layout

February 27, 2025

1 The Layout Problem

[1]:
```
import numpy as np
import random
from utils import *
from human_ai import MultiHumanAI
from model.mallows import Mallows
```

1.1 Known Ground-Truth

Suppose there are k types of humans, each with **heterogeneous** ground-truth rankings. Denote the ground-truth ranking of human i as π_h^i.

If the algorithm has full knowledge of $\{\pi_h^i\}$, it can ensure that each type of human benefits from the collaboration by adopting the following strategy:

- *Always presenting a fixed set of items to the humans. Specifically, the presented k items are the k top items according to humans' ground-truth rankings.*

In the following experiment,

- we consider `num_of_humans` types of humans, each of whom has a random ground-truth ranking.
- These ground-truth rankings are **known** to the algorithm.
- It takes the strategy by setting the k top items as its first k items of its ground-truth.

We can see **every human is beneficial from the collaboration**.

[2]:
```
m = 10
phi = 1
for num_of_humans in range(2, 6):
    D_hs = []
    for _ in range(num_of_humans):
        pi_h_star = list(range(1, m + 1))
        random.shuffle(pi_h_star)
        D_hs.append(Mallows(m, phi, pi_h_star))

    joint_system = MultiHumanAI(m, num_of_humans, D_hs, None)
    benefits = joint_system.interaction_with_known_distribution()
```

```
[INFO] Sythesizing a layout..
[INFO] benefits: [0.3593302714709977, 0.36627696535629906]
[INFO] Sythesizing a layout..
[INFO] benefits: [0.11385074163951325, 0.33785074163951323,
0.3298507416395132]
[INFO] Sythesizing a layout..
[INFO] benefits: [0.2908507416395133, 0.35685074163951325,
0.2638507416395133, 0.20185074163951322]
[INFO] Sythesizing a layout..
[INFO] benefits: [0.09885074163951324, 0.07785074163951322,
0.1858507416395132, 0.33985074163951323, 0.01985074163951328]
```

1.2 Unknown Ground-Truth

However, the ground-truth rankings may not always known in advance to the algorithm, especially in scenarios that protect user privacy.

To learn about humans' preference, algorithm usually adopt query-based learning to learn humans' preference. We suppose the humans are interacting with the algorithm in the following way:

- At time t, a human comes with a type-i human arriving with a probability of p_i.
- The algorithm presents a set of items S_t to that human. She selects her favourite one from the items (but she sometimes would make mistakes). The human will get a **postive** review if the item is perfect to her and a **negative** review otherwise.
- The algorithm updates S_t by always picking the items that human like the most

[3]:
```python
m = 10
phi = 1
for num_of_humans in range(2, 6):
    info(f"Number of humans {num_of_humans}")
    D_hs = []

    ## The probability of every type person arriving.
    ps = np.array([random.random() for _ in range(num_of_humans)])
    ps /= np.sum(ps)
    info("p_i: {}".format(ps))

    ## Generating ground-truth
    for _ in range(num_of_humans):
        pi_h_star = list(range(1, m + 1))
        random.shuffle(pi_h_star)
        D_hs.append(Mallows(m, phi, pi_h_star))

    ## 1000 interactions between the algorithm and these humans
    joint_system = MultiHumanAI(m, num_of_humans, D_hs, ps)
    joint_system.interaction_with_unknown_distribution(1000, 200)
```

```
[INFO] Number of humans 2
```

```
[INFO] p_i: [0.62851802 0.37148198]
[INFO] t: 0, benefits: [-0.6321492583604867, -0.6321492583604867]
[INFO] t: 200, benefits: [0.36627696535629906, -0.6321492583604867]
[INFO] t: 400, benefits: [0.36627696535629906, -0.6321492583604867]
[INFO] t: 600, benefits: [0.36627696535629906, -0.6321492583604867]
[INFO] t: 800, benefits: [0.3593302714709977, -0.6321492583604867]
[INFO] Number of humans 3
[INFO] p_i: [0.45437628 0.05007939 0.49554433]
[INFO] t: 0, benefits: [0.3348507416395132, -0.6321492583604867,
-0.6321492583604867]
[INFO] t: 200, benefits: [0.2908507416395133, -0.6321492583604867,
0.35785074163951325]
[INFO] t: 400, benefits: [0.2988507416395133, -0.6321492583604867,
0.35585074163951325]
[INFO] t: 600, benefits: [0.24885074163951326, 0.35085074163951324,
0.3288507416395132]
[INFO] t: 800, benefits: [0.2718507416395133, 0.35885074163951325,
0.3298507416395132]
[INFO] Number of humans 4
[INFO] p_i: [0.33539883 0.27490237 0.1368927  0.2528061 ]
[INFO] t: 0, benefits: [-0.6321492583604867, -0.6321492583604867,
-0.6321492583604867, -0.6321492583604867]
[INFO] t: 200, benefits: [0.1918507416395132, 0.1828507416395132,
0.35885074163951325, 0.35285074163951324]
[INFO] t: 400, benefits: [0.1788507416395133, 0.25985074163951327,
0.08985074163951323, 0.34585074163951324]
[INFO] t: 600, benefits: [0.20585074163951322, 0.2768507416395133,
0.3208507416395132, 0.33785074163951323]
[INFO] t: 800, benefits: [0.1868507416395132, 0.2808507416395133,
0.21885074163951324, 0.09085074163951323]
[INFO] Number of humans 5
[INFO] p_i: [0.3467792  0.34676603 0.01986551 0.14630446 0.14028479]
[INFO] t: 0, benefits: [0.20385074163951322, 0.06185074163951321,
0.2808507416395133, 0.3098507416395132, 0.1828507416395132]
[INFO] t: 200, benefits: [0.34685074163951324, 0.09485074163951324,
-0.6321492583604867, 0.33885074163951323, 0.1748507416395133]
[INFO] t: 400, benefits: [0.34085074163951323, 0.11685074163951326,
-0.6321492583604867, 0.33785074163951323, 0.1768507416395133]
[INFO] t: 600, benefits: [0.20385074163951322, 0.08985074163951323,
0.25185074163951326, 0.3058507416395132, 0.21185074163951323]
```

```
[INFO] t: 800, benefits: [0.34185074163951323, 0.07685074163951322,
-0.6321492583604867, 0.35285074163951324, 0.1558507416395133]
```

1.3 Limitation

However, the above query-based learning algorithm still has limitation. If a p_i is very small, then algorithm has a very low probability of meeting a type-i human. Thus, the algorithm cannot learn the top item of the i-th human very well.

For example, in the above experiment with 5 agents. The third human has a probability of 0.0198 to appear, while is relatively low compared to other humans. As a result, her utility is still negative (-0.632) at the end of interaction. So it is still possible that that human **only gets hurt** from the collaboration.

This problem also has widespread manifestations in real-world scenarios. For example, for elderly users who do not frequently use smartphones, how can AI-algorithm ensure that their preferences are met?