# Video streaming model

Richard G. Clegg, Raul Landa, David Griffin, Miguel Rio

This document describes the mathematical underpinnings of the work on the video streaming system. The simulation is split into a number of separate models:

* Demand model – when and where do users join the system and when do they leave.
* Stream model – how do individual users assign themselves to a stream which communicates by video (do they all join the same stream, join several individual streams...)
* Network model – where are data centres located in the world.
* Server model – given a number of data centres and a set of user demand split into “streams” which data centres can the users actually use.
* Routing model – given a number of communicating users and allowed data centres, how will traffic between the users be routed.
* Quality of Experience (QoE) model – given set of users and a set of routes for the traffic between them, what QoE will the users experience.
* Cost model – given a set of users and a set of routes for the traffic between them what will the cloud operator charge for that dat.

## Demand model

The demand model has two components: a system which determines when and where arrivals will occur and a system which determines how long those arrivals will stay in the system (and hence where departures will occur. Some core simplifying assumptions are made:

1. Arrivals can be modelled as a series of time varying Poisson processes (processes which are Poisson with an arrival rate which varies as a function of time) at a finite number of fixed geographical locations.
2. Departures are modelled by assuming a lognormal distribution of session times. That is, an arrival stays for a time with a lognormal distribution. These session times are assumed independent of the arrival characteristics.
3. The arrival process is further broken down into a geographical and a temporal component and these are independent.

This model will produce as an output, arrival and departure times with associated latitude and longitudes.

Assume *N* locations on the Earth’s surface are specified in terms of latitude and longitude and also an associated rate multiplier so that each location has a triple (*xi,yi,li*) where *xi* is the latitude *yi*is the longitude and *li*is the rate multiplier for that location. Further assume a daily periodicity where a day is split into *M* equal time units and each time unit has an associated rate multiplier *hi* (this will be referred to as “hour” although any time period can be used). Let *h(t)* be the time period associated with time *t*. Finally, let *λ* be the unmodified rate of arrival (without multipliers). So the rate of arrival at the *i*th location during the *j*th time period is given by . For the *j*th time period then the total arrival rate is . Because of the independent nature of the Poission process then at time *t* the next arrival can be calculated by the following algorithm.

1. Set the time and “hour” T:= t, H:=h(t)
2. Calculate the current global rate .
3. Draw uniform random number *U* which has exponential distribution rate *r*.
4. If *T+U* is within the time period *H* then *T+U* is the time of the next arrival. Exit.
5. Set *H:=H+1* and *T* to the start time of period *H+1*.
6. Go to step 2).

Having calculated the time at which the arrival occurs the location of the arrival is calculated with random selection between the locations with location *i* having probability .

The departure time for this arrival is set by drawing a number from a lognormal distribution where and are parameters. The distribution has mean , variance

and median . The lognormal distribution is often used for session durations.

Together the arrival process and the session length distribution imply a mean population at a given time.

### Inputs related to demand model

The demand model is completely specified by the following inputs:

– the set of lat, longs and arrival rate multipliers for locations 1 to *N*. However, an additional marker for the continental zone can be useful in calculating the delay (as will be described later).

– the set of arrival rate multipliers for time periods (hours) 1 to *M*.

– the overall arrival rate.

– the parameters for the lognormal session duration process.

### Determination of data-driven static parameters

The parameters will be calculated on two steps, so that where is an scenario-independent rate representative of the difference in density of users for various geographical locations, and corresponds to the proportion of users in locale participating in scenario.

We calculate as follows. First, using the *CIA Internet Users dataset* [CIA] we can obtain what proportion of total Internet users are associated to each country. In particular, if we define as the total number of Internet users globally, we define as where corresponds to the proportion of Internet users living in country and locale . We can hence write

,

where is the proportion of Internet users residing in country , and is the proportion of Internet users from country living in locale . For our purposes, we will define each locale as one of the top cities by population in the *GeoNames database* [GeoNames]. Since this database contains 122,952 place bindings representing cities with 1000 inhabitants or more, we are confident that it can provide sufficient resolution for our purposes. As we explained before, we will obtain directly from available data. Conversely, we will estimate from our extensive delay measurement dataset [Networking, ICCCN]. To achieve this, we posit that corresponds to the proportion of DNS servers in country that are located in locale .

Regarding , we will derive it from data (if available) in the following way. First, we define as , that is, the proportion of users participating in the scenario that live in country and locale . If we could get access to datasets in which is directly recoverable for the set of locales that we are using, this definition would be enough for our purposes. Unfortunately, most datasets only include data dis-aggregated to the country level. In this case, we assume that the proportion of Internet users on a given locale that participate in the scenario is equal to the proportion of Internet users in country that participate in the scenario, where of course locale is located within country . Formally, this means that we assume that , and hence, that . Hence, we have that , where the country is implied by the locale .

As before, we express in terms of conditional probabilities to obtain that

.

### Parameters for the Poker model

For the poker scenario, we use the *Online Poker Database of the University of Hamburg* [OPD-UHH] to estimate . This dataset includes data on more than 6 million users spread over more than 150 countries, making it one of the most extensive studies of its kind. Hence, we consider it is broadly representative of global online poker use, and extract our conditional probabilities from it directly.

#### Simulation of Arrival Rates

The poker model parameters are generated in the following way.

This gives the triple for a given location.

The arrival rates for time periods are taken from observations of the Poker site and BT call distributions

The parameters are tuned to match the mean session length and other data about the scenario described in a later section.

Finally, the parameter is tuned to give the correct overall mean population.

### Parameters for the MOOC model

Unfortunately, there are no studies on MOOC usage that can be considered representative of global MOOC usage. Instead, there are a number of small-scale studies for specific courses. Hence, we propose an algorithm to take the outcomes of these partial studies and integrate them into a single frequency distribution that can be used to estimate .

Most previous studies [FPRO, MOOC, ATR1, ATR2, KNCT, EDXC, IHTS, CS50] report the Top-N most frequent countries that the participants report as their country of residence. Hence, most studies only report country information for a fraction of their probability mass. Our algorithm proceeds in two steps.

First, for each one of these course-specific datasets we allocate the probability mass outside the Top-N to those countries not reported in the dataset. To account for Internet penetration and other related factors, we do this proportionally to the number of DNS servers in each country as revealed by [ICCCN, Networking]. However, a simple proportional allocation could lead to a probability mass assignment that violates the Top-N ranking provided in the original studies. Hence, we propose a mechanism to do this in such a way that we ensure that the Top-N ranking provided in the original study is maintained. We denote this mechanism as *partial distribution estimation*.

Second, we combine all of these datasets into a single per-country distribution of MOOC users. We denote this mechanism as *total distribution synthesis*. We now explain these two algorithms.

#### Partial Distribution Estimation

Let the Top-N list used as a data source account for a proportion of the total probability mass, and let the list of countries not included in the Top-N, sorted in decreasing order of DNS server population, be denoted as . We denote the -th element in . Further, let each element be associated with a probability mass that represents the proportion of MOOC users residing in country . Let the proportion of global DNS servers in country be denoted , and let the probability mass associated with the last element in the Top-N list be denoted as . Hence, in order to preserve the Top-N ranking, we require that for all country ranks . We now propose an algorithm to find in such a way that it is proportional do but also respects the Top-N ranking.

do

{

}

Intuitively, the algorithm iteratively assigns a proportion of the remaining probability mass to the country . It does so in a continuously decreasing fashion, thus ensuring that the Top-N ranking is respected, and proportionally to the proportion of DNS servers . The probabilities arising from this procedure for each one of the partial datasets [FPRO, MOOC, ATR1, ATR2, KNCT, EDXC, IHTS, CS50] are then used as an input for the algorithm described in the following section.

