Piazza Q & A System Simulation

CS482/IE413
University of Urbana - Champaign

Anirudh Viswanath (viswnth3)
Yuan Shen (yshen47)
Zehua Li (zehuali2)

Table of Content

Problem Description **2**
- System Description 2
- Purpose 2

Simulation Design **3**
- Model Description 3
  - Data 3
  - Assumptions 3
  - Entities and Attributes 4
  - Traced Variables 5
  - Variables 5
  - Events 5
- Model Analysis 7
  - Model Validation 7
  - Model Verification 8

Result Analysis **8**
- Simulation Setup 8
- Outcome Analysis 8

Conclusion **10**
- Possible Sources of Error 10
- Improvement 10

Appendices **11**
- Source of Data 11
- Python Program 11
- Reference for Statistical Analysis 11
- List of Graphs and Outputs 12

Problem Description

System Description

Piazza is a widely used online Q & A platform for instructors (including lecturers and TAs) and students to share course-related resources and to interact with each other. Piazza provides rudimentary statistics feedbacks (such as average responding time, top viewed posts, top active members, etc.) for each course, partly for promoting its efficiency. However, it doesn't distinguish between responses from students and instructors, nor does it assess the efficiency of the any particular instructor, or address the concerns of whether the course needs more/less instructors, or more/less frequent instructor interactions.

Purpose

Our team would like to sample a few accessible courses that use Piazza, and focus on the Q & A aspect of the platform (disregard the course information and notes). We will evaluate the distribution of inquiries, and the response latency distributions of the instructors and students. Via modeling and simulation, our main goal is to deduce some meaningful results regarding the efficiency of the instructor schedule for these courses, and possible suggestions for improvements. Our measure of efficiency included the number of unresolved questions throughout the entire simulation process, the waiting time for a question to be answered, and the fraction of time instructors are idle. Using these three metrics, we hope to we wish to discover some patterns regarding the system, and ascertain the optimal amount of instructors to use.

Simulation Design

Model Description

Data

In order to model the system we first examined the data available to us. Via the Piazza-API implemented by Hamza Faran, we are able to gather information including but not limited to:

1. The entire content of a given post
2. The type of a given post (i.e. is it a note or a question?)
3. The category a post belongs to (i.e. is it related to homework, quiz, lab, or test?)
4. The time a post is posted (and edited)
5. The responses to the post
6. The time a response is posted
7. The author of the response (which can be anonymous, if by choice of student)

Data collection is done via a custom-written python program. It collected the time interval at which new questions arrive by recording and comparing their creation timestamps in order. It collected the latencies for instructors to resolve the questions. It collected the percentage of questions answered by students and those answered by instructors. It also collected information such as the time of the day a question is posted, and the distribution of questions throughout the span of a certain event (hw, test, lab, etc.), but not all of this information was used in the modeling.

Assumptions

Unlike a conventional queuing system we see in physical world, or in industries, there are no set schedules for instructors to check and resolve the questions online, and we have no way of knowing when the instructors are checking the platform. This is a common characteristic of online platforms, and is one of our major challenges in interpreting the data and modeling the system. **We had to assume that the instructors are always available**. The potential inaccuracy introduced is modulated by the distribution of latency for an instructor to resolve a question. This, however, means that although we have detailed distribution of the time of a day a question is more likely to be posted, we cannot easily gain any statistically significant insights into the schedule of instructors within our model at that level of detail.

Even though we collected data regarding the individual types of events (machine problems, labs, homeworks, and tests), due to the size restraints of sigma, we were unable to model the separate distributions of questions generated by these events, and instead **resorted to making them a single event, and assume no difference in the ways that an instructor answers these questions**. In order to counter the increased complexity introduced to the system, and increase accuracy, we resorted to using the course with the most data available, CS 241, which has around 4000 posts in a given semester, and running the simulation in trace mode, with the timing data from the most recent iteration of this course.

Due to the fact that students can choose to be anonymous, **we model all the students to be indistinguishable**. Due to the fact that Sigma has a limitation to the complexity, although we can analyze the instructors individually, **we model them as indistinguishable as well.**

Realistically, a certain amount of questions are never getting answered, due to various reasons such as them not being valid questions, being taken back, or being valid yet remained unanswered until it would be relevant. In this model, the percent of questions unanswered can also come from insufficient number of instructors, and we can't simply use a probability to safely discount questions, therefore **we assume all questions to be answerable, and remain in the queue until answered**.

