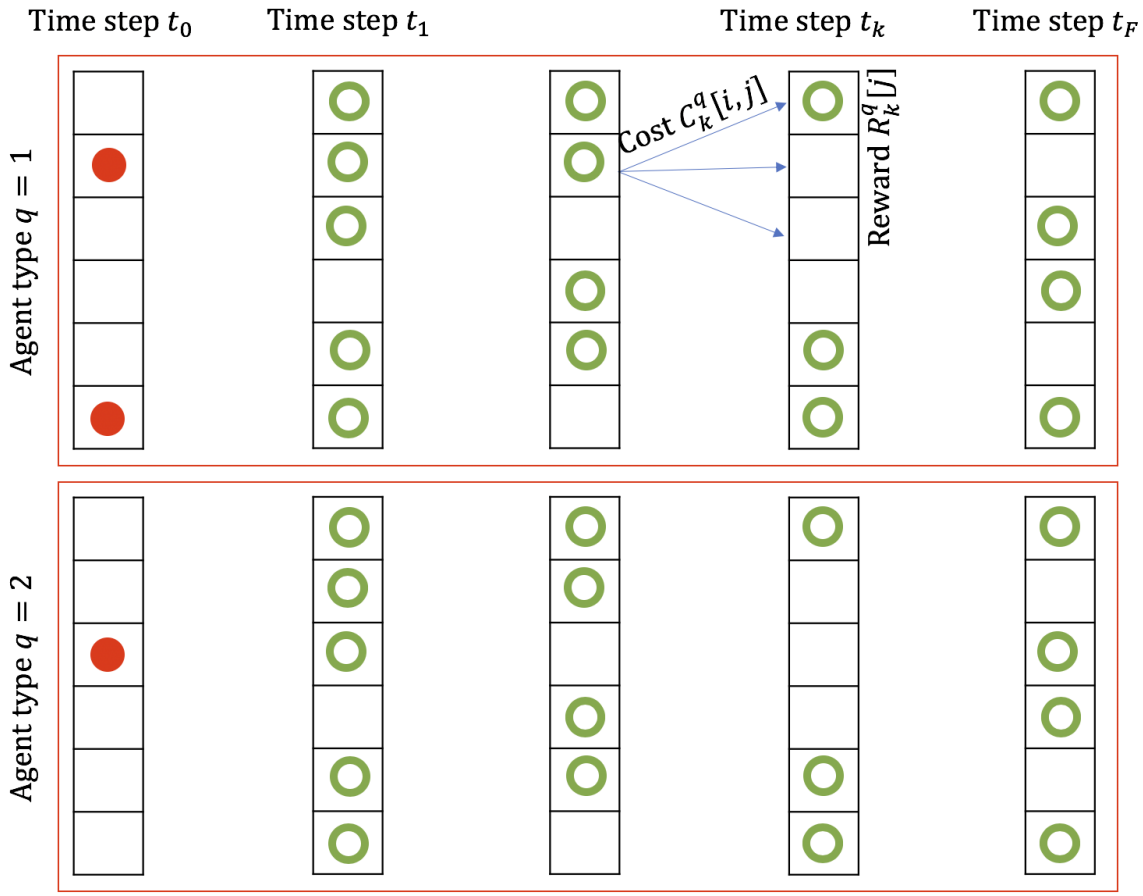

Figure 3. Heterogeneous Multiple-Step Assignment Problem

Let $x_k^q[i,j]$ represent the indicator variable that an agent of type q moves from cell i in time step $t_{(k-1)}$ to cell j in time step t_k.

Let $A^q[i]$ represent the indicator variable that an agent of type q is present in cell i at time step t_0.

Our objective is the maximize the (reward - cost) gained by the agents over multiple time steps.

