

Quality Assurance for Human Computation Based Recommendation

Master Defense Presentation

Michael Schwarz

June 13, 2017

Graz University of Technology

Introduction

Motivation

- **Recommender systems** are ubiquitous

Motivation

- **Recommender systems** are ubiquitous
- These systems are usually based on **knowledge**

Motivation

- **Recommender systems** are ubiquitous
- These systems are usually based on **knowledge**
- Reliable but **expensive** if entered by small number of **experts**

Motivation

- **Recommender systems** are ubiquitous
- These systems are usually based on **knowledge**
- Reliable but **expensive** if entered by small number of **experts**
- **Unreliable** but cheap if entered by regular **users**

Motivation

- **Recommender systems** are ubiquitous
- These systems are usually based on **knowledge**
- Reliable but **expensive** if entered by small number of **experts**
- **Unreliable** but cheap if entered by regular **users**
- **Combine approaches** to reliably and cheaply collect knowledge

- **Design** and **implement** a web-based generic recommender platform

- **Design** and **implement** a web-based generic recommender platform
- Add mechanisms to **collect data** from regular users

- **Design** and **implement** a web-based generic recommender platform
- Add mechanisms to **collect data** from regular users
- Develop techniques to ensure the **quality** of the collected data

- **Design** and **implement** a web-based generic recommender platform
- Add mechanisms to **collect data** from regular users
- Develop techniques to ensure the **quality** of the collected data
- Efficiently **distribute** tasks to users to improve the knowledge base

Recommender Systems

Recommender Systems

- There are different **types** of recommender systems

Recommender Systems

- There are different **types** of recommender systems
- They all **recommend** products/items...

Recommender Systems

- There are different **types** of recommender systems
- They all **recommend** products/items...
- ...but use different **techniques** to find the best item(s)

Recommender Systems

- There are different **types** of recommender systems
- They all **recommend** products/items...
- ...but use different **techniques** to find the best item(s)
- Three types of systems are **commonly** used

Content-based Systems



- Collect information about the **items** (e.g., keywords)

Content-based Systems



- Collect information about the **items** (e.g., keywords)
- Find items similar to ones the user liked in the past

Content-based Systems

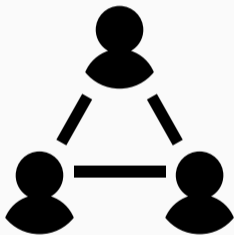


- Collect information about the **items** (e.g., keywords)
- Find items similar to ones the user liked in the past
- Idea: user preferences do not change

Content-based Systems

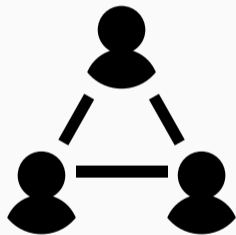


- Collect information about the **items** (e.g., keywords)
- Find items similar to ones the user liked in the past
- Idea: user preferences do not change
- Advantage: independent of other users



- Collect information about the **user**

Collaborative Filtering Systems



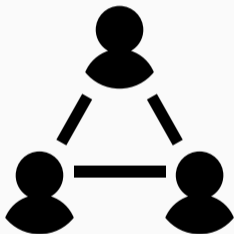
- Collect information about the **user**
- Find similar users

Collaborative Filtering Systems

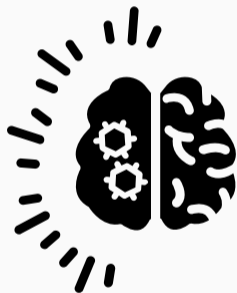


- Collect information about the **user**
- Find similar users
- Idea: people who liked the same things will like the same in the future

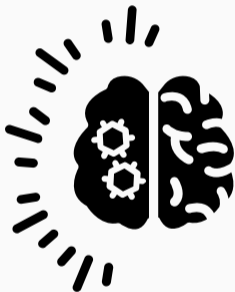
Collaborative Filtering Systems



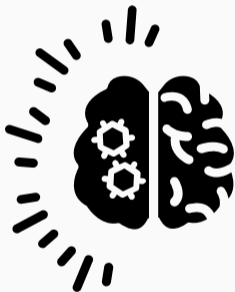
- Collect information about the **user**
- Find similar users
- Idea: people who liked the same things will like the same in the future
- Advantage: no understanding of the items necessary



