# Algorithms for Online Labour Marketplaces


**Stefano Leonardi**

Sapienza University of Rome


Based on work with


| | |
|---|---|
| **Aris Anagnostopoulos** | (Sapienza Univ.) |
| **Carlos Castillo** | (UPF, Barcelona) |
| **Adriano Fazzone** | (Sapienza Univ.) |
| **Aris Gionis** | (Aalto Univ.) |
| **Evimaria Terzi** | (Boston University) |

# Online labor marketplaces

- We will see an increase in the sophistication of systems that use and guide user actions

- Require models and algorithms to capture the human elements

  - What skills people have
  - Efficiency
  - Time availability
  - Human-human relationship
  - Incentive and behavioral issues
  - Human errors / disagreements
  - Work organization

# Online collaborative systems

Several success stories indicate that much more is possible:

- Tagging/geotagging systems:

- Content creation systems:
- Online labor markets:
- Crowdsourcing:
- Polymath project:

- Open source community:

# This lecture

We will look at two specific problems:

- **How can we form teams** of experts online when compatibility between experts is modelled by a social network

- How can we decide online when to use outsourced workers, when to hire workers in a team and when to fire inactive workers

# This lecture

We like to solve the above problems while achieving:

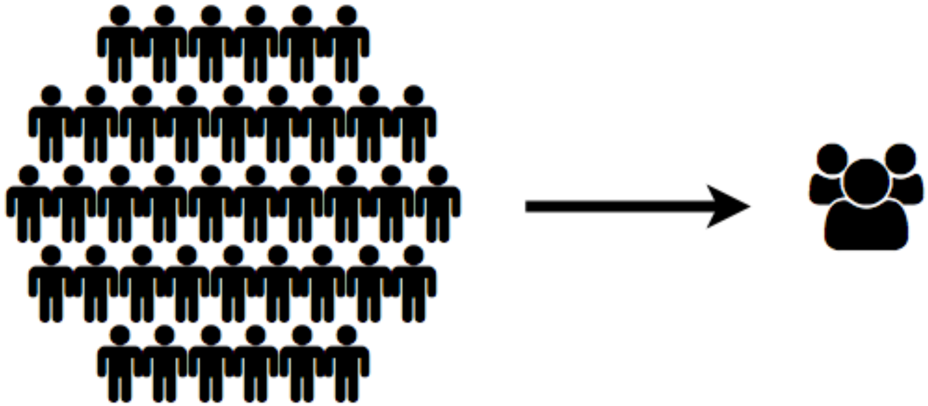
- Good performance of formed teams on allocated tasks
- Fair distribution of the task load between experts
- Low  coordination overhead within a team
- Good trade-offs between outsourcing and hiring/salary cost

# Online collaborative systems

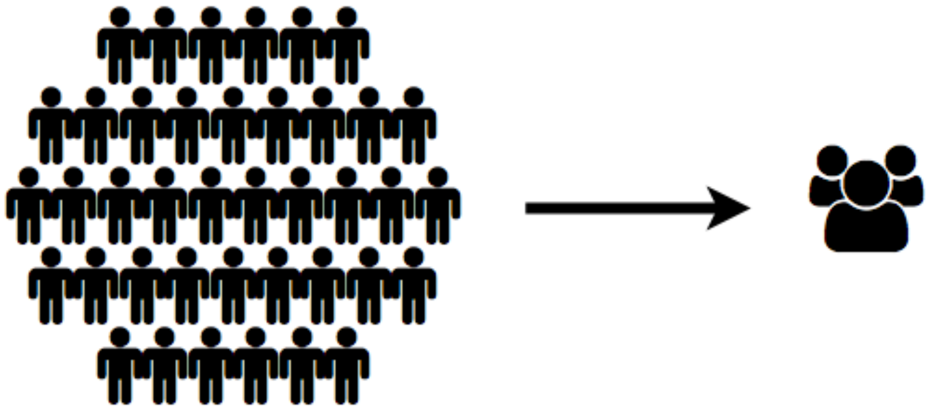
Several success stories indicate that much more is possible:

- Tagging/geotagging systems:  
- Content creation systems:
- Online labor markets:
- Crowdsourcing:
- Polymath project: 
- Open source community:

# Team formation



# Team formation

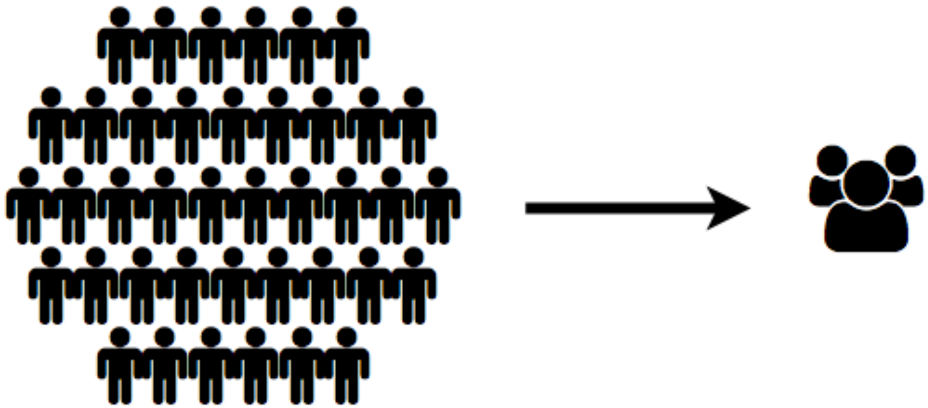


**Industrial and business settings**

**Cluster hires**: Which experts should be hired?

**Online collaborations**: Can teams really work online?

# Team formation

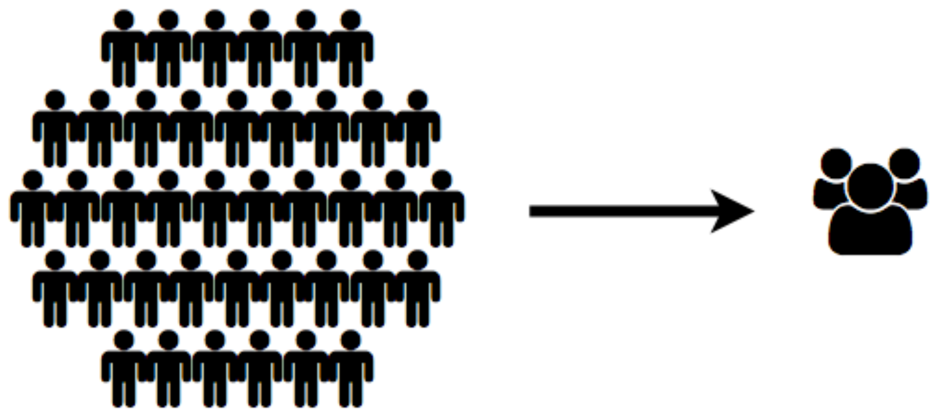


**Educational settings**

**Traditional classroom**: How to create good study groups?

**Massive Online Courses (MOOCS)**: How to bring in social aspects?

# Team formation



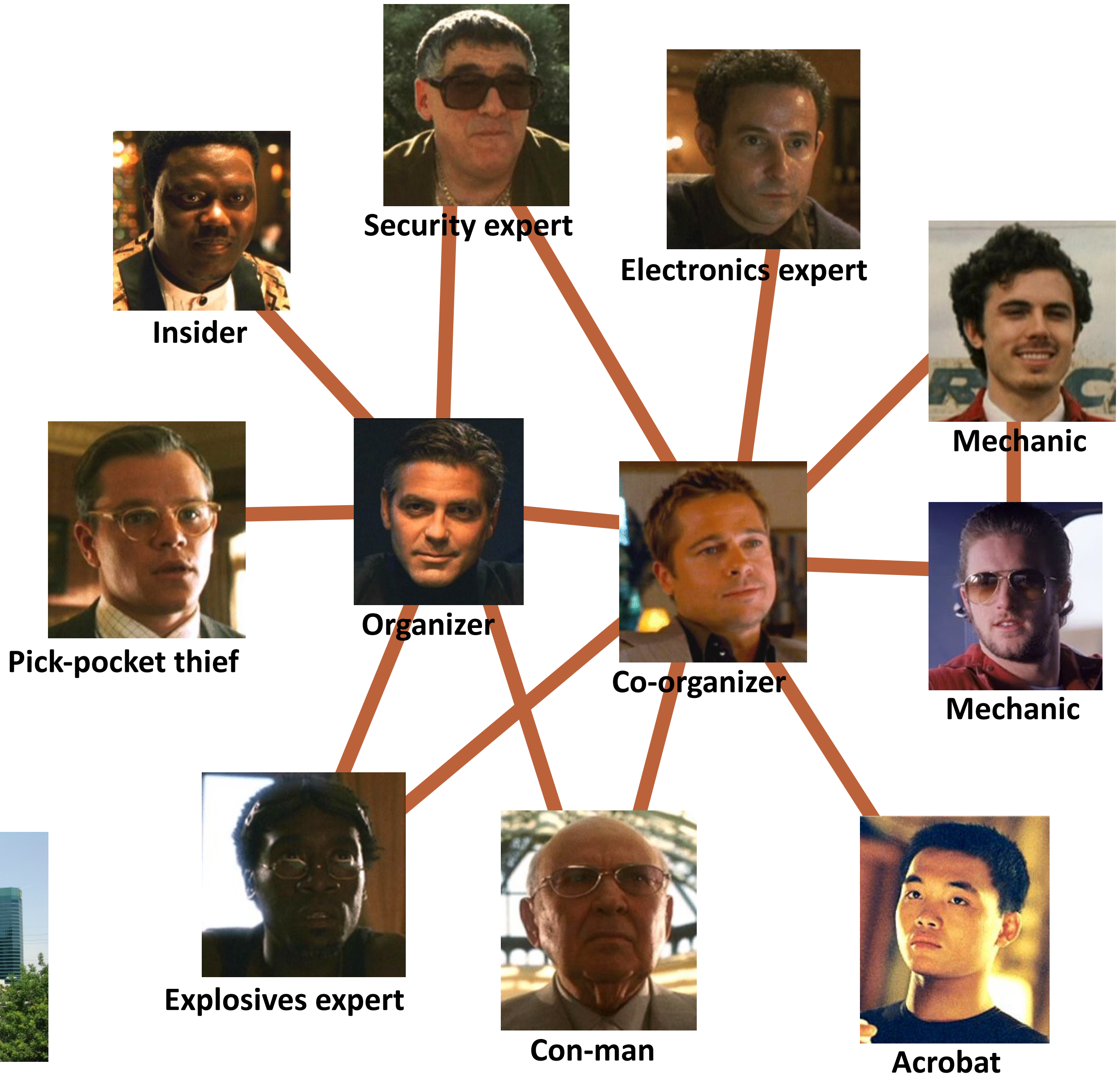
**Research environments**

Writing proposals with others
Cluster hires with diversity
Collaborative problem solving


george clooney
matt damon
andy garcia
brad pitt
julia roberts
OCEAN'S ELEVEN
Place your bets.


MIRAGE
GRAND


george clooney
matt damon
andy garcia
brad pitt
julia roberts
OCEAN'S ELEVEN
Place your bets.


Security expert
Electronics expert
Insider
Mechanic
Pick-pocket thief
Organizer
Co-organizer
Mechanic
Explosives expert
Con-man
Acrobat


GRAND


george clooney
matt damon
andy garcia
brad pitt
julia roberts
OCEAN'S ELEVEN
Place your bets.


george clooney
matt damon
andy garcia
brad pitt
and julia roberts
OCEAN'S ELEVEN
Place your bets.


That Big One!!
FRANK SINATRA !! DEAN MARTIN
SAMMY DAVIS JR. PETER LAWFORD
ANGIE DICKINSON
"OCEAN'S 11"
Nobody else would have
dared it because nobody else would
have the nerve! Just Danny Ocean and his 11 pals –
the crazy night they blew all the lights in Las Vegas!...
TECHNICOLOR PANAVISION WARNER BROS.
RICHARD CONTE · CESAR ROMERO · PATRICE WYMORE · JOEY BISHOP


george clooney
matt damon
andy garcia
brad pitt
julia roberts
OCEAN'S ELEVEN
Place your bets.


