Option swapping model (proposed)

For a two-option (i.e., a and b) item, the maximum combination of order is two, present it as an array, should be [0, 1], and then possible values inside the array could be [a, b] and [b, a].

Table 1. possible option order combinations of a two-option item.

[0]	[1]
a	b
b	a

For a three-option (i.e., a, b, and c) item, the maximum combination of order increases from 2 to 6:

[a, b, c] [a, c, b]

[b, a, c] [b, c, a]

[c, a, b] [c, b, a]

Or can be presented it in an array table below:

Table 2. possible option order combinations of a three-option item.

[0]	[1]	[2]
a	b	c
a	c	b
b	a	c
b	c	a
c	a	b
c	b	a

However, for a practical perspective, we do not need to include all the possibilities above, instead we may consider adopting an alternative route to make a significant only swapping, for example, to let the first option (a) goes through each of the positions:

[a, b, c] [b, a, c] [b, c, a]

The above example is a possible solution for making first option (a) a significant move/swap, but how about the second and third options?

For tackling this issue, let us rearrange the combinations as below:

Table 3. rearranged order combinations of a three-option item.

[0]	[1]	[2]
a	b	c
b	a	b
c	c	a

Although table 3 is a little bit different from the example before, it shares the same feature that the first option (a) goes through each of the positions (3 in total) in the array above.

For addressing the issue, take second (b) and third (c) options into consideration, we can see each of them goes through two out of three (positions) only, then we add one more storage column [3] and make sure that b and c do go through all three positions (see table 4 below).

Table 4. proposed model works for three-option item.

[0]	[1]	[2]	[3]
a	b	c	a
b	a	b	c
c	c	a	b

For [1-3], a significant change will be observed while performing a swap merely once (75% or twice 93.75%).

The code logic (i.e., typescript) could be written accordingly as below:

```
swap=1? b: (swap=2? c: a)

swap=1? a: (swap=3? c: b)

swap=2? a: (swap=3? b: c)
```

To be continued…

Option swapping model (Cont.)

For a four-option (i.e., a, b, c, and d) item, the maximum combination of order increases dramatically from 6 to 24:

[a, b, c, d] [b, a, c, d] [c, a, b, d] [d, a, b, c] [a, b, d, c] [b, a, d, c] [c, a, d, b] [d, a, c, b]

[a, c, b, d] [b, c, a, d] [c, b, a, d] [d, b, a, c] [a, c, d, b] [b, c, d, a] [c, b, d, a] [d, b, c, a]

[a, d, b, c] [b, d, a, c] [c, d, a, b] [d, c, a, b] [a, d, c, b] [b, d, c, a] [c, d, b, a] [d, c, b, a]

In that case, include all the possibilities may not be a worthwhile deal, with the proposed model above, we take the advantage of both significance and efficiency, though the memory or hardware issue may not be a problem today, as it shortens the code and enhances readability to same extent.

Table 5. proposed model works for four-option item.

[0]	[4]	[5]	[6]
a	b	c	d
b	a	d	c
c	d	a	b
d	c	b	a

For [4-6], a significant change will be observed while performing a swap merely once (75% or twice 93.75%), similar to table 4 above but not exactly the same since table 5 utilizes [0] as well.

The code logic (i.e., typescript) could be written accordingly as below:

```
swap=4? b: (swap=5? c:(swap=6? d: a))

swap=4? a: (swap=5? d:(swap=6? c: b))

swap=4? d: (swap=5? a:(swap=6? b: c))

swap=4? c: (swap=5? b:(swap=6? a: d))
```

The abovementioned model is actually not that simple while you induce some conditionals (i.e., if an 'all of the above' option appears, how to handle it? etc.), we will look into details in the coming episode(s).

Stay Tuned!