MoDaS

Mob Data Sourcing

Mob Data Sourcing

Daniel Deutch

Tova Milo

אוניברסיטת בן-גוריון בנגב
Ben-Gurion University of the Negev

TEL AVIV UNIVERSITY אוניברסיטת תל-אביב

Outline

- Crowdsourcing

- Crowd data-sourcing

- Towards a principled solution

- Conclusions and challenges

Warning: some (tasteful) nudity ☺

Outline

- Crowdsourcing 
- Crowd data-sourcing 
- Towards a principled solution 
- Conclusions and challenges 

Warning: some (tasteful) nudity ☺

CrowdSourcing

- Main idea: Harness the crowd to a “task”
  - Task: solve bugs
  - Task: find an appropriate treatment to an illness
  - Task: construct a database of facts

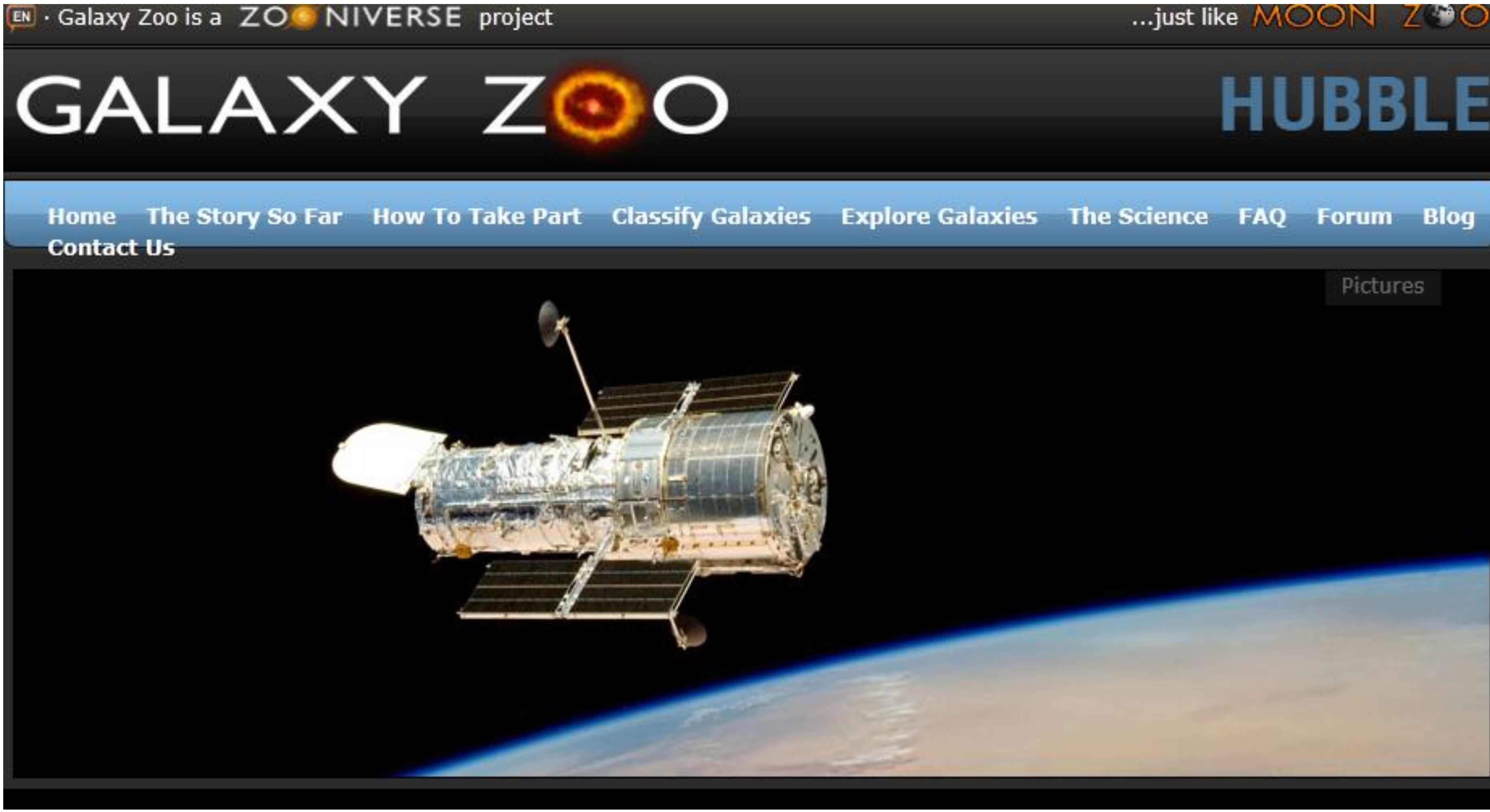
  …

- Why now?
  - Internet and smart phones …

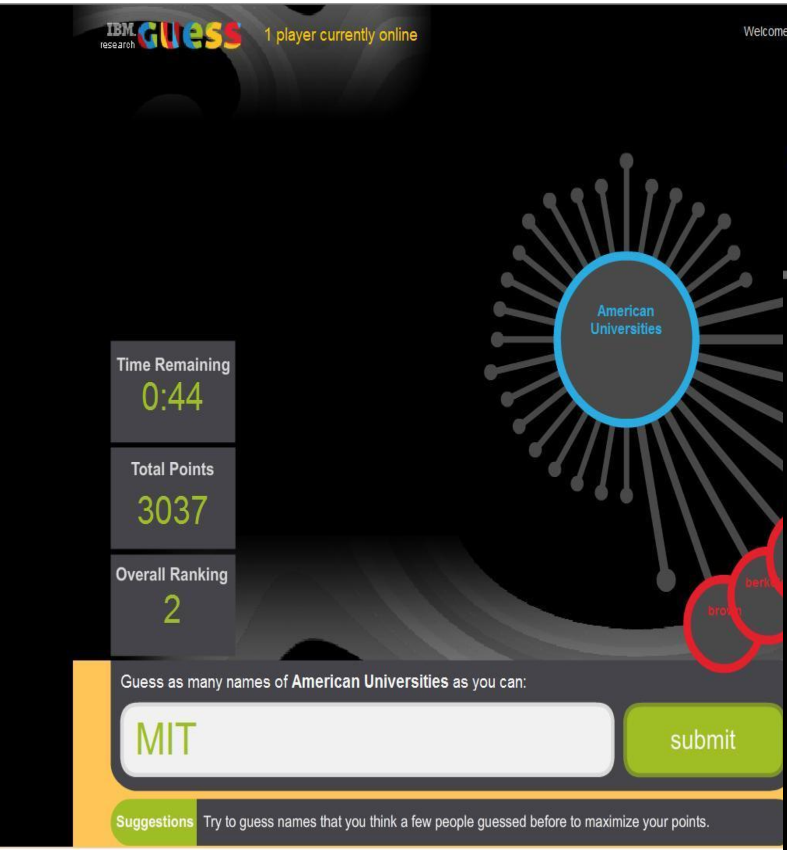
    We are all connected, all of the time!!!