The assignment problem is given by the following ILP:

$$\underset{\substack{x_k^q[i,j],\ \forall i,j \in \{1,\ldots,n_{\text{cell}}\},\\ \forall k \in \{1,\ldots,F\},\ q \in \{1,\ldots,Q\}}}{\text{maximize}} \quad \sum_{k\in\{1,\ldots F\}} \left(\left(\sum_{j\in\{1,\ldots,n_{\text{cell}}\}} \left(\sum_{q\in\{1,\ldots,Q\}} \left(R_k^q[j] \sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k^q[i,j] \right) \right) \right) - \left(\sum_{q\in\{1,\ldots,Q\}} \left(\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} \sum_{j\in\{1,\ldots,n_{\text{cell}}\}} C_k^q[i,j]\, x_k^q[i,j] \right) \right) \right) \tag{27}$$

subject to

$$x_k^q[i,j] \in \{0,1\} \qquad \forall i,j \in \{1,\ldots,n_{\text{cell}}\},\ \forall k \in \{1,\ldots,F\},\ q \in \{1,\ldots,Q\} \tag{28}$$

$$\sum_{q\in\{1,\ldots,Q\}} \sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k^q[i,j] \le 1 \qquad \forall j \in \{1,\ldots,n_{\text{cell}}\},\ \forall k \in \{1,\ldots,F\} \tag{29}$$

$$\sum_{j\in\{1,\ldots,n_{\text{cell}}\}} x_1^q[i,j] = A^q[i] \qquad \forall i \in \{1,\ldots,n_{\text{cell}}\},\ q \in \{1,\ldots,Q\} \tag{30}$$

$$\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k^q[i,j] = \sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_{k+1}^q[i,j] \qquad \forall j \in \{1,\ldots,n_{\text{cell}}\},\ \forall k \in \{1,\ldots,F-1\},\ q \in \{1,\ldots,Q\} \tag{31}$$

Eq. (29) ensures that at most 1 agent of any type can be in any cell at any time step.

Eq. (31) ensures that equal number of agents of a given type are entering and exiting a cell. This is used to conserve the number of agents.

We could also state this problem as a LP:

$$\underset{\substack{x_k^q[i,j],\ \forall i,j \in \{1,\ldots,n_{\text{cell}}\},\\ \forall k \in \{1,\ldots,F\},\ q \in \{1,\ldots,Q\}}}{\text{maximize}} \quad \sum_{k\in\{1,\ldots F\}} \left(\left(\sum_{j\in\{1,\ldots,n_{\text{cell}}\}} \left(\sum_{q\in\{1,\ldots,Q\}} \left(R_k^q[j] \sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k^q[i,j] \right) \right) \right) - \left(\sum_{q\in\{1,\ldots,Q\}} \left(\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} \sum_{j\in\{1,\ldots,n_{\text{cell}}\}} C_k^q[i,j]\, x_k^q[i,j] \right) \right) \right) \tag{32}$$

subject to

$$x_k^q[i,j] \ge 0 \qquad \forall i,j \in \{1,\ldots,n_{\text{cell}}\},\ \forall k \in \{1,\ldots,F\},\ q \in \{1,\ldots,Q\} \tag{33}$$

$$x_k^q[i,j] \le 1 \qquad \forall i,j \in \{1,\ldots,n_{\text{cell}}\},\ \forall k \in \{1,\ldots,F\},\ q \in \{1,\ldots,Q\} \tag{34}$$

$$\sum_{q\in\{1,\ldots,Q\}} \sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k^q[i,j] \le 1 \qquad \forall j \in \{1,\ldots,n_{\text{cell}}\},\ \forall k \in \{1,\ldots,F\} \tag{35}$$

$$\sum_{j\in\{1,\ldots,n_{\text{cell}}\}} x_1^q[i,j] = A^q[i] \qquad \forall i \in \{1,\ldots,n_{\text{cell}}\},\ q \in \{1,\ldots,Q\} \tag{36}$$

$$\sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_k^q[i,j] = \sum_{i\in\{1,\ldots,n_{\text{cell}}\}} x_{k+1}^q[i,j] \qquad \forall j \in \{1,\ldots,n_{\text{cell}}\},\ \forall k \in \{1,\ldots,F-1\},\ q \in \{1,\ldots,Q\} \tag{37}$$

Eq. (33) and (34) convert the integer constraint in Eq. (28) into linear constraints.

We DONOT think the solution of this LP is the same as the above ILP, because of the flow mixing in Eq. (31). But this fraction solution might be good enough since the tasks are also probabilistic.

Note that this formulation already accepts predictive tasks and deterministic tasks.