#### Total Distribution Synthesis

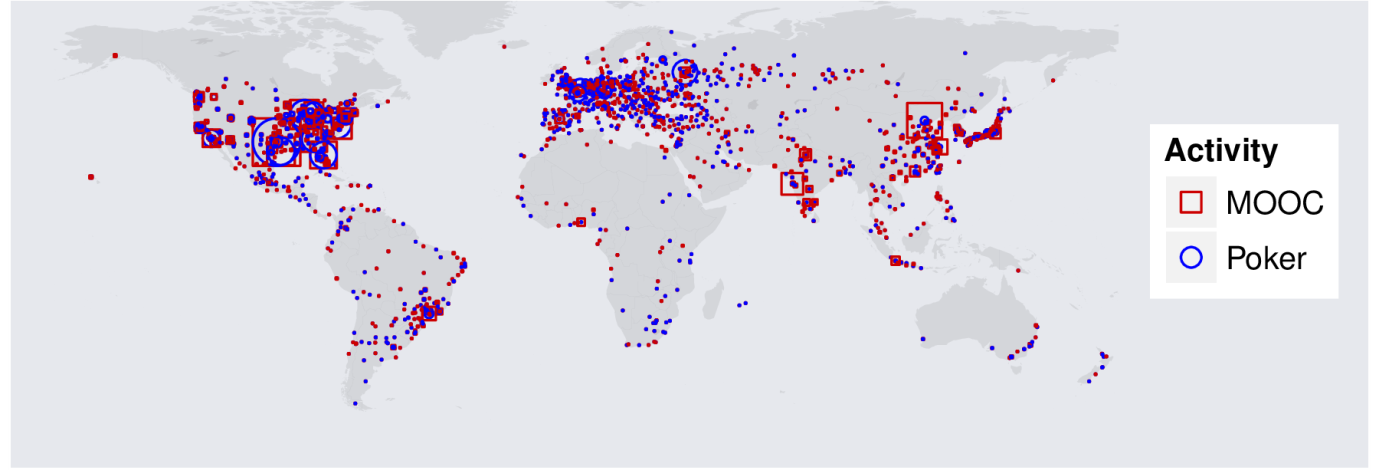
Once that a complete set of probabilities is available for each study , we simply incorporate it into a single probability by means of a convex combination. Formally, if we extend our notation so that denotes the set of probabilities estimated for dataset , have that

Where corresponds to a weight so that . In our case, we define as the total proportion of the combined MOOC student sample that corresponds to dataset . Hence, datasets with greater number of participating students will have greater weight in the determination of . Once this probability has been ascertained, we proceed as in the previously described poker model.

#### Overall rate

The overall rate *λ* allows the tuning of the system overall to produce a mean number of users per day which allows experimentation over a variety of ranges.

#### Overall demand by location



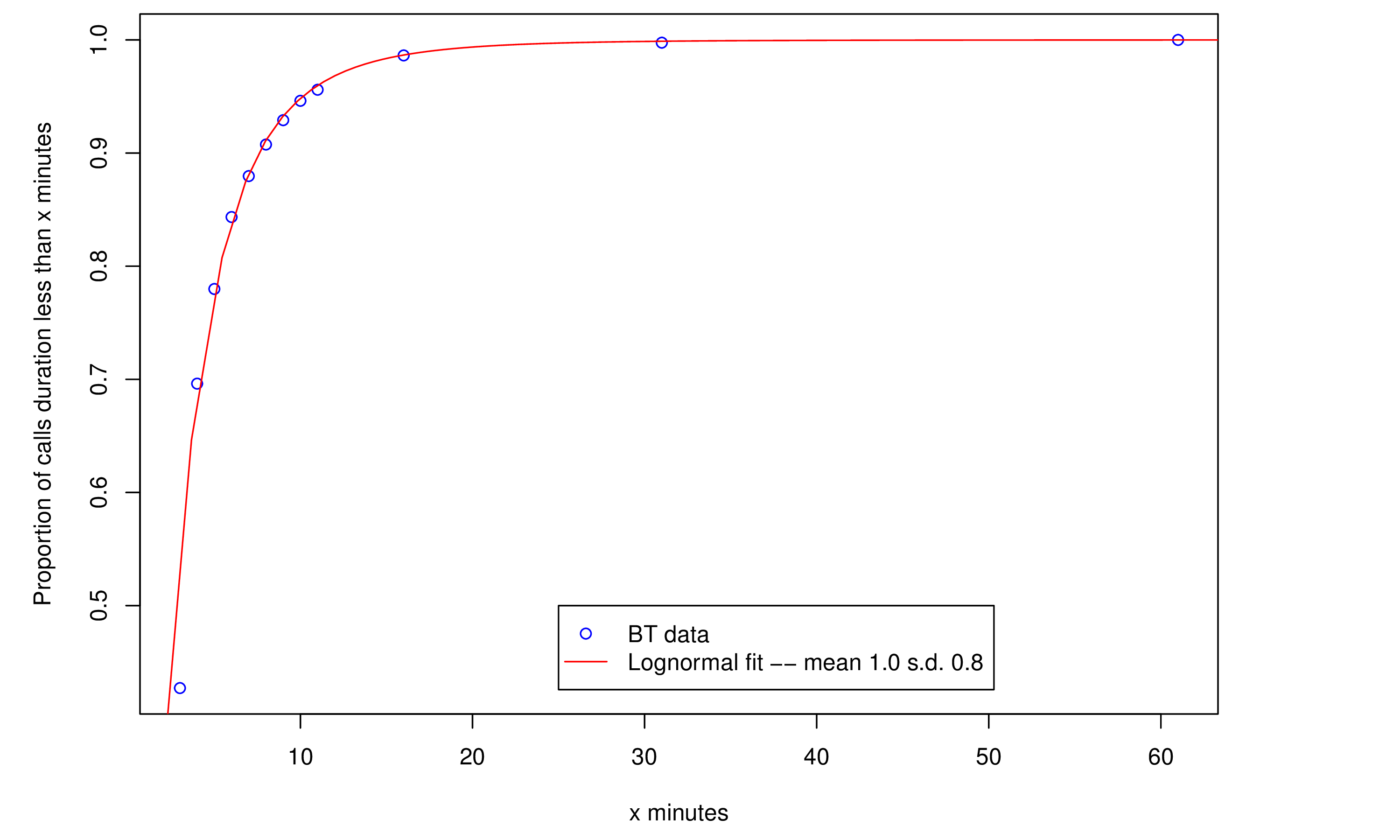
This graph shows the locations and rates for the MOOC and Poker demand models. The red and blue shapes represents demand for MOOC (red square) or Poker (blue circle). The size of the shape represents the relative size of the demand. So, large demand can be seen in the east and west coast of the US and some cities in Europe for both Poker and MOOC. Large demand for MOOC but not Poker can be seen in some Chinese and Indian cities.

#### Session durations and time of day behaviour – Poker

The following informed the session durations for the Poker model. A poker player was studied in [POKER]. His average session length was 2 hours. 10% of his sessions were 30 minutes or less. 2% of his visits are > 5 hours. This was fitted as well as possible to a lognormal distribution. A distribution with a mean of 2 hours and a shape parameter of 0.6 had 10% of sessions 46 minutes or less and 2% of sessions longer than 5.7 hours.

Time of day data was taken for the number of users on a single day. The final parameters chosen for the lognormal were such that the mean was 2 hours and a shape parameter of 0.6. This assumption will need revisiting for published work.

#### Session durations and time of day behaviour – MOOC

The following informed the session durations for the MOOC model. Data of calls durations and call frequencies from BT were used. This gave a profile of calls at a given time of day and call duration. The session durations extremely well fit a lognormal model – the data from BT is in blue, the theory line red. The data was found to fit a lognormal with scale of 1.0 and a shape of 0.8

Unfortunately, the mean session duration for the calls was too short to be usable. Hence a session duration was taken from a study of Skype data which showed a mean of 27 minutes.

## Session model

This is an application specific model which assigns users to “sessions”. These “sessions” represent users which are considered to be in the same chat. Only users in the same “session” exchange data. Two such models are produced here.

### Poker session model

Users are assigned to sessions which represent poker tables. This means that there is a specified maximum number of players (a single table is governed by the number of cards available). The rules of this are as follows. A minimum number of people to form a table is defined. A maximum number of people who can be in a room is defined.

A new arrival has the following rules applied in order:  
1) If a table with space exists the arrival is assigned randomly to any table with space and joins the session for that table.

2) The user is assigned to the waiting list. If the waiting list reaches the number to form a new table (the minimum number for a table) then the waiting list forms a new session.

A departure has the following rules applied in order:

1) The departure leaves their room – this may cause users from the waiting list to join that room.

2) If the room falls below the minimum number of users the remaining user(s) join the waiting list.

### MOOC session model

This model represents students chatting about a single online course. Hence all users join the same session which grows or shrinks accordingly.

## Network model

This model simply represents the data centres present in the system which can host video routers. For simplicity the data centres are assumed to have enough capacity that the streaming video system will not “overwhelm” them (quests for new video router instances are never refused). Three data centres models are used:

### Amazon ec2

The location of the first six amazon ec2 servers is used. The costs are used from the amazon website.

### Distributed cloud model

2507 data centres with their latitude and longitude are used. These data centres are taken from <http://www.datacentermap.com/>

The costs between each are taken from the ec2 model and the costs out for each are identical and the same as the cheapest ec2 data centre.

