Fair Virtual Conference Scheduling

Vaibhav Saxena Roll: 20CS60R57
Ravi Pratap Singh Roll: 20CS60R60
Sahil Jain Roll: 20CS60R64
Prithwish Jana Roll: 20CS60R66

{saxenavaibhav, ravips, sahil1206, jprithwish}@kgpian.iitkgp.ac.in

Under mentorship of: **Gourab Kumar Patro**
Under guidance of: **Prof. Bivas Mitra**

Department of Computer Science & Engineering,
IIT Kharagpur

26-Feb-2021

PROBLEM DESCRIPTION

- Pandemic-induced restrictions on travel and social gatherings
  - Conferences going virtual
- Report a Schedule such that:

Challenges Involved

  - People from around the globe can attend (as per **Participant-Availability**)
  - Maximum # of talks can be attended (as per **Participant-Interests**)
  - It is **Fair** to both the participants & speakers
- Data at Hand:

AVLBTY	Slot-1	Slot-2	...	Slot-N
Prcpnt-1	0.8	0.0	...	0.4
Prcpnt-2	0.4	0.6	...	0.9
...	...	...	...	...
Prcpnt-P	0.7	1.0	...	0.3

How much AVAILABLE is the Pth Participant in the Nth slot ?

INTEREST	Talk-1	Talk-2	...	Talk-M
Prcpnt-1	0.5	0.3	...	0.3
Prcpnt-2	0.9	0.7	...	0.3
...	...	...	...	...
Prcpnt-P	0.4	1.0	...	0.5

How much INTERESTED is the Pth Participant to attend Mth talk ?

[1] Image source: https://www.colourbox.com/vector/online-tutor-concept-vector-illustration-vector-40140428

Related Work & Our Contributions

- Different from Job and Network Scheduling [2] (one stakeholder) and Event Scheduling [3] (binary availability, non-attendee's interest not considered)
- Fair Virtual Conference Scheduling [4]

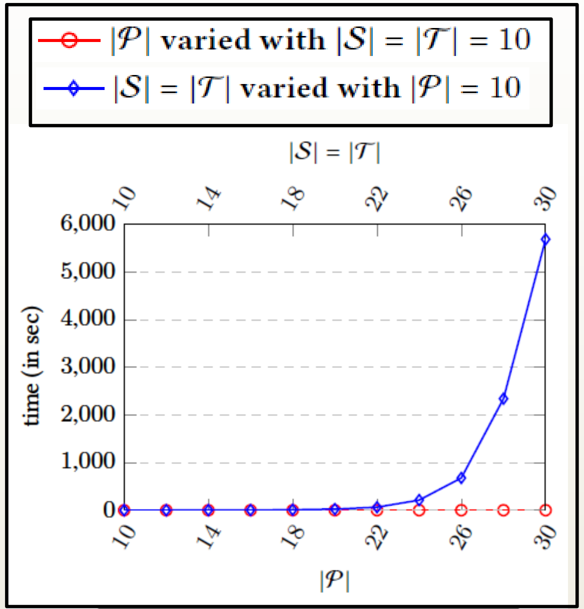
$$\underset{X}{\operatorname{argmax}} \quad \underbrace{\frac{1}{mn}\sum_{t\in\mathcal{T}}\sum_{p\in\mathcal{P}}\sum_{s\in\mathcal{S}} V_p(t)\cdot A_p(s)\cdot X_{t,s}}_{\text{Maximizing Total Expected Participation}} + \underbrace{\lambda_1\left[\min_{p_j\in\mathcal{P}}\sum_{t\in\mathcal{T}}\sum_{s\in\mathcal{S}}\frac{V_{p_j}(t)\cdot A_{p_j}(s)}{ICG(p_j)}\cdot X_{t,s} - \max_{p_i\in\mathcal{P}}\sum_{t\in\mathcal{T}}\sum_{s\in\mathcal{S}}\frac{V_{p_i}(t)\cdot A_{p_i}(s)}{ICG(p_i)}\cdot X_{t,s}\right]}_{\text{Minimizing Participant Unfairness}} + \underbrace{\lambda_2\left[\min_{t_j\in\mathcal{T}}\sum_{p\in\mathcal{P}}\sum_{s\in\mathcal{S}}\frac{V_p(t_j)\cdot A_p(s)}{IEC(t_j)}\cdot X_{t_j,s} - \max_{t_i\in\mathcal{T}}\sum_{p\in\mathcal{P}}\sum_{s\in\mathcal{S}}\frac{V_p(t_i)\cdot A_p(s)}{IEC(t_i)}\cdot X_{t_i,s}\right]}_{\text{Minimizing Speaker Unfairness}}$$

s.t.

