Structure of the Talk

- Definition
- Mechanical Turk
- Quality Control
- CrowdDB
What is Crowdsourcing?

- The practice of obtaining needed services, ideas, or content by soliciting contributions from a large group of people, and especially from an online community, rather than from traditional employees or suppliers.
- Allows for large-scale and on-demand invocation of human input for data-gathering and analysis.
- Distinct from outsourcing in that the work comes from an undefined public rather than from a specific group.
Crowdsourcing Overview

- **Requester:**
  - People who submit tasks and collect answers

- **Platform:**
  - Performs task management

- **Worker:**
  - People who work on tasks
Problem:

- Manual Evaluation of quality is slow and expensive

Crowdsourcing:

- Low costs of non-experts, $0.10 to translate a sentence
- High agreement between experts and non-experts
- Good framework for complex tasks like human-assisted translation edit rate (i.e. how much editing a human would have to perform to change a system output so that it exactly matches a reference translation)

*Li, Guoliang, Crowdsourcing @ HotDB2012*
Painting Similarity

How similar is the artistic style in the paintings above?

- Very Similar
- Similar
- Somewhat Dissimilar
- Very Dissimilar

Image Search

Tingxin Yan, Vikas Kumar, Deepak Ganesan: CrowdSearch: exploiting crowds for accurate real-time image search on mobile phones. MobiSys 2010:77-90
Examples of Crowdsourcing Platforms

- Mechanical Turk: Marketplace for (usually small) tasks
- CrowdDB: Uses crowd to answer DB queries
When to Crowdsource

- Computers cannot do the task (e.g. translation)
- A single person cannot do the task
- The work can be split into many small tasks
Different Slide Deck
Different Slide Deck
Relational Database Fail

SELECT market_capitalization FROM company
WHERE name = "I.B.M.";

Query returns an empty answer if the company table instance in the database does not contain a record for "I.B.M."

- Why?
- Could have been deleted by accident
- Could be under I.B.N.
- Could be under International Business Machines
Issues with Relational Databases

- Closed World Assumption
  - Information not in database is either false or nonexistent

- Relational databases are extremely literal
  - Expect data to have been properly cleaned and validated before entry; no native tolerance of inconsistency in data or queries
Issues with Relational Databases

Let’s say you were to run a query like the one below:

```
SELECT image FROM picture
WHERE topic = "Business Success"
ORDER BY relevance LIMIT 1;
```

Unless someone had previously sorted the pictures by specific topic, there is no good way to run a query like this.
CrowdDB

- Use the crowd to answer DB queries
  - Find missing data
  - Make a subjective comparison
- Recognize patterns
- Main operations
  - Join
  - Sort
CrowdSQL

- An SQL extension that supports crowdsourcing (and is therefore the language for crowdDB)
- Involve missing data and subjective comparisons
- For traditional databases, equivalent to SQL
- Developers don’t have to be aware that their code involves crowdsourcing
**CrowdSQL**

- SQL DDL Extensions
- Specific attributes of tuples can be crowdsourced
- Entire tuples can be crowdsourced
- Keyword: CROWD
  - CrowdDB does not impose any limitations with regard to SQL types and integrity constraints
  - CROWD tables must have a primary key
Column "url" marked as crowdsourced

CREATE TABLE Department (university STRING, name STRING, url CROWD STRING, phone STRING, PRIMARY KEY (university, name));
Sample Code

"Professor" table to be crowdsourced

CREATE CROWD TABLE Professor (  
name STRING PRIMARY KEY,  
email STRING UNIQUE,  
university STRING,  
department STRING,  
FOREIGN KEY (university, department)  
REF Department(university, name) );
Comparisons

- **CROWDEQUAL** – ask the crowd if two objects are equal

  ```sql
  SELECT profile FROM department
  WHERE name ≈ "CS";
  ```

- **CROWDORDER** – ask the crowd to arrange the objects in order of importance

  ```sql
  CREATE TABLE picture (
    p IMAGE,
    subject STRING
  );
  SELECT p FROM picture
  WHERE subject = "Golden Gate Bridge"
  ORDER BY CROWDORDER(p, "Which picture visualizes better %subject");
  ```
User Interface Generation

- Automatically generates user interfaces
- Two-step process in CrowdDB
- User interfaces are in HTML and JavaScript
What the worker sees

- The title of the HTML is the name of the table
- Fields ask the worker to input the missing information
- Copies the known field values into the HTML form
- Generates JavaScript code to check for correct types of input

Figure 2: Example User Interfaces Generated by CrowdDB
Multi-Relation Interfaces

- Foreign key references a non-crowdsourced table
- Generated user interface shows a drop-down box
- CrowdDB supports two types of user interfaces
  - Normalized
  - Denormalized
Crowd Operators

- Implements all operators of the relational algebra, just like any traditional database system
- Initialized with a user interface template and the standard HIT parameters
- Quality control carried out by majority vote
CrowdDB has three crowd operators

- CrowdProbe: Crowdsources missing information of crowd columns
- CrowdJoin: Implements an index nested-loop join over two tables
- CrowdCompare: Implements the CROWDEQUAL and CROWDORDER functions
CrowdSQL in practice

- Minimal extension to SQL
- CrowdSQL changes the closed-world to an open-world assumption
- Cost and response time of queries can be unbounded
- Provide a way to define a budget for a query – using the LIMIT operator
  - Constrains the number of tuples returned as a result of the query
  - Implicit constraint on cost and result time of query
Shortcomings of CrowdSQL

- No explicit constraint on budget
  - LIMIT only constrains number of responses to query
- No accounting for lineage
  - Turker #5 is a spammer. Currently no way to identify and remove all data from him
- No entity resolution of crowdsourced data.
  - Not a problem if all the workers use exactly the same literals
  - In general, makes data from different sources difficult to clean
Workers were asked to fill in missing data for a table with two crowdsourced columns:

For 3607 business names in 40 cities, the turkers had to find the phone # and address of the business.

CREATE TABLE businesses (name VARCHAR PRIMARY KEY, phone_number CROWD VARCHAR(32), address CROWD VARCHAR(256));
Experiment: Vary hit groups, track response time

- Response times decrease dramatically as size of HIT groups increases
Experiment: Vary hit groups, track response time

But, there is a tradeoff between size of HIT group and how much of that HIT group is actually completed.

Figure 5: Completion (%): Vary Hit Group (1 Asgn/HIT, 1 cent Reward)
Experiment: Responsiveness, vary reward

For the particular task the experimenters assigned, paying the turkers more resulted in increased performance.

Figure 6: Completion (%): Vary Reward
(100 HITs/Group, 5 Asgn/HIT)
Experiment: Worker affinity and quality

- Analysis of the distribution of work among workers and answer quality
- Some workers begin to specialize in a particular requester’s requests
- This does not decrease error frequency
- Reward and group size also has no effect on error frequency

![Graph showing the distribution of Total HITs submitted and Incorrect HITs submitted](image)
Observations

- Crowd resources have long-term memory that impact performance
  - If the requester rejects too many HITs, workers stop working for requester
  - Bugs leading to error messages can alarm the turkers
- User interface design and precise instructions can greatly increase reliability of results