Human-powered Sorts and Joins

Summary

In this paper the authors describe how they integrate crowds into a declarative workflow engine called Qurk to reduce the burden on workflow designers. The author focus on how to use people to compare items for sorting and joining data in DBMSs. They describe basic query interface and the user interface of the task they send to MTurk. The author also provide a number of optimizations including task batching, replacing pairwise comparisons with numerical ratings, etc. These optimizations reduce overall cost of running sorts and joins on the crows. Specifically, the authors extended the sort and join tasks in Qurk system, a declarative query processing system designed to run queries over a crowd of workers, with crowd-based filter, join, and sort operators. The executor in Qurk combines human computation and traditional relational processing (e.g., filtering images by date before presenting them to the crowd). Qurk’s declarative interface enables platform independence with respect to the crowd providing work.

Pros

(1) This paper is the first study that propose the approach for implementing the two popular operators, join and sort, of the DBMSs in a crowdsourcing database. (2) The author proposed two batching strategies to reduce the number of HITs for matching operations which is good in database operation.

Cons

(1) The approach proposed in this paper assume that all human beings are alike (i.e., simple model of human errors) which are not correct since different people regarding to age, knowledge, etc will have different behaviors which will lead to different impact on the crowdsourcing. (2) I think that the paper is not good in handling the scalability issue of the database system. In other words, the approach will suffer when the number of records is huge. For example, with the modest data size of 10,000 records with the HIT of size k = 20, the approach will require 5000000 and 250000 HITs.

Suggestion

I would recommend that the author should first have a look the scalability issue of the the proposed approach. Consider Map-Reduce to see whether it is suitable for scalability. If possible, please refer to F1 System from Google for scalability issue.

The second thing I would recommend is to apply the approach with the attention that there will be a more complex human errors instead of simple model of human errors.
Max Algorithms in Crowdsourcing Environments

Summary

This paper examines the problem of retrieving the maximum item from a set in crowdsourcing environments. They first develop parameterized families of max algorithms which take as input a set of items and output an item from the set that is believed to be the maximum. They then propose strategies that select appropriate max algorithm parameters. The framework is reported to facilitate various human error and cost models and they investigate many of them in the experimental section. They evaluate under many metrics, both analytically and via simulations, the tradeoff between three quantities: (1) quality, (2) monetary cost, and (3) execution time. The authors also give the deep concepts on the effectiveness of the strategies in selecting appropriate max algorithm parameters.

Pros

(1) The proposed an effective max algorithm that is capable of handling user mistakes or variability. This feature of the algorithm has much impact on current crowdsourcing environment. For example their method can tackle the case that different input from humans in a comparison task, i.e., pick different item as the best. (2) The algorithms in general provides efficient heuristics and validate them empirically and this can be tuned using parameters like execution time, cost, and quality of the result, i.e., the trade-off between three quantities.

Cons

(1) In order to make the algorithm to work efficient, there are several parameters to tune. This will make the algorithm difficult in using in a specific situation. It may be better if using some technique to automatically searching for the optimizing parameters. (2) The second thing is the paper will be more convincing if the authors can provide some results of the algorithm on real-world databases.

Suggestions

I would suggest that the author should provide more experimental results of the algorithms on real-world databases in order to see the effectiveness of the algorithms.