That Big One!!
FRANK SINATRA !! DEAN MARTIN
SAMMY DAVIS JR. PETER LAWFORD
ANGIE DICKINSON
"OCEAN'S 11"
Nobody else would have dared it because nobody else would have the nerve! Just Danny Ocean and his 11 pals – the crazy night they blew all the lights in Las Vegas!...
TECHNICOLOR PANAVISION WARNER BROS.
RICHARD CONTE · CESAR ROMERO · PATRICE WYMORE · JOEY BISHOP


VITTORIO GASSMAN • CLAUDIA CARDINALE
MARCELLO MASTROIANNI • RENATO SALVATORI TOTO'
i soliti ignoti
MARIO MONICELLI


george clooney
matt damon
andy garcia
brad pitt
and julia roberts
OCEAN'S ELEVEN
Place your bets.


That Big One!!
FRANK SINATRA !! DEAN MARTI
SAMMY DAVIS JR. PETER LAWFOR
ANGIE DICKINSON
OCEAN'S 11
Nobody else would have dared it because nobody else would have the nerve! Just Danny Ocean and his 11 pals – the crazy night they blew all the lights in Las Vega
TECHNICOLOR PANAVISION
WARNER BROS.
RICHARD CONTE · CESAR ROMERO · PATRICE WYMORE · JOEY BISHOP


VITTORIO GASSMAN • CLAUDIA CARDINALE
MARCELLO MASTROIANNI • RENATO SALVATORI TOTO'
i soliti ignoti
MARIO MONICELLI


TONY LE STEPHANOIS EST EXACT AU RENDEZ-VOUS...
...IL VA Y AVOIR
DU RIFIFI CHEZ LES HOMMES
UN FILM DE JULES DASSIN
d'après le roman d'Auguste le Breton
édité par la librairie Gallimard
Musique de GEORGES AURIC
avec
JEAN SERVAIS
CARL MOHNER
ROBERT MANUEL
et par ordre alphabétique
JANINE DARCEY · PIERRE GRASSET · ROBERT HOSSEIN
MARCEL LUPOVICI · MAGALI NOËL · MARIE SABOURET · CLAUDE SYLVAIN
PRODUCTION INDUSFILMS · PRIMA FILM · S.N. PATHÉ CINÉMA


george clooney
matt damon
andy garcia
brad pitt
julia roberts
OCEAN'S ELEVEN
Place your bets.

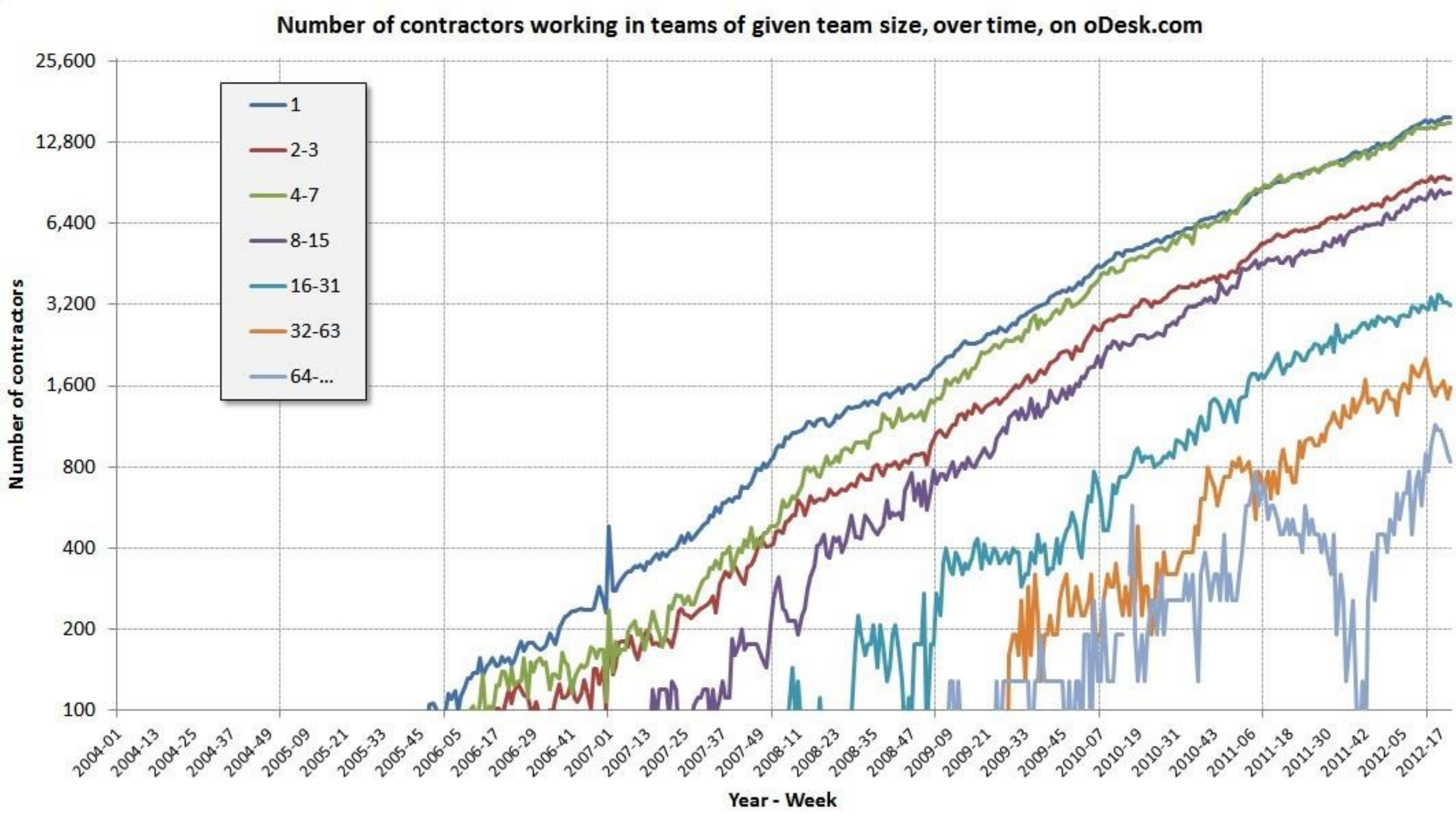

That Big One!!


VITTORIO GASSMAN • CLAUDIA CARDINALE
MARCELLO MASTROIANNI • RENATO SALVATORI TOTO'
i soliti ignoti
MARIO
MONICELLI


CULT ΒΙΝΤΕΟΘΗΚΗ 80s
ΓΙΩΡΓΟΣ ΚΩΝΣΤΑΝΤΙΝΟΥ
ΜΑΡΚΟΣ ΛΕΖΕΣ
ΓΙΑΝΝΗΣ ΒΟΥΡΟΣ
011374
THE
ΚΟΠΑΝΟΙ
ΣΕΝΑΡΙΟ-ΣΚΗΝΟΘΕΣΙΑ:
ΓΙΩΡΓΟΣ ΚΩΝΣΤΑΝΤΙΝΟΥ
DVD
VIDEO


TONY LE STEPHANOIS
EST EXACT AU
RENDEZ-VOUS...
...IL VA Y AVOIR
DU
RIFIFI
CHEZ LES
HOMMES
UN FILM DE JULES DASSIN
d'après le roman d'Auguste le BRETON
édité par la librairie GALLIMARD
Musique de
GEORGES AURIC
avec
JEAN SERVAIS
CARL MOHNER
ROBERT MANUEL
et par ordre alphabétique
JANINE DARCEY · PIERRE GRASSET · ROBERT HOSSEIN
MARCEL LUPOVICI · MAGALI NOËL · MARIE SABOURET · CLAUDE SYLVAIN
PRODUCTION INDUSFILMS · PRIMA FILM · S.N. PATHÉ CINÉMA

# oDesk – Team sizes over time



# The Online Team Formation Problem

# Related work

### Business & Management Science

[Lau et al. 1998]
[Li et al. 2005]
[Choi et al. 2010]
[Thatcher et al. 2003]
[Molleman 2005]
[Polzer et al. 2006]
[Bezrukova et al. 2009]
[Pearsall et al. 2008]
[Jehn et al. 2010]
[Gratton et a. 2007]
[Shaw 2004]

### Education Sciences

[Slavin 1987]
[Kulik 1982]
[Kerchoff 1986]
[Kulik et al. 1992]
[Mislevy 1983]
[Lazarowitz et al. 1995]
[Vygotsky et al. 1978]

### Social Research

[DeGroot 1974]
[Friedkin et al. 1990]
[Jackson et al. 2008]
[Friedkin et al. 1999]

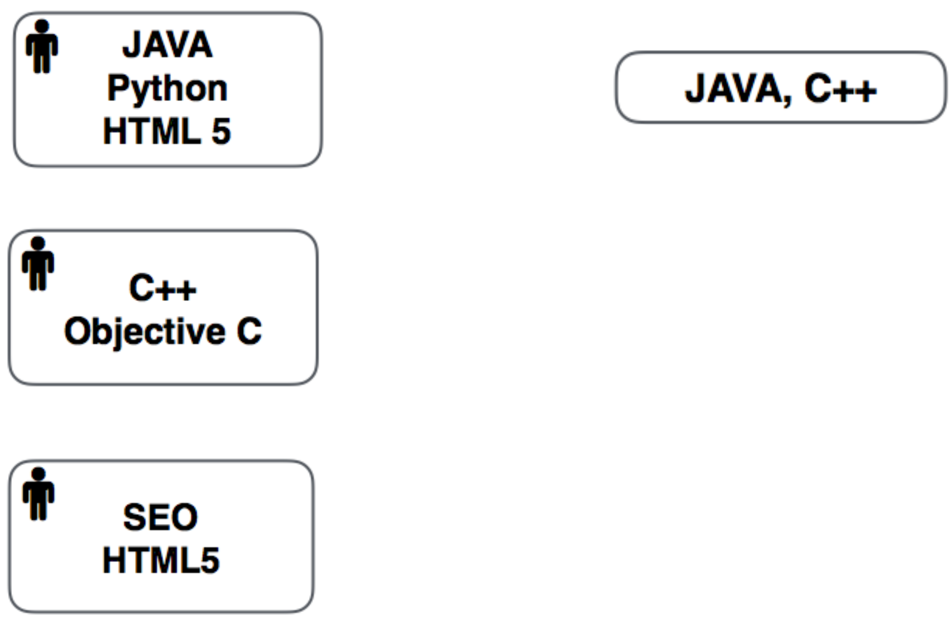
### Computer Science

[Anagnostopoulos et al. 2010]
[Okimoto et al. 2015]
[Agrawal et al. 2014]
[Lappas et al. 2009]
[Sozio et al. 2010]
[Gajewar et al. 2012]
[Anagnostopoulos et al. 2012]
[Yildiz et al. 2013]
[Kargar et al. 2013]
[Dorn et al. 2010]
[Kargar et al. 2011]
[Li et al. 2010]
[Bell, 2007]
[Majumder 2012]
[Golshan et al. 2014]