$$X_{t,s}\in\{0,1\}\ \forall t\in\mathcal{T}, s\in\mathcal{S}$$

$$\sum_{s\in\mathcal{S}} X_{t,s} = 1,\ \forall t\in\mathcal{T}$$

$$\sum_{t\in\mathcal{T}} X_{t,s} \le 1,\ \forall s\in\mathcal{S}$$



Computation Time

- Too slow on direct conversion to ILP → 26 Talks, 48 Slots, 32 Partcpnts → 30+ Hours!
- With LP-Repeated-Rounding heuristic, we explore various graph clustering techniques:
  - K-Spanning Tree, Shared Nearest Neighbours, Betweenness Centrality, Maximal Clique Enumeration

Participant Profile Clustering

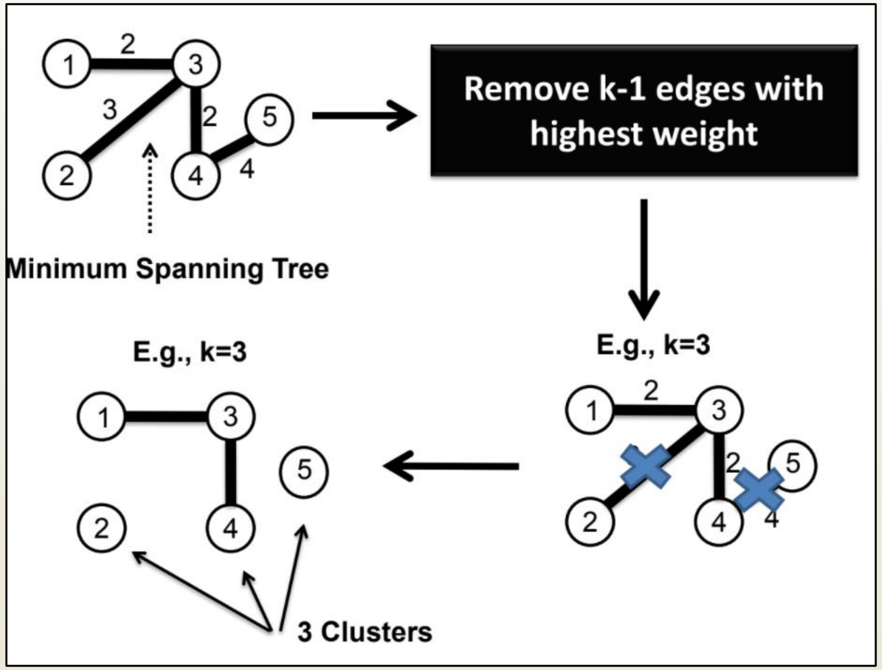
[2] J.P. Lozi, et al. (2016). "The Linux Scheduler: A Decade of Wasted Cores." *Proceedings of the Eleventh European Conference on Computer Systems*.
[3] H. Lee and A. Goel (2016). Probabilistic Matrix Inspection and Group Scheduling. In *IJCAI* (pp. 322-328).
[4] G.K. Patro, A. Chakraborty, N. Ganguly and K.P. Gummadi (2020). On Fair Virtual Conference Scheduling: Achieving Equitable Participant and Speaker Satisfaction. arXiv preprint arXiv:2010.14624.

