LEVEL 1: GOAL REGRESSION

Given a set of entities $e \in E$, preferences $R = \{r_1, r_2, r_3, \ldots, r_n\}$, and a value function over preferences $h(e, r) \in [0, 1]$, it is possible to produce an ideal match over the set of entities $q(e_i, e_j)$. To do this, a few functions need to be defined:

- properties, $P(e, r) \in \{0, 1\}$, denotes if an entity e has property r,
- value function $h(e, r) \in [0, 1]$ is the level of disappointment e will experience when matched with property r,
- directional quality of match $q(e_i, e_j)$, which expresses the disappointment e_i experiences when matched with e_j: $q(e_i, e_j) = \sum\limits_{\{r|P(e_j,r)=1\}} h(e_i, r)$,
- a match indicator function, which indicates that a match has been assigned: $m(e_i, e_j) = \begin{cases} 1, & \text{match assigned \& } b(e_i, e_j) = 0 \\ 0, & \text{not matched} \end{cases}$ $(m(e_i, e_j) = m(e_j, e_i))$,
- a value $b(e_i, e_j) \in \{0, 1\}$ expresses if an entity e_i or e_j is too busy to be matched.

Given these match characteristics and quality values, two matrices may be maintained:

$$Q = \begin{bmatrix} q(e_1, e_1) & & \cdots & & \\ & \ddots & & & \\ \vdots & & q(e_i, e_j) & & \\ & & & \ddots & \vdots \\ & & \cdots & & q(e_n, e_n) \end{bmatrix} \tag{1}$$

and

$$M = \begin{bmatrix} m(e_1, e_1) & & \cdots & & \\ & \ddots & & & \\ \vdots & & m(e_i, e_j) & & \\ & & & \ddots & \vdots \\ & & \cdots & & m(e_n, e_n) \end{bmatrix}. \tag{2}$$

Given a set E, then the cost of a match arrangement M is

$$C(Q, M) = QM^T \tag{3}$$

and the optimization goal is to assign matches to unmatched entities:

$$M_{t+1} \leftarrow \frac{\arg\max_M \sum_{m \in M} M\alpha(Q, M)}{QM^T\,(1 - \alpha(Q, M))} \tag{4}$$

where α is a scaling factor, $\alpha(Q, M) \in [0, 1]$, which can emphasize quantity of matches or quality of matches. Specifically:

$\alpha(Q, M) = 0 \ \rightarrow$ only the cost of matches is significant. Trivially, $\sum\limits_{m \in M} m| = \varnothing$ will always satisfy this condition.

$\alpha(Q, M) = 1 \rightarrow$ only the amount of assignments matter. Trivially, $\sum_{m\in M} m = \sum_{m\in M} 1$ satisfies this requirement.

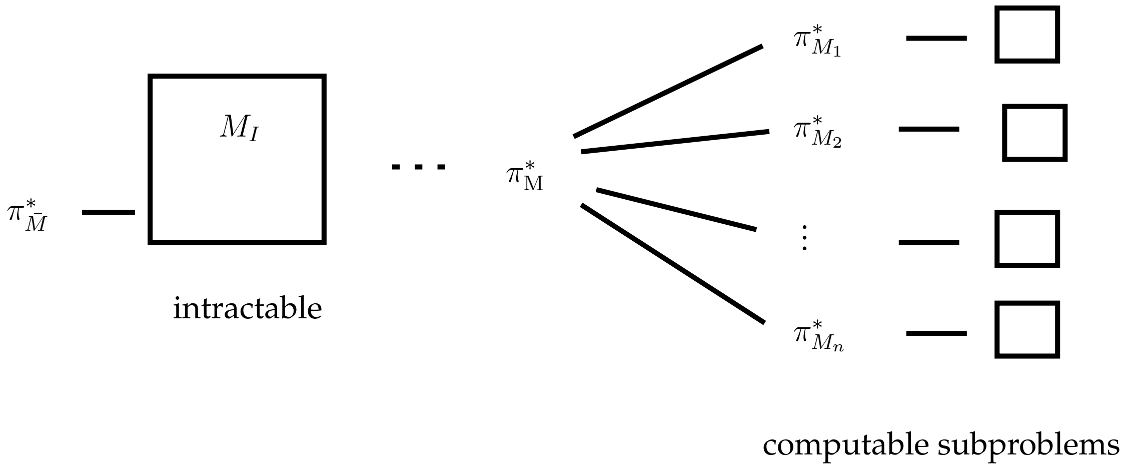
Thus, the ideal matching characteristics depend on an optimal value α. For algorithmic stability and quantity, low values of α should be attempted, which will prioritize quality over quantity.

