Team Formation in Online Social Networks

Stefano Leonardi

Sapienza University of Rome and
Google NYC

Based on work with

Aris Anagnostopoulos
Luca Becchetti
Sapienza Univ. of Rome

Carlos Castillo
Qatar Computing Research Institute

Aris Gionis
AALTO Univ. - Helsinki
Online Collaborative Social Systems

Success stories of online collaborative systems indicate that much more is possible:

Tagging

Geotagging

Wikipedia

Fold It

Polymath
Do you have ...?

... too many papers/proposals to review?
... or too many candidates to interview?

Paper review workload for last year: ~60 papers
Setting

- Pool of people with different skills
- Stream of tasks/jobs arriving online
- Tasks have some skill requirements
- Create teams on-the-fly for each job
  - Select the right team
  - Satisfy various criteria
Criteria

- **Fitness**
  - E.g. if fitness is success rate, maximize expected number of successful tasks
  - Depends on:
    - People skills
    - Ability to coordinate

- **Efficiency**
  - Do not load people very much

- **Fairness**
  - everybody should be involved in roughly the same number of tasks

- **Trade-offs may appear: do you see how?**
Framework

- Jobs/Tasks (k)
- People (n)
- Skills (m)
- Teams (k)
- Distance between people
- Team coordination cost
- Score/fitness
- Load

\[ J = \{ J^j ; j = 1, 2, \ldots, k \} \]
\[ P = \{ p^j ; j = 1, 2, \ldots, n \} \]
\[ S = \{ 0, 1 \}^m \quad \text{or} \quad S = [0, 1]^m \]
\[ Q^j \subseteq P \]
\[ d(p^i, p^j) \]
\[ c(Q^j) \]
\[ s(Q^j, J^j) \]
\[ L(p) = |\{ j ; p \in Q^j \}| \]
Properties

• Non decreasing performance
  – Adding people to a team does no harm
    \[ Q^j \subseteq Q^i \Rightarrow s(Q^j, J) \leq s(Q^i, J) \]

• Pareto-dominant profiles
  – Hiring a more expert person can only improve

• Nonincreasing marginal utility
  – The value of adding a person to a smaller team is bigger
    that adding the person to a larger team

• Job monotonicity
  – If a task requires strictly more skills then a team can only
    perform worse
Properties (cont.)

• Non decreasing performance
  – c.f. Brooks' Law: “adding manpower to a late software project makes it later”

• Non-increasing marginal utility
  – May not hold e.g. if all skills are required

• Job monotonicity
  – Compare team with skills X on two jobs
    • Job 1: Requires Y (disjoint from X)
    • Job 2: Requires X ∪ Y
Team profiles

• Maximum skill

\[ q_\ell = \max_j p_{\ell}^j \]

• Additive skills

\[ q_\ell = \min\{1, \sum_{j=1}^{\left|Q\right|} p_{\ell}^j\} \]

• Multiplicative skills

\[ q_\ell = (1 - \prod_{j=1}^{\left|Q\right|} (1 - p_{\ell}^j)) \]
Score functions

• Fraction of skills possessed

\[ s(q, J) = \frac{|\{\ell; J_\ell > 0 \land q_\ell \geq J_\ell\}|}{|\{\ell; J_\ell > 0\}|} \]

• is sub-modular: greedy method provides an approximation within a constant factor

• In other applications all skills are required: covering problem
Binary Profiles

In this talk (and most the work): Binary skill profiles

\[ S = \{0, 1\}^m \]

- A person either has a skill or not
- Team has a skill if a person has it
- A job either requires it or not
- Score of a team \( Q \) for task \( J \)

\[ s(Q, J) = \begin{cases} 
1, & \text{if } Q \text{ has all the skills of } J, \\
0, & \text{otherwise.}
\end{cases} \]

- Covering problem
- Other options are available
Balanced task covering

• Cover all the jobs
  \[ s(Q^j, J^j) = 1 \quad \forall j = 1, \ldots, k \]

• Objective
  \[ \min \max_j L(p^j) \]

• NP-hard problem even with \( k = 2 \)
  – Reduction from MSAT (a clause for each skill of each of the two jobs, experts are variables: expert assigned to job 1 if positive literal is true, to job 2 if negative literal is true)

• Offline setting has a randomized approx. algo. that succeeds with prob 1 - \( \delta \) with ratio
  \[ O \left( \log \left( \frac{mk + n}{\delta} \right) \right) \]
Balanced task covering – Online

• Evaluate by **competitive ratio**
  – Compare with optimal offline assignment
  – Offline has full information

• Simple heuristics
  – Assemble the team of minimum size
  – Assemble the team that minimize the maximum load of a person: \( \max_{p \in Q} L^t(p) \)
  – Assemble the team that keeps the minimize the sum of the loads of the team: \( \sum_{p \in Q} L^t(p) \)
  – Competitive ratios are bad: \( \Omega(n), \Omega(k), \Omega(\sqrt{m}) \)