Profile Clustering: Using K-Spanning Tree

- **Minimum Spanning Tree** : Spanning tree of a graph with minimum possible sum of edge-weights (edge weights ≡ distance)
- **Clustering (an application of MST)** : Given an undirected graph and distance between each node pair d(u,v). Here, d(u,v) can be actual distance or dissimilarity between nodes.
- **Goal :** Divide n nodes into k groups so that, the minimum distance/dissimilarity between items in different groups is maximized

ALGORITHM:

1. Construct Dissimilarity matrix using the Interest and Availability matrix of the users, where similarity measure is Jaccard Similarity.
2. From the above matrix, construct a graph G.
3. Find MST of graph G.
4. Now, to get **k** clusters, remove (**k-1)** most dissimilar edges from MST
5. Return the reduced Interest and Availability Matrices.



[5] Image Courtesy: Lecture Notes of Prof. B. S. Panda, Dept. of Maths, IIT Delhi. https://web.iitd.ac.in/~bspanda/graphclustering.pdf

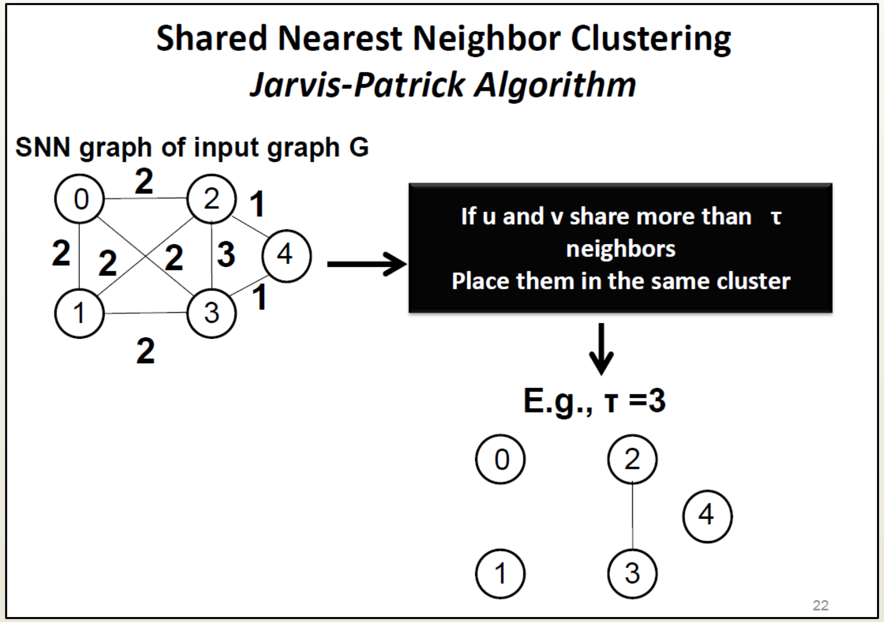
Profile Clustering: Using Shared Nearest Neighbors

- Vertices **(Sparsified similarity graph (G))** :
  - V = {P_i } : { P_i } = { Availability$_i$ *concat.* Interest$_i$ }
- Edges **(Sparsified similarity graph (G))** :
  - Wt = similarity(vi,vj), if similarity(vi,vj) ≥ threshold

    = ∞ , otherwise

ALGORITHM:

1. Obtain the count of shared neighbours among the set of all pairs of connected participants (P_i) in G using union find.
2. Replace the edge weights in G by the count of shared neighbours among vertices of each edge (v_i,v_j).
3. If vertices u and v share more than τ neighbors, place them in the same cluster.

