Allocation of slots in position auctions

Value of an agent i = $P_{a_i}(\hat{e}_i \cdot \theta_i) = v_i(a, \theta_i)$

where $a = (a_1, \ldots, a_n)$ is the allocation, a_i is the slot allocated to i.

Pick allocations $a^* \in \underset{a \in A}{\operatorname{argmax}} \sum_{i \in N} v_i(a, \theta_i)$ efficient allocation

Claim: An allocation of slots is efficient iff it is rank-by-expected revenue mechanism.

Proof sketch: maximizing the weighted sum problem. Sum is maximized when maximum weight is put on maximum value.

The slot allocation problem is a sorting problem - hence computationally tractable.

Allocation decision is done, need payments to make it DSIC.

natural candidate: VCG [used in Facebook]

Note: actual implementation in practice might be different. Here we discuss only an abstract notion of how it can be done.

VCG in position auction

Given bids $(b_1, \ldots, b_n)$ [note, $\hat{\theta}_i$: reported type and b_i are same]

WLOG ordered such that $\hat{e}_1 b_1 \geqslant \hat{e}_2 b_2 \geqslant \cdots \geqslant \hat{e}_n b_n$

- allocation a^* is s.t. $a_i^* = i$.

• define $\underline{a}_{-i}^* \in \arg\max_{a \in A} \sum_{j \neq i} v_j(a, \theta_j)$

note: allocations of the agents after i, i.e., $i+1$ to n get one slot better.

hence $p_i^{VCG}(b) = \sum_{j \neq i} v_j(\underline{a}_{-i}^*, \theta_j) - \sum_{j \neq i} v_j(a^*, \theta_j)$

$$= \sum_{j=i}^{n-1} p_j \left(\hat{e}_{j+1}\, b_{j+1}\right) - \sum_{j=i}^{n-1} p_{j+1} \left(\hat{e}_{j+1}\, b_{j+1}\right)$$

$$= \sum_{j=i}^{n-1} (p_j - p_{j+1}) \left(\hat{e}_{j+1}\, b_{j+1}\right), \quad \forall\, i = 1, \dots, n-1$$

$p_n^{VCG}(b) = 0.$

This is the total expected payment. To convert this to the pay-per-click: $\frac{1}{p_i\, \hat{e}_i}\, p_i^{VCG}(b).$