Computation and Economics - Spring 2012
Assignment #7: Programming the Crowd

Professor Sven Seuken
Department of Informatics, University of Zurich
Out: Friday, April 20, 2012
Due: **14:00** sharp: **Monday, April 30, 2012**
Submissions should be made to `econcs.uzh.spring12@gmail.com`.
For submission format, check description below.

[**Total: BSc 100 Points, MSc 120 Points**] This is a group assignment to be completed by groups of **up to 2 students each**. While you are permitted to discuss your ideas with all students as much as you like, each group must write their own code and explanations. If you want a partner and don't have one, post to piazza as early as possible. Your submissions should be made via email to the course email address as attachments: Code in .java-files and writeup as PDF, sending .zip-archives is also ok.

Goal

Your task in this assignment is to learn about Amazon's Mechanical Turk (MTurk), one of the major Crowsourcing platforms, and then write some small Java programs to solve tasks using the crowd that could hardly be solved by a computer alone.

Introduction

1. [**10 Points**] Reading about MTurk

   In this first exercise you will learn about MTurk and get acquainted with the basic terminology. Read about MTurk from sources of your choice. Here are a few pointers to start with:

   - `http://aws.amazon.com/en/mturk/`
   - `http://aws.amazon.com/documentation/mturk/`
   - `http://en.wikipedia.org/wiki/Amazon_Mechanical_Turk`.

   Briefly answer the following questions.

   (a) [**6 Points**] Why was the name Mechanical Turk chosen? Comment in a few sentences on the slogan "Artificial Artificial Intelligence".

   (b) [**4 Points**] What is a HIT? A worker? A requester?

Setup

Here you will create the accounts on Amazon to be able to create HITs on MTurk. Additionally, you will have to set up a Java project using CrowdLang such that you can start programming your algorithms. CrowdLang is a Java library that allows you to easily manage HITs on the MTurk market place by abstracting away the lower level communication such that you can focus more on the logic of your program. CrowdLang is a prototype developed here at IFI. For more information, go to `http://www.ifi.uzh.ch/ddis/research/CrowdLang.html`.

2. [**6 Points**] Creating accounts and doing some work

   (a) [**1 Points**] Go to `http://aws.amazon.com/` and create an AWS account. The most important information of this account are your security credentials, namely your "Access Key ID" and your "Secret Access Key", which will be necessary to define HITs in your program. These can be found at `https://aws-portal.amazon.com/gp/aws/securityCredentials`.

   (b) [**1 Points**] Also, make an MTurk worker account, as well as an MTurk requester account, at Amazon's MTurk homepage `https://www.mturk.com/mturk/welcome`.

   In addition, make an MTurk worker and requester account for the **sandbox** at `https://workersandbox.mturk.com/mturk/welcome` and `https://requestersandbox.mturk.com/`. You will need these for testing purposes later.

   (c) [**3 Points**] Try to work for 15 minutes on MTurk and make as much money as you can. Share your experiences by providing some evidence/details of your work.

   (d) [**1 Points**] Bachelor students will get a budget of \$12, Master students of \$16, reflecting that Master students have to solve one additional task. Load the correct amount to your requestor account through the VISA number that has been provided.[1]

3. [**2 Points**] Setting up a Java project

   Here, we will walk through the process of creating a Java project and including the CrowdLang library. We recommend using the Eclipse IDE for this assignment, because it makes programming in Java very convenient. Therefore, the following steps describe the setup process for Eclipse. But of course, you are free to use any other method and/or Java IDE if you wish.

   (a) [**1 Points**] If you haven't done so, download and install Eclipse at `http://www.eclipse.org/downloads/`. Open Eclipse and create a new Java project. Copy the files you downloaded from the class homepage into the project, namely `Images.java`, `Nuclei.java` and `crowdlang-marketside-alpha-0-5-1.jar`.

   (b) [**1 Points**] Right-click on the project and select `Properties`. From the list on the left-hand side, select `Java Build Path`, then click on `Libraries` and `Add JARs` and choose the crowdlang jar. Now you should be able to use the CrowdLang library in your Java code.

[1]We will give a way to add funds to your MTurk account soon.

Problem Set

4. [**34 Points**] Counting nuclei

   Some tasks are computationally hard yet astonishingly easy for humans. One such example is image analysis for biological studies. In many cases, biologists get their data in the form of images which need to be processed to extract useful information.

   In this real-world case, we will use Mechanical Turk to count nuclei in an image depicting many different cells. The code resides in `NucleiCount.java`. When ready, it will first create a HIT that will post an image with many cells[2] and will ask workers to count the nuclei. When all responses have arrived, it will create new HITs to validate the answers that were given by the turkers. Finally, it processes the results to give an estimate of the number nuclei on the image.

   (a) [**2 Points**] Familiarize yourself with the code. Provide your access key ID and your secret access key at the right place in the program.

   (b) [**2 Points**] Decide on the design of the counting task. For this, you need to define how much you are willing to pay per accepted HIT and how many turkers you want to assign this task. Fill in the respective piece of code.

   (c) [**8 Points**] For every HIT submission from (b), you have to decide whether to accept or reject it. Implement a rule in the code for this. To do this, you might want to incorporate information about the value that was submitted, but also about the amount of time that was spent on the HIT.

   (d) [**2 Points**] It is usually the case that HIT responses are of low quality, especially for tasks that do not require 'MTurk qualifications'. To alleviate this, for every accepted answer that is provided, we generate a second HIT in which workers are presented the number from the first HIT and then have to vote if they agree with it or not. Again, define the reward and the number of workers that you want to assign this task. Fill in the respective piece of code.