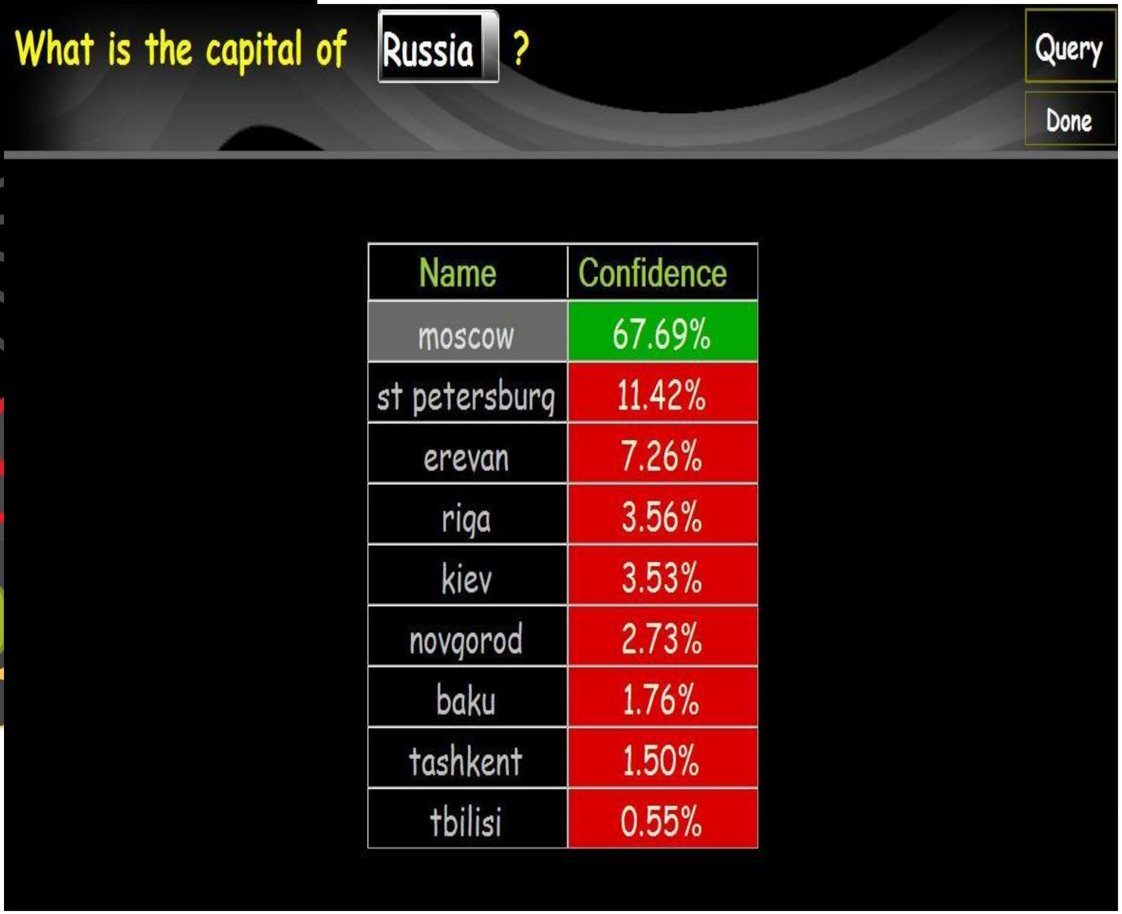
The classical example

Galaxy Zoo

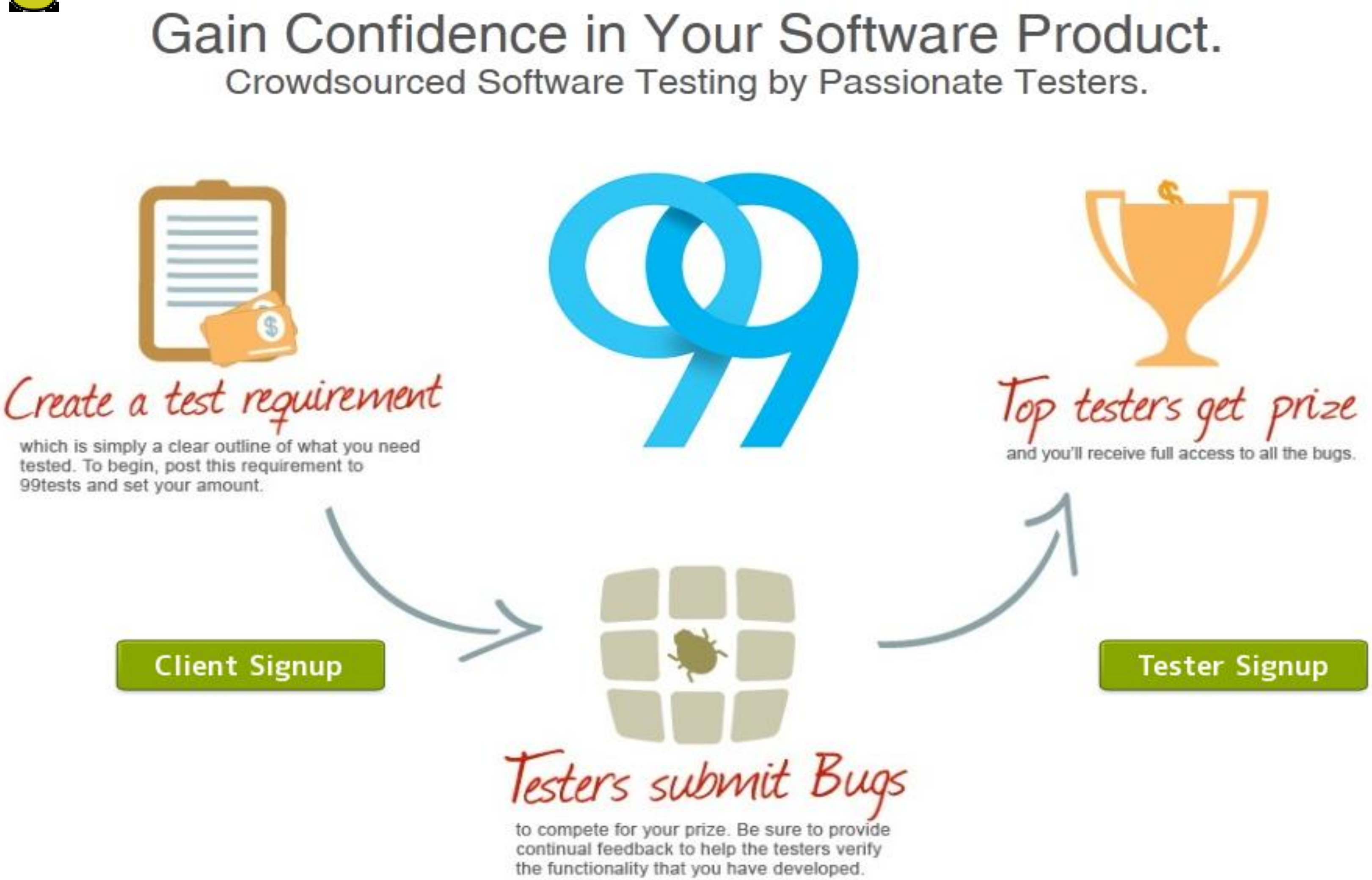

EN · Galaxy Zoo is a ZOONIVERSE project
...just like MOON ZOO
GALAXY ZOO
HUBBLE
Home
The Story So Far
How To Take Part
Classify Galaxies
Explore Galaxies
The Science
FAQ
Forum
Blog
Contact Us
Pictures

Playing Trivia

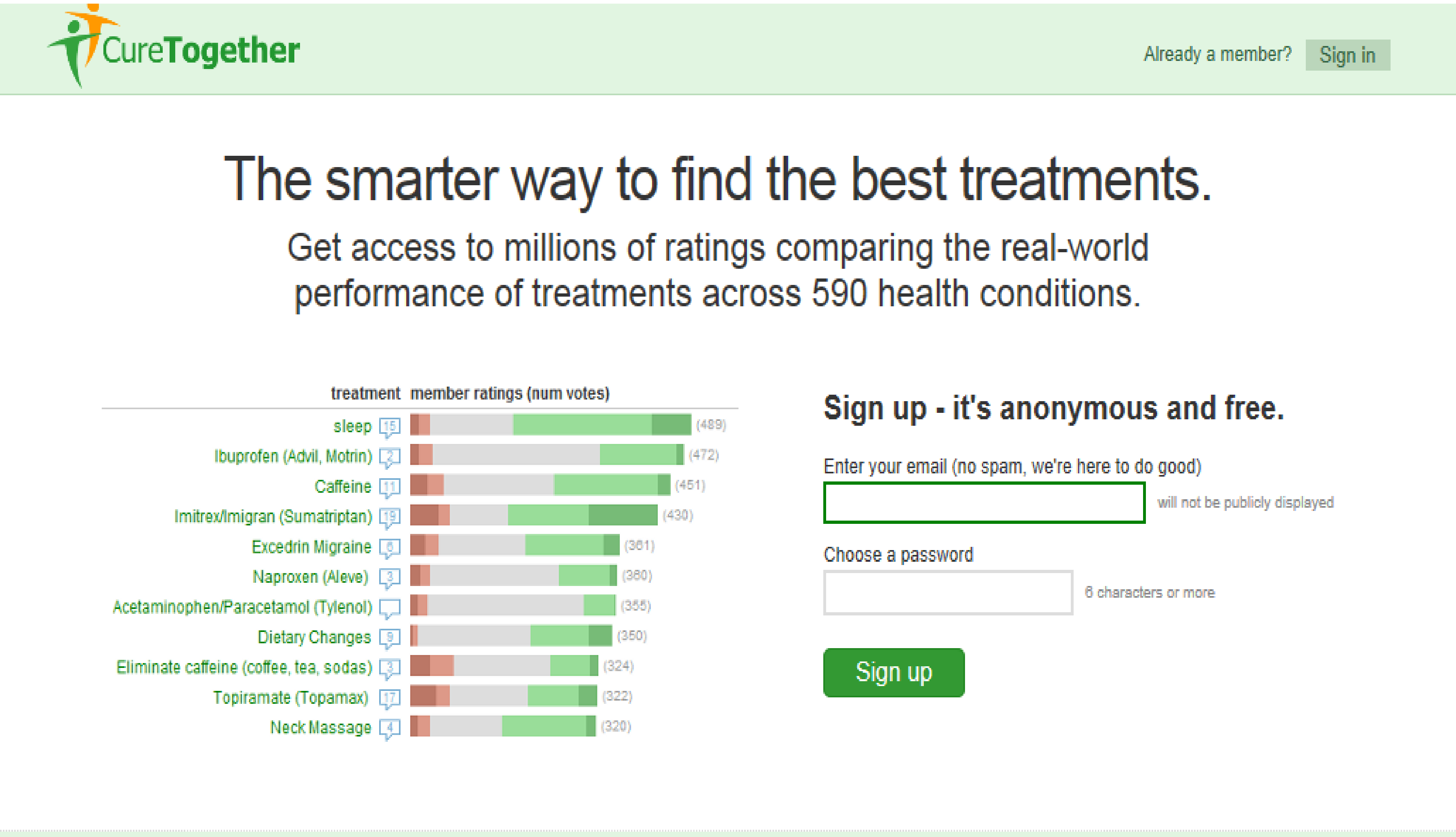




Collaborative Testing



Curing Together



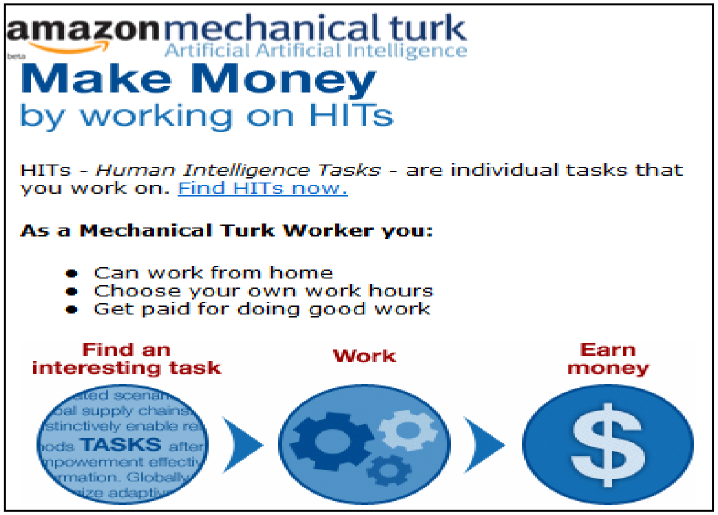
CrowdSourcing: Unifying Principles

- Main goal
  - "Outsourcing" a task to a crowd of users
- Kinds of tasks
  - Tasks that can be performed by a computer, but inefficiently
  - Tasks that can't be performed by a computer
- Challenges
  - How to motivate the crowd? **Next (very briefly)**
  - Get data, minimize errors, estimate quality **Rest of this tutorial**
  - Direct users to contribute where is most needed \ they are experts