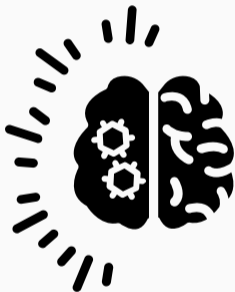
- Explicit information about the **items** and **user**



- Explicit information about the **items** and **user**
- Find items that fulfill the user-given constraints



- Explicit information about the **items** and **user**
- Find items that fulfill the user-given constraints
- Idea: recommendation boils down to a constraint satisfaction problem



- Explicit information about the **items** and **user**
- Find items that fulfill the user-given constraints
- Idea: recommendation boils down to a constraint satisfaction problem
- Advantage: no history of the user is necessary

A Generic Framework

Web-based Client-Server Model

- Subdivided into **frontend** and **backend**

Web-based Client-Server Model

- Subdivided into **frontend** and **backend**
- Backend is based on the Spring Framework (**Java**)

Web-based Client-Server Model

- Subdivided into **frontend** and **backend**
- Backend is based on the Spring Framework (**Java**)
- Frontend is mobile-friendly **HTML5**

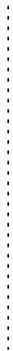
Web-based Client-Server Model

- Subdivided into **frontend** and **backend**
- Backend is based on the Spring Framework (**Java**)
- Frontend is mobile-friendly **HTML5**
- Parts are **loosely coupled**

Message Passing

Client

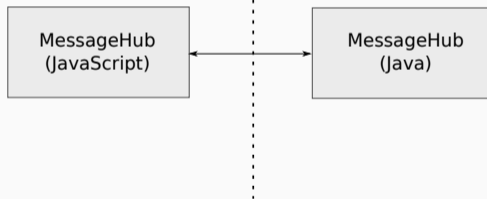
Server



Message Passing

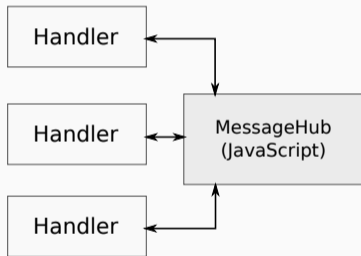
Client

Server

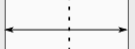
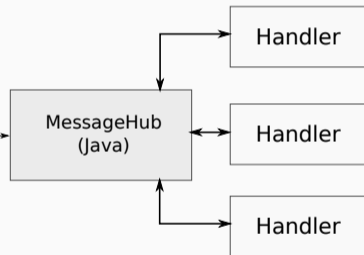


Message Passing

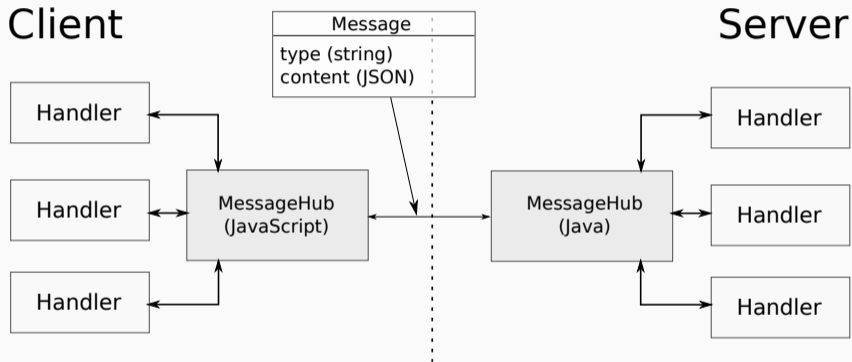
Client



Server



Message Passing



Message Format

Message to register a new user

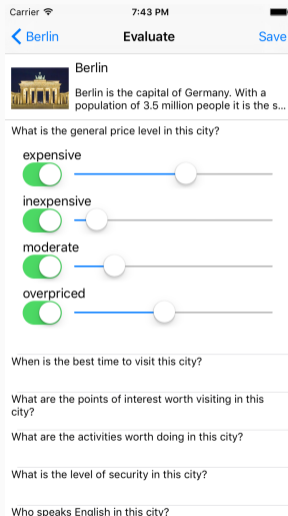
```
1 {  
2   type: "register",  
3   content : {  
4     username: "michael",  
5     password: "12345678",  
6     email: "michael.schwarz@noreply.com"  
7   }  
8 }
```