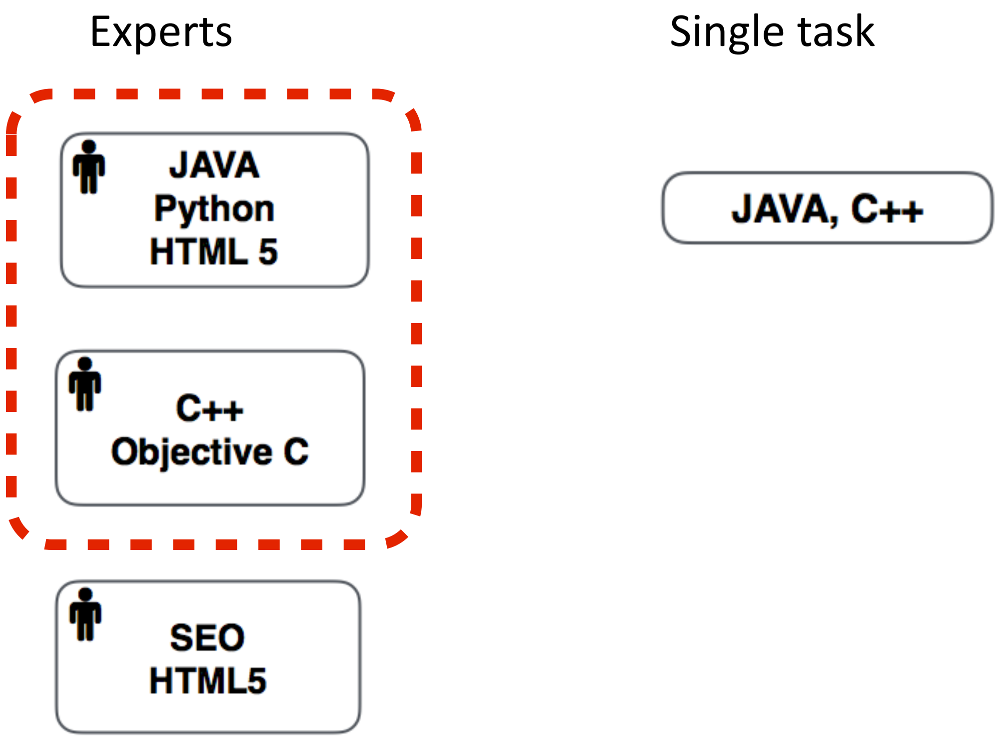
# Set-cover view of team formation

Experts

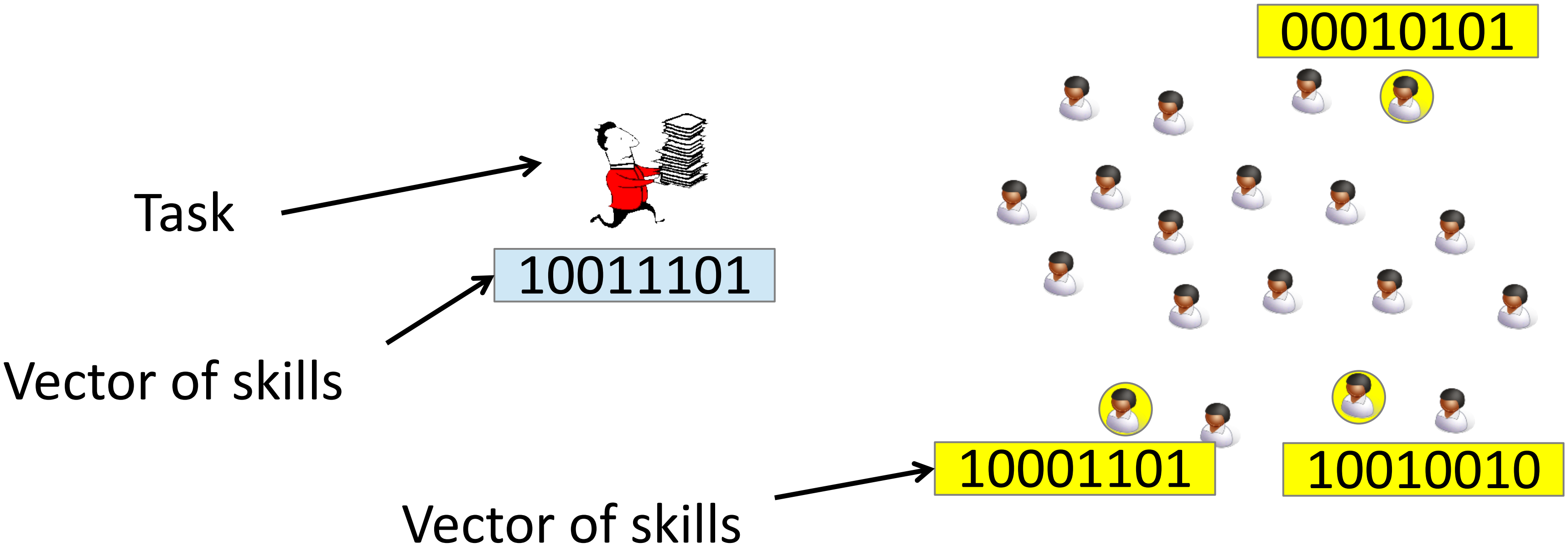
Single task



# Set-cover view of team formation



# Basic formulation: set cover



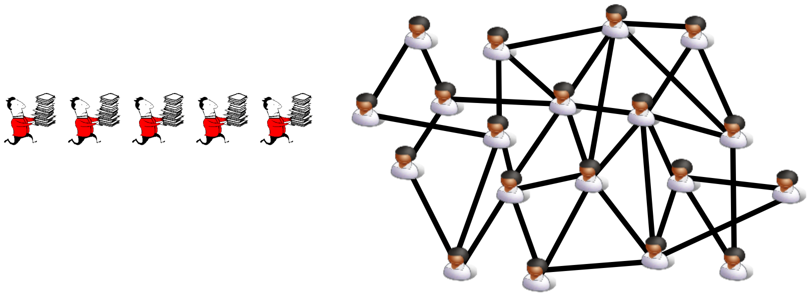

**Problem:** Given a pool of experts, a single task hire the minimum-cost subset of experts that can complete (i.e., cover) the task

**Facts:**

- The problem is NP-hard
- Greedy algorithm is a a good approximation algorithm

# Setting

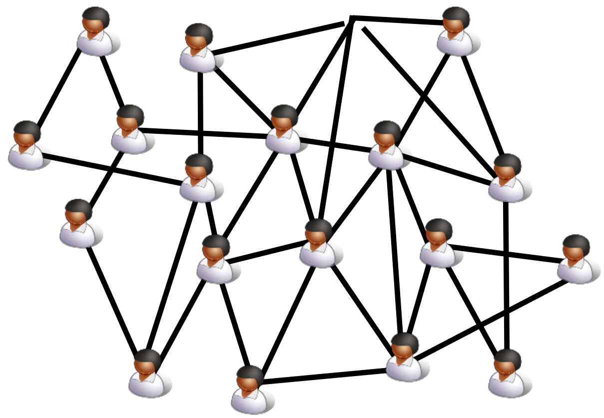
- Pool of people with different skills
- Stream of tasks/jobs arriving online
- Tasks have some skill requirements
- Create teams on-the-fly for each job
  - Select the right team
  - Satisfy various criteria



# Criteria

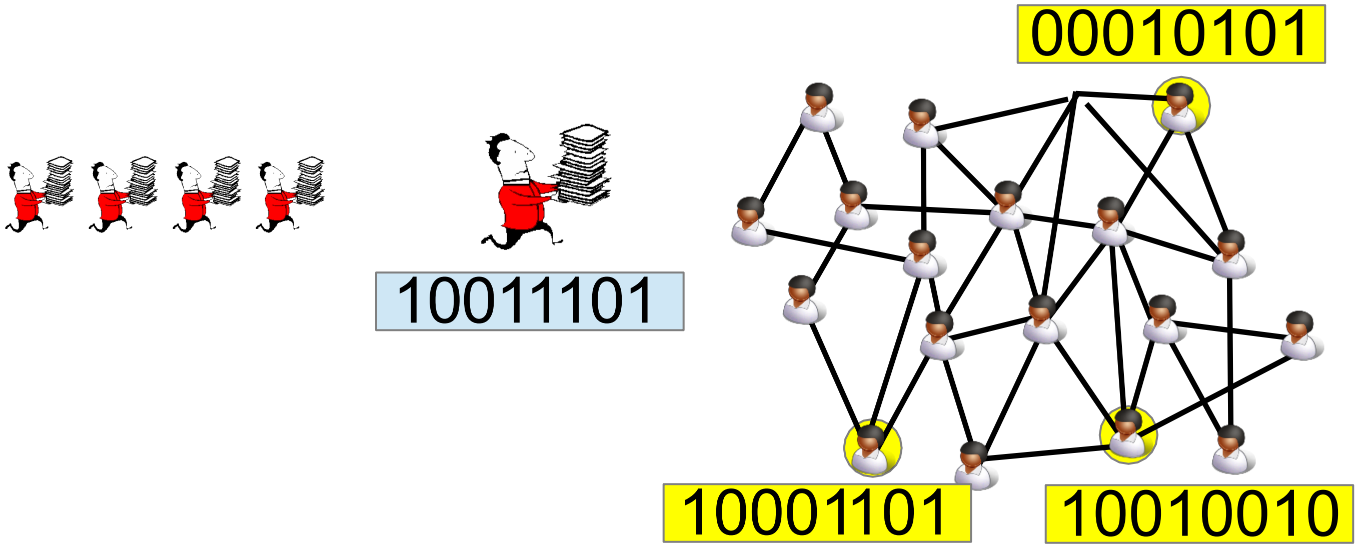
- Fitness
  - E.g. success rate, maximize expected number of successful tasks
  - Depends on:
    - People skills
    - Ability to coordinate
- Fairness: everybody should be involved in roughly the same number of tasks
- Efficiency:
  - Cost of outsourced tasks vs cost of hired workers
- Trade-offs may appear: do you see how?

# Basic formulation: Skills and people



- *n* People/Experts $\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^n$
- *m* Skills $\mathcal{S} = \{0, 1\}^m$
- Each person has some skills $\mathbf{p}^i \in \mathcal{S}$

# Basic formulation: jobs & teams

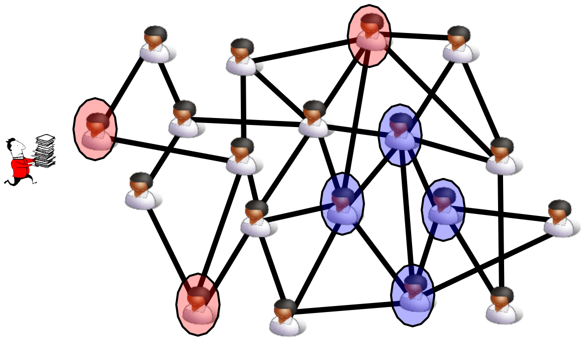



- Stream of *k* Jobs/Tasks $\mathbf{J}^1, \mathbf{J}^2, \ldots, \mathbf{J}^k$
- A job requires some skills $\mathbf{J}^j \in \mathcal{S}$
- *k* Teams are created online $Q^j \subseteq \{\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^n\}$
- A team must **cover** all job skills

Load: $L(\mathbf{p}) = |\{j;\, \mathbf{p} \in Q^j\}|$

# Coordination cost

- Coordination cost measures the compatibility of the team members
- Example of $d(\mathbf{p}^i, \mathbf{p}^j)$:
  - Degree of knowledge
  - Time-zone difference
  - Past collaboration



- Select teams that minimizes coordination cost $c(Q)$:
  - Steiner-tree cost
  - Diameter
  - Sum of distances

# Framework

- Jobs/Tasks (*k*) $\mathcal{J} = \{\mathbf{J}^j;\ j = 1, 2, \ldots, k\}$
- People (*n*) $\mathcal{P} = \{\mathbf{p}^j;\ j = 1, 2, \ldots, n\}$
- Skills (*m*) $\mathcal{S} = \{0, 1\}^m$ or $\mathcal{S} = [0, 1]^m$
- Teams (*k*) $Q^j \subseteq \mathcal{P}$
- Distance between people $d(\mathbf{p}^i, \mathbf{p}^j)$
- Team coordination cost $c(Q^j)$
- Score/fitness $s(Q^j, \mathbf{J}^j)$
- Load $L(\mathbf{p}) = |\{j;\ \mathbf{p} \in Q^j\}|$

# Binary Profiles

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

$$\mathcal{S} = \{0, 1\}^m$$

- 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 $\mathbf{J}$

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

- Covering problem
- Other options are available

# Online Balanced Task Covering

# 1. Balanced task covering

- Cover all the jobs $s(Q^j, \mathbf{J}^j) = 1 \quad \forall j = 1, \ldots, k$
- Objective $\min \max_j L(\mathbf{p}^j)$
- NP-hard problem even with *k* $= 2$
- Offline setting has a randomized approx. algo.
  That succeeds with prob 1 – δ with ratio $O\left(\log\left(\frac{mk+n}{\delta}\right)\right)$
- Does it exist an O(1)-APX?