Entities and Attributes

Our simulation model contain the following entities (**see appendix page 12 for diagram**) :
Temporary: Piazza question posts (modeled by question queues)
Permanent: Number of Instructors (modeled as indistinguishable servers), students (who answers questions, modeled as a single server).

Their attributes are as follows:
Interarrival time for question posts (Trace of historic data).
Instructor's time interval (seconds) [see page 14 of appendix for graph] to answer a question, distributed lognormal with mean = log(1726.1971) = 7.454, standard deviation = 2.235, and a positive shift of 15.345, which compensates for the fact that minimum time needed for answering the question is not 0 second.
Instructor state: idle or busy answering question (modeled with number of idle instructors).

Probability of a question getting answered by students. Some Piazza posts can fully rely on students to answer. If not answered (correctly) by a student, then they requires instructors to answer.
Student's time interval (seconds) to answer a question, distributed lognormal with mean = log(1169.5476) = 3.068, standard deviation = 1.755, and a positive shift of 4.722, which compensates for the fact that minimum time needed for answering the question is not 0 second.

Traced Variables

We traced the number of unresolved questions throughout the whole simulation process, which can measure the efficiency of the instructors' schedules.
We traced the number of idle instructor, which can measure the efficiency of the allocation of instructors.
We also traced the waiting time for each question to be answered by instructors, again, to measure the efficiency of the instructor schedule.

Variables

IRMN	Description: number of idle instructors	Type: INTEGER
MPQ	Description: question queue	Type: INTEGER
FLAG	Description: temporary U(0,1) random variable	Type: REAL
RATIO	Description: The ratio of posts answered by student	Type: REAL
IMPQ	Description: instructor question queue	Type: INTEGER
PSINDEX	Description: index for PARRIVAL array	Type: INTEGER
PEINDEX	Description: index for PEND array	Type: INTEGER
PAVG	Description: average waiting time for a post to be answered	Type: REAL
PARRIVAL	Description: an array of post arrival time	Type: REAL
PEND	Description: an array of post answered time	Type: REAL

Events

Name: **INIT**
Description: Initialization, Schedule event P_QA, Schedule S_start
State Changes: IRMN = 0,PSINDEX = 0,PEINDEX = 0, FLAG = 0, RATIO = 0.35, IMPQ = 0

Name: **P_QA**

Description: Post arrival, Schedule P_QA in Disk{POST.DAT, 0}
State Changes: PARRIVAL[PSINDEX]=CLK,PSINDEX=PSINDEX+1

Name: **S_start**
Description: Students start to answer the posts and decide if the post need instructors participate to answer. Schedule S_end if FLAG > RATIO. If it is not, then schedule event I_P_QA
State Changes: FLAG=RND

Name: **S_end**
Description: Students finish answering posts.
State Changes: PEND[PEINDEX]=CLK-PARRIVAL[PEINDEX], PSUM=PSUM+PEND[PEINDEX], PAVG=PSUM/(PEINDEX+1), PEINDEX=PEINDEX+1

Name: **I_P_QA**
Description: Enqueue a post to instructor answering queue. Schedule I_start if IRMN > 0
State Changes: IMPQ=IMPQ+1

Name: **I_start**
Description: Instructors start to answer posts. Schedule I_end in Disk{IA.DAT, 0} hours.
State Changes: IRMN=IRMN-1, IMPQ=IMPQ-1, ITIME=DISK{IA3.DAT;0}, PEND[PEINDEX]=CLK+ITIME-PARRIVAL[PEINDEX], PSUM=PSUM+PEND[PEINDEX], PAVG=PSUM/(PEINDEX+1), PEINDEX=PEINDEX+1

Name: **I_end**
Description: Instructors finish answering a post.
Schedule I_start if IRMN > 0 and IMPQ>0
State Changes: IRMN=IRMN+1

Model Analysis

Model Validation

We use historic data of question posting times to preserve the distribution driven by events that occur during the semester. This is a fixed schedule and should have a relatively good representation for the particular course (CS 241) we chose. However, the results from this model may not be generalizable to any courses on piazza, and to simulate for other courses, data native to those classes should be used.

We fit the distribution from historic data for the latency of instructor answering, latency of student answering, and probability of student answering a question and subsequently doesn't require instructors to answer. The data used to calculate the distribution are gathered from Piazza. Posts are filtered to see if they are questions, and if they get a corresponding answer type, the latency will subsequently be recorded for the type of the answer. For calculation of the probability, a boolean counter checks to see if a question is answered by a student, and not answer by an instructor.