Multiple Frontends

- Loose coupling and easy API allows easy implementation of **new frontends**

Multiple Frontends

- Loose coupling and easy API allows easy implementation of **new frontends**
- Bachelor Thesis: Implementation of a native iOS client



The screenshot shows an iOS app interface for evaluating Berlin. At the top, the status bar displays "Carrier", signal strength, Wi-Fi, and battery icons, along with the time "7:43 PM". The app header includes a back arrow labeled "Berlin", the title "Evaluate", and a "Save" button. Below the header is a section for "Berlin" featuring a small image of the Brandenburg Gate and a text description: "Berlin is the capital of Germany. With a population of 3.5 million people it is the s...". The main content area contains several evaluation questions, each with a slider control:

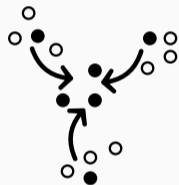
- What is the general price level in this city?**
 - expensive: Slider is positioned towards the right end.
 - inexpensive: Slider is positioned towards the left end.
 - moderate: Slider is positioned in the middle.
 - overpriced: Slider is positioned towards the right end.
- When is the best time to visit this city?**
- What are the points of interest worth visiting in this city?**
- What are the activities worth doing in this city?**
- What is the level of security in this city?**
- Who speaks English in this city?**

Knowledge Acquisition



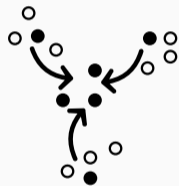
- Users do not like lengthy tasks

Acquire Knowledge



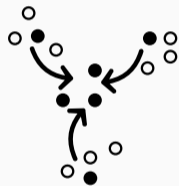
- Users do not like lengthy tasks
- Acquire knowledge from the user using small tasks (**microtasks**)

Acquire Knowledge



- Users do not like lengthy tasks
- Acquire knowledge from the user using small tasks (**microtasks**)
- Microtask has only **one question**

Acquire Knowledge




- Users do not like lengthy tasks
- Acquire knowledge from the user using small tasks (**microtasks**)
- Microtask has only **one question**
- 6 different types of microtasks

Microtask #1

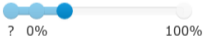
Item's support regarding one specific attribute

City



Item »Mumbai«, attribute »Price Level«: How well does it support »moderate«?

moderate



Don't show questions for this recommender

✕ Skip


✓ Next

Microtask #2


Best matching item regarding one specific attribute

City

Which item fits the answer »Museums« of the attribute »Sights« better?



Paris



Mumbai

Museums


? 0% 100%

Don't show questions for this recommender

Microtask #3

Best matching answer regarding one specific attribute

City




Item »Berlin«: Which answer fits the attribute »Activities« best?

Nightlife Shopping

Dining Hiking

Swimming


How well?  ? 0% 100%

Don't show questions for this recommender

Microtask #4

Weighted answers regarding one specific attribute

City



How would you evaluate the attribute »High Season« for the item »Sydney«?

January-March ? 0% 100%

April-June ? 0% 100%

July-September ? 0% 100%

October-December ? 0% 100%

Don't show questions for this recommender



Season	0%	100%
January-March	?	100%
April-June	?	100%
July-September	?	100%
October-December	?	100%

Microtask #5

Implicit CAPTCHA

City

Which item belongs to the recommender »City«?


 

Don't show questions for this recommender

Microtask #6

Binary decision

City



Does the item »Beijing« belong to the recommender »City«?

Yes No

Don't show questions for this recommender

Quality Assurance



- Users have to “earn” trust



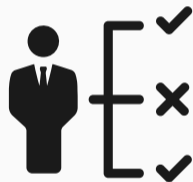
- Users have to “earn” trust
- Score is influenced by CAPTCHAs, user behavior, etc.