# Our modeling approach

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

Load of person i

$$\min \max_i L(\mathbf{p}^i)$$

$$Q^j \text{ covers } \mathbf{J}^j \qquad \forall j$$

$$c(Q^j) \leq B \qquad \forall j.$$

Team j covers job j

Bounded coordination cost

- Must concurrently solve various combinatorial problems:
  - Set cover
  - Steiner tree
  - Online makespan minimization

# Our modeling approach

| Job | $p_1$ | $p_2$ | $p_3$ | $p_4$ | $p_5$ | $p_6$ | $p_7$ | $Q_j$ |
|---|---|---|---|---|---|---|---|---|
| 1 | | ✓ | | ✓ | ✓ | | | $Q_1 = \{p_2, p_4, p_5\}$ |
| 2 | ✓ | | | ✓ | | ✓ | | $Q_2 = \{p_1, p_4, p_6\}$ |
| 3 | | | ✓ | ✓ | | | | $Q_3 = \{p_3, p_4\}$ |
| 4 | ✓ | | | | ✓ | | ✓ | $Q_4 = \{p_1, p_5, p_7\}$ |
| 5 | | ✓ | ✓ | ✓ | ✓ | | | $Q_5 = \{p_2, p_3. p_4, p_5\}$ |
| 6 | | | ✓ | | ✓ | ✓ | | $Q_6 = \{p_3, p_5, p_6\}$ |
| 7 | ✓ | ✓ | | | | | | $Q_7 = \{p_1, p_2\}$ |
| 8 | ✓ | ✓ | ✓ | ✓ | | | ✓ | $Q_8 = \{p_1, p_2, p_3, p_4, p_7\}$ |
| 9 | | | ✓ | ✓ | ✓ | | | $Q_9 = \{p_3, p_4, p_5\}$ |
| **Load** | 4 | 4 | 5 | 6 | 5 | 2 | 2 | |

$$\text{Competitive ratio} = \max_{I} \frac{\text{cost of alg's online solution on instance } I}{\text{best offline solution on instance } I}$$

# 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 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

Load of **p** at time *t*

When a task arrives at time *t*

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

- Select team $Q$ that covers all task skills and minimizes

$$\sum_{\mathbf{p} \in Q} (2n)^{L_t(\mathbf{p})}$$

- Weighted set cover problem

- **Theorem.** Competitive ratio = $O(\log m \log k)$

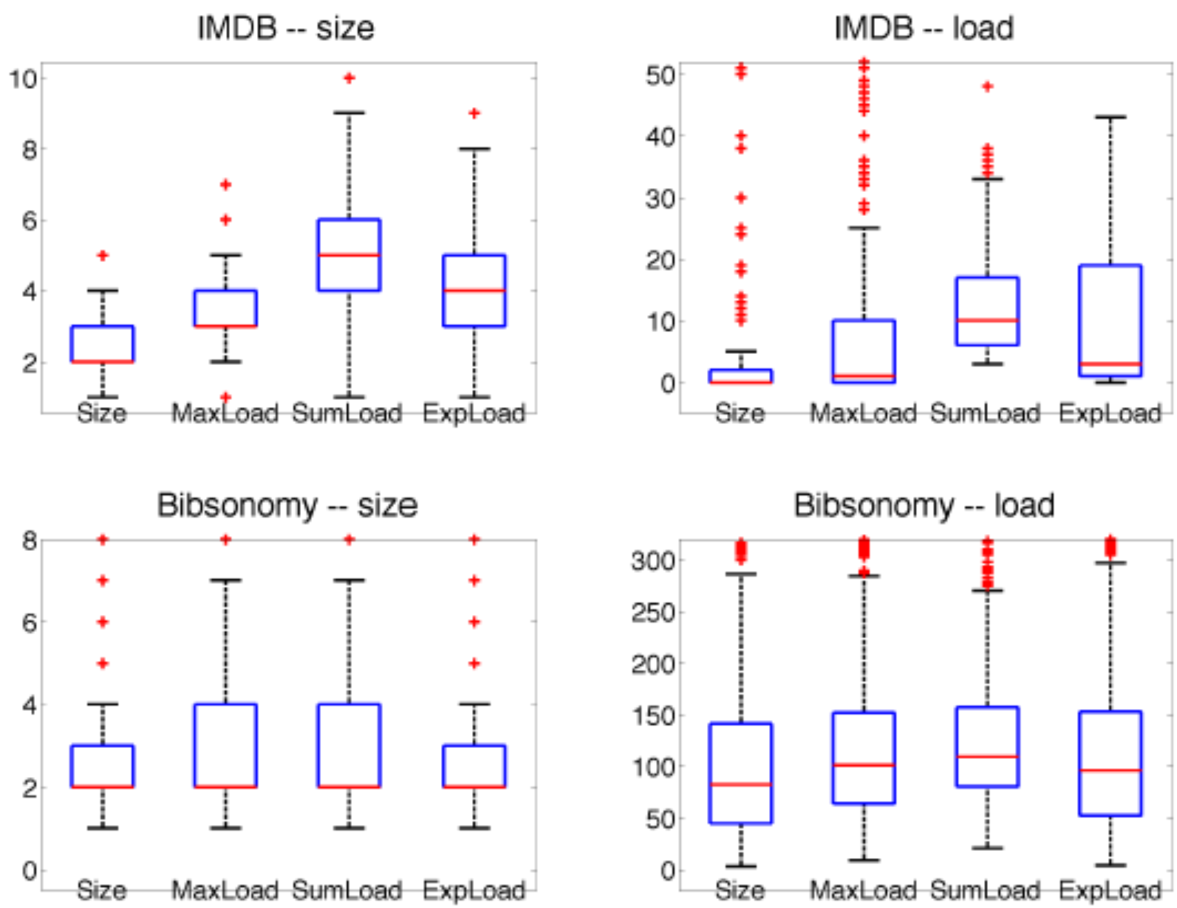
# Experiments

## Mapping of data to problem instances

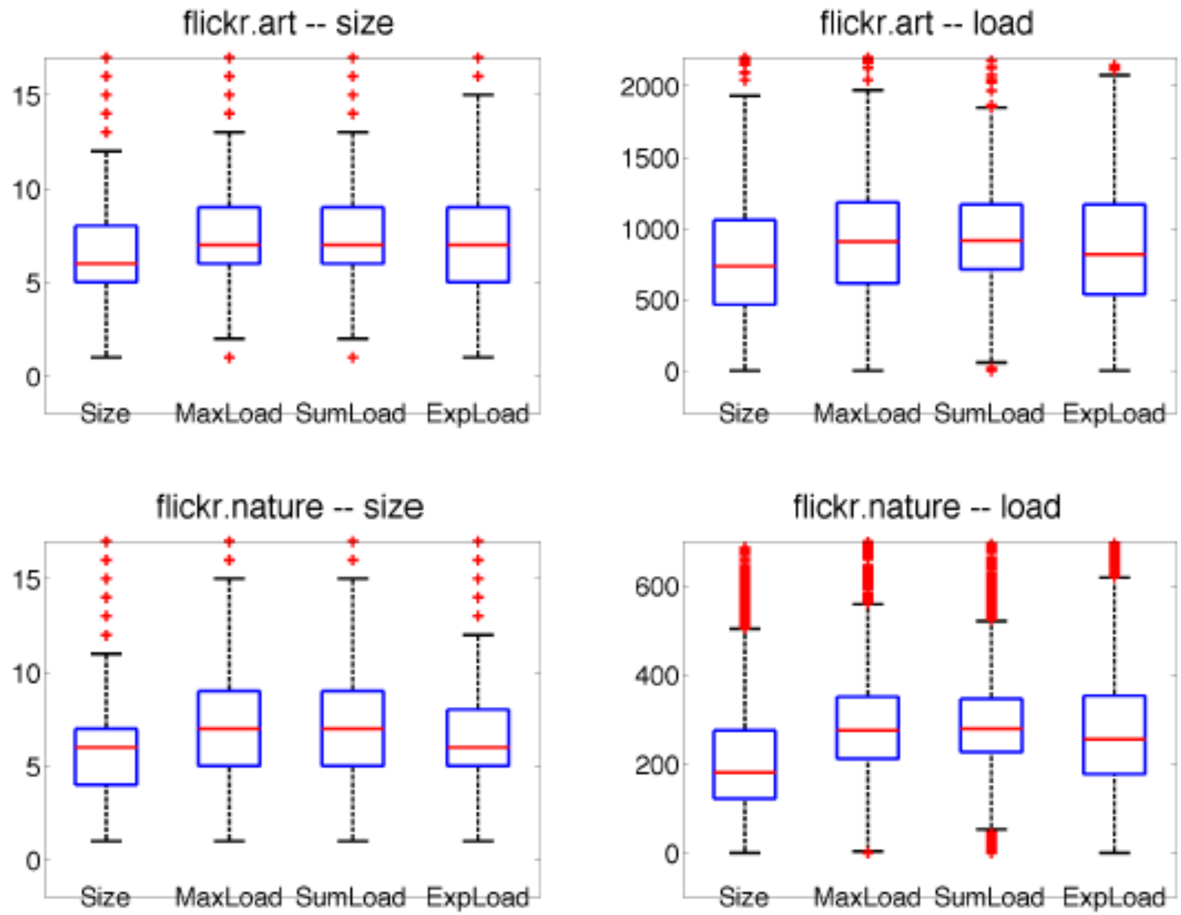
| Dataset | Experts | Tasks |
|---|---|---|
| IMDB | Movie directors | Audition actors |
| Bibsonomy | Prolific scientists | Interview scientists |
| Flickr | Prolific photographers | Judge photos |

## Summary statistics

| Dataset | Experts | Tasks | Skills | Skills/ expert | Skills/ task |
|---|---|---|---|---|---|
| IMDB | 725 | 2 173 | 21 | 2.96 | 11.10 |
| Bibsonomy | 816 | 35 506 | 793 | 7.64 | 4.44 |
| Flickr.art | 504 | 59 869 | 12 913 | 49.90 | 15.73 |
| Flickr.nature | 2 879 | 112 467 | 26 379 | 31.25 | 15.45 |



IMDB -- size
IMDB -- load
Bibsonomy -- size
Bibsonomy -- load
Size
MaxLoad
SumLoad
ExpLoad



flickr.art -- size
flickr.art -- load
flickr.nature -- size
flickr.nature -- load
Size
MaxLoad
SumLoad
ExpLoad

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.

| | Team size statistics | | | | Experts load statistics | | | |
|---|---|---|---|---|---|---|---|---|
| Method | mean | $\phi_{.9}$ | $\sigma_{.9}$ | max | mean | $\phi_{.9}$ | $\lambda_{.1}$ | max |
| **IMDB** | | | | | | | | |
| Size | 2.31 | 4 | 3 | 5 | 6.92 | 11 | 58 | 1260 |
| MaxLoad | 3.27 | 4 | 3 | 7 | 9.80 | 45 | 53 | 65 |
| SumLoad | 4.75 | 7 | 3 | 10 | 14.23 | 32 | 46 | 65 |
| ExpLoad | 3.80 | 5 | 3 | 9 | 11.38 | 32 | 47 | 64 |
| **Bibsonomy** | | | | | | | | |
| Size | 2.70 | 5 | 5 | 22 | 117.66 | 251 | 397 | 1417 |
| MaxLoad | 2.92 | 5 | 3 | 22 | 127.13 | 248 | 353 | 700 |
| SumLoad | 3.13 | 6 | 7 | 25 | 136.05 | 244 | 343 | 701 |
| ExpLoad | 2.83 | 5 | 4 | 22 | 123.27 | 258 | 365 | 700 |
| **Flickr.nature** | | | | | | | | |
| Size | 6.34 | 10 | 25 | 29 | 247.85 | 439 | 823 | 6645 |
| MaxLoad | 7.38 | 11 | 27 | 31 | 288.22 | 468 | 571 | 941 |
| SumLoad | 7.53 | 12 | 30 | 35 | 294.09 | 438 | 535 | 937 |
| ExpLoad | 7.08 | 11 | 28 | 34 | 276.60 | 475 | 587 | 964 |

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.

