Auction Matching Algorithm

Anna Bachstein and Benjamin Case

Clemson University

Algorithm

Input Weighted bipartite graph with non negative integer weights.
Output Maximum weight matching such that no bidder and no item appear more than once.

Algorithm

Initialization: For each good j set $p_j \leftarrow 0$ and $\text{owner}_j \leftarrow \text{null}$
Set queue Q for all bidders i
Fix $\delta = \dfrac{1}{n_g + 1}$ where n_g is the number of goods
algorithm: While Q is not empty do:
$i \leftarrow Q.deque()$
Find j that maximizes $w_{ij} - p_j$
If $w_{ij} - p_j \geq 0$ then:
    Enque current $owner_j$ into Q
    $\text{owner}_j \leftarrow i$
    $p_j \leftarrow p_j + \delta$
end

Correctness

Defintion We say the bidder i is δ-happy if one of the following is true

1. For some good j, $\text{owner}_j = i$ and for all good j' we have $\delta + w_{ij} - p_j \geq w_{ij} - p_{j'}$
2. For no good j does it hold that $\text{owner}_j = i$ and for all goods j we have $w_{ij} \leq p_j$

Correctness

- The key loop invariant is that all bidders, except that are in Q are δ happy. This is true since Q is initialized to all bidders.
- For the bidder i dequeued in an iteration, the loop exactly chooses the j that makes them happy, if one exists, and the δ-error is due to the final increase in p_j.
- In other words any increase in p_j for j that is not owned by by i' does not hurt the inequality while an increase for the j that was owned by i' immediately enqueues i'

Lemma

Lemma: If all bidders are δ-happy then every matching M' we have that $n\delta + \sum_{i=owner_j} w_{ij} \geq \sum_{(i,j)\in M'} w_{ij}$

- Fix bidder i, let j denote the good that they got from the algorithm and j' be what he got from the matching M'
- Since i is happy we have that $\delta + w_{ij} - p_j \geq w_{ij'} - p_{j'}$ (case 2 in definition). Thus over all i we get $\sum_{i=owner_j}(\delta + w_{ij} - p_j) \geq \sum_{(ij')\in M'}(w_{ij'} - p_{j'})$

Lemma

- If some j does not appear of the left-hand side then it was never picked by the algorithm so $p_j = 0$.
- So when we subtract $\sum_j p_j$ from both sides, the left sides becomes the lefthand side of the inequality in the lemma and same with right-hand side.

Running Time

- For each main loop execution: either p_j is increased or the number of bidders in Q is decreased by one (and never increased)
- No p_j can increase above $C = \max_{i,j} w_{i,j}$
- Thus total number of main loop executions is $Cn/\delta = O(Cn^2)$
- Each loop takes $O(n)$ time (trivially)
- Thus $O(Cn^3)$, which matches best known on dense graphs. For sparse graphs, heaps can be used to keep track of the max and achieve a speedup

Our timing

- nodes - 12, edges - 14, max cost - 20
  3.95 ms $\pm$ 734 μs per loop (mean $\pm$ std. dev. of 7 runs, 100 loops each)
- (asymmetric) nodes - 12, edges - 14, max cost - 20, source - 4, sink - 6
  2.06 ms $\pm$ 402 μs per loop (mean $\pm$ std. dev. of 7 runs, 1000 loops each)
- nodes - 200, edges - 1400, max cost - 10000
  5min 19s 3.66 $\pm$ μs per loop (mean $\pm$ std. dev. of 7 runs, 1 loop each)

References

- Demange, G., Gale, D., and Sotomayor, M. (1986). Multi-Item Auctions. Journal of Political Economy, 94(4), 863-872.
- *Auction Algorithm for Bipartite Matching*, 07.13.2009 , Word Press