- Users have to “earn” **trust**
- Score is influenced by CAPTCHAs, user behavior, etc.
- All **contributions** of the user are **weighted** with this score (0 % - 100 %)



- Users have to “earn” **trust**
- Score is influenced by CAPTCHAs, user behavior, etc.
- All **contributions** of the user are **weighted** with this score (0 % - 100 %)
- New or malicious users have minor to no influence on the knowledge base



- Depending on the human score, users get microtask with known answers (**ground truth**)



- Depending on the human score, users get microtask with known answers (**ground truth**)
- Similar to CAPTCHAs, but not seen as such by the user



- Depending on the human score, users get microtask with known answers (**ground truth**)
- Similar to CAPTCHAs, but not seen as such by the user
- Influence the human score (positively and negatively)



- Depending on the human score, users get microtask with known answers (**ground truth**)
- Similar to CAPTCHAs, but not seen as such by the user
- Influence the human score (positively and negatively)
- Classify an image, hard to do automatically



- We **model** the **time** it takes to answer a microtask

Timing Models



- We **model** the **time** it takes to answer a microtask
- Timings are matched using Kullback-Leibler distance



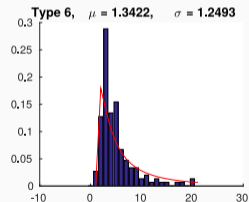
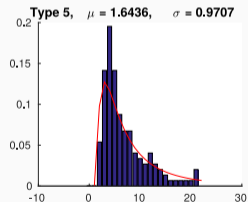
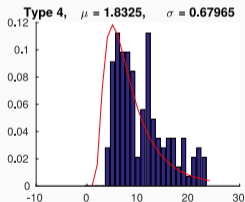
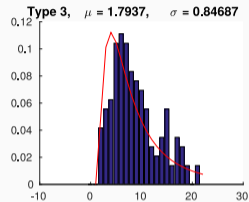
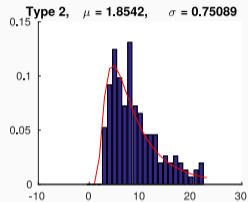
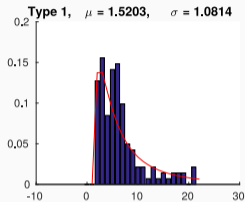
- We **model** the **time** it takes to answer a microtask
- Timings are matched using Kullback-Leibler distance
- Answers are **weighted** according to how well they fit

Timing Models



- We **model** the **time** it takes to answer a microtask
- Timings are matched using Kullback-Leibler distance
- Answers are **weighted** according to how well they fit
- **Non-matching** timings are **discarded** and decrease the human score

Microtask Timings





- Users can add new item, we have to cope with **spam**



- Users can add new item, we have to cope with **spam**
- CAPTCHAs only prevent automated spam



- Users can add new item, we have to cope with **spam**
- CAPTCHAs only prevent automated spam
- For a new item, we generate **verification microtasks**



- Users can add new item, we have to cope with **spam**
- CAPTCHAs only prevent automated spam
- For a new item, we generate **verification microtasks**
- If the community decides that an item does not belong to the recommender, it is removed



- We need **knowledge** for new items



- We need **knowledge** for new items
- Dynamic approach to calculate number of distributed microtasks



- We need **knowledge** for new items
- Dynamic approach to calculate number of distributed microtasks
- Loosely based on **local working set** algorithm for task scheduling



- We need **knowledge** for new items
- Dynamic approach to calculate number of distributed microtasks
- Loosely based on **local working set** algorithm for task scheduling
- Settle on minimum number of microtasks based on quality of the results

Evaluation



- We conducted a **worldwide study**



- We conducted a **worldwide study**
- Users had to complete microtasks, evaluate items, and use the recommender