The distributions for the two latencies are fitted lognormal with maximum likelihood estimation(MLE), and pass the Kolmogorv - Smirnov test:
KS Test for instructor answering latency for lognormal with parameter from MLE:
$D = 0.02233399717088777,\ p\ value = 0.46730865000455807,\ D_{0.05} = 0.035789473684210531$

The null hypothesis is not rejected.

KS Test for student answering latency for lognormal with parameter from MLE:
$D = 0.031182828374802463,\ p\ value = 0.22974685862538746,\ D_{0.05} = 0.040968315852756905$

The null hypothesis is not rejected.

The probability of student answering a question and subsequently doesn't require instructors to answer is calculated to be 35.002%

The lognormal distribution is used in many natural and scientific analysis, such as neuron firing rate, and length of post online. More fittingly, it is hypothesized to be the dwell time for users on online articles. Since these conditions are fairly similar to those we are interested in for this simulation, it provides us with strong reasons to model the latency for answering piazza posts with the lognormal distribution.

Model Verification

We completed a couple of steps to verify our code and ensure that our model is accurate and realistic. After analyzing our code thoroughly to make sure that we hadn't made any mistake, we carried on with further verification techniques. We tracked 3 variables as a test of the model, and the event/state changes made during the simulation run were normal, and therefore, we could safely assume that our code was indeed working. As a further test, the simulation was also put through extreme values, but the code did not act unreasonably. We also ensured that the priorities of each event was set correctly to avoid having "phantom" posts. Hence, after performing all the above techniques, we can confidently assume that our code is correct and our model will perform as we expect.

Result Analysis

Simulation Setup

We ran the simulation with 5, 10, 15, 20, and 25 instructors, each with 3 different trials.
Each trial has a different seed for generating the answering time for instructors and students, and for whether a question is answered by an instructor or a student. The timing data for question arrival is the trace from the spring 2017 iteration of CS 241, up to April 28th.

The random variable for the probabilities are generated uniform(0,1), and the answering times are generated lognormal with Python's scipy.stats API's lognormal random number generator. To generate lognormal random without it, one can first generate normal random variables, then take the exponential to obtain the lognormal random numbers.

Outcome Analysis

We plotted and calculated the change in the number of idle instructors, the number of questions waiting to be answered, and the mean time the question required to be resolved, using a batched mean (100 recordings per batch, 100 batches) approach for better accuracy and less variance.

The percent of time that all instructors are busy is

# instructors	5	10	15	20	25
% all busy	35.4576%	3.1667%	0.0287%	0.00000%	0.0000%

The mean (recorded and calculated in Sigma) of time required to resolve a question is

# instructors	5	10	15	20	25
Time (hours)	10.893	4.132	4.079	4.104	4.087

The average number of questions waiting to be answered. Range of Confidence Interval 95%

# instructors	5	10	15	20	25
Mean Range	(29.01517, 37.03483)	(0.010592, 0.349408)	(0.147798, 0.14509)	(0.146294, 0.153706)	(0.156806, 0.163194)

There are a few interesting findings from our simulation:

1. Increasing the number of instructors stopped making significant impact on the mean answering time, after it reached 10.
2. The percent of time instructors are all busy converges to 0 after the number of instructors is higher than 15, meaning a good number of instructors won't be actively participating after that point.
3. Increasing the number of instructors also stopped making a significant impact on the number of questions waiting to be answered, after it reached 15. (10 is very chaotic, possibly because that the instructors gets really busy some of the times and really free some other times, making the variance very high.)
4. This leads us to conclude that most of the time (except for when especially difficult assignments take place, which are reflected on sudden increases of queued questions), 10 instructors are enough to make sure that there is at least one idle instructor. We deduce from the above, that the current number of instructors (38 of them) is a lot more than necessary from the point of maintaining the Piazza Q & A queue. Without considering other duties for TAs, a possible suggestion is to deploy around 10 TAs responsible for the Piazza platform, and 5 TAs as graders, who comes to help with the platform only when difficult assignments are released.

5. Additionally, the average response time calculated by Piazza disagrees with the mean found in our collected data (which agrees with the result from our simulation). To be more specific, Piazza's value, around 12 minutes, is way shorter than our value, around 4 hours. Piazza's calculates of the average response time is closer, however, to the mode after interpolating, or the median with sampling or filtering. This does not discount either of these statistics, but shows different approaches that can be applied on the same set of data. Piazza's value is certainly more appealing to the users, but does not reflect the fact that many posts take a very long time to be answered.