Level 2: Hyperparameter optimization

Given a goal regression problem G_i, we wish to optimize both the quality of assignments C_{G_i}, and the number of matches, $\sum_{m\in M_{G_i}} m$. The problem G_i based around a geographical region and pertains to a fixed set of entities. In general for a goal problem G_i, we may consider the problem of optimization as an online stably stochastic process, which can be represented with a Markov Decision Process, $\bar{M}$:

$S_i = \{p_1, p_2, \ldots, p_n\}$ — a list of control parameters $p_x \in [0, 1]$
$A_i = \{a_1^+, a_1^=, a_1^-, a_2^+, a_2^=, a_2^-, \ldots, a_n^+, a_n^=, a_n^-\}$ — an action sequence which alters hyperparameters
$T = P(s'|a, s)$ — the likelihood of transiting between states given an action
$R(s) = \frac{QM^T}{\sum_{m\in M} m}$ — reward, specified as quality per match

This MDP, in general, can be seen as intractable for all but non-trivial problems. Various approaches are used to decompose this MDP into tractable subproblems as covered in [Concurrent Markov Decision Processes, 2,3]. The result of this Decomposition Process is a family of MDPs, $\mathrm{M} = \{M_1, M_2, \ldots, M_i, \ldots\}$ s.t. a gating function π^*_{M} representing the policy for the MDP family is interchangeable with the policy $\pi^*_{\bar{M}}$, $\pi^*_{\mathrm{M}} \sim \pi^*_{\bar{M}}$.



Level 3: Solution Combination

Given a set of Goal Regression Problems, and a set of online Hyperparameter Optimization Problems, the operation result is a set of Goal Regression Problems and associate Hyperparameters that are effective for the optimization of independent GR problems. In addition, over time, the characteristics will increase due to Hyperparameter Optimization.

This optimization can be run as an online process, and coded using Dynamic Programming Principles. The assumption is that each region G_i outputs a set of locally optimal matches m_i. The combination function $C_b(m_i, m_j)$ outputs a higher quality match matrix m_{ij}, such that the total cost is reduced, s.t.

$$m_{ij}(e_a, e_b) \leftarrow \arg\min \begin{matrix} m_i(a) & m_i(b) \\ m_j(a) & m_j(b) \end{matrix} \tag{5}$$

Given all matches $(x, y) \in M_i$ and all matches $(a, b) \in M_j$, it is possible that $a, b \in M_i$ and that $x, y \in M_j$. Thus, the combination of $M_i \cup M_j = M_{ij}$ may form a bi-partite graph, where each set (a, b) may have duplicate values for a or b.

In the most complex case, the graph is fully converti**(???)**. The goal of this optimization problem is to choose the best over all matches such that no entity receives a duplicate match.

Since the graph is bi-partite, it is possible to consider exactly two scenarios:

a) the case where odd vertices are selected;
b) the case where even vertices are selected.

$\therefore$ if limiting to these cases, the optimal matches are

$$\arg\min_{\text{INDICES}} \sum_{k \in \text{INDICES}} q(e_i, e_j) + q(e_j, e_i)\,, \qquad \text{s.t. INDICES} \in \{\text{ODD}, \text{EVEN}\} \tag{6}$$