Motivating the Crowd

Altruism

Fun



Money

Outline

- Crowdsourcing

- Crowd data-sourcing

- Towards a principled solution

- Conclusions and challenges

Crowd Data Sourcing

- The case where the task is collection of data
- Two main aspects **[DFKK’12]**:
  - Using the crowd to create better databases
  - Using database technologies to create better crowd datasourcing applications

Our focus

[DFKK’12]: Crowdsourcing Applications and Platforms: A Data Management Perspective, A.Doan, M. J. Franklin, D. Kossmann, T. Kraska, VLDB 2011

Data-related Tasks (that can be) Performed by Crowds

- Data cleaning
  - E.g. repairing key violations by settling contradictions
- Data Integration
  - E.g. identify mappings
- Data Mining
  - E.g. entity resolution
- Information Extraction

[**Internet- Scale Collection of Human- Reviewed Data ,** Q. Su, D. Pavlov, J. Chow, W.C. Baker, WWW ’07]

[**Matching Schemas in Online Communities: A Web 2.0 Approach,** R. McCann, W. Shen, A. Doan**,** ICDE '08]

[**Amplifying Community Content Creation with Mixed Initiative Information Extraction,** R. Hoffman, S. Amershi, K. Patel, F. Wu., J. Fogarty, D. Weld, CHI ’09]

Information Extraction

Luis von Ahn

From Wikipedia, the free encyclopedia

Dr. Luis von Ahn (born in 1979 in Guatemala City, Guatemala) is an entrepreneur and an associate professor in the Computer Science Department at Carnegie Mellon University.[2] He is known as one of the pioneers of the idea of crowdsourcing. He is the founder of the company reCAPTCHA, which was sold to Google in 2009.[3] As a professor, his research includes CAPTCHAs and human computation,[4] and has earned him international recognition and numerous honors. He was awarded a MacArthur Fellowship (a.k.a., the "genius grant") in 2006,[5][6] the David and Lucile Packard Foundation Fellowship in 2009, a Sloan Fellowship in 2009, and a Microsoft New Faculty Fellowship in 2007. He has also been named one of the 50 Best Brains in Science by Discover Magazine, and has made it to many recognition lists that include Popular Science Magazine's Brilliant 10, Silicon.com's 50 Most Influential People in Technology, Technology Review's TR35: Young Innovators Under 35, and FastCompany's 100 Most Innovative People in Business.

Siglo Veintiuno, a leading newspaper in Guatemala, chose him as the person of the year in 2009. In 2011, Foreign Policy Magazine in Spanish named him the most influential intellectual of Latin America and Spain.[7]

Contents [hide]

Luis von Ahn

Born	1979 (age 32–33)[citation needed] Guatemala City, Guatemala
Residence	United States
Institutions	Carnegie Mellon University
Alma mater	Carnegie Mellon University Duke University
Doctoral advisor	Manuel Blum
Known for	CAPTCHA, reCAPTCHA,

Biography

[edit]

Main Tasks in Crowd Data Sourcing

- What questions to ask?
- How to define correctness of answers?
- How to clean the data?
- Who to ask? how many people?
- How to best use resources?

Declarative Framework!

Probabilistic Data!

Data Cleaning!

Optimizations and Incremental Computation

Outline

- Crowdsourcing
- Crowd datasourcing
- Towards a principled solution
- Conclusions




Platforms for Crowdsourcing

Qurk (MIT)

CrowdDB (Berkeley and ETH Zurich)

CrowdForge (CMU)

Deco (Stanford and UCSC)

MoDaS (Tel Aviv University)

…

[and many more, please forgive us if your project is not listed!]

Qurk

- Main observation: Tasks aided by Mturk can be expressed as workflows, with
  - Queries on existing data
  - “Black boxed” (User Defined Functions) that are tasks (HITs) to be performed by the turker

Crowdsourced Databases: Query Processing with People, A. Marcus, E. Wu, D. R. Karger, S. Madden, R. C. Miller, CIDR 2011

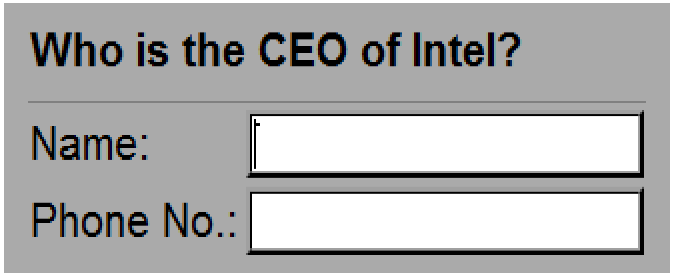
Qurk Example

SELECT companyName,
findCEO(companyName).CEO,
findCEO(companyName).Phone
FROM companies

companies

Company	CEO Name	CEO Phone Number
Microsoft	?	?
Intel	?	?

TASK findCEO(String companyName)
RETURNS (String CEO,String Phone):
TaskType: Question
Text: ``Who is the CEO of %s?“, companyName
Response: Form((``Name'',String), (``Phone No.'',String))



Contradictions?

- The same form is presented to multiple users
  - Not everyone will have the answer to every question
- But then contradictions may rise
  - E.g. multiple CEOs to the same companies
  - Can be identified as a key violation
- In Qurk one can choose a combiner to aggregate the answers
  - Out of a predefined set of options
  - E.g. Majority Vote

We will get back to this point!

Optimization Issues

- Cost of a HIT
  - Optimized statically or at runtime
- Given a limited number of HITs, choosing a subset
- Batch Predicates
- Asynchronous Implementation

CrowdDB

- A different declarative framework for crowd data sourcing

- Main difference: allows to crowd-source the generation of new tuples

CrowdDB: Answering Queries with Crowdsourcing,
M. J. Franklin, D. Kossmann ,T. Kraska, S. Ramesh, R. Xin
SIGMOD ‘11

CrowdForge

- A declarative framework inspired by MapReduce

- Provides a small set of task primitives (partition, map, and reduce) that can be combined and nested
  - Allows to break MTurk tasks to small tasks and combine the answers

- Sub-tasks are then issued to the crowd (turkers)

CrowdForge: Crowdsourcing Complex Work, A. Kittur, B. Smus S. Khamkar R. E. Kraut , UIST ‘11

How Well are We Doing?