Server model

The server model chooses which data centres will instantiate video routers to serve a set of one or more sessions. Three are currently used. Each requires a number of servers to be specified.

### nRandom

When the session begins the servers are picked randomly from all available servers. They never change. This is only useful as a basis for comparison.

### nStatic

A clustering technique is used to cluster the 1000 demand locations with the most demand into n clusters. The unpicked server closest in terms of delay to the cluster centroid is chosen – this is done in order according to the “weight” of the cluster (the sum of the demand of all nodes in that cluster).

These n servers are always chosen for every session. For poker the choice order is:

1) Ashburn (US east) 2) Dublin 3) Palo Alto (US west) 4) Tokyo 5) Singapore 6) Sao Paolo, Brazil

For MOOC it is:

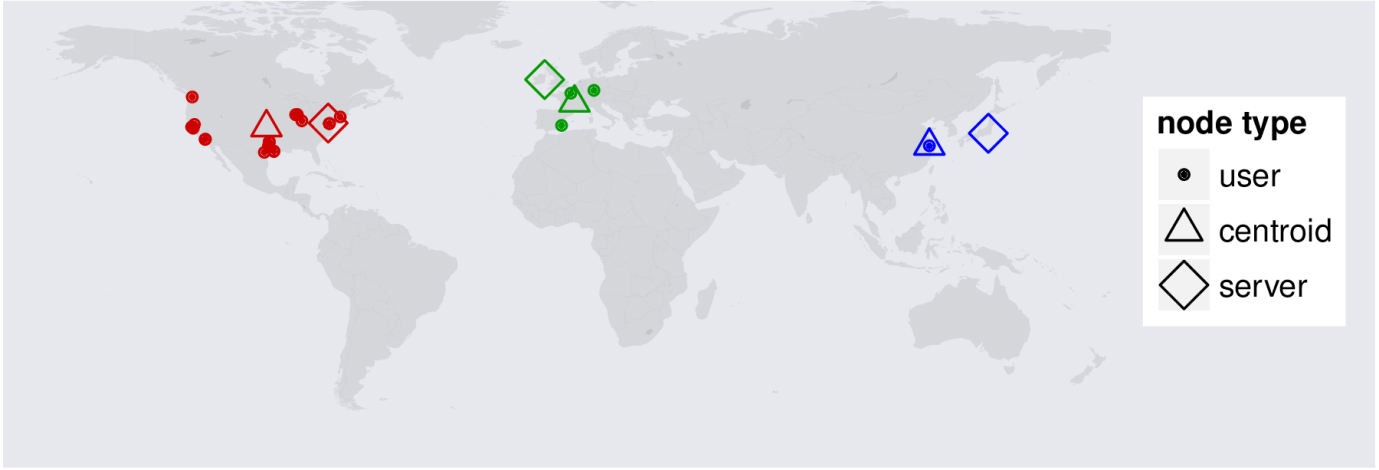
1) Ashburn (US east) 2) Dublin 3) Tokyo 4) Palo Alto (US west) 5) Singapore 6) Sao Paolo, Brazil

### nStatic \_time

The clustering technique is re-used but this is done for each time period, using the 1000 locations with the most demand in each time period (that is scaling the demand by the multiplied at a given time).

### nDynamic

All of the users in a session are clustered into n clusters and servers are picked as in nStatic. If users add or leave a session the servers are repicked. An example is shown with 25 users and three clusters. The dots show the users, the triangles the cluster centroid and the diamond the chosen server. Here then, one cluster represents North America, one (with only three users) represents Europe and one (with a single user) is China. The chosen servers are in the US east coast, Ireland and Japan.



## **Routing model**

This model picks the routes for traffic between users in streams. For every pair of users in the stream a route is picked for the traffic between them via one or more video routers which must be sited at a data centre chosen by the server model. It is assumed that all user traffic must go through at least one video router for co-ordination purposes. Traffic need not go through the same video router. The routing models used are listed below.

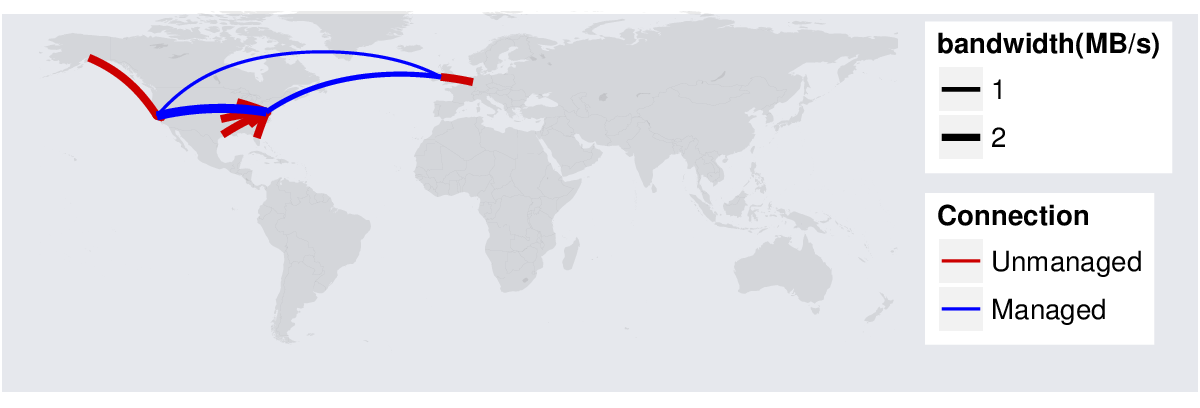
### Hot potato routing

This model aims to reduce delay to the minimum possible. For a pair of users the traffic is routed via the data centre which maximally reduces the delay along the path.

### Stay-on-route model

This model aims to reduce problems due to poor throughput. It is assumed implicitly in this model that bandwidth between data centres is usually running on better provisioned infrastructure than in the general internet. (For example amazon produce dedicated infrastructure for their data centres). This model therefore chooses for each of a pair of users the data centre closest to that user and routes the traffic between those two users via those two routers.

The graph below shows a typical session of poker with 12 users.



Quality of Experience (QoE) model

Given a set of users and routes this model tries to come up with proxies for the QoE that the users will experience. No good model of throughput across the globe is available so delay alone is used as a metric. Because throughput is important, the proportion of the journey spent on managed paths (between data centres) is also given as a QoE metric. The delay for a route is summed as the delay across each “hop” of the journey. The “hop” delay is calculated in two ways.

### Haversine distance

The Haversine distance between two points on the globe is the “great circle”, that is the shortest distance which can be travelled across the surface of the earth between those points. The delay is then estimated by a linear formula *d(x,y) = c + m h(x,y)* where *d(x,y)* is the delay between points *x* and *y* and *h(x,y)* is the haversine distance between them. If *d* is in ms and *h* is in km then the constants *c* and *m* are measured by experiment in [Networking].

### Distance corrected by subcontinental zone

This formula can be improved by modifications related to the subcontinental zone where the end points *x* and *y* are situated. The modelling for this is described in [Networking] .

Cost model

Given a set of users in each stream and the routes over which their traffic will travel this implies a certain cost that the cloud provider will charge of use of resources. This includes traffic into and out of data centres, traffic between datacentres and the cost of computational resources within the datacentres to run the required video routers. The following assumptions are made about traffic:

1) A “full” stream is 2.0 MB/s. A “compressed” stream is 0.125 MB/s.

2) Each user spends an equal amount of time being the person shown “full stream”.

3) A user will always send “full” to the server.

4) A server will send on data either full or compressed according to which users is “full” at that point. The average send rate for a user in a stream with “n” users is (2.0/n + (n-1)0.125/n) MB/s.

5) Servers send either full or compressed streams to all users receiving from them.

## Base case for results

The base case for all results was the poker model with the following characteristics:

1) An average of 10,000 users per day.

2) 2.0MB/s full stream 0.125 MB/s compressed stream.

3) Minimum of 4 maximum of 10 players in a “room” (session).

4) Six Amazon EC2 servers with the costs taken from amazon. This includes only costs for leaving the network and moving data between the network (these are the only data costs amazon charges).

5) CPU charges are assumed to be negligible by comparison to the network costs.

6) The “stayOnRoute” routing model.

7) The nRandom server choice model with 2 servers.

The base case for MOOC was the same with

1) No limit to room size all users in just one session.

2) Average of 1000 users per day (at 10,000 users per day the session size became unrealistically large).

## Results

The results are now presented for a number of scenarios. Each run was for 100 simulated days. At the end of each simulated day, parameters were gathered for delay and cost. The mean over each day was taken as well as the standard deviation between days. For these results the basic parameters taken were the cost per user in dollars, separated into components for cost from network to host (out cost) and between hosts (between cost).