Conclusion

Possible Sources of Error

We only used one semester of data for the trace. Although the structure of the class doesn't vary, it still means that the applicability of this simulation is limited. We also didn't consider impacts of follow-up discussions on the time-consumption of instructors and students. If a significant number instructors spend way more time on follow-ups than the questions, yet another group of them does the opposite, the distribution for the answering time may not reflect the real situation.

Improvement

While our simulation model ran fine, there are some avenues we identified where we could've improved the model. Had we done things differently, we would have been able to apply many different variance reduction techniques. For example, had we simulated a distinct, separate event for exams, MP's, homeworks etc., this condition would have given us a much greater insight into the different distributions. Also, if we had used a parametric probability model instead of using trace variables, we would've been able to run the model on a daily cycle, and therefore would've given us much better results. Finally, by distinguishing the servers, we would be able to know which post goes to which server, and therefore through conditioning, we would have been able to reduce variance.

Appendices

Source of Data

Piazza website, course CS 241 of UIUC (spring 2017). Retrieved April 18th 2017, https://piazza.com/

Hamza Faran's Github account (2016, September 12). Retrieved April 20th 2017, https://github.com/hfaran/piazza-api

Python Program

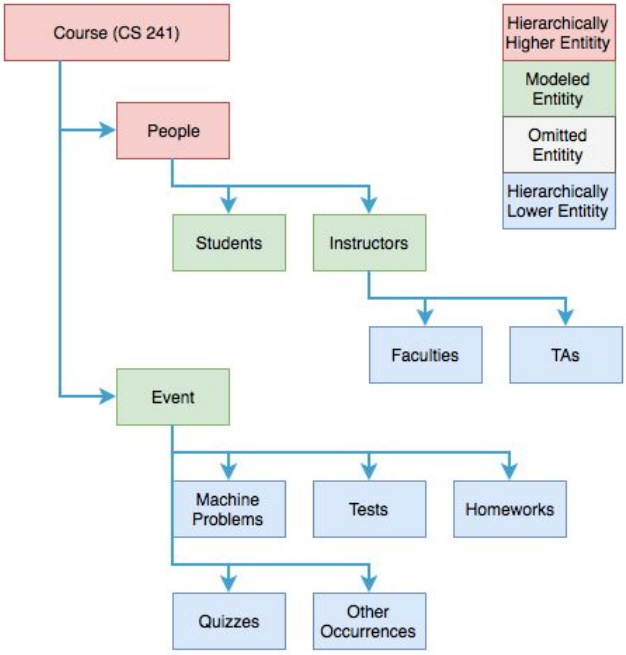
The used Python Program for this project is uploaded to here: https://github.com/Code-Omega/Piazza-analytics/blob/master/Piazza_analytics.ipynb

Reference for Statistical Analysis

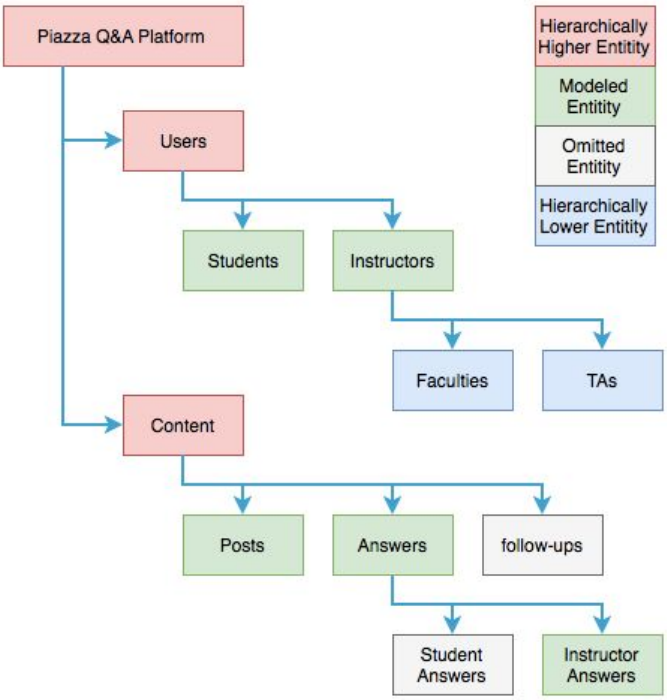
K-S Test. Retrieved April 29th 2017, http://www.itl.nist.gov/div898/handbook/eda/section3/eda35g.htm

