INTERVIEW SCHEDULING

Eshita Sharma
IIIT Bangalore
eshita.sharma@iiitb.org

Lubaina Machinewala
IIIT Bangalore
lubaina.machinewala@iiitb.org

Keshav Gupta
IIIT Bangalore
keshav.gupta@iiitb.org

Sujit Kumar Chakrabarti
IIIT Bangalore
sujitkc@iiitb.ac.in

CONTEXT

We present a method for automatically creating interview panels and scheduling them for research interviews.
The general selection process which is followed in most institutes for students in the research programmes involves inviting applications from candidates, shortlisting through application review, written test followed by interviews and final shortlisting.
The applicants have to choose the primary research domain under which they wish to apply. For each (research) domain, there is a domain representative (a faculty member listed under the domain) who does an initial screening of all the applications in his/her domain. A candidate can get shortlisted for final interview only if there is at least one faculty member who has reviewed his/her application and has shown his/her interest in supervising the candidate should he/she be finally selected. We see the following disadvantages of this process:

1. Labour Intensive
Domain representative's initial task is very labour intensive. He/she has to go through all the applications in his/her domain – which may run into several hundred – to assign reviewers. This must be done really quickly. Therefore, there are chances of error (applications going to irrelevant reviewers, and/or missing relevant reviewers).

2. Exclusive Domains
The applicant must choose a single primary domain. Though there is a facility to choose a secondary domain, this is mostly a choice made to the exclusion of all other choices. This is very restrictive. Quite often, the alignment of a candidate's research interests is not well aligned within the boundaries of a domain. There are many candidates whose interests cut across the boundaries of predesignated research domains. In such cases, an application may miss being noticed by a faculty member with matching research interest because of domains not matching.

For example, an applicant may be interested in working on applying formal methods for verifying machine learning algorithms. Which domain should he/she apply to – Data Science(DS), Computer Science(CS) or System Engineering(SE)? We believe that we should not force applicants to commit exclusively to one domain. Instead, they should be asked to mention their interests as a set of keywords. The review panel for the applicant and the panel of interviewers in case the applicant is shortlisted should then be composed based on keyword matching between the interests of the candidate and those of pertinent faculty members.

IMPLEMENTATION

The basic idea of our approach is simply to match applications with faculty members based on the keywords. In simple words, suppose that the candidate C's application has 3 matching topics with professor P1 and 2 with professor P2. All other things remaining the same, the application should be reviewed by P1 and not P2 if only one of them has to be chosen as the reviewer. This is the basic idea of our approach.

When applying this idea on a real data set, we encounter a number of other challenges:

1. Common keywords
There are a number of keywords which appear in a large number of applications, e.g. Machine Learning, Artificial Intelligence, etc. Applicants mention them not so much because they are interested in these topics, but because these are trending topics of the day.

2.Overloaded faculty members
For similar reasons, there are faculty members who would appear in an inordinately large number of review panels, simply because they too are interested in such trending topics as machine learning. Therefore, the automated reviewer assignment process must associate less importance to common keywords as compared to rarer keywords. Also, it must have a mechanism to avoid adding faculty members to review panels where they have a match through common keywords.

To formalise the discussion, we define the following:
1. Faculty set F.
2. Topic set T.
3. Application set A.
4. Faculty topic set $FT : F \rightarrow 2^T$.
5. Application topic set $AT : A \rightarrow 2^T$.

Now, in order to create mapping between professors and candidates, we have created a bipartite graph which represents the similarity in topics between any two nodes. An example of this data model is shown in the below figure:

In this diagram, 0,1,2.. represents professors and 100,101,102.. Represents candidates and the weight of the edges represents the number of common topics between them.