- What questions to ask? √
- How to define correctness of answers?
- How to clean the data?
- Who to ask? how many people?
- How to best use resources?

Declarative Framework!

Probabilistic Data!

Data Cleaning!

The Naked Truth?

Spencer Tunick

Errors, Contradictions and Motivation

- The solutions described so far propose declarative infrastructures for collecting data from crowds
- But how credible is the data?
  - It is likely to contain errors
  - As well as contradictions
- We need ways to
  - settle contradictions, and
  - estimate trust in users
- Also related to the incentives and budget
  - Can we reward correct users?

Deco (sCOOP project)

- A declarative platform based on 3 main concepts:
  1. Fetch: add tuples
     Fetch Rules (FR) procedures
  2. Resolve: resolve dependent attributes
     Resolution Rules (RR) procedures
  3. Join: Outerjoin of tables

Deco: Declarative Crowdsourcing, A. Parameswaran, H. Park, H.G. Molina, N. Polyzotis, J. Widom, Stanford Infolab Technical Report, 2011

[Deco slides based on slides presented in Crowd-Crowd 2011]

Fetch Rules

R (restaurant, address, [rating], [cuisine]), S (address, [city, zip])

LHS ⇨ RHS with procedure P

Given LHS value, procedure P can obtain RHS values
from external source(s)

restaurant,address ⇨ rating
restaurant ⇨ cuisine
address ⇨ city,zip

Resolution Rules

R (restaurant, address, [rating], [cuisine])

S (address, [city, zip])

A resolution rule per dependent attribute-group

restaurant,address ➔ rating (F=avg)

restaurant ➔ cuisine (F=dup-elim)

address ➔ city,zip (F=majority)

Designing Resolution Rules

- Average value? Majority vote?
- But some people know nothing about a given topic
- So maybe a "biased vote"?
- But how to bias?
- A "chicken or the egg" problem:

  To know what is true we need to know who to believe.
  But to know this we need to know who is usually right
  (and in particular, what is true..)

MoDaS

- Observation: two key aspects in the design of crowdsourcing applications
  - Uncertainty in data
  - Recursion in policies

- Approach: take declarative solutions further
  - Use probabilistic DBs for modeling uncertainty in data
  - Use datalog for modeling recursion

Example

- Start with some probability reflecting the trust in users (turkers)

- Gain confidence in facts based on the opinion of users that supported them
  - Choose probabilistically "believed" facts
  - Assign greater weight (in probability computation) to trusted users

- Then update the trust level in users, based on how many of the facts which they submitted, we believe

- Iterate until convergence

  Trusted users give us confidence in facts,

  and users that supported these facts gain our trust…

Declarative Approach

- That was one possible policy
- We want to have easy control on the employed policy
- We want to be able to design such policies for conflict resolution
- But also for
  - rewarding turkers, choosing which question to ask…
  - and for data cleaning, query selection, user game scores,…

Declarative Approach (cont.)

- We don't want to (re)write Java code (for each tiny change!)
- We want (seamless) optimization, update propagation,…

Database approach:

Define a **declarative language** for specifying policies

- Based on probabilistic databases and (recursive) datalog

[D., Greenshpan, Kostenko, M. ICDE'11 ,WWW'12]
[D., Koch, M. PODS'10]

Block-Independent Disjoint (BID) Tables

Name	Cuisine	Prob.
Alouette	French	0.7
Alouette	American	0.3
Mcdonald's	Fast food	1

Name	Cuisine
Alouette	French
Mcdonald's	Fast food

0.7

Name	Cuisine
Alouette	American
Mcdonald's	Fast food

0.3

Efficient Query Evaluation on Probabilistic Databases, N. Dalvi and D. Suciu, VLDB '04

Repair-Key

Restaurants

Rest	Cuisine	Support
Alouette	French	Alouette
Alouette	American	3
Mcdonald's	Fast food	1

REPAIR-KEY[Rest@ Support](Restaurants)

Rest	Cuisine
Alouette	French
Mcdonald's	Fast food

0.7

Rest	Cuisine
Alouette	American
Mcdonald's	Fast food

0.3

Approximating predicates and expressive queries on probabilistic databases, C. Koch, PODS '08

Proposed Language

- Enrich SQL with the REPAIR-KEY construct

- And a WHILE construct

- Semantics: Markov chain of DB instances.

  Return the Probability of a fact to hold in a give instance.

- Allows to easily express nicely common policies for cleaning, selection of questions, scoring answers

Recursion on Prob. Data!

The “while” language consists of 3 parts:

1. Update rules, to be evaluated repeatedly.
   Intuitively, rules to settle contradictions.

2. A boolean condition, deciding when to sample.
   Intuitively, when the DB includes no contradiction.

3. A query of interest, to be sampled.
   E.g. what kind of cuisine is Alouette?

Example

User	Confidence
Alice	6
Bob	2
Carol	2

Rest	Cuisine	User
Alouette	French	Alice
Alouette	French	Bob
Alouette	American	Carol
McDonalds	French	Carol
McDonalds	Fast Food	Bob

Example (cont.)

U	C
A	6
B	2
C	2

$$\frac{8 * 2}{10 * 4}$$

R	T
A	F
M	FF

U	C
A	7
B	3
C	3

…

…

…

$$\frac{8 * 2}{10 * 4}$$

P	T
A	F
M	F

$$\frac{2 * 2}{10 * 4}$$

P	T
A	A
M	F

…

$$\frac{2 * 2}{10 * 4}$$

P	T
A	A
M	FF

…

Example: Update Rules

U1

```
Drop BelievedRestaurants;
INSERT INTO BelievedRestaurants
REPAIR-KEY[Restaurant @ authority]
ON
   (SELECT name, cuisine, authority
    FROM Restaurants AS R, Users AS U
     WHERE R.user = U.user);
```