Error bars on all graphs are presented to show two standard deviations above and below the mean. If errors are normal this forms an approximate 95% confidence interval around the mean. The bars are slightly offset to the left and right of the x-axis in order that error bars are not printed over each other (so 1,000 users is printed very slightly to the left or right of 1,000 for each line).

### Poker base case varying number of users

Firstly the scaling was investigated, considering the number of users. The number of users was varied including 1,000, 2,000, 5,000 and 10,000. The case of using hot-potato routing was also considered here.



As can be seen, the cost per user does not increase with the number of users in the system. This might be expected but it is useful to understand this. The costs of data transfer between servers was low in this case (negligible compared with the cost for data out from server to user). This is for two reasons:

1) Bandwidth between servers is cheaper than from server to user – EC2 servers charge $0.01/GB for transfer between servers – the cost for transfer from server to user varies from $0.02/GB to $0.125/GB depending on the data centre used.

2) Sometimes the data does not travel between servers. In the base model chosen, this will often be the case. If the nearest server to two users is the same server then the traffic will go to that server and immediately to that user. In the case of two servers chosen randomly (which was the base case picked), this case will be very common. In fact it was found that more than 80% of users exchanged data via a single server rather than going over managed network.

The error bars are quite large on costs even when 10,000 users are used. This reflects the randomness inherent in this scenario. Users can find themselves paying for a server which is expensive as it has been randomly selected.

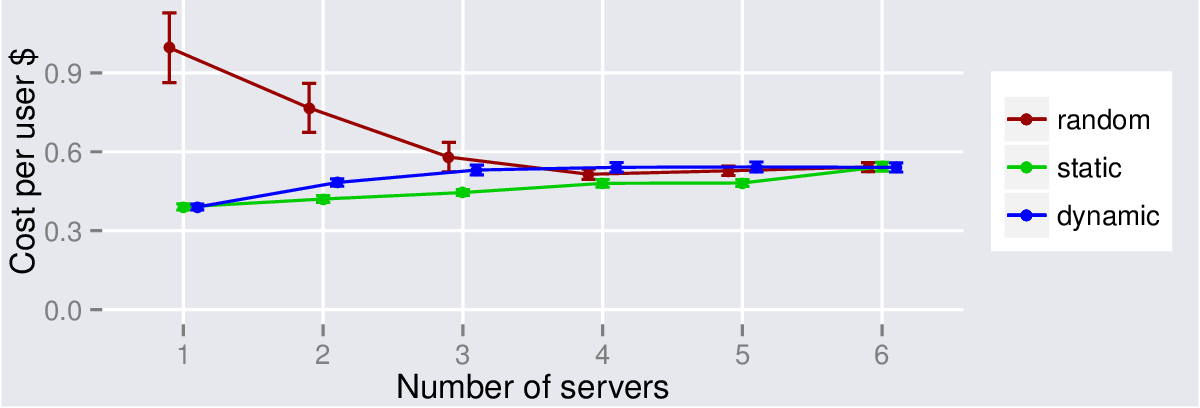
The same scenario using hot-potato routing is compared. The costs are not significantly different. The only cost difference with hot-potato routing should be the between costs which are negligible.



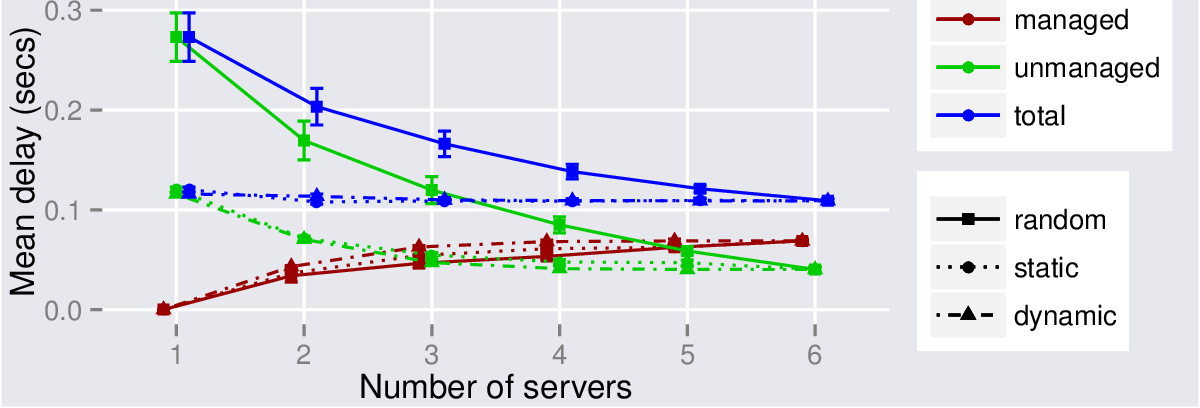
This graph shows the delays for the base case scenario. It shows the mean delay per connection between users. The red, green and blue lines show the delay over managed network, unmanaged network and total delay respectively. The black line shows the same scenario for hot potato routing. It can be seen that the delay does not depend on the number of users. The delay over managed network is low but this is mainly because most users are not, in fact, using managed network at all but instead connecting directly. The hot potato routing is lower delay on average. This is again, to be expected. The hot potato routing minimises the total delay and has no “managed network” component as the traffic never enters the managed network.

## Varying servers results

This experiment looks at the server selection strategies in terms of cost. The selection strategies are nRandom, nStatic and nDynamic. The number of servers is varied from 1 to 6. When 6 servers are selected then all servers are picked and all strategies are the same.



This graph shows the cost per users as the number of servers increases. For the random model this starts high and drops. The likely reason for this is that the random model could choose a server which has high data costs. These servers tend to be in regions with low demand (for example the most costly server is Sao Paolo in Brazil but this area has low demand in all models and will rarely be chosen by nStatic or nDynamic. The routing model will only choose that server when it is closest so when more servers are available it will be rarely used. The random model has wide error bars reflecting the fact that sometimes it picks expensive servers.

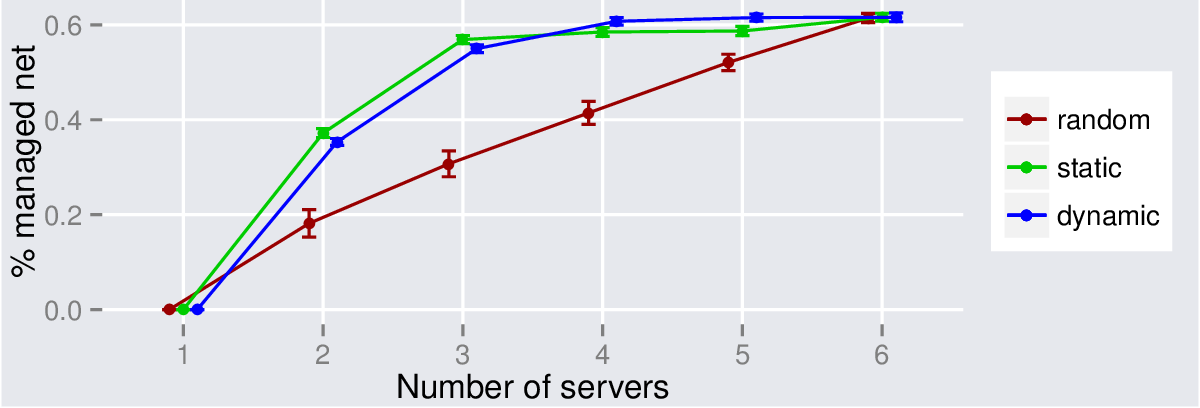


The delay model shows interesting behaviour. The graph should be interpreted by looking at colour (which indicates managed/unmanaged/total delay) and linestyle/point shape which indicates if the server selection algorithm is random, static or dynamic.

In all cases the mean delay over managed network increases as the number of servers increases. This reflects the fact that when few servers are available users often have no need to use managed network. The dynamic model is for obvious reasons, better at choosing appropriate servers and, hence, getting users onto managed network, however the static model is almost as good. The amount of time spent on decreases well for the dynamic and static strategies with the dynamic strategy being slightly better at getting users onto the managed network.