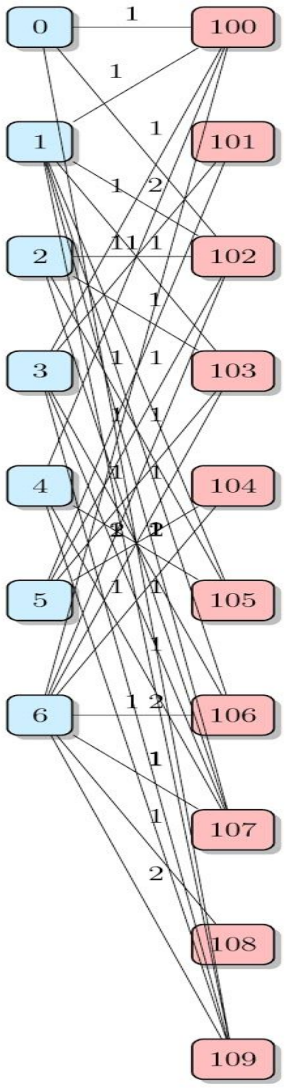

Fig 1: Faculty Candidate Mapping Graph

Using this graph we can then find the best match between the candidate and the professors while prioritizing the less common topics over the more common ones. This is done by assigning scores to faculty and topics which defines the priority of the same.We can calculate the scores as follows:

1.Match triples.

For faculty interest and applicant interest, we construct match triples (F × T × A) which are generated by merging the information available in FT and AT.

2. Faculty score SF .

The number of match triples a faculty member (a.k.a professor) appears in.

3. Topic score ST .

The number of match triples a topic appears in.

Topic score is indicative of how popular a particular topic is among the applicants. Faculty score is indicative of a faculty's involvement in popular topics. Topics will be checked by a large number of applicants. The review panels are computed using the following algorithm.

```
def calculate_review_panels(fac_apps):
  def second(t):
    _, b = t
    return b

  faculty_score, application_score, topic_score =
    calculate_scores(all_fac_apps)
  review_panels = {}
  for app in application_score:
    app_fac_list = []
    for fac in faculty_score:
      for f, a, ts in fac_apps:
        if f == fac and a == app:
          topic_weight = sum([1.0 / topic_score[t] for t in ts])
          match_score = topic_weight / faculty_score[fac]
          app_fac_list.append((fac, match_score))

    app_fac_list.sort(key=second, reverse=True)
    review_panels[app] = app_fac_list
  return review_panels
```

Fig 2 : Review Panel algorithm

EVALUATION

We compared the performance of our approach with that currently practiced in our institute as detailed in above section For comparison, we used the following metrics:

1. Matching efficiency:

We call the degree to which constituted review panels enable each application to be reviewed only by pertinent reviewers and conversely prevents unrelated applications to land on the desk of any professor with matching efficiency. Mathematically, we measure the matching efficiency of the panel formation process as the sum of weights of the edges in matching graph which are selected by the matching processes: higher the sum, greater is the matching efficiency.

2. Time span of Interviews:
The time elapsed between the start and end of the interview process (deducting the breaks in between) is a measure of calendar time required to conduct the interview. An interview scheduling process that exploits the concurrent schedulability of non-conflicting panels will be able to conduct the entire interview process in less time.

3. We used the data from one cycle of research application processing in our institute to compute the matching efficiency of the existing process. We constituted review panels using our algorithm using the same input. The algorithm was found to constitute review panels with overall higher matching efficiency than the existing process.

4. We also compared the elapsed time of the interviews that were conducted in the same admission cycle with the elapsed time that would arise from the interview schedule generated by our algorithm. The time span of the interview schedule generated by our algorithm was found to be significantly shorter than the one that happened in the admission cycle in question.

CONCLUSION

We have presented an automated system for research application processing. As an example, we have presented details of the current process followed in our institute, and have experimentally compared the performance of our approach with the existing predominantly manual process. From our experiments, our algorithm is seen to consistently outperform the existing method followed in our institute.

In this paper, for concreteness we have situated our work in the specific case of our institute. However, the process can be used by any institution for automating its application process with little or no modification. In fact, any process which involves processing a large number of applications can benefit from the ideas presented in this paper.

Our current preliminary implementation provides encouraging results. We are currently extending our work in the following specific directions: Firstly, we are exploring harnessing more sophisticated matching algorithms (eg. based on machine-learning) to constitute review panels for even better matching efficiency. Secondly, we are exploring alternative approaches (e.g. based on meta-heuristics like genetic algorithms) for generating interview schedules. With these modifications, we are confident that the automated application processing system will result in a significantly more efficient process yielding far improved utilisation of institutional resources.

REFERENCES

[1]http://www.academia.edu/Documents/in/Graph_Coloring
[2]http://kodu.ut.ee/~varmo/TM2010/slides/tm-reg.pdf
[3]https://www.researchgate.net/publication/228057670_Coloring_heuristics_for_register_allocation