Compute a subset of believed facts based on user authorities

Boolean condition: Name is a key in *BelievedRestaurants*

U2

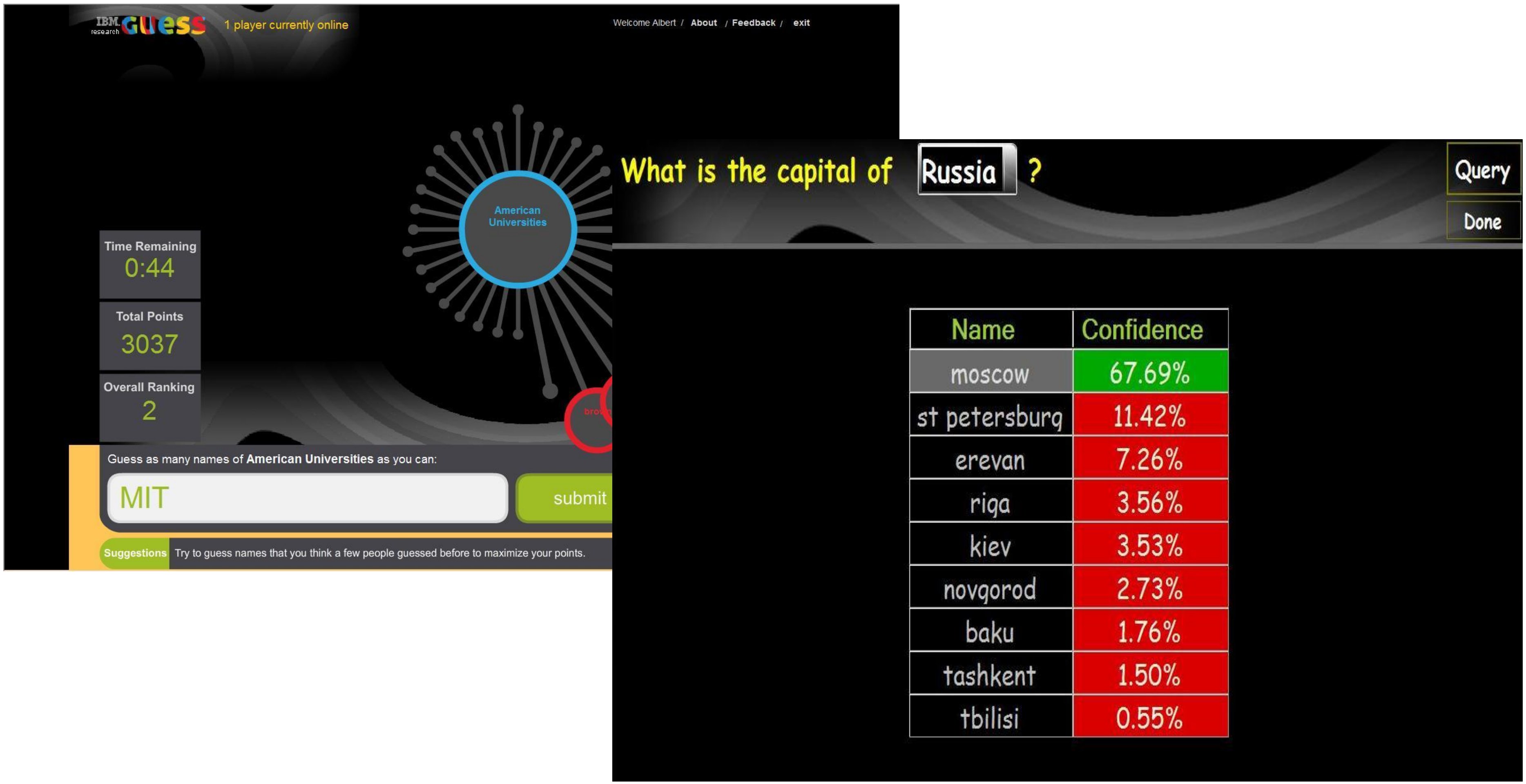
```
UPDATE Users
SET Authority =
(SELECT CorrectFacts
FROM Q1
WHERE Q1.user = Users.user)

Q1 = SELECT user, COUNT(DISTINCT name)
AS CorrectFacts FROM Q2
GROUP BY user;

Q2 = SELECT user, name, cuisine
FROM UserRest UR
WHERE EXISTS
(SELECT * FROM BelievedRestaurants  BR
WHERE BR.name = UR.name AND
BR.cuisine = UR.cuisine);
```

Update user authorities according to number of believed facts

TriviaMasster



Some Complexity Results

Formal problem: Given a Markov Chain of database instances and an SQL query on the database (“what is Alouette’s cuisine ?”), compute the probabilities of the different answers.

- Theorem: Exact computation is #P-hard
- Theorem: If Markov Chain is **ergodic**, computable in EXPTIME
  - Compute the stochastic matrix of transitions
  - Compute its fixpoint
  - For ergodic Markov Chain corresponds to correct probabilities
  - Sum up probabilities of states where the query event holds
- Theorem: In general, 2-EXPTIME
  - Apply the above to each connected component of the Markov Chain
  - Factor by probability of being in each component

Some Complexity (cont.)

Approximations:

- Absolute approximation: approximates correct probability ±ε
- Relative approximation: approximates correct probability up to a factor in-between (1- ε), (1+ ε).

[Relative is harder to achieve]

Language	Exact computation	Relative approx	Absolute approx
(Linear) datalog	#P-hard In PSPACE	NP-hard	In PTIME
Inflationary fixpoint	#P-hard In PSPACE	NP-hard	In PTIME
Non-inflationary fixpoint	#P-hard In (2)EXP-TIME	NP-hard	NP-hard; PTIME in input size and mixing time

Sampling

Algorithm induced by the (operational) semantics:

- Perform a random walk on the Markov Chain of database states
- Sample the query results on observed states
- Upon convergence, report the fraction of states in which a tuple was observed in the query result, as an approximation of its probability

Convergence?

- Guaranteed to converge to absolute ($\pm\varepsilon$) approximation
- However the time until convergence depends on the MC structure
  - Polynomial in the database size and MC mixing time

Still Lots of Open Questions

- How (and when) can we evaluate things fast enough?
- How to store the vast amount of data?
    - Distributed Databases? Map-reduce?
- The data keeps changing. How to handle updates?
- …

How Well are We Doing?

- What questions to ask? √
- How to define correctness of answers? √
- How to clean the data? √
- Who to ask? how many people?
- How to best use resources?