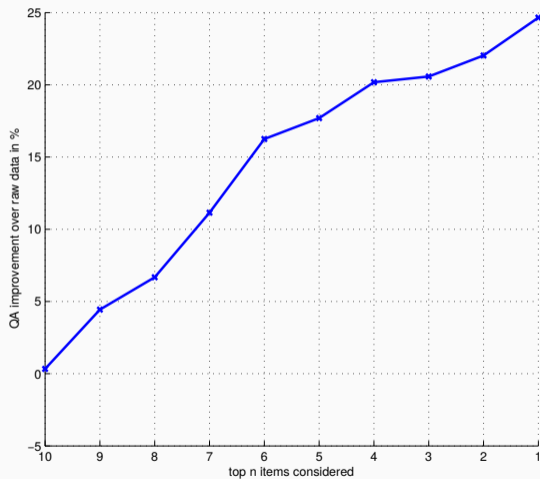


- We conducted a **worldwide study**
- Users had to complete microtasks, evaluate items, and use the recommender
- 1307 users (90.9 %) completed all tasks



- We conducted a **worldwide study**
- Users had to complete microtasks, evaluate items, and use the recommender
- 1307 users (90.9 %) completed all tasks
- Quality assurance led to recommendation **improvement** of **>20 %**

Recommendation Quality Improvement



Conclusion

- We developed a **generic recommender framework** for knowledge-based recommenders

- We developed a **generic recommender framework** for knowledge-based recommenders
- We showed that users are willing to contribute through **small tasks**

- We developed a **generic recommender framework** for knowledge-based recommenders
- We showed that users are willing to contribute through **small tasks**
- We presented automatic ways to ensure the **quality** of user content

Human Computation Based Acquisition of Financial Service Advisory Practices

Alexander Felfernig, Michael Jeran, Martin Stettinger, Thomas Absenger, Thomas Gruber, Sarah Haas, Emanuel Kirchengast, Michael Schwarz, Lukas Skofitsch, Thomas Ulz

FINREC'15

Peopleviews: Human computation for constraint-based recommendation

Alexander Felfernig, Thomas Ulz, Sarah Haas, Michael Schwarz, Stefan Reiterer, Martin Stettinger

ACM RecSys 2015 CrowdRec Workshop

Human computation for constraint-based recommenders

Thomas Ulz, Michael Schwarz, Alexander Felfernig, Sarah Haas, Amal Shehadeh, Stefan Reiterer, Martin Stettinger

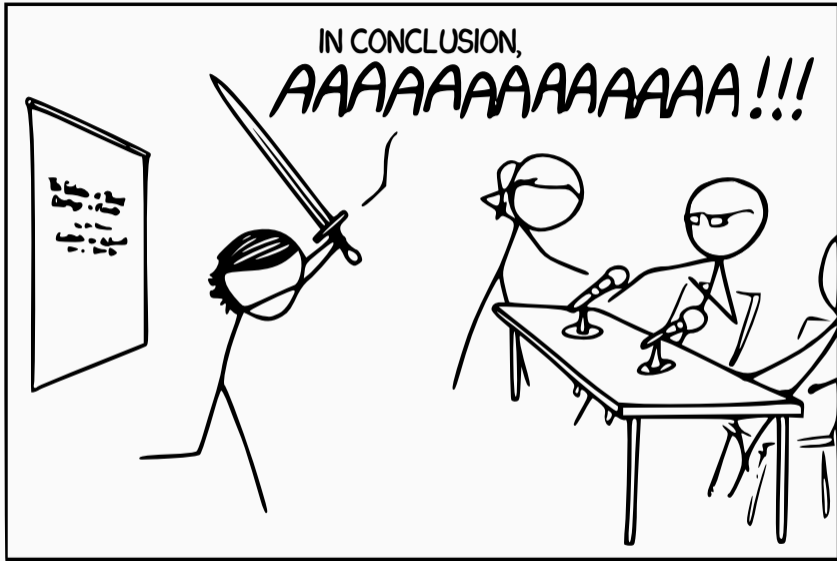
Journal of Intelligent Information Systems 2016

A Short Overview of the PeopleViews Mobile User Interface

*Angela Promitzer, Alexander Felfernig, Michael Schwarz, Thomas Ulz, Amal Shehadeh,
Sarah Haas*