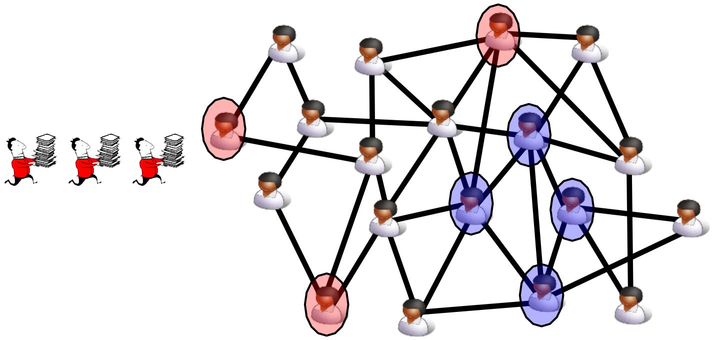
| | Team size statistics | | | | Experts load statistics | | | |
|---|---|---|---|---|---|---|---|---|
| Method | mean | $\phi_{.9}$ | $\sigma_{.9}$ | max | mean | $\phi_{.9}$ | $\lambda_{.1}$ | max |
| | | | | IMDB | | | | |
| Size | 2.31 | 4 | 3 | 5 | 6.92 | 11 | 58 | 1260 |
| MaxLoad | 3.27 | 4 | 3 | 7 | 9.80 | 45 | 53 | 65 |
| SumLoad | 4.75 | 7 | 3 | 10 | 14.23 | 32 | 46 | 65 |
| ExpLoad | 3.80 | 5 | 3 | 9 | 11.38 | 32 | 47 | 64 |
| | | | | Bibsonomy | | | | |
| Size | 2.70 | 5 | 5 | 22 | 117.66 | 251 | 397 | 1417 |
| MaxLoad | 2.92 | 5 | 3 | 22 | 127.13 | 248 | 353 | 700 |
| SumLoad | 3.13 | 6 | 7 | 25 | 136.05 | 244 | 343 | 701 |
| ExpLoad | 2.83 | 5 | 4 | 22 | 123.27 | 258 | 365 | 700 |
| | | | | Flickr.nature | | | | |
| Size | 6.34 | 10 | 25 | 29 | 247.85 | 439 | 823 | 6645 |
| MaxLoad | 7.38 | 11 | 27 | 31 | 288.22 | 468 | 571 | 941 |
| SumLoad | 7.53 | 12 | 30 | 35 | 294.09 | 438 | 535 | 937 |
| ExpLoad | 7.08 | 11 | 28 | 34 | 276.60 | 475 | 587 | 964 |

# Online Balanced Task Covering with Coordination Cost

# 2. Coordination cost

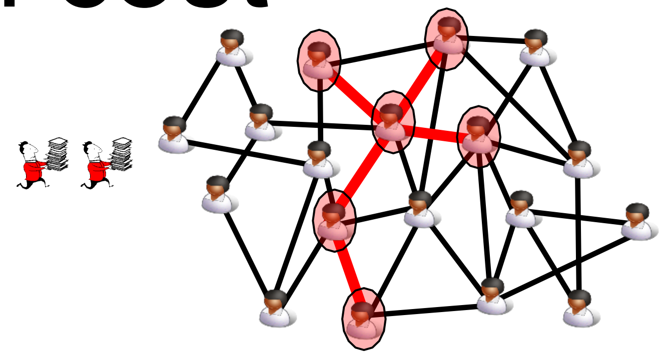
- Have not taken into account **coordination cost**

- Distance between people $d(\mathbf{p}^i, \mathbf{p}^j)$
- Team coordination cost $c(Q^j)$
- Select teams that minimizes $c(Q^j)$
  - Steiner-tree cost
  - Diameter
  - Sum of distances

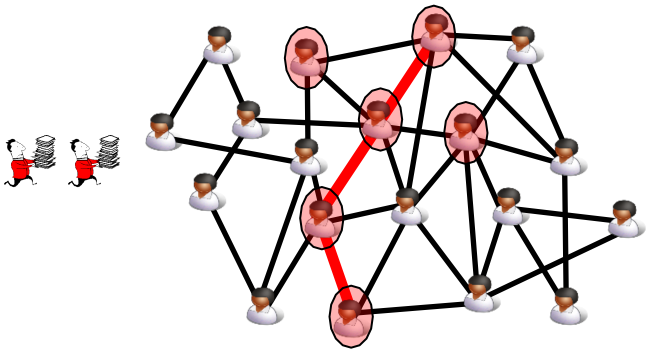


# Coordination cost

- Steiner-tree cost



- Diameter



- Sum of distances

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

# Conflicting goals

- We want solutions that minimize
  - **Load**
  - **Coordination cost**

and satisfy each job.

# Our modeling approach

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

$$\begin{aligned} &\min \max_i L(\mathbf{p}^i) \\ &s(\mathbf{J}^j, Q^j) = 1 \qquad \forall j \in \mathcal{J} \\ &c(Q^j) \leq B \qquad \forall j \in \mathcal{J}. \end{aligned}$$

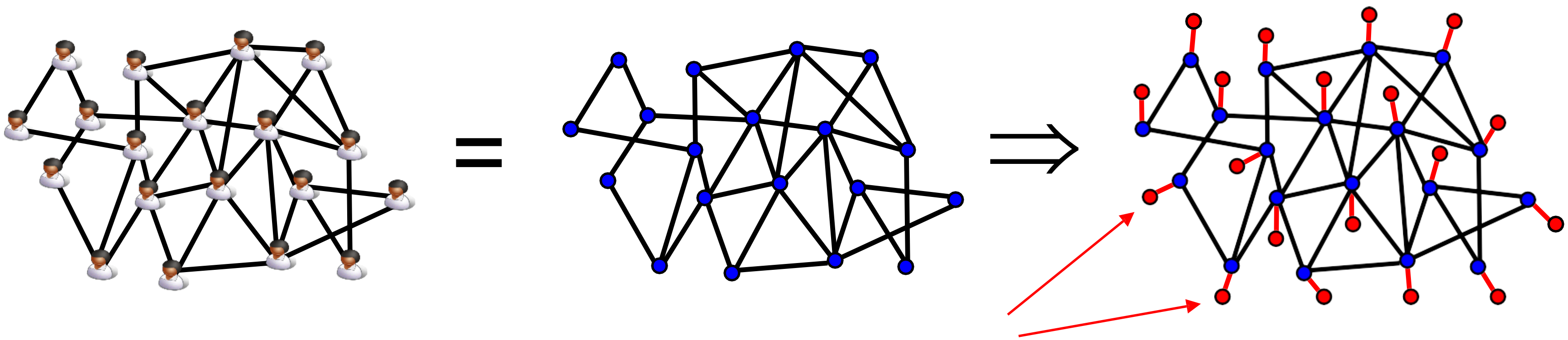
- 3 different problems for the 3 different coordination costs
- This talk: focus on Steiner tree coordination cost

# 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 $\sum_{\mathbf{p}\in Q} (2n)^{L_t(\mathbf{p})}$
- How?

# At every step t



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

# Result

We wanted:

$$\min_i \max L(\mathbf{p}^i)$$
$$s(\mathbf{J}^j, Q^j) = 1 \qquad \forall j \in \mathcal{J}$$
$$c(Q^j) \leq B \qquad \forall j \in \mathcal{J}.$$

**Theorem.** The algorithm satisfies:

$\alpha$-approximates

$$\min_i \max L(\mathbf{p}^i)$$
$$s(\mathbf{J}^j, Q^j) = 1 \qquad \forall j \in \mathcal{J}$$
$$c(Q^j) \leq \beta B \qquad \forall j \in \mathcal{J}.$$

- Can obtain ***α***, ***β*** = 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 $\mathbf{J}_l$ 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 $\mathbf{p}^j$
cost-effectiveness: $\frac{\text{gain}(\mathbf{p}^j)}{\text{loss}(\mathbf{p}^j)}$

$\text{gain}(\mathbf{p}^j)$ # newly covered skills

$\text{loss}(\mathbf{p}^j)$ distance to experts selected so far plus $\lambda$ * ExpLoad of the expert

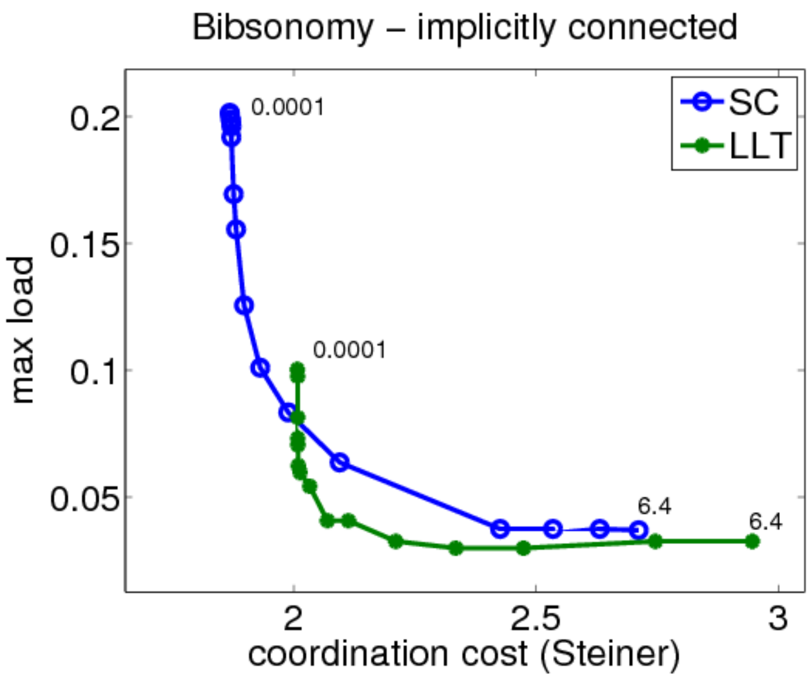
# Experiments Bibsonomy

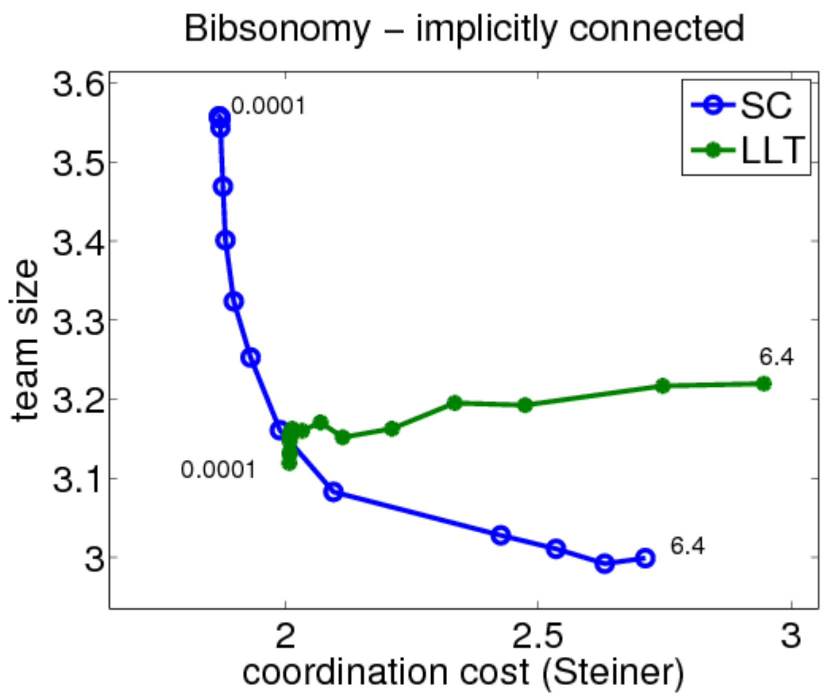
Experts = prolific authors

Task = interview scientists

Distance = f( #collaborations )