Maximum Likelihood Estimation (2017, April 21). Retrieved April 29th 2017, https://en.wikipedia.org/wiki/Maximum_likelihood_estimation

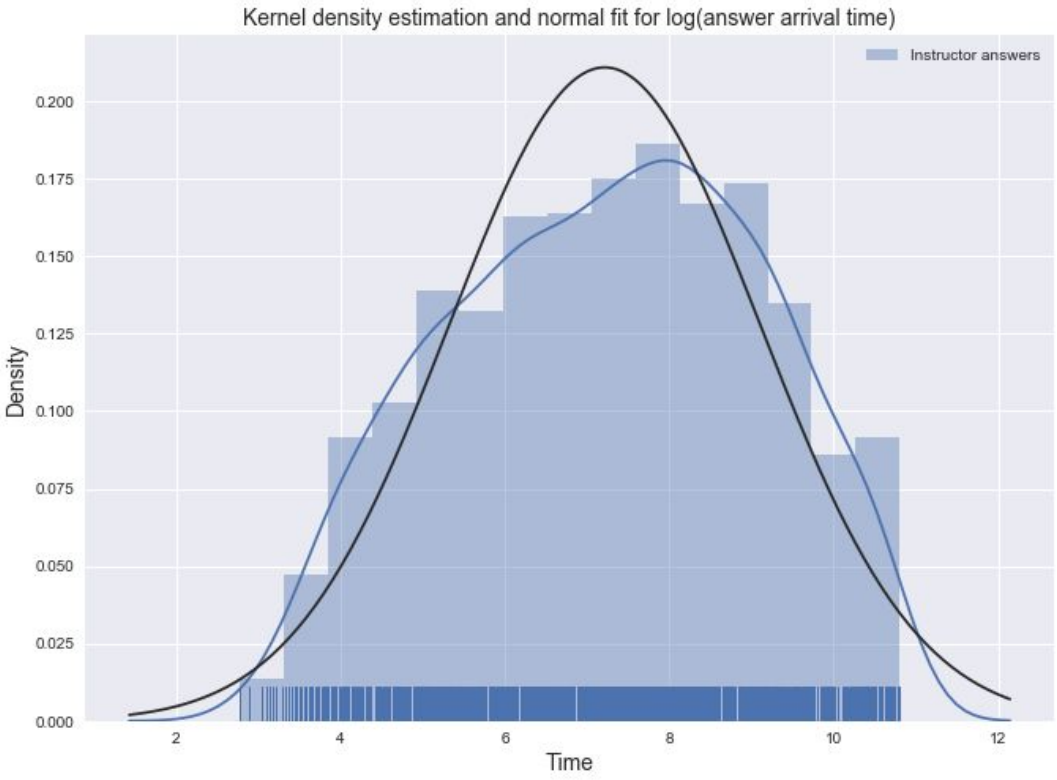
List of Graphs and Outputs

