State of the Art in Crowdsourcing

Richard Bayerle
Technical University of Vienna
Studentnr. 1025259
e1025259@
student.tuwien.ac.at

Peter Klein
Technical University of Vienna
Studentnr. 8251105
e8251105@
student.tuwien.ac.at

Alexander Kumbeiz
Technical University of Vienna
Studentnr. 1228689
e1228689@
student.tuwien.ac.at

ABSTRACT

This paper provides an overview in the state of the art in crowd sourcing applications. A classification of crowd sourcing systems is shown according to the collaboration provided by users and the tasks performed by them. Then most important issues in building crowd sourcing systems such as recruiting and retaining users, define tasks for the users, combining results provided by users and evaluating their results are presented.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Internet Computing

Keywords

ACM proceedings, LaTeX, text tagging

1. OVERVIEW

Crowd sourcing, which is utilizing a crowd of humans to solve a task that is typically easy to solve for humans but hard for machines, has to handle following 4 major tasks.

- Recruiting and retaining users
- Define what the users should do
- Combining users results
- Evaluate the results

First step in evaluation and categorization of crowd sourcing systems is to classify them according to the type of collaboration and the task performed.

1.1 User Collaboration

Explicit collaboration requires users to willingly participate in creation of artifacts or solving of problems, such as writing Wikipedia articles or reviewing and voting products on Amazon

Implicit collaboration means that users are collaborating by working on other tasks such as playing games or browsing websites.

1.2 Tasks

1.2.1 Explicit Systems

Evaluation Systems let users analyze and comment on items such as books, movies or webpages. The result can be given in textual comments, number scores or tags.

Sharing Systems lets users share items such as products, services or textual and structured knowledge. Systems that share products and services are YouTube, programmableweb.com or CPAN. Systems that share textual knowledge are QA web sites (such as Yahoo Answers) and customer support systems (such as Ask Jeeves' AnswerPoint). Systems that share structured knowledge include Swivel, Google Fusion Tables and many science web sites (such as galaxyzoo.org). Sharing systems can be central such as YouTube or working in a peer-to-peer fashion such as Orchestra.

Building Artifacts Such systems allow users to build textual knowledge bases, structured knowledge bases and software. Example of a textual knowledge base is Wikipedia where users contribute texts and web pages and edit and merge each others contributions. Structured knowledge bases include DBpedia that is based on Wikipedia info boxes, Freebase.com that builds a structured database and Google Fusion Tables that lets users contribute tabular data and collaborate on it by merging tables and commenting on them. Building a structured knowledge base requires selecting a set of data sources, extracting data from them and integrating and merging them. Users can contribute by selecting data sources, integrating structured data, resolving ambiguities and adding inferences. The most well known artifact besides Wikipedia is open source software whether users contribute by writing source code, documentation and test of the software.

Task Execution Systems let users perform collaborative tasks such as searching for missing people on images or running

election campaigns to mobilize voters. A number of crowd sourcing platforms on the web such as Amazons Mechanical Turk help distribute pieces of tasks to a crowd and collect the results.

1.2.2 Implicit Systems

Standalone Systems provide a service so that users utilizing this service collaborate to solve a problem. E.g. the ESP game lets users play a game of guessing common words that describe images, those are then used to label the images. IMDB lets users import movies in a way that it motivates users to rate this movies. reCAPTCHA asks users to solve captchas to prove they are human and uses the results for handwritten digit recognition.

Piggyback Systems utilize user traces of other systems to perform a task. Many such systems have been built on top of search engines such as Google or Bing to predict epidemics or find synonyms. Other examples use traces of purchases to recommend products or use click logs to improve presentation of a web site.

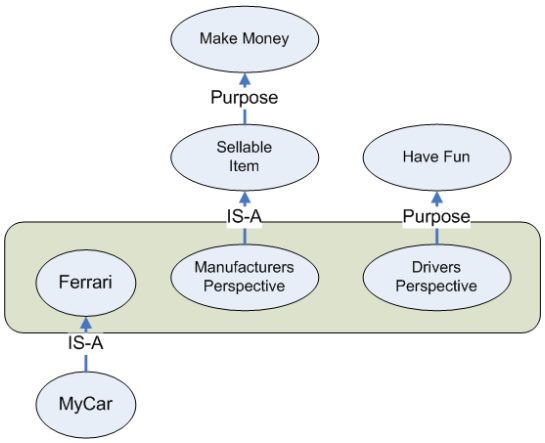

Figure 1: Logic Formulas

2. KEY ISSUES

2.1 Recruiting and Retaining Users

This is the most important crowd sourcing challenge to solve when running such a system. Following solutions can be distinguished.