• In practice some are OK
Algorithm ExpLoad

When a task arrives at time $t$

- Weight each person $p$ by $(2n)^{L_t(p)}$

- Select team $Q$ that covers all task skills and minimizes
  $$\sum_{p \in Q} (2n)^{L_t(p)}$$

- Weighted set cover problem

- **Theorem.** Competitive ratio $= O(\log m \log k)$
Experiments
### Mapping of data to problem instances

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Experts</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMDB</td>
<td>Movie directors</td>
<td>Audition actors</td>
</tr>
<tr>
<td>Bibsonomy</td>
<td>Prolific scientists</td>
<td>Interview scientists</td>
</tr>
<tr>
<td>Flickr</td>
<td>Prolific photographers</td>
<td>Judge photos</td>
</tr>
</tbody>
</table>

### Summary statistics

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Experts</th>
<th>Tasks</th>
<th>Skills</th>
<th>Skills/expert</th>
<th>Skills/task</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMDB</td>
<td>725</td>
<td>2173</td>
<td>21</td>
<td>2.96</td>
<td>11.10</td>
</tr>
<tr>
<td>Bibsonomy</td>
<td>816</td>
<td>35506</td>
<td>793</td>
<td>7.64</td>
<td>4.44</td>
</tr>
<tr>
<td>Flickr.art</td>
<td>504</td>
<td>59869</td>
<td>12913</td>
<td>49.90</td>
<td>15.73</td>
</tr>
<tr>
<td>Flickr.nature</td>
<td>2879</td>
<td>112467</td>
<td>26379</td>
<td>31.25</td>
<td>15.45</td>
</tr>
</tbody>
</table>
We report mean, maximum, and additional columns as follows: $\phi_{.9}$ denotes the 90% quantile; $\sigma_{.9}$ is the maximum team size that an algorithm allocates provided that each task is covered only up to 90% of the required skills; finally, $\lambda_{.1}$ is the mean load of the 10% more loaded experts.

<table>
<thead>
<tr>
<th>Method</th>
<th>Team size statistics</th>
<th>Experts load statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mean  $\phi_{.9}$ $\sigma_{.9}$ max</td>
<td>mean  $\phi_{.9}$ $\lambda_{.1}$ max</td>
</tr>
<tr>
<td><strong>IMDB</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>2.31  4 3 5</td>
<td>6.92  11 58 1260</td>
</tr>
<tr>
<td>MaxLoad</td>
<td>3.27  4 3 7</td>
<td>9.80  45 53 65</td>
</tr>
<tr>
<td>SumLoad</td>
<td>4.75  7 3 10</td>
<td>14.23 32 46 65</td>
</tr>
<tr>
<td>ExpLoad</td>
<td>3.80  5 3 9</td>
<td>11.38 32 47 64</td>
</tr>
<tr>
<td><strong>Bibsonomy</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>2.70  5 5 22</td>
<td>117.66 251 397 1417</td>
</tr>
<tr>
<td>MaxLoad</td>
<td>2.92  5 3 22</td>
<td>127.13 248 353 700</td>
</tr>
<tr>
<td>SumLoad</td>
<td>3.13  6 7 25</td>
<td>136.05 244 343 701</td>
</tr>
<tr>
<td>ExpLoad</td>
<td>2.83  5 4 22</td>
<td>123.27 258 365 700</td>
</tr>
<tr>
<td><strong>Flickr.nature</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>6.34  10 25 29</td>
<td>247.85 439 823 6645</td>
</tr>
<tr>
<td>MaxLoad</td>
<td>7.38  11 27 31</td>
<td>288.22 468 571 941</td>
</tr>
<tr>
<td>SumLoad</td>
<td>7.53  12 30 35</td>
<td>294.09 438 535 937</td>
</tr>
<tr>
<td>ExpLoad</td>
<td>7.08  11 28 34</td>
<td>276.60 475 587 964</td>
</tr>
</tbody>
</table>
We report mean, maximum, and additional columns as follows: $\phi_{.9}$ denotes the 90% quantile; $\sigma_{.9}$ is the maximum team size that an algorithm allocates provided that each task is covered only up to 90% of the required skills; finally, $\lambda_{.1}$ is the mean load of the 10% more loaded experts.