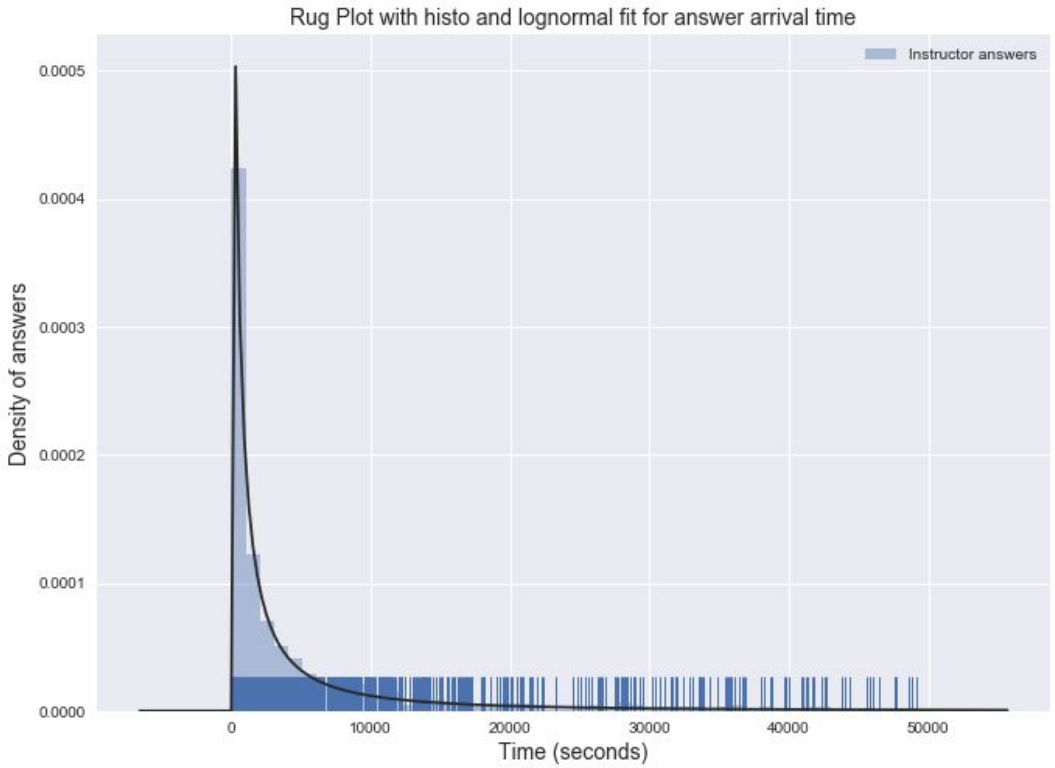


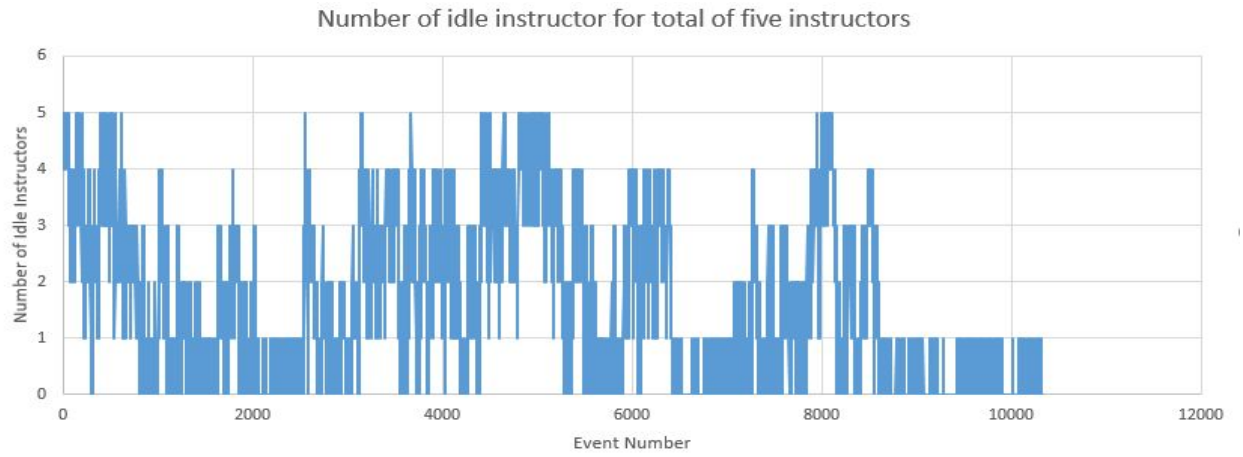
ENTITY HIERARCHY CHART

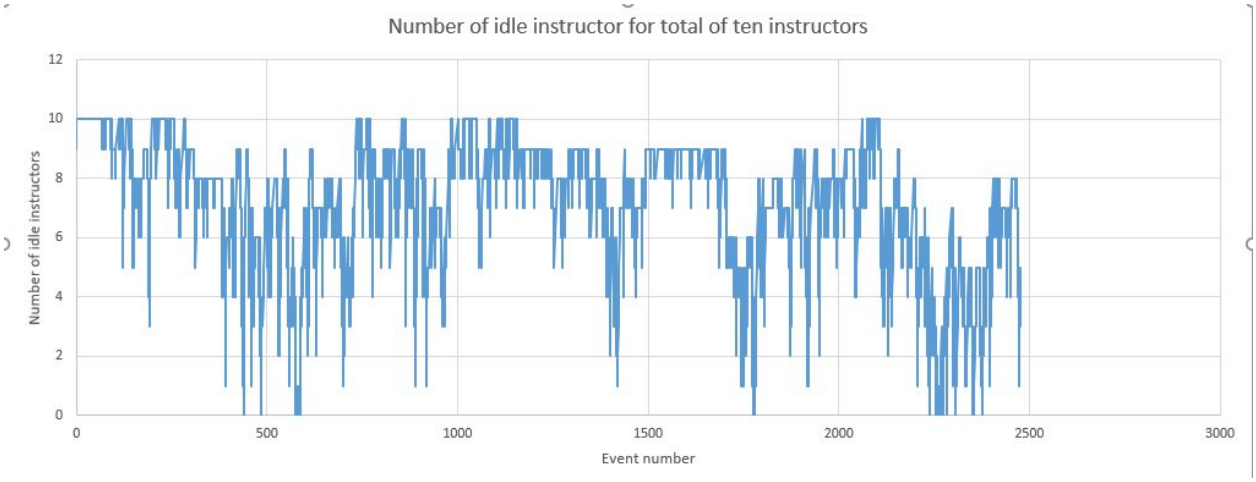


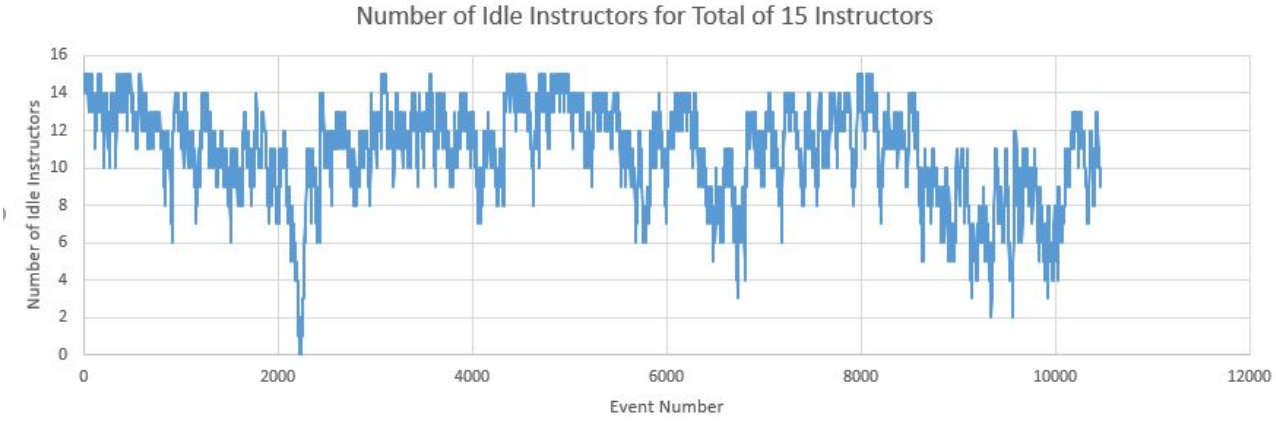
ENTITY HIERARCHY CHART