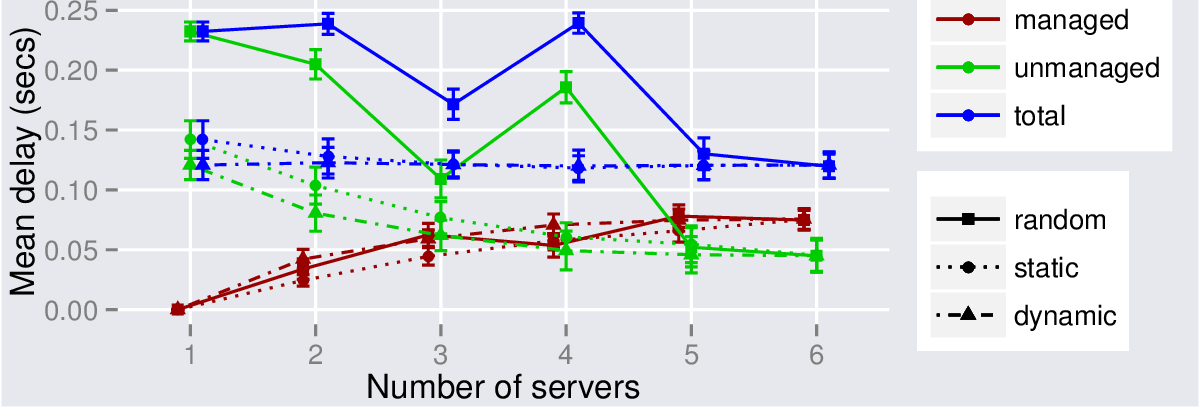
The behaviour of the random strategy is in some ways the reverse. The amount of time spent on unmanaged network greatly reduces as more servers are available, reflecting the fact that this strategy can choose very inappropriate servers.

The behaviour of the nStatic strategy is very similar to that of the dynamic model. The graph below shows the proportion of users which use managed network in each strategy.

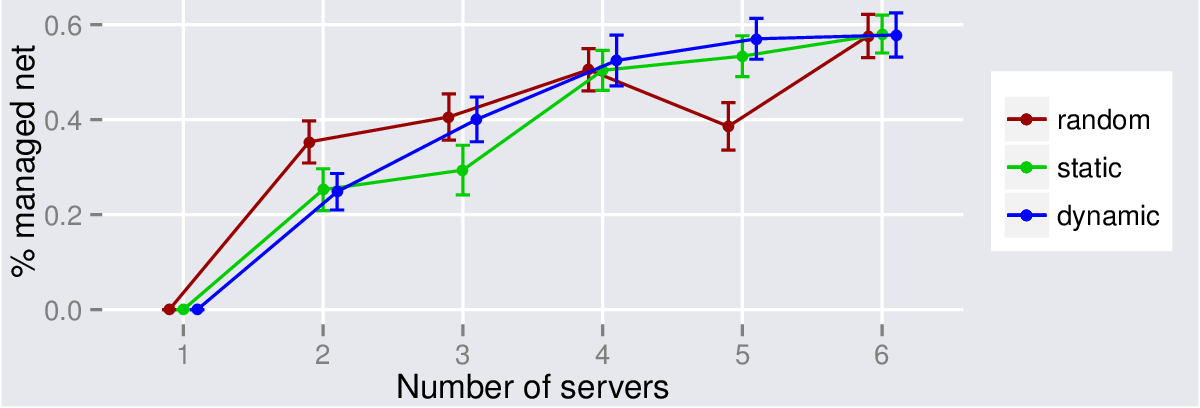


### MOOC modelling

Because the user base for the MOOC modelling is smaller (1,000 users per day not 10,000) then the error bars are larger, particularly for price. The models are indistinguishable in cost except for the random model. This has high cost when the model (randomly) chooses high cost servers. Because the MOOC model only has one session, servers are selected once only. In this case for the one server case the MOOC model has selected high cost servers. As will be seen later, this is not a repeatable result. The error bars show the standard deviation between days (which is high) but when servers are randomly reselected the graph shape is different for random.



The performance of the three models is not that different except that Random sometimes has anomalous high delay.



### Model repeatability

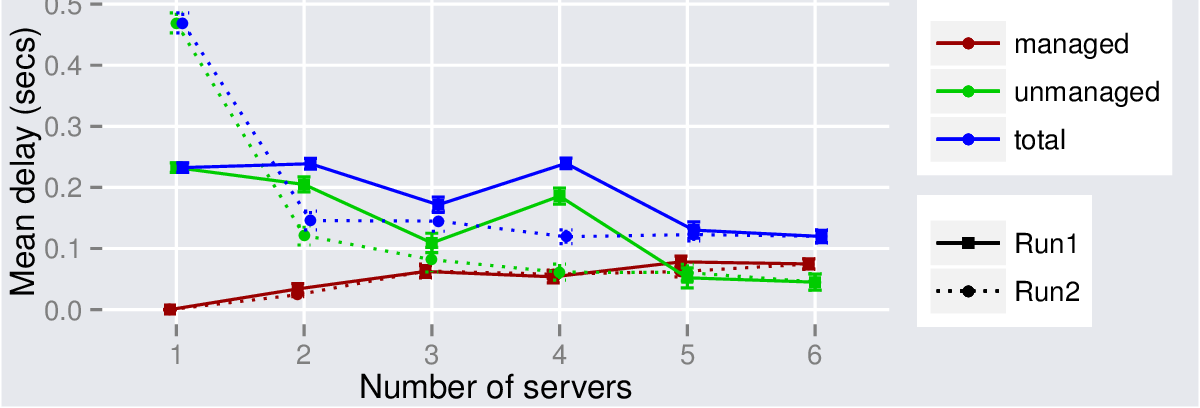
Because the MOOC model only has a single session then repeatability is a large issue for the nRandom model. Servers are selected once only – if poor servers are selected this selection remains. Two identical runs are performed the differences between the runs are:

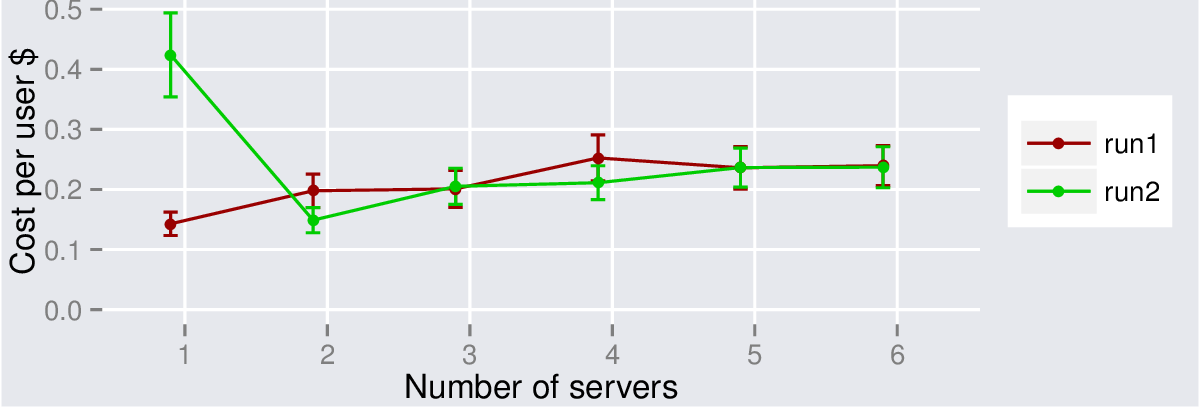
1) The locations of the users randomly selected by the demand model

and

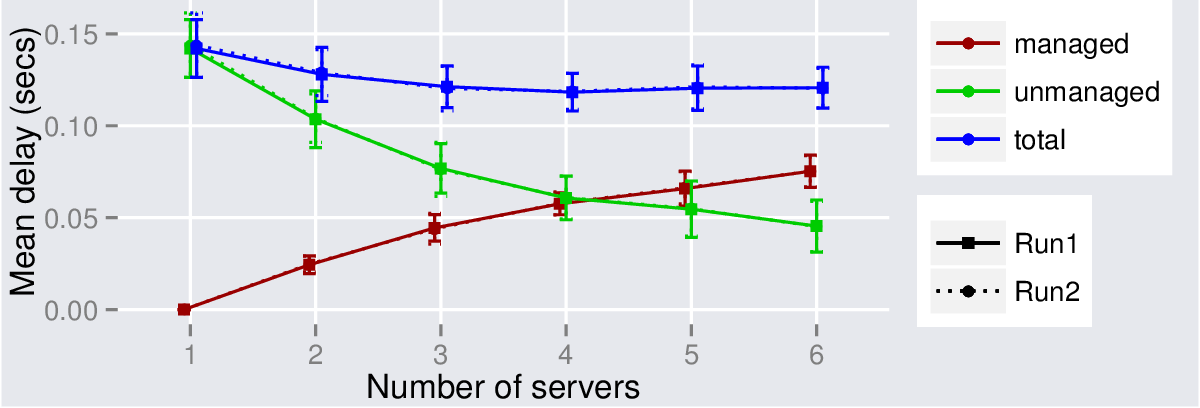
2) In the nRandom server selection model, the servers selected.