<table>
<thead>
<tr>
<th>Method</th>
<th>Team size statistics</th>
<th>Experts load statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mean</td>
<td>$\phi_{.9}$</td>
</tr>
<tr>
<td><strong>INDB</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>2.31</td>
<td>4</td>
</tr>
<tr>
<td>MaxLoad</td>
<td>3.27</td>
<td>4</td>
</tr>
<tr>
<td>SumLoad</td>
<td>4.75</td>
<td>7</td>
</tr>
<tr>
<td>ExpLoad</td>
<td>3.80</td>
<td>5</td>
</tr>
<tr>
<td><strong>Bibsonomy</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>2.70</td>
<td>5</td>
</tr>
<tr>
<td>MaxLoad</td>
<td>2.92</td>
<td>5</td>
</tr>
<tr>
<td>SumLoad</td>
<td>3.13</td>
<td>6</td>
</tr>
<tr>
<td>ExpLoad</td>
<td>2.83</td>
<td>5</td>
</tr>
<tr>
<td><strong>Flickr</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>6.34</td>
<td>10</td>
</tr>
<tr>
<td>MaxLoad</td>
<td>7.38</td>
<td>11</td>
</tr>
<tr>
<td>SumLoad</td>
<td>7.53</td>
<td>12</td>
</tr>
<tr>
<td>ExpLoad</td>
<td>7.08</td>
<td>11</td>
</tr>
</tbody>
</table>
Coordination cost

- Have not taken into account **coordination cost**
- Distance between people $d(p^i, p^j)$
- Team coordination cost $c(Q^j)$
- Select teams that minimizes $c(Q^j)$
  - Steiner-tree cost
  - Diameter
  - Sum of distances
Coordination cost

- Steiner-tree cost
- Diameter
- Sum of distances

$$\sum_{p^i, p^j \in Q} d(p^i, p^j)$$
Conflicting goals

• We want solutions that minimize
  – Load
  – Unfairness
  – Coordination cost

and satisfy each job.
Our modeling approach

- Set a desirable coordination cost upper bound $B$
- Online solve

$$\min \max_i L(p^i)$$

$$s(J^j, Q^j) = 1 \quad \forall j \in J$$

$$c(Q^j) \leq B \quad \forall j \in J.$$ 

- 3 different problems for the 3 different coordination costs
- This talk: focus on Steiner tree coordination cost
Algorithm

At every step $t$:
- Combine ExpLoad with coordination cost constraint $\Rightarrow$
- Find a team that:
  - Covers all required skills
  - Satisfies $c(Q) \leq B$
  - Minimizes $\sum_{p \in Q} (2n)^{L_t(p)}$
- How?
At every step $t$

- Incorporate to the graph $\lambda(2n)^{L_t(p)}$
- Create a family of graphs
- Solve a variant of Steiner tree. Get a solution that
  - Covers all required skills
  - Satisfies $c(Q) \leq \beta B$
  - $\alpha$-approximates $\sum_{p \in Q} (2n)^{L_t(p)}$
- Different graphs in the family tradeoff between $\alpha, \beta$
Result

We wanted:  \[
\min_i \max_i L(p_i)
\]
\[
s(J^j, Q^j) = 1 \quad \forall j \in J
\]
\[
c(Q^j) \leq B \quad \forall j \in J.
\]

**Theorem.** The algorithm satisfies:

- \(\alpha\)-approximates \[
\min_i \max_i L(p_i)
\]
  \[
s(J^j, Q^j) = 1 \quad \forall j \in J
\]
  \[
c(Q^j) \leq \beta B \quad \forall j \in J.
\]

- Can obtain \(\alpha, \beta = O(\log(n m k))\)
Group Steiner Tree

• Group Steiner Tree: Construct a Steiner tree that connects at least one node for each group

• Heuristics for Group Steiner Tree:

1. LLT [Lappas, Liu, Terzi, KDD 2009]
   
   – Connect each skill $J_l$ to all experts that own the skill
   
   – Construct a Steiner tree connecting all skills of $J$
Group Steiner tree

2. Set Cover (SC): Cover all skills with experts.

At each step select the most effective expert cost-effectiveness:

\[
\frac{\text{gain}(p^j)}{\text{loss}(p^j) + \lambda \text{ExpLoad of the expert}}
\]

- \(\text{gain}(p^j)\): \# newly covered skills
- \(\text{loss}(p^j)\): distance to experts selected so far plus \(\lambda \text{ExpLoad of the expert}\)
Experiments Bibsonomy

Experts = prolific authors
Task = interview scientists
Distance = \( f(\text{#collaborations}) \)
Optimize over \( \lambda \)

Bibsonomy – implicitly connected

Diagram showing the relationship between max load, coordination cost, and team size for different values of \( \lambda \). The plots illustrate the performance of the SC and LLT algorithms.
Experiments Bibsonomy

Experts = prolific authors
Task = interview scientists
Distance = $f(\#\text{collaborations})$
Experiments IMDB

Experts = directors
Task = find a cast
Distance = $f(\text{#common actors directed})$
Related work

• Lots of works on matching and scheduling problems
• Lots of works on finding one expert
  – IR-style and SN-style
• T. Lappas, K. Liu, E. Terzi. Finding a team of experts in social networks, KDD'09.
  – Focuses on communication costs
  – Only one task
Conclusions and Future work

• Gave a framework for online collaboration
• Obtain competitive online algorithms
• Balance between various contradicting objectives
• Perform well in practice

Future work
• Profile learning
• Learn coordination based on performance
• Train people
• Incentives for participation
Thanks!