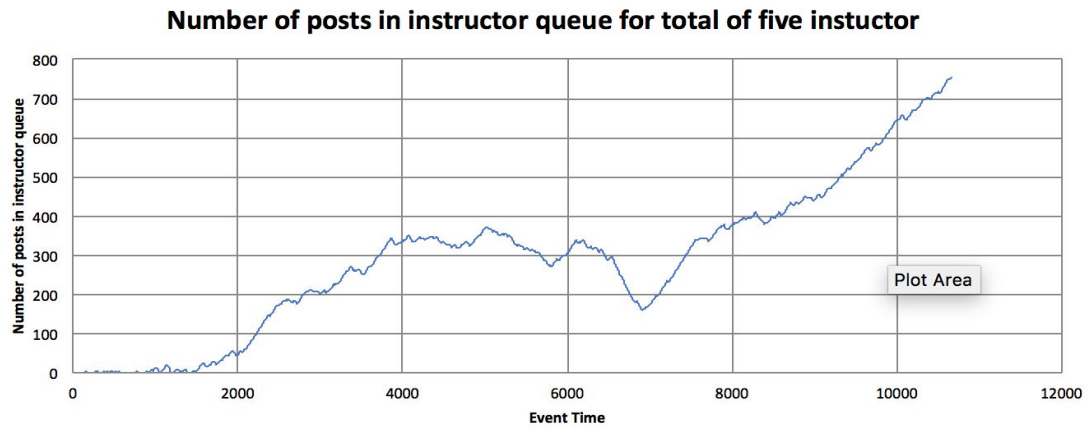

Kernel density estimation and normal fit for log(answer arrival time)
Instructor answers
Density
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
2
4
6
8
10
12
Time