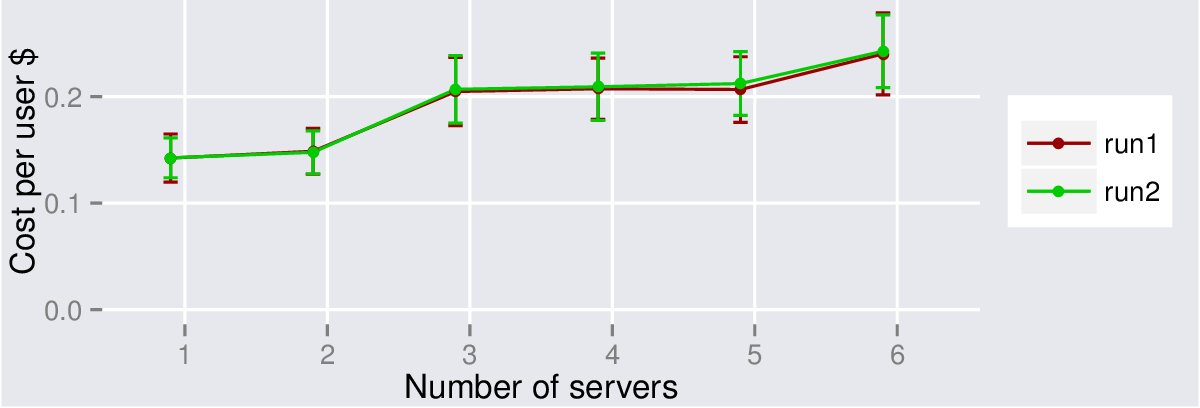
The next two graphs show that for the MOOC scenario this selection is critical. Selecting the “wrong” servers can increase both cost and delay.



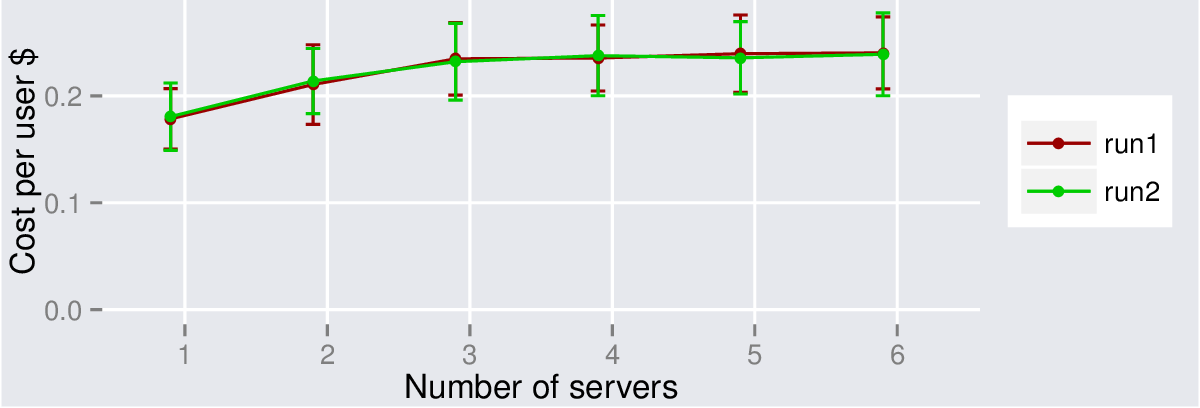


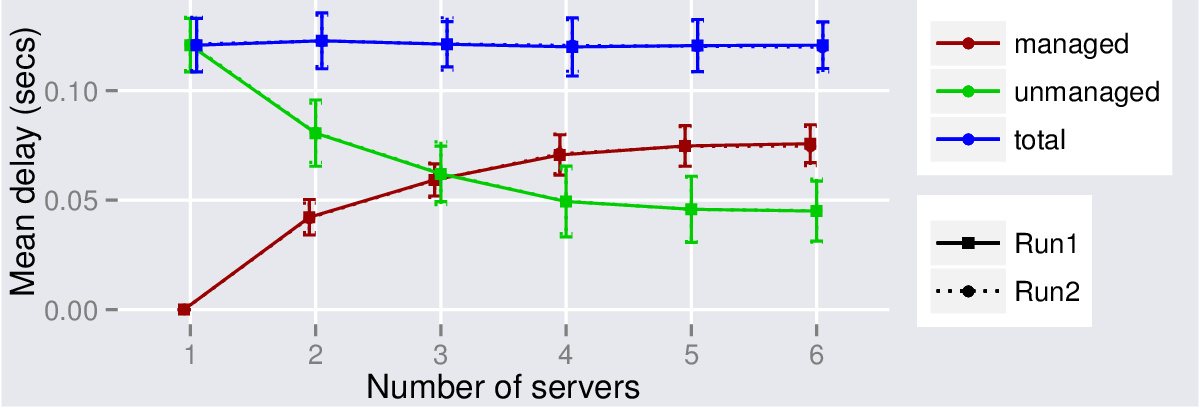
However, for the nStatic model, the picture is different. Two runs produce extremely similar results for costs and for delays. The pattern seems to be a repeatable result from the servers selected by the clustering algorithm. The cost increases as more expensive servers are brought online and more traffic goes over internal links.





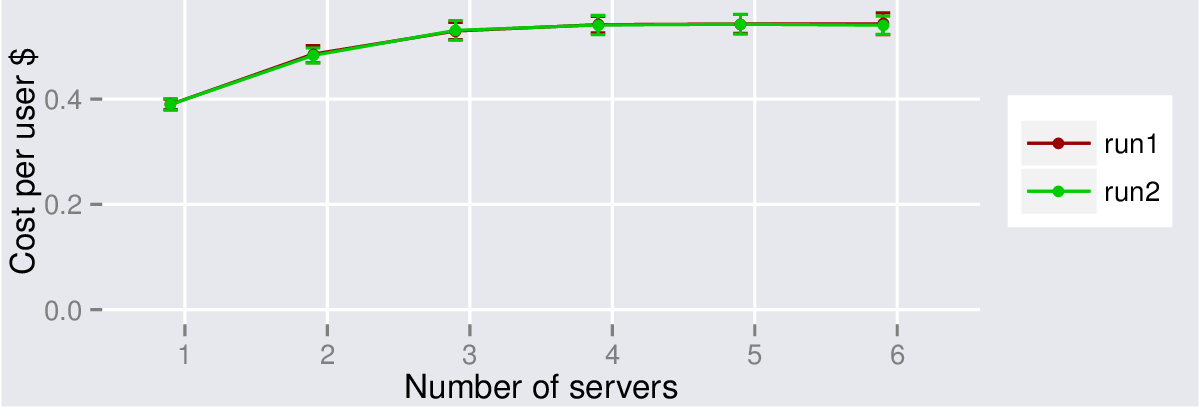
Similarly for the nDynamic model the results are extremely repeatable with the two runs almost exactly overlying each other for both cost and delay.