Optimize over $\lambda$



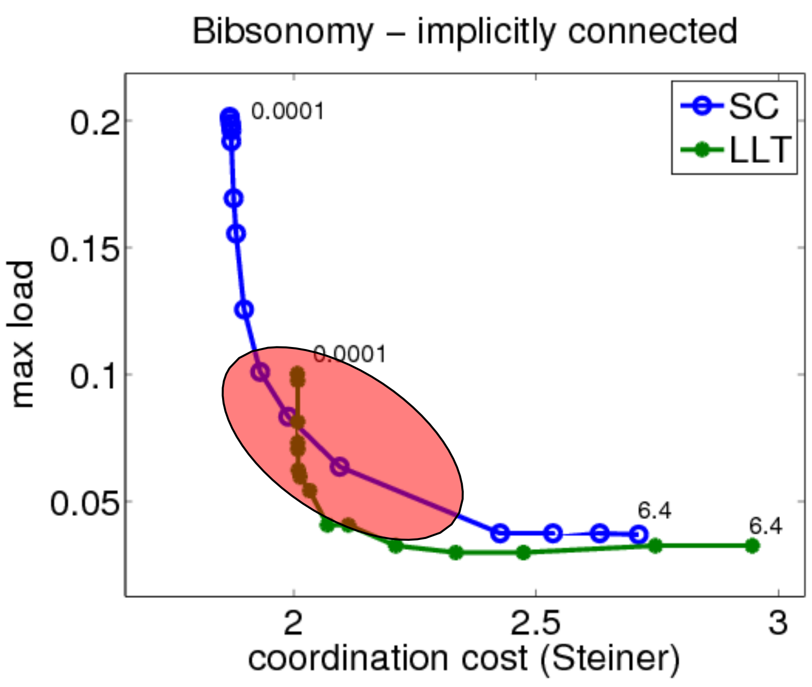



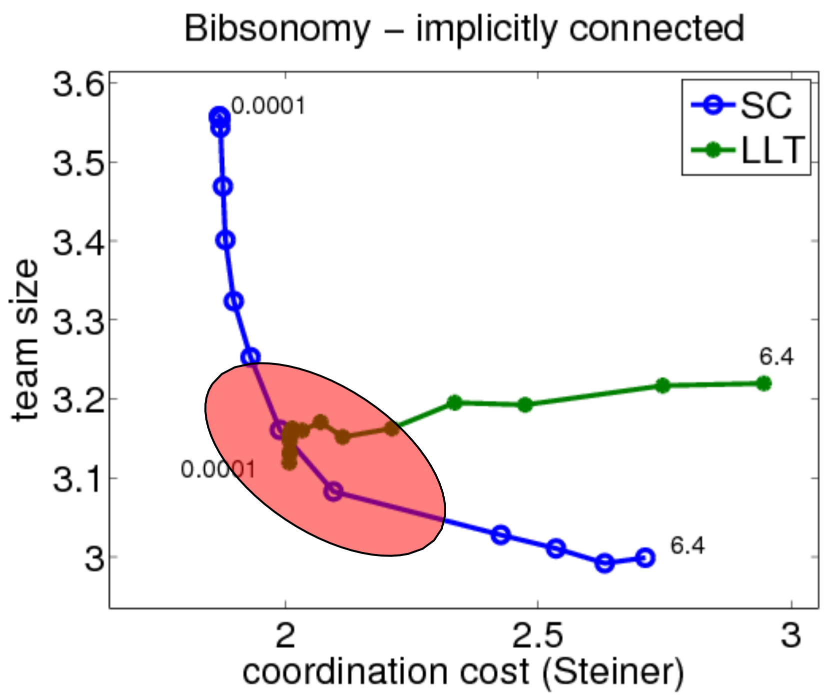
# Experiments Bibsonomy

Experts = prolific authors

Task = interview scientists

Distance = f( #collaborations )



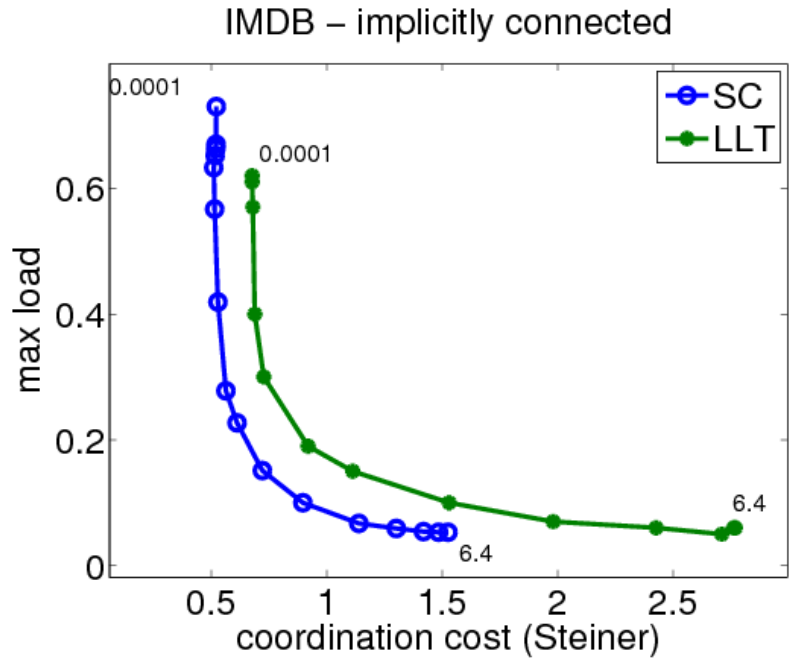



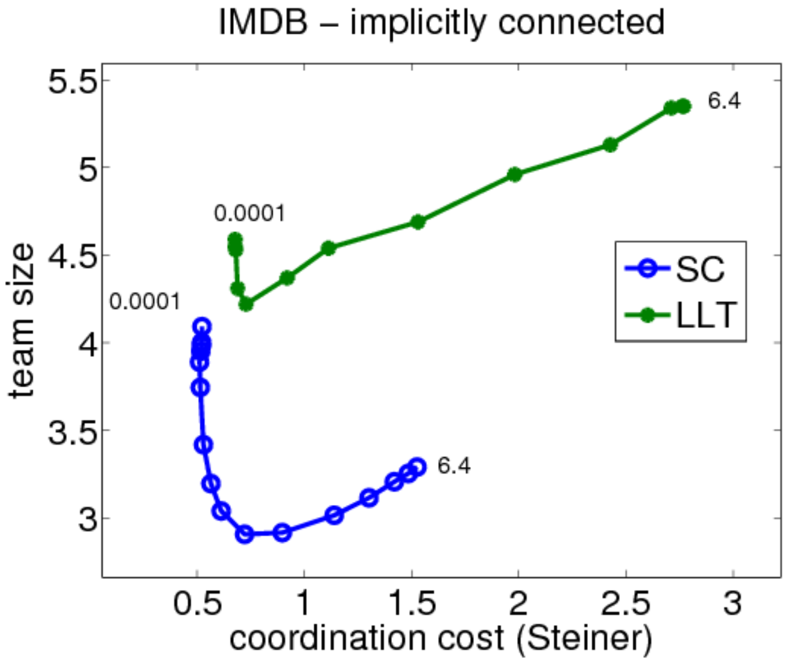
# Experiments IMDB

Experts = directors

Task = find a cast

Distance = f( #common actors directed )



# Online Team Formation with Outsourcing

# Team Formation with Outsourcing

- Create teams of workers for solving tasks/jobs that arrive online
- Tasks and workers are represented as a set of skills
- At each time step a new task arrives
- A team must be created to cover all the task skills
- Each member of the team can be either hired or a freelance worker
- Each worker $w_r$ has a hiring ($C_r$), salary ($\sigma_r$), and outsourcing ($\lambda_r$) cost
- **Goal:** design an online, cost-minimizing algorithm for hiring, firing, and outsourcing.

# Team Formation with Outsourcing

- Create teams of workers for solving tasks/jobs that arrive online
- Tasks and workers are represented as a set of skills
- At each time step a new task arrives
- A team must be created to cover all the task skils
- Each member of the team can be either hired or a freelance worker
- Each worker $w_r$ has a hiring ($C_r$), salary ($\sigma_r$), and outsourcing ($\lambda_r$) cost
- **Goal:** design an online, cost-minimizing algorithm for hiring, firing, and outsourcing.

$w_1 = \{s_1\}$
$w_2 = \{s_2, s_4\}$
$w_3 = \{s_2, s_3\}$

# Team Formation with Outsourcing

- Create teams of workers for solving tasks/jobs that arrive online
- Tasks and workers are represented as a set of skills
- At each time step a new task arrives
- A team must be created to cover all the task skils
- Each member of the team can be either hired or a freelance worker
- Each worker $w_r$ has a hiring ($C_r$), salary ($\sigma_r$), and outsourcing ($\lambda_r$) cost
- **Goal:** design an online, cost-minimizing algorithm for hiring, firing, and outsourcing.

$w_1=\{s_1\}$
$w_2=\{s_2, s_4\}$
$w_3=\{s_2, s_3\}$


$J^1=\{s_1, s_2\}$

# Team Formation with Outsourcing

- Create teams of workers for solving tasks/jobs that arrive online
- Tasks and workers are represented as a set of skills
- At each time step a new task arrives
- A team must be created to cover all the task skils
- Each member of the team can be either hired or a freelance worker
- Each worker $w_r$ has a hiring ($C_r$), salary ($\sigma_r$), and outsourcing ($\lambda_r$) cost
- **Goal:** design an online, cost-minimizing algorithm for hiring, firing, and outsourcing.

$w_1=\{s_1\}$

$w_2=\{s_2, s_4\}$

$w_3=\{s_2, s_3\}$


$J^1=\{s_1, s_2\}$

*Hired:* —

*Outsourced:* $w_1, w_2$

*Cost:* $\lambda_1+\lambda_2$

# Team Formation with Outsourcing

- Create teams of workers for solving tasks/jobs that arrive online
- Tasks and workers are represented as a set of skills
- At each time step a new task arrives
- A team must be created to cover all the task skils
- Each member of the team can be either hired or a freelance worker
- Each worker $w_r$ has a hiring ($C_r$), salary ($\sigma_r$), and outsourcing ($\lambda_r$) cost
- **Goal:** design an online, cost-minimizing algorithm for hiring, firing, and outsourcing.

$w_1=\{s_1\}$
$w_2=\{s_2, s_4\}$
$w_3=\{s_2, s_3\}$


$J^1=\{s_1, s_2\}$ $J^2=\{s_1, s_3\}$

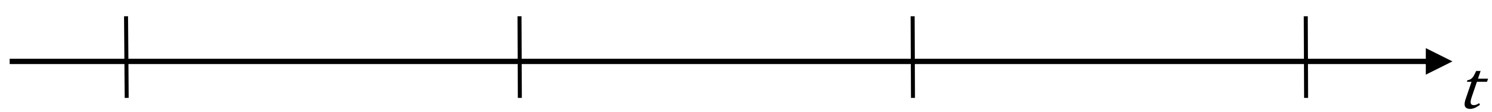
*Hired:* –

*Outsourced:* $w_1, w_2$

*Cost:* $\lambda_1+\lambda_2$

# Team Formation with Outsourcing

- Create teams of workers for solving tasks/jobs that arrive online
- Tasks and workers are represented as a set of skills
- At each time step a new task arrives
- A team must be created to cover all the task skils
- Each member of the team can be either hired or a freelance worker
- Each worker $w_r$ has a hiring ($C_r$), salary ($\sigma_r$), and outsourcing ($\lambda_r$) cost
- **Goal:** design an online, cost-minimizing algorithm for hiring, firing, and outsourcing.