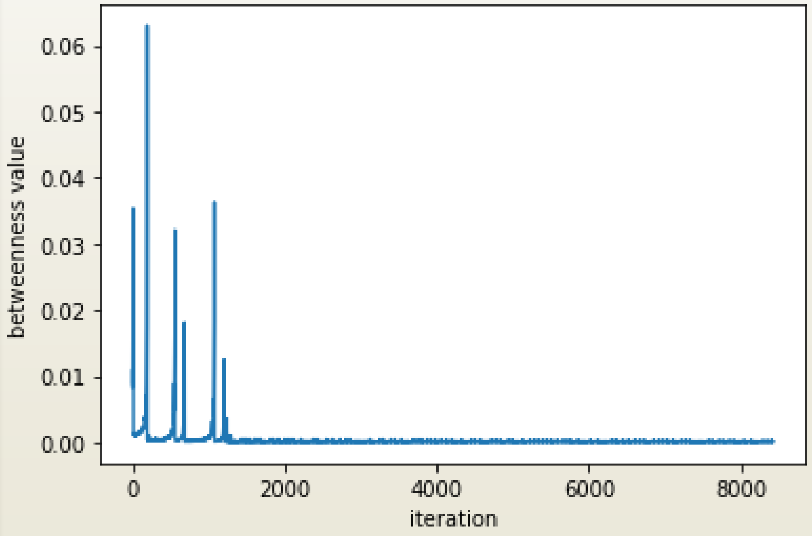


[5] Image Courtesy: Lecture Notes of Prof. B. S. Panda, Dept. of Maths, IIT Delhi. https://web.iitd.ac.in/~bspanda/graphclustering.pdf

Profile Clustering: Using Betweenness Centrality

- The betweenness centrality [6] of a node v is given by the expression: $c_B(e) = \sum_{s,t \in V} \frac{\sigma(s,t|e)}{\sigma(s,t)}$
- Here,
  - V = set of nodes
  - $\sigma(s,t)$ = no. of shortest (s,t)-paths
  - $\sigma(s,t \mid e)$ = number of those paths passing through edge e
- Vertices and Edges with High Betweenness form good starting points to identify clusters

```
runGirvanNewman() is
    while True:
        init_ncomp = number_connected_components(G)
        ncomp = init_ncomp
        while ncomp <= init_ncomp:
            bw = edge_betweenness_centrality(G)
            max_edge = max(bw)
            G.remove_edge(max_edge)
            ncomp = number_connected_components(G)
        if ncomp == cut_off
            break
```

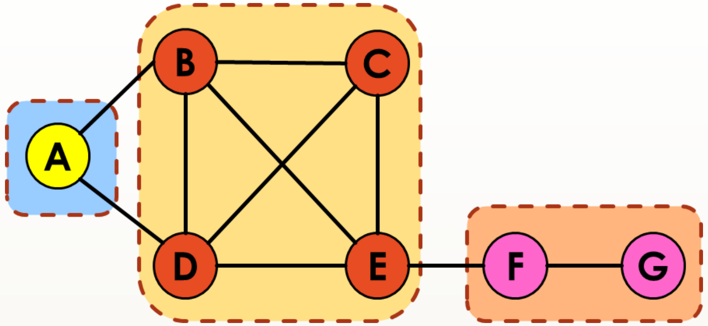


[6] U. Brandes. On Variants of Shortest-Path Betweenness Centrality and their Generic Computation. Social Networks 30(2):136-145, 2008. http://www.inf.uni-konstanz.de/algo/publications/b-vspbc-08.pdf

Profile Clustering: Using Maximal Clique Enumeration

- Undirected Graph
  - $V = \{ v_i \}$: $\{ P_i \} = \{ \text{Availability}_i \textit{ concat. } \text{Interest}_i \}$
  - $E = \{ (v_i, v_j) \}$: $Wt = \begin{cases} sim(v_i, v_j), \text{ if } sim(v_i, v_j) \geq \text{threshold} \\ \infty \text{, otherwise} \end{cases}$

Sparsification



- Using Bron and Kerbosch Algorithm with pivoting [5]:

```
BronKerbosch(R, P, X) is
    if P and X are both empty:
        report R as a maximal clique
    choose pivot vertex u in P ∪ X, having max |N(u) ∩ P |
    for each vertex v in P \ N(u) do
        BronKerbosch(R ∪ {v}, P ∩ N(v), X ∩ N(v))
        P := P \ {v}
        X := X ∪ {v}
```

R : vertices already chosen for MaxClique
P : vertices that can be selected next
X : vertices that cannot be selected next

Start with:
(R = Ø, P = V, X = Ø)

- Choose max-sized maximal clique recursively
  - Terminate when all vertices chosen