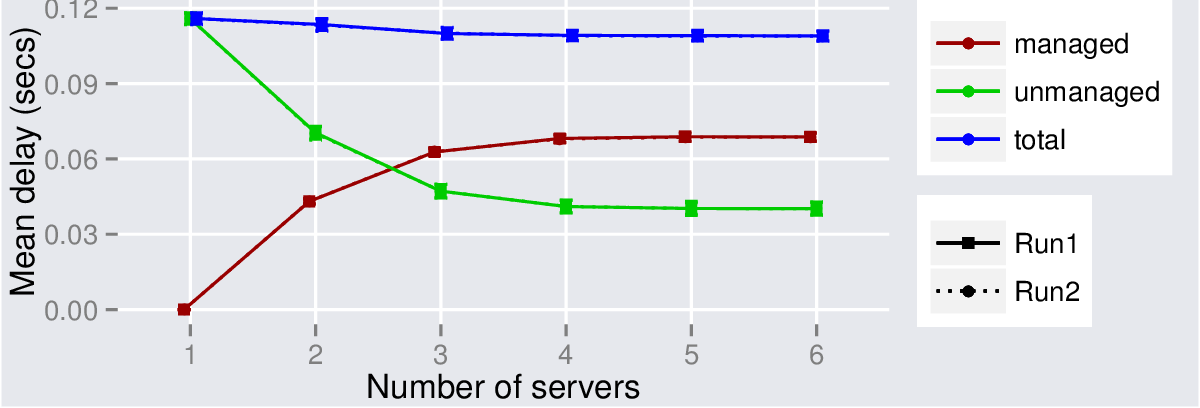




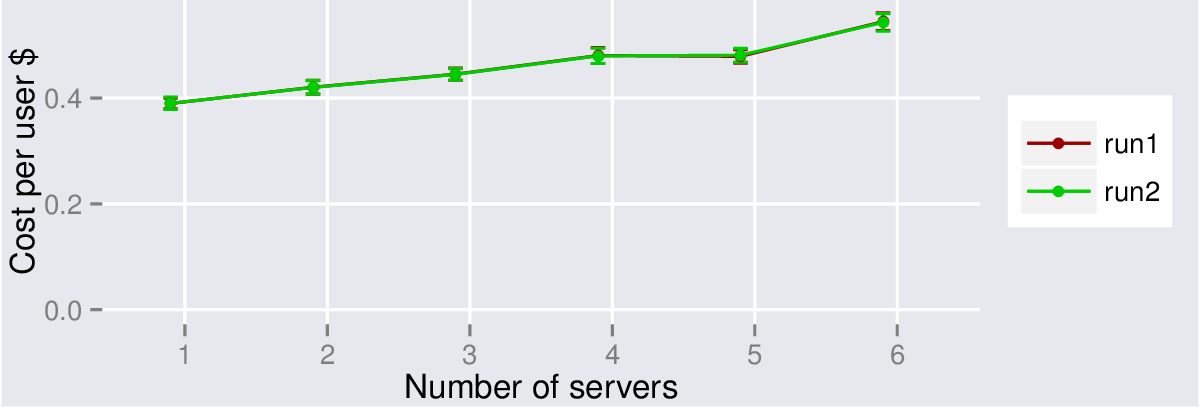
Repeatability runs have also been made for the Poker model:

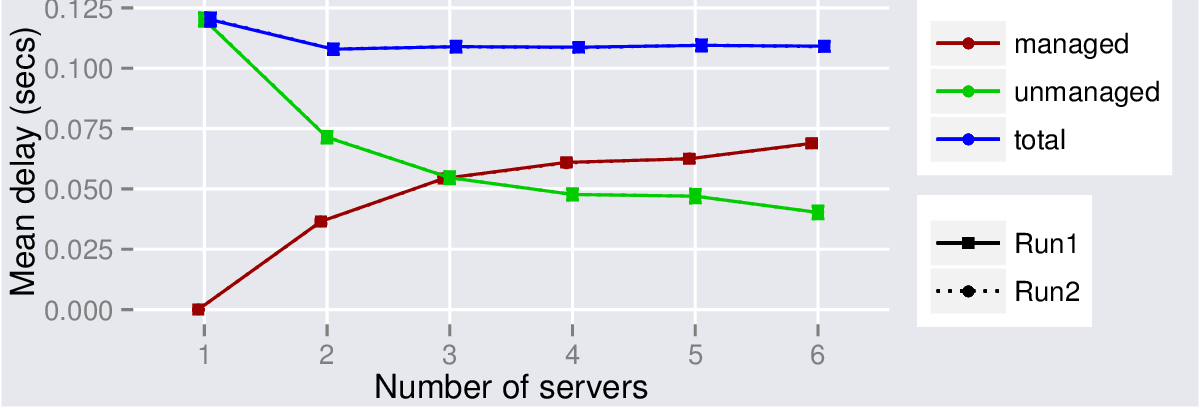
In this case for the nDynamic strategy the lines overlie each other so closely that the second line cannot be seen.



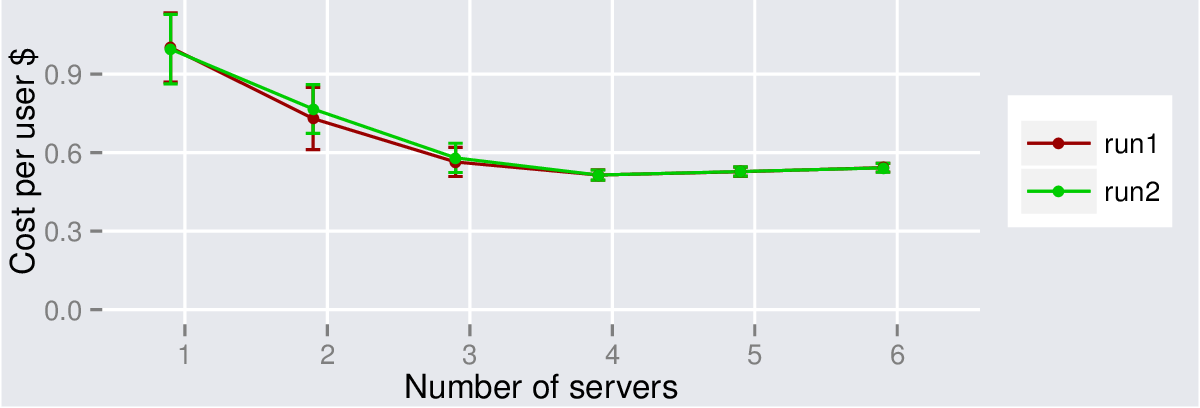


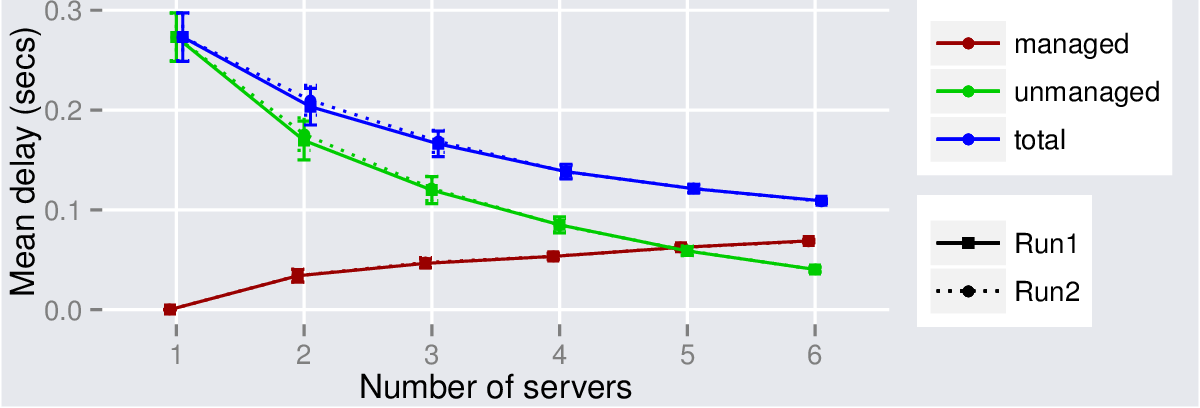
This is also the case for the nStatic and nRandom cases. The overlap is almost exact. nStatic is shown first:





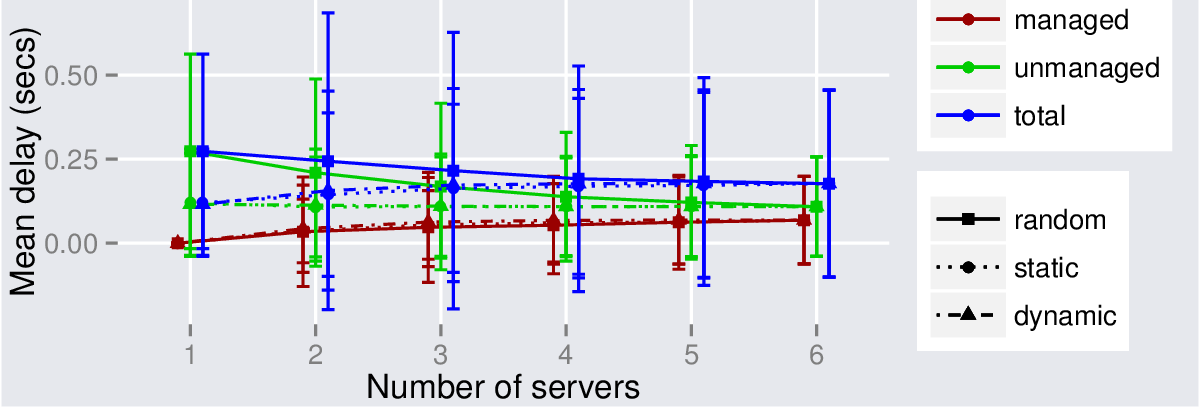
Then nRandom – the results are not quite as close as the others but still extremely close.