   (e) [**8 Points**] After you have filtered out some bad submissions in (c) and (d) (and hopefully ended up with high quality data), do some post-processing of the remaining submissions to compute an estimate of the number of nuclei.

   (f) [**12 Points**] Now test your code in the sandbox. Use the link provided from the console to navigate to your HIT and solve it by yourself. If that works, run your program on the real market place.

   Explain briefly how your complete algorithm works. Your answer should include the resulting estimate of the number of nuclei. Also provide evidence that your result was indeed created on MTurk (e.g. csv reports, snapshots etc.).

   Note: Before going to the real market place, be sure to check the correctness of your program such that you don't waste your money. Also, take into account that you have a limited budget. So decide on the payment of your HITs carefully. You shouldn't use more than 30% of your budget up to this point.

[2] The URL of the image is `http://www.ifi.uzh.ch/ce/teaching/spring2012/nuclei1.jpg`

5. [**BSc 48 Points, MSc 68 Points**] Image Sorting

   Here, you are presented a slightly more complex problem. You are given a list of images of eight very famous people in this order: 1. George W. Bush, 2. Mother Theresa, 3. Diego Maradona, 4. Madonna, 5. Lady Diana, 6. Nicole Kidman, 7. Mahatma Gandhi, and 8. Albert Einstein.[3] Your task is to sort them according to their date of birth (oldest first) with the help of MTurk.

   If you want good quality in sorting the images, you will need to have a lot of workers and a clever way to process responses (remember MTurk output is noisy). However, more workers will cost you more and you need to stay on budget. Therefore, before you start, you will need to think about how many comparisons will be needed, how many workers per comparison to request for, and how much to pay. Also, time might be a constraint. If you process your HITs sequentially, you might have to wait for a long time until your algorithm can proceed to the next step. Therefore, think of how you can parallelize your HITs. For example, one (not so clever) idea would be to first get the complete ranking of all eight people by asking workers on all pairwise comparisons (possibly several times per comparison). Then you could sort the images based on this information (Do you see why this is approach wastes time and money?). However, by refining this idea you can design more efficient algorithms. Of course, you are free to chose a completely different approach.

   (a) [**10 Points**] Have a look at the sample file `Images.java`. It creates HITs that ask workers to select the older of two people and then, based on the majority rule, decides which one is indeed older. You might want to run it in the sandbox to see how it works. (NB. We have included the names of the people plus the number (e.g. 1: George W. Bush) below the images to avoid confusion among the workers. This way, you should obtain better results.)

       Now think of how you want to tackle the sorting problem. Provide a short description of your algorithm.

   (b) [**38 Points**] Implement your algorithm in the class `ImageSort.java`. Test it in the sandbox. Then decide carefully on the costs and run your program. Your answer should include the ordered list of the above people as the result from your experiment. Also provide evidence that your result was indeed created on MTurk (e.g. csv reports, snapshots etc.).

   (c) [**MSc 20 Points**] Now that you have some experience sorting images with MTurk, your task here is to implement a sorting procedure that uses a rank aggregation algorithm. Rank aggregation integrates the individual complete orders of the turkers into one global order. For this, you need to create a HIT that asks the workers to order *all eight people* at once, i.e. they need to return a list like "$Bush < Theresa < Gandhi < Einstein < Kidman < Diana < Maradona < Madonna$". After you have collected enough of such single rankings, you have to combine them into a global order. To aggregate the information, use one of the rules described in chapter 12 of the lecture notes, i.e. Borda, Dodgson or Kemeney-Young.

[3] The images can be found at `http://www.ifi.uzh.ch/ce/teaching/spring2012/picture1.jpg`, where picture1 can be replaced by picture2 etc. to get the other images.

Of course, the payment for the workers has to be higher for this task since it requires more effort. Also, you need to decide how many HITs to create to get useful results.

Provide a short description of your algorithm. Write your code in the class `RankSort.java`. Your answer should include the ordered list of the above people as the result from your experiment. Also provide evidence that your result was indeed created on MTurk (e.g. csv reports, snapshots etc.).

Hint: A good way to obtain the ordered list from the workers might be to create a textbox (similar to the nuclei task) and ask them two write the numbers associated with the names in the pictures in the correct sequence, e.g. "$Bush < Theresa < Gandhi < Einstein < Kidman < Diana < Maradona < Madonna$" would result in $1, 2, 7, 8, 6, 5, 3, 4$.

Tips, Comments

- As a requester, please accept *all* tasks completed in a reasonable way. Otherwise, this will affect your reputation as a requester and possibly other fellow students who will be running similar experiments.
- After you have implemented your program for a specific task, be sure to test it in the **sandbox**.
- Your budget is quite big. This is to prevent you from running out of money too early should you make a mistake (e.g. bugs in your code, writing unclear HITs and getting rubbish data back). So you can afford **one** failure and start again. Nevertheless, before you go to the real market think carefully about how you formulate your HITs, how much money you want to spend per HIT and how many HITs you want to create.
- Don't underestimate the time it can take until you get the results from your HITs back. So start early with your experiments.
- If you are under time pressure, and you want workers to complete your tasks more quickly, incentivize them with more money (the more you pay, the faster you will find workers). But make sure you get all tasks completed with the given budget.
- You can manually inspect (and manage) your HITs at `https://requester.mturk.com/mturk/manageHITs`.
- Please document your code well. Use clear variable names and make comments at points where it is hard to understand what you are doing.
- Remember to provide evidence of work on MTurk in order to receive full grade. We are looking for careful designs and completely automated executions on MTurk. No manual work should be used to solve the problems. In particular, do not hardcode any knowledge about the images into your code, of course.
- Please report any bugs on NB. We will try to fix them as quickly as possible.
- If anything is unclear, ask a question on NB.