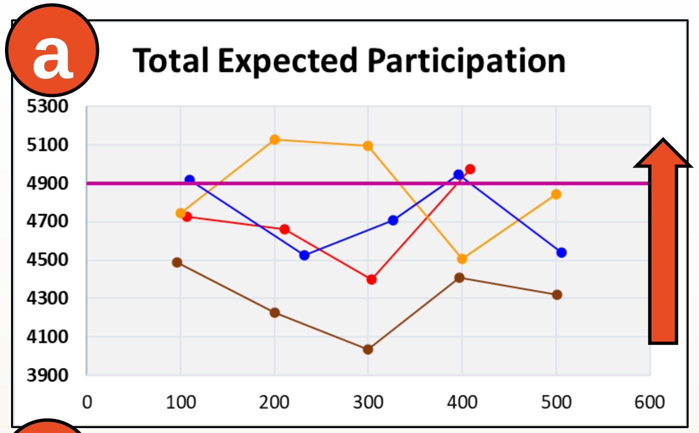
[7] E. Tomita, A. Tanaka, & H. Takahashi. (2006). The Worst-case Time Complexity for generating all Maximal Cliques and Computational Experiments. *Theoretical Computer Science*, 363(1), 28-42.

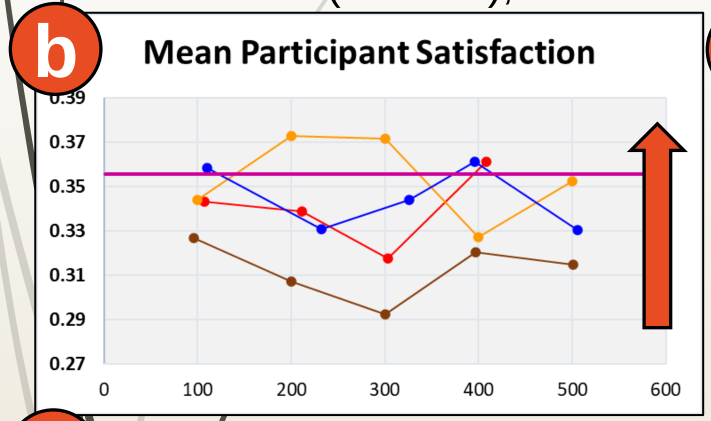
EVALUATION METRICS & RESULTS

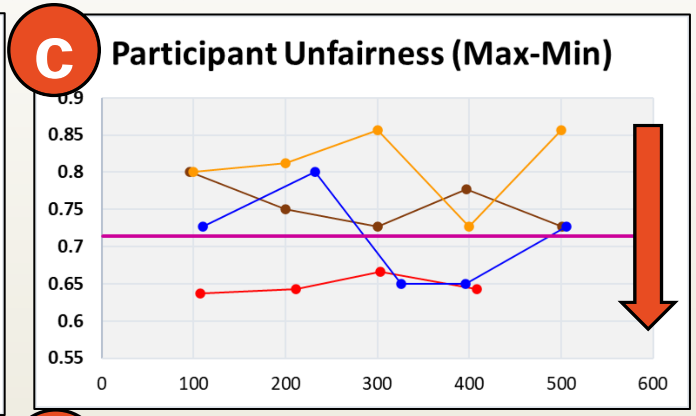
- 7 Evaluation Metrics:
  - (a) Social Welfare
  - (b-d) Participant-side
  - (e-g) Speaker-side
- Tested on RecSys-2017 [8] data: 26 Talks, 48 Slots, 1112 Partcp.
- X-axis: #(clusters), Y-axis: metric value