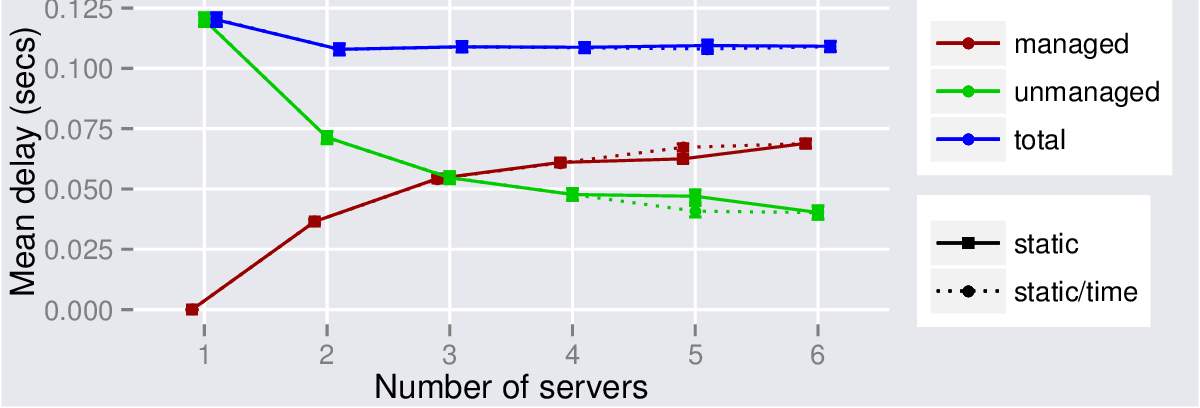
### Error bars between sessions

If we consider the error bars indicating the standard deviations between sessions within a day rather than the standard deviation between days then the error bars are much much larger. Indeed the error bars overwhelm the graph. There is a great deal of variability between sessions within a day.



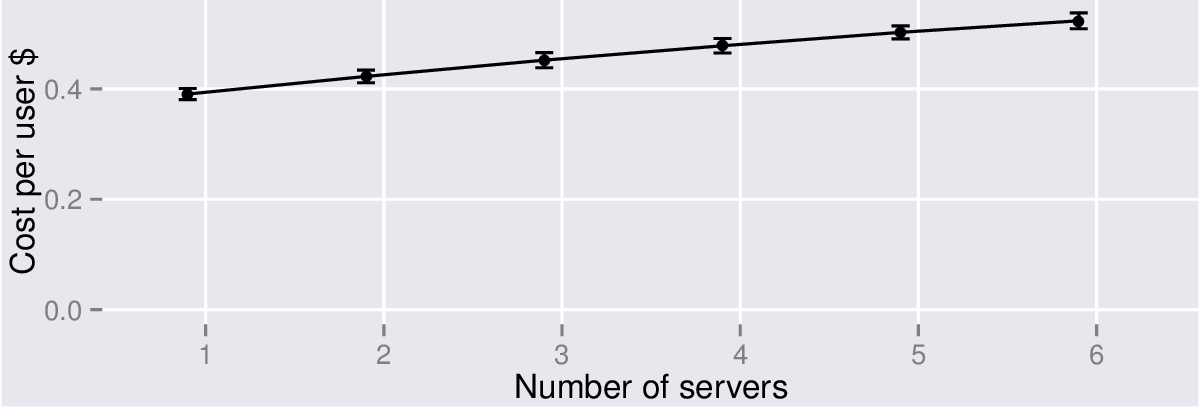
### Reclustering nstatic by time of day

An experiment was tried where nstatic formed different clusters at different times of day. In fact this seems to make little difference to the results – the reason this makes so little difference is not known, however the static algorithm alone is very good.

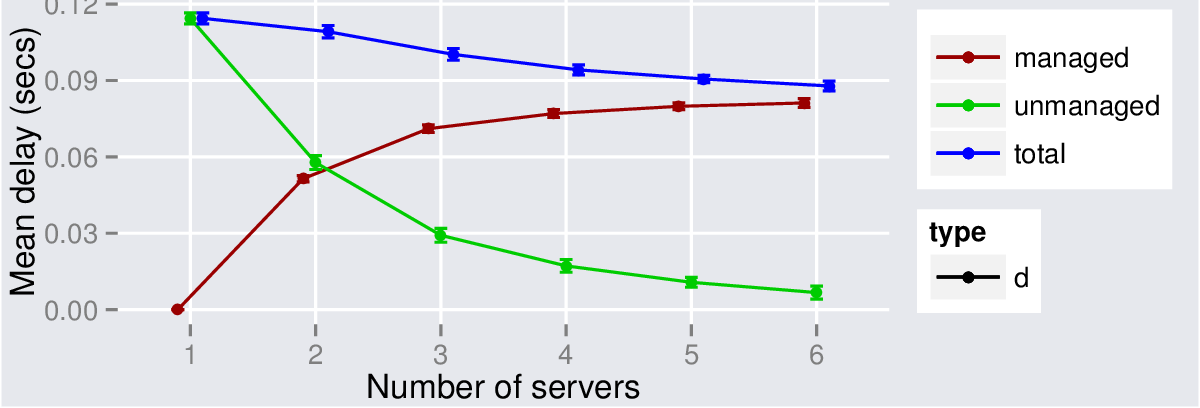


## Large number of data centres

In the large number of data centre case the cost per user increases slightly as more users use managed network more often and hence that small additional charge is used. In this case the costs for servers are all considered identical so the only difference in cost comes from increased use of managed network.



The QoE results are slightly different with the larger number of datacentres the reduction in managed network used is greater for the dynamic model.



## Follow on work

To get each graph in this paper takes approximately eight hours of computing time – but several can be run simultaneously.

## Making this into a paper

Some work is needed to make this into a paper.

1) The Poker session durations needs to be somehow firmed up.

2) Runs above need to be completed.

3) Presentation of the PDF of delays should somehow be incorporated.

### References

[CIA] The World Factbook. Country comparison: Internet users. https://www.cia.gov/library/publications/the-world-factbook/rankorder/2153rank.html.

[GeoNames] GeoNames. http://www.geonames.org, 2012.

[Networking] R. Landa, J. T. Araujo, R. G. Clegg, E. Mykoniati, D. Griffin, M. Rio, “The Large-Scale Geography of Internet Round Trip Times”, to appear in Proceedings of the 12th International IFIP TC 6 Networking Conference (NETWORKING), May 2013, New York, USA.

[ICCCN] R. Landa, R. G. Clegg, J. T. Araujo, E. Mykoniati, D. Griffin, M. Rio, “Measuring the Relationships between Internet Geography and RTT”, to appear in Proceedings of the 22nd IEEE International Conference on Computer Communications and Networks (ICCCN), July/August 2013, Nassau, Bahamas (pdf).

[OPD-UHH] Ingo Fiedler and Ann-Christin Wilcke, The Market for Online Poker. Unlv Res. & Rev. J. 7--9 (2012).

[FPRO] “Functional Programming Principles in Scala”, EPFL, Coursera. http://docs.scala-lang.org/news/functional-programming-principles-in-scala-impressions-and-statistics.html

[MOOC] University of Edimburg report on MOOC usage, 2013. https://www.era.lib.ed.ac.uk/bitstream/1842/6683/1/Edinburgh%20MOOCs%20Report%202013%20%231.pdf

[ATR1] “Computational Investing, Part I”, Coursera, 2012. http://augmentedtrader.wordpress.com/2013/05/25/mooc-student-demographics-fall-2012/

[ATR2] “Computational Investing, Part I”, Coursera, 2012. http://augmentedtrader.wordpress.com/2013/01/27/mooc-student-demographics/

[KNCT] “Introduction to Data Journalism”, the Knight Center for Journalism in the Americas, 2013. https://knightcenter.utexas.edu/00-14152-nearly-4000-students-62-countries-participate-knight-center%E2%80%99s-mooc-introduction-data-journa

[EDXC] “6.002x: Circuits and Electronics”, MIT/Harvard University, edX, 2012. http://www.rpajournal.com/dev/wp-content/uploads/2013/05/SF2.pdf

[IHTS] “Internet, History, Technology and Security”, University of Michigan, Coursera, 2012. http://www.dr-chuck.com/coursera/insidetheinternet/2012-001/maps/where.html

[CS50] “CS50x: Introduction to Computer Science”, Harvard Uinversity, edX, 2012. https://blog.cs50.net/2013/05/01/0/

[POKER]   Moderate Expectations: The Case of a Normal, Level-Headed Semi-Professional Online Poker Playerand His Family, Global Media Journal – Polish Edition Fall 2011, No. 2 (8)