$w_1=\{s_1\}$
$w_2=\{s_2, s_4\}$
$w_3=\{s_2, s_3\}$

| | $J^1=\{s_1, s_2\}$ | $J^2=\{s_1, s_3\}$ |
|---|---|---|
| *Hired:* | – | $w_1$ |
| *Outsourced:* | $w_1, w_2$ | $w_3$ |
| *Cost:* | $\lambda_1+\lambda_2$ | $C_1 + \sigma_1 + \lambda_3$ |

# Team Formation with Outsourcing

- Create teams of workers for solving tasks/jobs that arrive online
- Tasks and workers are represented as a set of skills
- At each time step a new task arrives
- A team must be created to cover all the task skils
- Each member of the team can be either hired or a freelance worker
- Each worker $w_r$ has a hiring ($C_r$), salary ($\sigma_r$), and outsourcing ($\lambda_r$) cost
- **Goal:** design an online, cost-minimizing algorithm for hiring, firing, and outsourcing.

$w_1=\{s_1\}$
$w_2=\{s_2, s_4\}$
$w_3=\{s_2, s_3\}$

$t$

| | $J^1=\{s_1, s_2\}$ | $J^2=\{s_1, s_3\}$ | $J^3=\{s_2, s_3\}$ |
|---|---|---|---|
| *Hired:* | – | $w_1$ | |
| *Outsourced:* | $w_1, w_2$ | $w_3$ | |
| *Cost:* | $\lambda_1+\lambda_2$ | $C_1 + \sigma_1 + \lambda_3$ | |

# Team Formation with Outsourcing

- Create teams of workers for solving tasks/jobs that arrive online
- Tasks and workers are represented as a set of skills
- At each time step a new task arrives
- A team must be created to cover all the task skils
- Each member of the team can be either hired or a freelance worker
- Each worker $w_r$ has a hiring ($C_r$), salary ($\sigma_r$), and outsourcing ($\lambda_r$) cost
- **Goal:** design an online, cost-minimizing algorithm for hiring, firing, and outsourcing.

$w_1=\{s_1\}$
$w_2=\{s_2, s_4\}$
$w_3=\{s_2, s_3\}$

| | | | | |
|---|---|---|---|---|
| | $J^1=\{s_1, s_2\}$ | $J^2=\{s_1, s_3\}$ | $J^3=\{s_2, s_3\}$ | $t$ |
| Hired: | – | $w_1$ | $w_1, w_3$ | |
| Outsourced: | $w_1, w_2$ | $w_3$ | – | |
| Cost: | $\lambda_1 + \lambda_2$ | $C_1 + \sigma_1 + \lambda_3$ | $\sigma_1 + C_3 + \sigma_3$ | |

# Team Formation with Outsourcing

- Create teams of workers for solving tasks/jobs that arrive online
- Tasks and workers are represented as a set of skills
- At each time step a new task arrives
- A team must be created to cover all the task skils
- Each member of the team can be either hired or a freelance worker
- Each worker $w_r$ has a hiring ($C_r$), salary ($\sigma_r$), and outsourcing ($\lambda_r$) cost
- **Goal:** design an online, cost-minimizing algorithm for hiring, firing, and outsourcing.

$w_1=\{s_1\}$
$w_2=\{s_2, s_4\}$
$w_3=\{s_2, s_3\}$

$t$

| | $J^1=\{s_1, s_2\}$ | $J^2=\{s_1, s_3\}$ | $J^3=\{s_2, s_3\}$ | $J^4=\{s_2, s_3, s_4\}$ |
|---|---|---|---|---|
| Hired: | – | $w_1$ | $w_1, w_3$ | |
| Outsourced: | $w_1, w_2$ | $w_3$ | – | |
| Cost: | $\lambda_1+\lambda_2$ | $C_1+\sigma_1+\lambda_3$ | $\sigma_1+C_3+\sigma_3$ | |

# Team Formation with Outsourcing

- Create teams of workers for solving tasks/jobs that arrive online
- Tasks and workers are represented as a set of skills
- At each time step a new task arrives
- A team must be created to cover all the task skils
- Each member of the team can be either hired or a freelance worker
- Each worker $w_r$ has a hiring ($C_r$), salary ($\sigma_r$), and outsourcing ($\lambda_r$) cost
- **Goal:** design an online, cost-minimizing algorithm for hiring, firing, and outsourcing.

$w_1=\{s_1\}$
$w_2=\{s_2, s_4\}$
$w_3=\{s_2, s_3\}$

| | $J^1=\{s_1, s_2\}$ | $J^2=\{s_1, s_3\}$ | $J^3=\{s_2, s_3\}$ | $J^4=\{s_2, s_3, s_4\}$ |
|---|---|---|---|---|
| Hired: | – | $w_1$ | $w_1, w_3$ | $w_3$ |
| Outsourced: | $w_1, w_2$ | $w_3$ | – | $w_2$ |
| Cost: | $\lambda_1+\lambda_2$ | $C_1 + \sigma_1 + \lambda_3$ | $\sigma_1 + C_3 + \sigma_3$ | $\sigma_3 + \lambda_2$ |

$t$

# Quality of online algorithms

**Goal**:

Design a polynomial-time online algorithm for the TFO problem with a small competitive approximation ratio.

Competitive approximation ratio of an online algorithm:

$$\max_{Stream\ of\ Tasks} \frac{Cost\ of\ the\ Algorithm}{Cost\ of\ an\ Optimal\ Algorithm}$$

# Methodology

1. **TFO-LumpSum**: no salary and no firing.
   Design a polynomial time online algorithm with a logarithmic competitive approximation ratio.

2. **TFO**: full version.
   Design a polynomial time online algorithm with a logarithmic competitive approximation ratio, by modifying the algorithm for TFO-LumpSum.

# Methodology

1. **TFO-LumpSum**: no salary and no firing.
   Design a polynomial time online algorithm with a logarithmic competitive approximation ratio.

2. **TFO**: full version.
   Design a polynomial time online algorithm with a logarithmic competitive approximation ratio, by modifying the algorithm for TFO-LumpSum.

# TFO-LumpSum: Online primal–dual technique

- $x_r = 1$ if worker $W^r$ is hired, 0 otherwise.
- $f_{rt} = 1$ if worker $W^r$ is outsourced for performing task $J^t$, 0 otherwise.

Linear program for LumpSum:

$$\min \sum_{r=1}^{n} \left( C_r x_r + \lambda_r \sum_{t=1}^{T} f_{rt} \right)$$

subject to: $\forall t = 1, \ldots, T, \ell \in J^t$ :

$$\sum_{W^r \in P_\ell} (x_r + f_{rt}) \geq 1$$

$\forall t = 1, \ldots, T, r = 1, \ldots, n$:

$$x_r, f_{rt} \geq 0$$

| | |
|---|---|
| $C_r$ | Hiring fee, paid when worker $r$ is hired. |
| $\lambda_r$ | Outsourcing fee, paid every time $r$ performs a task. |

# TFO-LumpSum: Online primal–dual technique

- $x_r = 1$ if worker $W^r$ is hired, 0 otherwise.
- $f_{rt} = 1$ if worker $W^r$ is outsourced for performing task $J^t$, 0 otherwise.

Linear program for LumpSum:

$$\min \sum_{r=1}^{n} \left( C_r x_r + \lambda_r \sum_{t=1}^{T} f_{rt} \right)$$

subject to: $\forall t = 1, \ldots, T, \ell \in J^t$ :

$$\sum_{W^r \in P_\ell} (x_r + f_{rt}) \geq 1$$

$\forall t = 1, \ldots, T, r = 1, \ldots, n$:

$$x_r, f_{rt} \geq 0$$

The dual of the linear program for LumpSum:

$$\max \sum_{t=1}^{T} \sum_{\ell \in J^t} u_{\ell t}$$

subject to: $\forall r = 1, \ldots, n$:

$$\sum_{t=1}^{T} \sum_{\ell \in J^t \cap W^r} u_{\ell t} \leq C_r$$

$\forall t = 1, \ldots, T, r = 1, \ldots, n$:

$$\sum_{\ell \in J^t \cap W^r} u_{\ell t} \leq \lambda_r$$

$\forall t = 1, \ldots, T, \ell \in J^t$:

$$u_{\ell t} \geq 0,$$

| | |
|---|---|
| $C_r$ | Hiring fee, paid when worker $r$ is hired. |
| $\lambda_r$ | Outsourcing fee, paid every time $r$ performs a task. |

# TFO-LumpSum: Algorithm

When job $J^T$ arrives:

Step 1: Increase potentials:

**for each** skill $\ell \in J^T_{\mathcal{F}}$:

  **while** $\sum_{W^r \in P_\ell} \left(\tilde{x}_r + \tilde{f}_{rT}\right) < 1$:

   $u_{\ell t} \leftarrow u_{\ell t} + 1$

   **for each** $W^r \in P_\ell$: $\tilde{x}_r \leftarrow \tilde{x}_r \left(1 + \frac{1}{C_r}\right) + \frac{1}{nC_r}$

   **for each** $W^r \in P_\ell$: $\tilde{f}_{rT} \leftarrow \tilde{f}_{rT} \left(1 + \frac{1}{\lambda_r}\right) + \frac{1}{n\lambda_r}$

Step 2: Perform randomized rounding to decide which worker to hire and to whom to outsource

**repeat** $\rho$ times:

  **for each** $W^r \in P_T^{\mathcal{F}}$

   with probability $\Delta\tilde{x}_r$:

    hire worker $W^r$ (set $x_r \leftarrow 1$)

   with probability $\tilde{f}_{rT}$:

    outsource worker $W^r$ (set $f_{rT} \leftarrow 1$)

*n:* *total number of workers.*

*m:* *total number of skills.*

*C*:* *maximum hiring cost.*

Running time:

$$O\left(n\left(|J^T|\log n + \log m + \log C^*\right)\right)$$

Competitive approximation ratio:

$$O(\log n(\log m + \log C^*))$$

# Methodology

1. **TFO-LumpSum**: no salary and no firing.
   Design a polynomial time online algorithm with a logarithmic competitive approximation ratio.

2. **TFO**: full version.
   Design a polynomial time online algorithm with a logarithmic competitive approximation ratio, by modifying the algorithm for TFO-LumpSum.

# TFO

**Theorem.** There exists a a polynomial time online algorithm for TFO with competitive approximation ratio

$$O((\log m + \log C^* + \log T^*) \log n)$$

**Proof.** Use online primal–dual schema with a more complicated set of integer and linear programs.

*m: total number of skills.*
*C*: maximum hiring cost.*
*T*: number of tasks in the stream.*
*n: total number of workers.*

# TFO

**Theorem.** There exists a a polynomial time online algorithm for TFO with competitive approximation ratio

$$O((\log m + \log C^* + \log T^*) \log n)$$

**Proof.** Use online primal–dual schema with a more complicated set of integer and linear programs.

Linear program for TFO:

$$\min \sum_{r=1}^{n} \left[ \sum_{I \in \mathcal{I}} C_r x(r, I) + \sum_{t=1}^{T} \lambda_r f_{rt} + \sum_{t=1}^{T} \sigma_r g_{rt} \right]$$

subject to

$\forall t = 1 \ldots T, \ell \in J^t$ :

$$\sum_{W^r \in P_\ell} \left( f_{rt} + \sum_{I \in \mathcal{I} : t \in I} x(r, I) \right) \geq 1.$$

$\forall t = 1 \ldots T, r = 1 \ldots n$ :

$$\sum_{I \in \mathcal{I} : t \in I} x(r, I) \leq g_{rt}$$

$\forall t = 1 \ldots T, r = 1 \ldots n, I \in \mathcal{I}$ :

$$x(r, I), f_{rt}, g_{rt} \geq 0$$

*m: total number of skills.*
*C*: maximum hiring cost.*
*T*: number of tasks in the stream.*
*n: total number of workers.*