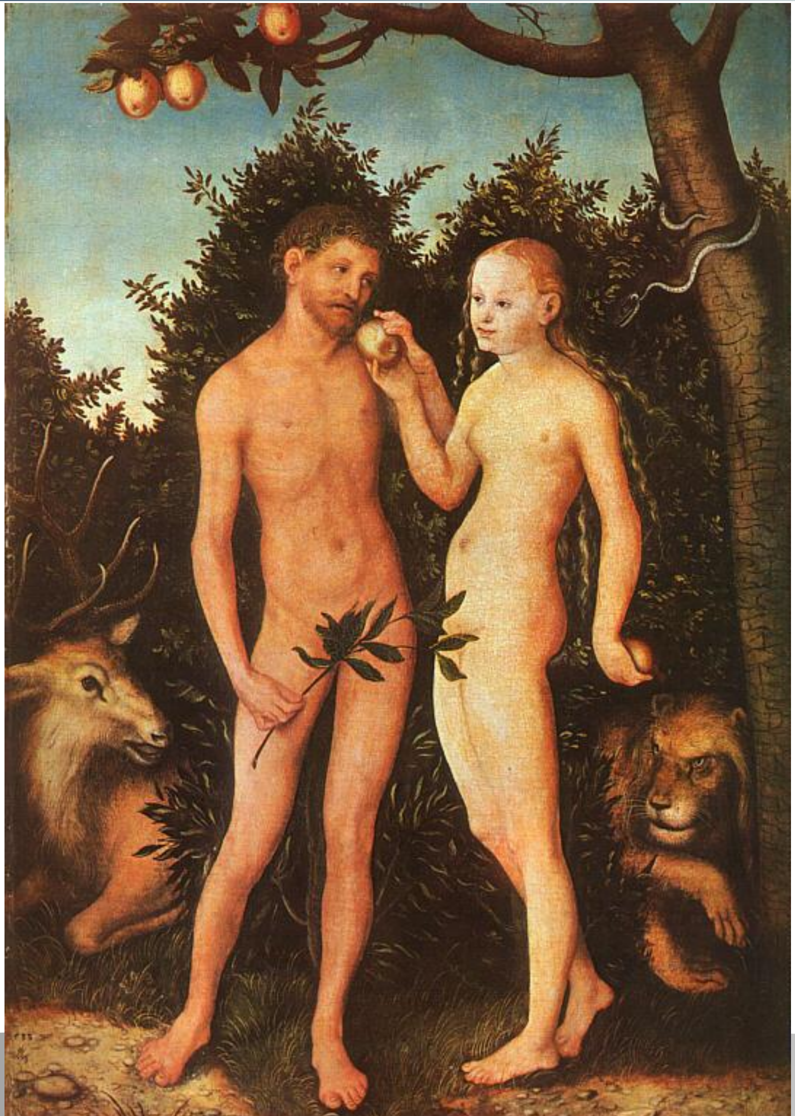
Declarative Framework!

Probabilistic Data!

Data Cleaning!

Optimizations and Incremental Computation

The Tree of Knowledge



Partial Knowledge

	q1	q2	q3	q4	q5	q6	…		
u1	a	5		b					
u2	a		3						
u3		5	3	b					
u4	b	2	3						
u5	c		3	a					
…									

- Goal: Compute an aggregate function f for each query, e.g.
  - Some metric of the distribution (e.g. entropy)
  - Most frequent answer
  - Aggregated value (e.g. average)

Increasing Knowledge

- Limited overall resources
- Limited user availability
- Bounded resources per question

Which cells to resolve?

[Boim, Greenshpan, M., Novgorodov, Polyzotis, Tan. ICDE’12,…]

Quantifying Uncertainty

- Assume t answers suffice for computing f for q
- Comp(q): all possible completions of q's column
- Dist(r – r'): distance between two results of f
- Uncertainty(q): max{ Dist(f(X) - f(Y)) | X,Y in Comp(q) }

  i.e. the largest distance between possibly completions

Quantifying Uncertainty (cont.)

- Uncertainty measures for a Users-Answer matrix M
  - **Max-uncertainty(M)**
  - **Sum-uncertainty(M)**

- **Problem statement (X-uncertainty Reduction)**

  Given a matrix M, a choice x ϵ {max,sum}, and a set of constraints, identify a set C of empty cells that satisfy the constraints and where

  Max $_{M' \in M_C}$ **X-uncertainty(M')** is minimized.

  Where M_C contains all possible matrices that we can derive from M by resolving solely the cells in C.

Examples

- Target function f
  - Entropy, majority-vote, average,…
- Constraints
  - A: bound k on the over number of cells
  - B: also a bound k’ on questions per users
  - C: here k’ is a bound on users per question

Some Complexity Results

- **max-Uncertainty Reduction**

  in PTIME for all constraints classes
  - Greedy algo for constraints class A (and C)
  - Using Max-flow for constraints class B