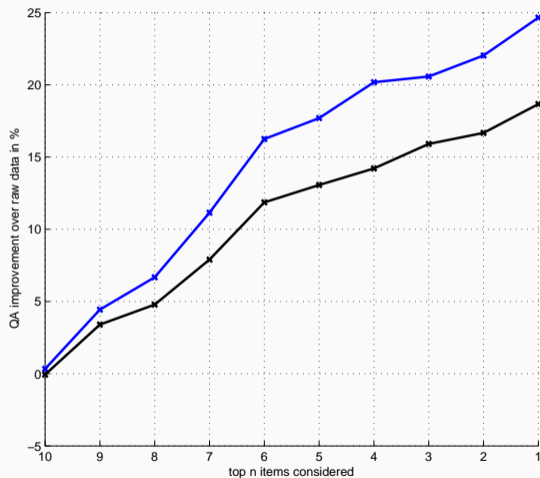
TU Graz Technical Report 2016

Thank you for your attention!



THE BEST THESIS DEFENSE IS A GOOD THESIS OFFENSE

Recommendation Quality Improvement without Ground Truth



Human Score Calculation Example

User	Human Score	Support			
		Answer 1	Answer 2	Answer 1 (weighted)	Answer 2 (weighted)
User 1	1	0.8	0.3	0.8	0.3
User 2	0.5	0.9	0.4	0.45	0.2
User 3	0.5	0.6	0.5	0.3	0.25
User 4	0	0.2	0.7	0	0
<i>Sum</i>	2	2.5	1.9	1.55	0.75
<i>Average</i>	-	$\frac{2.5}{4} = 0.625$	$\frac{1.9}{4} = 0.475$	$\frac{1.55}{2} = 0.775$	$\frac{0.75}{2} = 0.375$

Table 1: Four different users and their support values for Answer 1 and Answer 2.

Optimal Number of Microtasks Example

	# of microtasks	Answered	Data is good	New # of microtasks
Cycle 1	10	4	no	$10 \times 1.5 = 15$

Goal: 5 answers

Cycle 1 Start with 10 tasks → not enough, increase to 15

Optimal Number of Microtasks Example

	# of microtasks	Answered	Data is good	New # of microtasks
Cycle 1	10	4	no	$10 \times 1.5 = 15$
Cycle 2	15	11	yes	$15 \times 0.75 = 11$

Goal: 5 answers

Cycle 1 Start with 10 tasks → not enough, increase to 15

Cycle 2 15 was enough, decrease to $15 \cdot 0.75 = 11$ tasks

Optimal Number of Microtasks Example

	# of microtasks	Answered	Data is good	New # of microtasks
Cycle 1	10	4	no	$10 \times 1.5 = 15$
Cycle 2	15	11	yes	$15 \times 0.75 = 11$
Cycle 3	11	6	yes	$11 \times 0.75 = 8$

Goal: 5 answers

Cycle 1 Start with 10 tasks → not enough, increase to 15

Cycle 2 15 was enough, decrease to $15 \cdot 0.75 = 11$ tasks

Cycle 3 11 was enough, decrease to $11 \cdot 0.75 = 8$ tasks

Optimal Number of Microtasks Example

	# of microtasks	Answered	Data is good	New # of microtasks
Cycle 1	10	4	no	$10 \times 1.5 = 15$
Cycle 2	15	11	yes	$15 \times 0.75 = 11$
Cycle 3	11	6	yes	$11 \times 0.75 = 8$
Cycle 4	8	4	no	$8 \times 1.5 = 12$