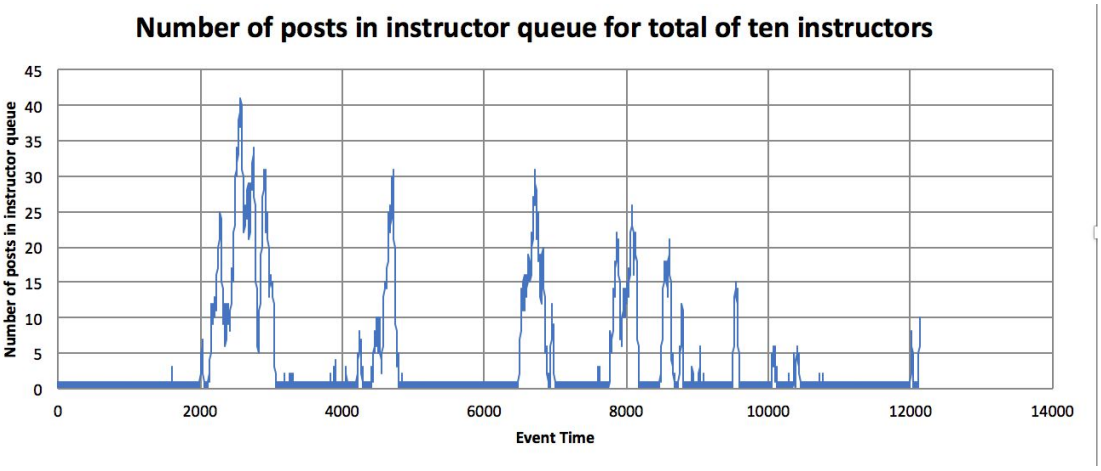

Rug Plot with histo and lognormal fit for answer arrival time
Instructor answers
Density of answers
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
10000
20000
30000
40000
50000
Time (seconds)

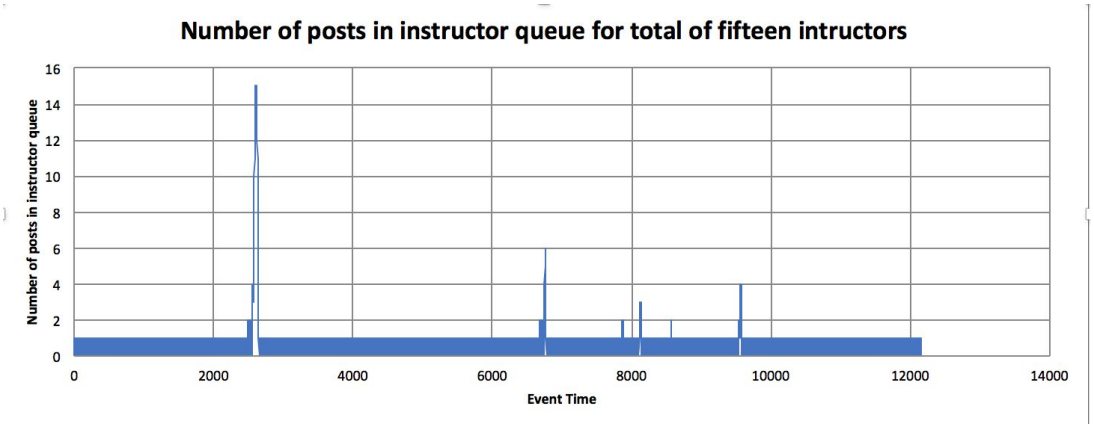

Number of idle instructor for total of five instructors
Number of Idle Instructors
Event Number
0
1
2
3
4
5
6
0
2000
4000
6000
8000
10000
12000


Number of idle instructor for total of ten instructors
Number of idle instructors
Event number
0
2
4
6
8
10
12
0
500
1000
1500
2000
2500
3000


Number of Idle Instructors for Total of 15 Instructors
Number of Idle Instructors
Event Number
0
2
4
6
8
10
12
14
16
0
2000
4000
6000
8000
10000
12000


Number of posts in instructor queue for total of five instuctor
Number of posts in instructor queue
800
700
600
500
400
300
200
100
0
0
2000
4000
6000
8000
10000
12000
Event Time
Plot Area


Number of posts in instructor queue for total of ten instructors
Number of posts in instructor queue
45
40
35
30
25
20
15
10
5
0
0
2000
4000
6000
8000
10000
12000
14000
Event Time


Number of posts in instructor queue for total of fifteen intructors
Number of posts in instructor queue
16
14
12
10
8
6
4
2
0
0
2000
4000
6000
8000
10000
12000
14000
Event Time