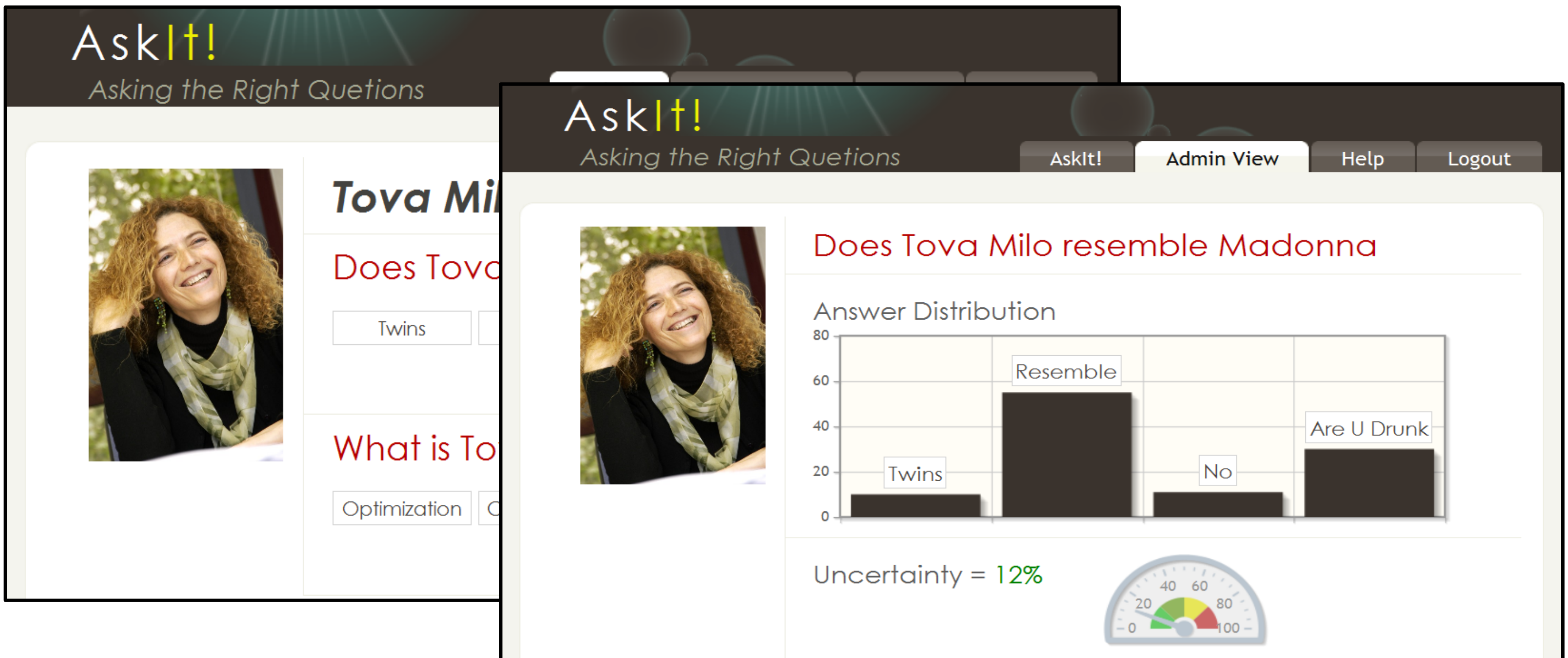
- **sum-Uncertainty Reduction**

  in PTIME for constraint classes A and C
  - Dynamic programming

  NP-COMPLETE for constraints class B
  - Reduction for perfect 3 set cover

AskIt (ICDE'12 demo)

- Gather information (scientific as well as fun) on ICDE'12 authors, participants, papers, presentations,…



Lots of Open Questions

- Use prior knowledge about users/answers
    - Predict answers
    - Predict who can/will answer what

  [Collaborative Filtering-style analysis is useful here]

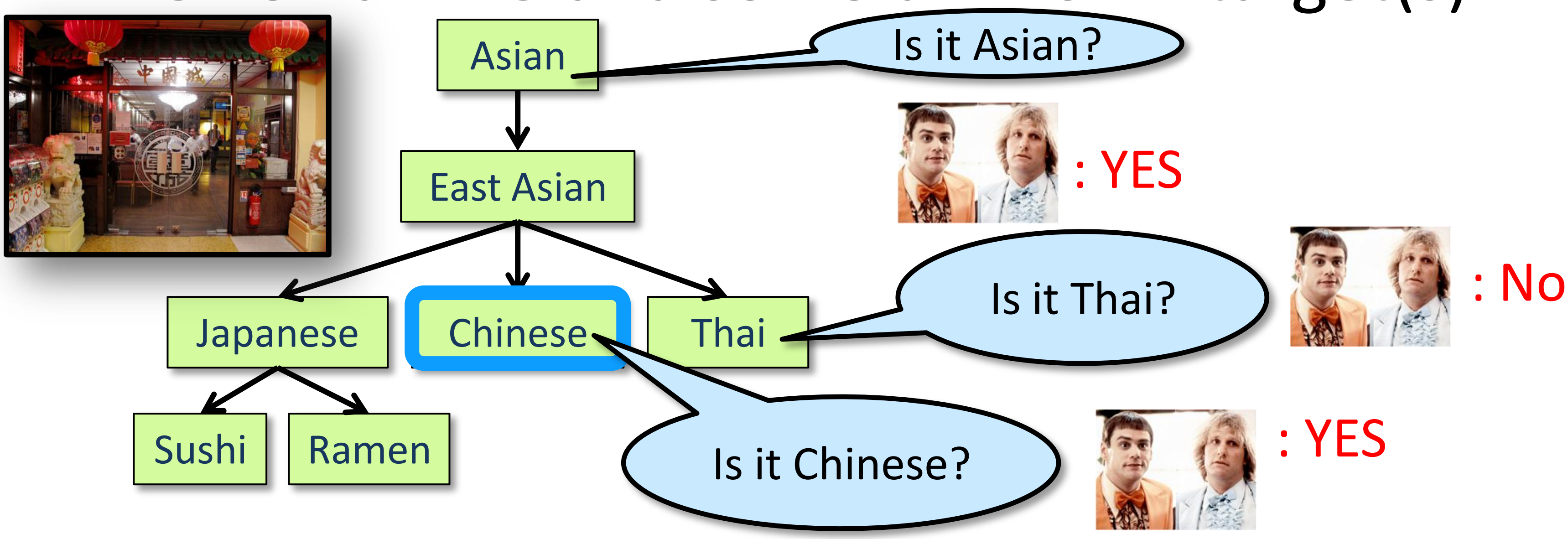
- Worse-case analysis vs. expected error

- Incremental computation & optimization

…

Best use of resources: Human Assisted Graph Search

- Given a DAG and some unknown target(s)



- We can ask **YES/NO** questions
  - E.g. reachability

HumanAssisted Graph Search: It's Okay to Ask Questions, A. Parameswaran, A. D. Sarma, H. G. Molina, N. Polyzotis, j. Widom, VLDB '11

The Objective

- Find an optimal set of questions to find the target nodes
  - **Optimize cost:** Minimal # of questions
  - **Optimize accuracy:** Minimal # of possible targets

- **Challenges**
  - Answer correlations (Falafel → Middle Eastern)
  - Location in the graph affects information gain (leaves are likely to get a NO)
  - Asking several questions in parallel to reduce latency

Problem Dimensions

- Single target/Multiple targets
- Online/Offline
  - Online: one question at a time
  - Offline: pre-compute all questions
  - Hybrid approach
- Graph structure

More in this SIGMOD!

- **CrowdScreen: Algorithms for Filtering Data with Humans** [Parameswaran, García-Molina, Park, Polyzotis, Ramesh, Widom]
  – Deterministic and probabilistic algorithms to optimize expected cost (number of questions) and error.

- **So Who Won? Dynamic Max Discovery with the Crowd** **[**Guo, Parameswaran, García-Molina]
  – Algorithms for finding max-ranked (top-1) element in a set by asking questions.

Outline

- Crowdsourcing

- Crowd datasourcing

- Towards a principled solution

- Conclusions and challenges

Conclusions

- All classical issues:

  Data models, query languages, query processing, optimization, HCI
- Database techniques are very useful
  - "Classical" as well as new
- BUT
  - (Very) interactive computation
  - (Very) large scale data
  - (Very) little control on quality/reliability

Challenges

- Open vs. closed world assumption
- Asking the right questions
- Estimating the quality of answers
- Incremental processing of updates

More Challenges

- Distributed management of huge data
- Processing of textual answers
- Semantics
- More ideas?

Thank You!   תודה!

Thanks: Serge Abiteboul, Yael Amsterdamer, Rubi Boim, Susan Davidson, Ohad Greenshpan, Yael Grossman, H. V. Jagadish, Slava Novgordov, Ilia Lotosh, Neoklis Polyzotis, Pierre Senellart, Wang-Chiew Tan…