- *Requires Users* to make contributions. E.g. in an in house system management may order users to enter their contributions.
- *Pay Users* This is a very simple solution but not effective in economic sense if the services provided by the users are not very valuable. E.g. Amazons Mechanical Turk provides a way to pay users.
- *Ask for Volunteers* This is the most popular solution since it is free and easy to execute but it is hard to predict and achieve that enough contributors are found for an application.
- *Let Users "pay" for the Service* This works for implicit systems where users are only allowed to use the real service behind if they have contributed to the crowd sourcing system before.
- *Piggyback on Existing Systems* This gives a steady stream of users but only can only be applied to problems that can be solved by the existing user traces.

When the users have been acquired the next step is to encourage and retain them. Popular schemes are *Instant Gratification* to give users immediate feedback on their contributions and show them the positive effects, *Enjoyable Experience* e.g. to allow playing a game for a contribution,*Reputation* to establish, measure and show fame and trust, this can be a very strong means to encourage users,*Set up Competitions* e.g. show top rated users and provide *Ownership Situations* where users feel they own part of the system.

2.2 User Contributions

In many crowd sourcing system contributions users can made are limited to simple tasks like reviewing, rating or tagging but in more complex systems users do much more cognitively challenging tasks, e.g providing inference rules or resolving controversial issues in building structured knowledge bases.Often users are classified in groups such as guests, regulars, editors and administrators. Low ranking users provide easy contributions such as editing a paragraph or flagging incorrect ones where as high ranking users take the hard parts.

In addition it important to quantify the impact user contributions have on the overall artifact. Flagging an incorrect piece of data has far fewer impact than contributing a wrong inference rule. Only high ranking users should be allowed to make such contributions. Not all contributions in a crowd sourcing system will come from humans, often a split of tasks between them and machines is performed. Humans should concentrate on tasks where they outperform machines, e.g. determining whether two textual or image descriptions are equal to each other.

Finally the user interface should be optimized for the tasks performed by users. Natural language input (e.g. in openmind.com)is easy for users but hard to translate to the crowd sourcing application, formal language is just the other way around.

2.3 Combining User Results

Many crowd sourcing systems do not combine contributions or combine them only in a simple way using numeric formulas. Other systems, e.g. those that build structured knowledge bases or software employ more sophisticated concepts. A key issue is what to do if user contributions differ, e.g. one user is asserting "A" and the other "not A". Automatic solutions employ voting mechanism where contributions are weighted with the trustworthiness of each user. Manual systems let users resolve differences by themselves, unresolved issues are escalated to higher ranking users. They are still the preferred way to solve difficult issues. Especially if also machines contributions resolving issues can become very tricky since machines cannot participate in human discussions to resolve it.

2.4 Evaluating User Results

Crowd sourcing systems have to manage malicious users. Typical techniques are to detect, block and deter them, e.g. only trustworthiness users are allowed to make high impact contributions. Malicious users and contributions can be detected by manual techniques, e.g by users flagging bad contributions. Automatic systems ask e.g. questions where the answer is already knwon and then uses the users answer to compute a reliability measure. Other automatic system base a users trustworthiness on the fame and reputation of the user. Malicious users may also be deterred by "public shaming" where such a user is flagged in view of the other users. Another track is to detect users can provide exceptionally good results and get them rewarded and promoted to higher ranks.

If bad content has creeped into a crowd sourcing system there must be ways to undo it. E.g. in Wikipedia changes are local and logged and can be undone if a malicious contribution is detected. In other cases, such as knowledge bases with inference rules it is much harder to undo changes that had already influenced other contributions without reverting completely back.

3. CONCLUSIONS

To be added.

4. REFERENCES

[1] M. Bowman, S. K. Debray, and L. L. Peterson. Reasoning about naming systems. *ACM Trans. Program. Lang. Syst.*, 15(5):795–825, November 1993.

[2] J. Braams. Babel, a multilingual style-option system for use with latex's standard document styles. *TUGboat*, 12(2):291–301, June 1991.

[3] M. Clark. Post congress tristesse. In *TeX90 Conference Proceedings*, pages 84–89. TeX Users Group, March 1991.

[4] M. Herlihy. A methodology for implementing highly concurrent data objects. *ACM Trans. Program. Lang. Syst.*, 15(5):745–770, November 1993.

[5] L. Lamport. *LaTeX User's Guide and Document Reference Manual*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1986.

[6] S. Salas and E. Hille. *Calculus: One and Several Variable*. John Wiley and Sons, New York, 1978.