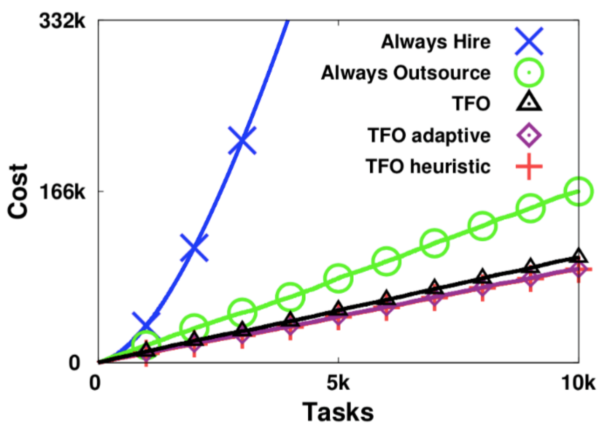
# Experiments: Datasets

| Dataset | upwork | freelancer | guru |
|---|---|---|---|
| Skills ($m$) | 2,335 | 175 | 1,639 |
| Workers ($n$) | 18,000 | 1,211 | 6,119 |
| Tasks ($T$) | *50,000* | 992 | 3,194 |
| ... distinct | *50,000* | 600 | 2,939 |
| ... avg. similarity (Jaccard) | *0.095* | 0.045 | 0.018 |
| Average Skills/worker | 6.29 | 1.45 | 13.07 |
| Average Skills/task | *41.88* | 2.86 | 5.24 |

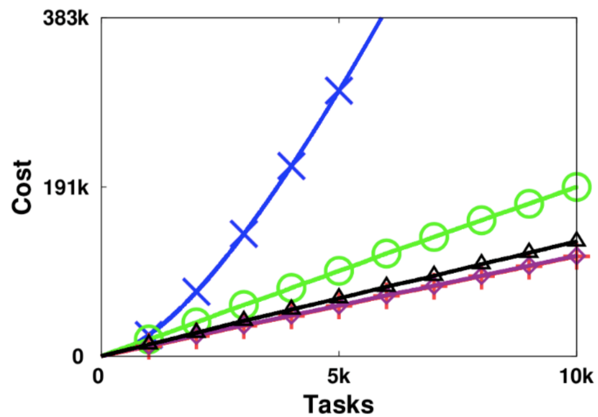
Generation of the stream of tasks:

- Pick a random task as pivot.
- With probability ***1-1/p***, pick the next task within those whose Jaccard similarity with the pivot is at least 0.5.
- With probability ***1/p***, pick another random task as a pivot.

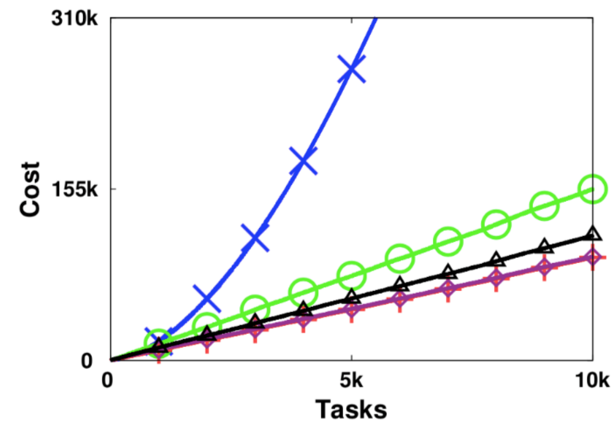
# Experiments: TFO vs. Heuristics




(b) TFO UpWork




(d) TFO Freelancer



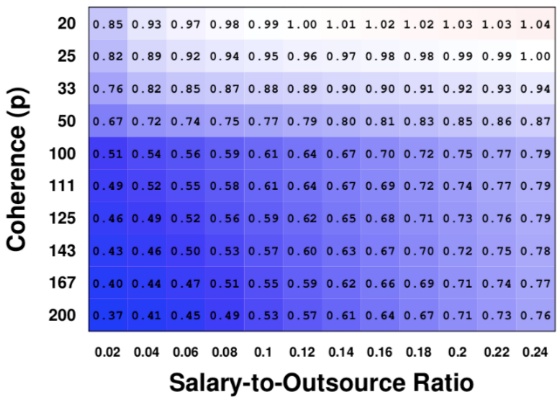

(f) TFO Guru

$$C_r = 4\lambda_r \quad \sigma_r = \lambda_r/10 \qquad p = 100$$

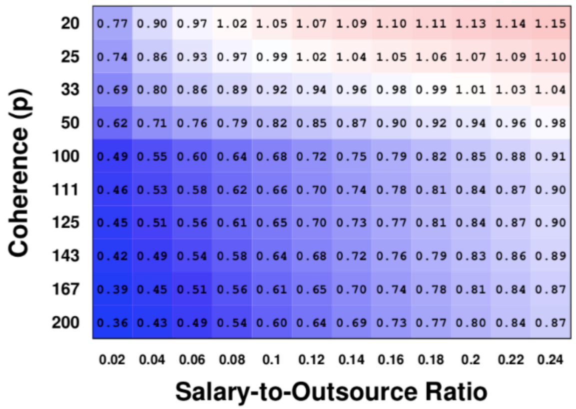
Generation of the stream of tasks:

- Pick a random task as pivot.
- With probability ***1-1/100***, pick the next task within those whose Jaccard similarity with the pivot is at least 0.5.
- With probability ***1/100***, pick another random task as a pivot.

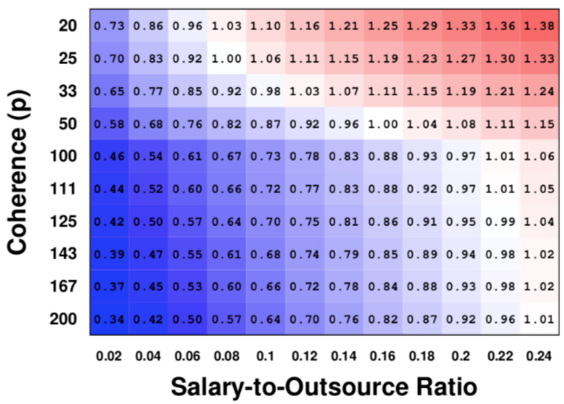
# Experiments: TFO vs. Always Outsource




(a) UpWork: TFO vs. Always-Outsource

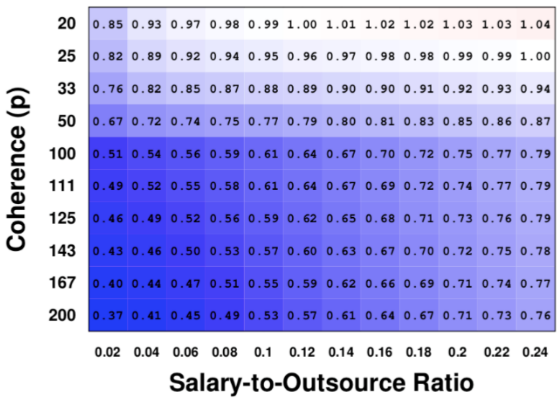



(c) Freelancer: TFO vs. Always-Outsource

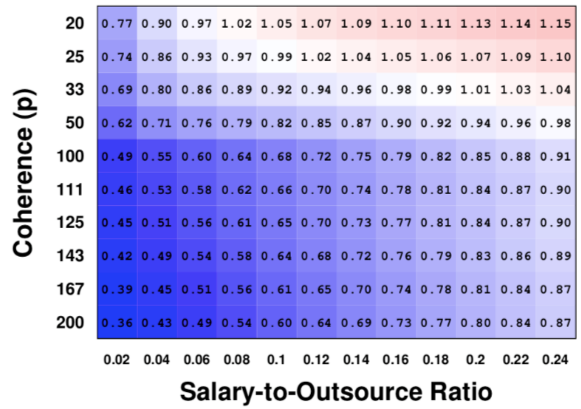



(e) Guru: TFO vs. Always-Outsource

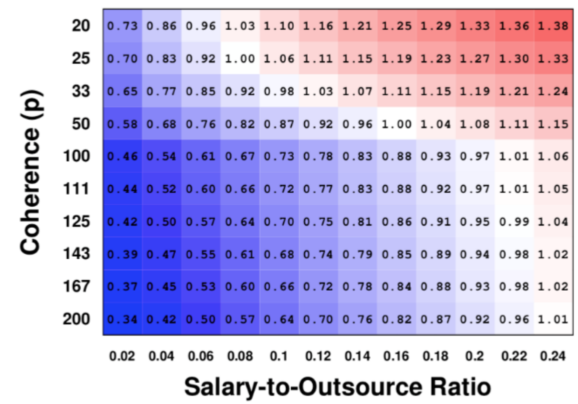
# Experiments: TFO vs. Always Outsource



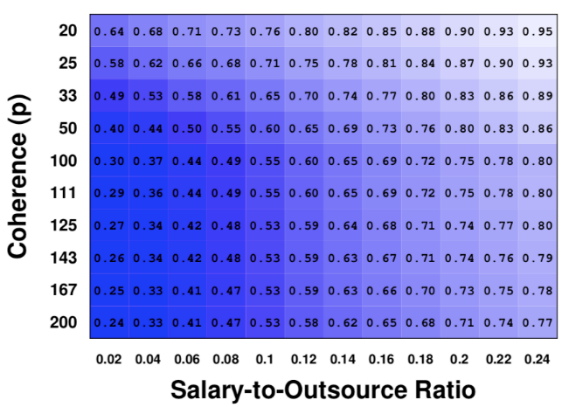

(a) UpWork: TFO vs. Always-Outsource



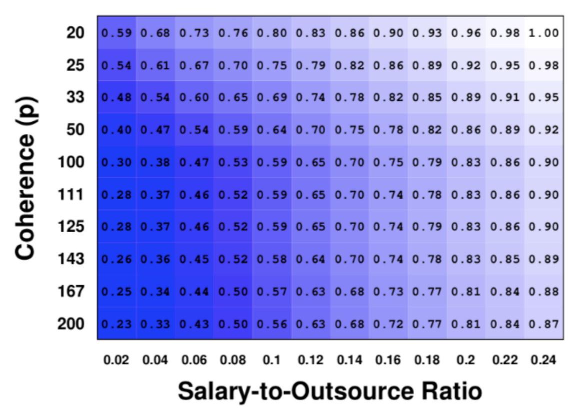

(c) Freelancer: TFO vs. Always-Outsource



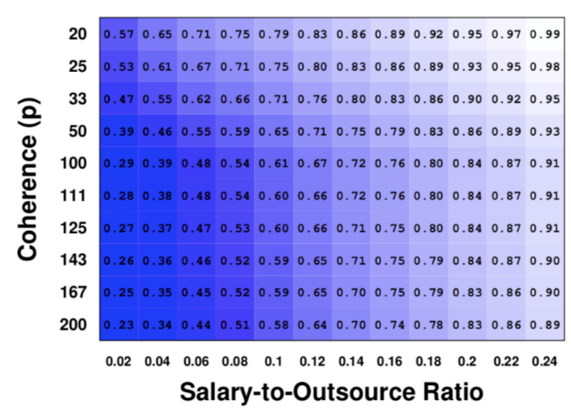

(e) Guru: TFO vs. Always-Outsource




(b) UpWork: TFO-Adaptive vs. Always-Outsource




(d) Freelancer: TFO-Adaptive vs. Always-Outsource




(f) Guru: TFO-Adaptive vs. Always-Outsource

# Conclusions

- Defined a novel online team formation problem in a hire-or-outsource setting
- Designed polynomial-time online algorithms with competitive approximation ratios
- Shown the applicability of our algorithmic solutions, by performing experiments using data from online outsourcing marketplaces
- Showed the practical use of the online primal–dual schema

**Future work:**

- Relax/test some of the modeling assumptions
- k-TFO: # of hired workers can be at most a fixed number k

# Future directions

**Modeling**

- Several human elements: capabilities, cooperation, etc.
- Application dependent

**Learning**

- Learning profiles of experts
- Learn coordination based on performance

**Algorithmic**

- Matching problems
- How to train experts
- Explore-exploit tradeoff

# Future directions

**Game-theoretic**

- Incentives for participation and rewarding mechanisms
- Issues on cooperation / altruism / trust

# Thanks!

**Questions, comments, etc.:**

**Stefano:** http://www.dis.uniroma1.it/~leon