Goal: 5 answers

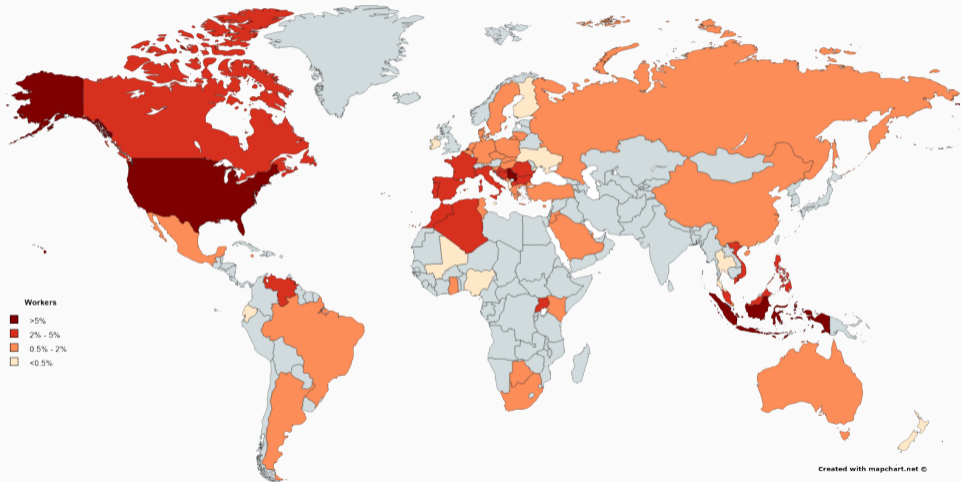
Cycle 1 Start with 10 tasks → not enough, increase to 15

Cycle 2 15 was enough, decrease to $15 \cdot 0.75 = 11$ tasks

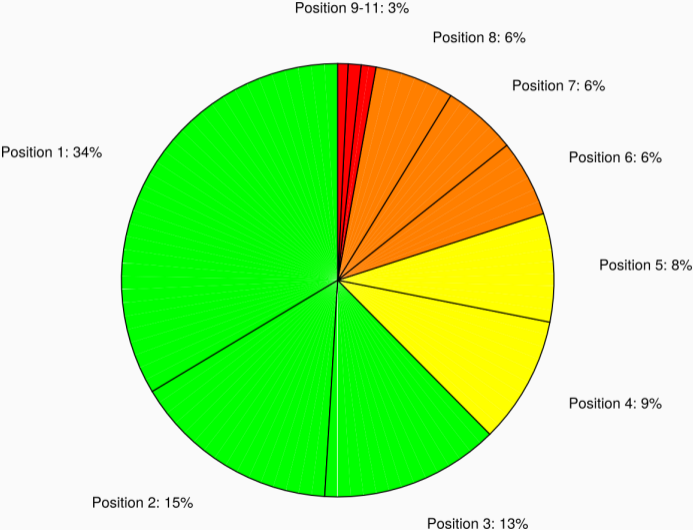
Cycle 3 11 was enough, decrease to $11 \cdot 0.75 = 6$ tasks

Cycle 4 8 was not enough, increase to $8 \cdot 1.5 = 12$ tasks

Worker Distribution



Position of Chosen Item



Recommendation Screen

Refine by

▾ Price Level

inexpensive

moderate

expensive

overpriced

▸ High Season

▾ Sights

Historic Sites

Museums

Religious Sites

Sport Venues

Scenic Places

▸ Activities

▸ Security

▾ English Speaking Population

Everyone

Most People

Only Young People









Nearly No One

▸ Public Transport

"City"

[← Back](#)

[⚡ Play against the community](#)

Item	Compare
 New York City The City of New York, often called New York City or simply New York, is the most populous city in the United States.[1] ...	<input checked="" type="checkbox"/> Best in 2 attributes
 Paris Paris, the cosmopolitan capital of France, is one of the largest agglomerations in Europe, with 2.2 million people livin...	<input checked="" type="checkbox"/> Best in Museums
 Sydney Sydney is the state capital of New South Wales and the most populous city in Australia and Oceania. Located on Australia...	<input type="checkbox"/>
 Berlin Berlin is the capital of Germany. With a population of 3.5 million people it is the second most populous city and the se...	<input type="checkbox"/>
 Moscow Moscow is the capital and the largest city of Russia. It is a major political, economic, cultural, and scientific center...	<input type="checkbox"/>
 Mumbai Mumbai (also known as Bombay, the official name until 1995) is the capital city of the Indian state of Maharashtra. Mumb...	<input type="checkbox"/>
 Beijing Beijing is the capital of the People's Republic of China and one of the most populous cities in the world.	<input type="checkbox"/>
 Tokyo Tokyo, officially Tokyo Metropolis, is one of the 47 prefectures of Japan, and is both the capital and largest city of J...	<input checked="" type="checkbox"/> Worst in Everyone

« 1 - 8 9 - 11 »

 Compare items

 Save filter