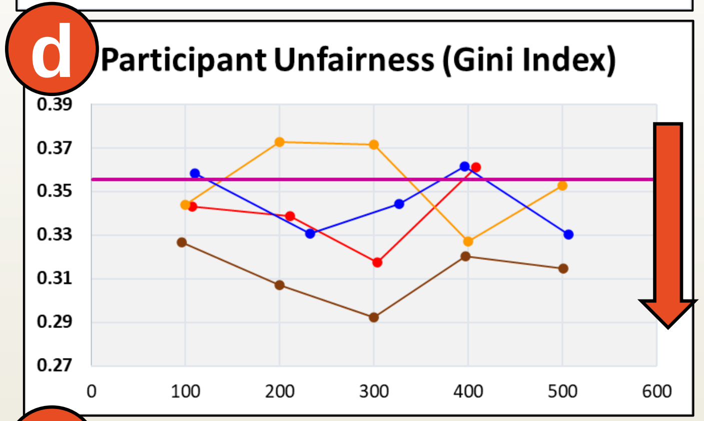
= Higher the better

= Lower the better

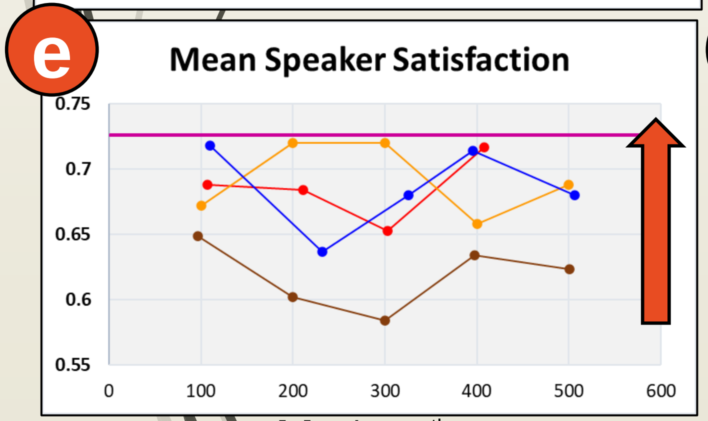


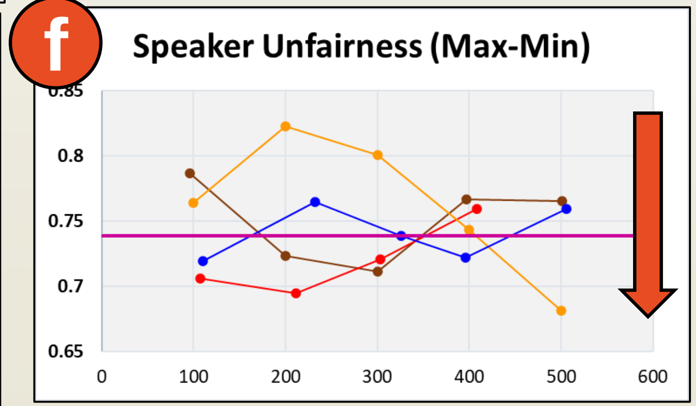


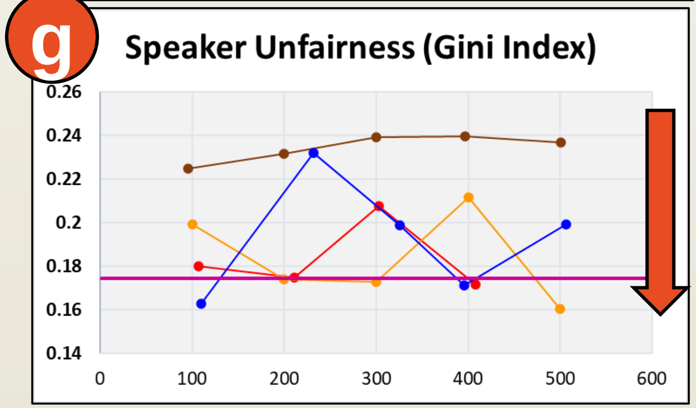




Participant-side Metrics







Speaker-side Metrics

[8] The 11th ACM Recommender Systems Conference (RecSys-2017), Como, Italy. https://recsys.acm.org/recsys17/

Epilogue & Further Plans towards Completion

- Similarity metrics (Jaccard coeff., Cosine, tailored, etc.) affect cluster and scheduling performance?

Feature	Slot-1	Slot-2	Slot-3	Slot-4	…	Slot-N	Talk-1	Talk-2	Talk-3	Talk-4	…	Talk-M
Prcpnt-1												
Prcpnt-2												

Jaccard Coeff. if substantial slots match in a 24-hour time-span, else 0 + Jaccard Coeff. for interest score for various research-tracks, taken together

- Try for even larger conferences like ICML (2200+ attendees), CVPR (9500+ attendees), etc.
- Compare how different similarity metrics and different thresholds (for sparsification) affect the same method's results.
- Try other clustering methods
- Think if any other LP-heuristics can improve overall computation time.

Thank You!