KUNGL TEKNISKA HÖGSKOLAN
Royal Institute of Technology
Numerical Analysis and Computing Science

TRITA-NA-D9712 • CID-23, KTH, Stockholm, Sweden 1997

Personal TV Program Schedules:

an Introduction to Usability and Experiments with Neural Networks as a Proposed Solution

Stefan Eriksson

CID
Centre for
User Oriented IT Design

Stefan Eriksson

Personal TV Program Schedules:
an Introduction to Usability and Experiments
with Neural Networks as a Proposed Solution

Report number: TRITA-NA-D9712, CID23

ISSN number: ISSN 1403-073X

Publication date: November 1997

E-mail of author: stefan1@e.kth.se

URL of author: http://www.e.kth.se/~stefan1/

Reports can be ordered from:

CID, Centre for User Oriented IT Design

Nada, Dept. Computing Science

KTH, Royal Institute of Technology

S-100 44 Stockhom, Sweden

telephone: + 46 8 790 91 00

fax: + 46 8 790 90 99

e-mail: cid@nada.kth.se

URL: http://www.nada.kth.se/cid/

Abstract

Digital television will reveal hundreds of TV-channels. TV-viewers cannot possibly survey all these channels. Personal TV program schedules might serve as an aid. This paper investigates personal TV program schedules from the viewers needs and then goes on with some practical experiments. Agents are introduced as a tool to provide personal program schedules and suggested as a universal TV helping hand. A simple neural network is used to show that viewers TV habits can be predicted, up to some level, with relatively simple means.

Sammanfattning

Införandet av digital television kommer att ge oss hundratals TV-kanaler. Den normala TV-tittaren kan omöjligt skapa sig en överblick av vad som sänds på alla dessa kanaler. Personliga programtablåer kan vara ett tänkbart hjälpmedel för tittaren. Denna rapport undersöker personliga programtablåer utifrån tittarnas behov och beskriver hur personliga programtablåer kan tas fram med hjälp av neurala nätverk. Agenter beskrivs som ett verktyg för att ta fram personliga programtablåer och som ett universellt hjälpmedel för TV-tittaren. Med hjälp av ett neuralt nätverk så visas att TV tittarnas vanor kan förutses till en viss grad med enkla hjälpmedel.

Acknowledgements

This report is the product of a master's thesis performed at the Centre for User Oriented IT Design, CID, a competence centre at NADA, KTH. The thesis project is a part of the research area Worlds on Interactive TV.

Special thanks to Hans Mandorff and Mediamätning i skandinavien AB, MMS. They have provided all necessary data about TV viewers used in my experiments.

I also wish to thank SVT, TV3, TV4, Kanal 5, TV6 and ZTV for their help with program schedules.

Contents

Part I Agents, usability and everything

Part II Experiments with neural networks

References

Appendices

Part I Agents, usability and everything

1 Introduction

The introduction of digital TV will reveal hundreds of TV channels. The major problem with all these channels is the lack of tools to help the viewer navigate through the plethora of programs. Channel surfers will have problems surfing through all channels before the first channel visited has changed its contents. And there will be little hope for those obsessed with reading schedules for all programs available.

To help lost and resigned viewers in their search for entertainment and information, personal computer-based so called agents have been proposed as an aid for those who want to surf above the mud.

These agents can be taught viewers' preferences and habits. Instead of spending hours with a 50 page program schedule, viewers will have their own personal TV program schedule which will contain only programs that are supposed to be interesting to the viewer.

A TV agent can also be used as a help for the TV-set itself and for other services like home-shopping and various forms of entertainment.

This report will focus on personal TV program schedules.

2 What is an agent?

A lot of hard work is being carried out around the world trying to define what agents are and what they should do. A very brief explanation could be made by saying that an agent is a system that has some sort of intelligence; that it acts on behalf of someone; has goals that it tries to bring about; has knowledge about its environment; solves problems that arise, and works autonomously (Wooldridge & Jennings 1995).

An important difference between an agent and a plain program is that an agent doesn't always know how to accomplish its tasks or if they are at all solvable. One cannot be sure that data given as input to the agent will be enough for the agent's level of problem solving, even if it can be proved to be enough theoretically. For a non agent program it is always possible to define a minimum input set that will produce the correct result.

The result of an agent isn't always reproducible. If the agent has made new experiences it might solve an old problem in a new and more efficient way, with a slightly different result. A non agent program is fully deterministic and always produces the same result as long as input remains unchanged.

The definition of what properties a program should have to be called an agent is still a subject of much debate. Nwana (1996) notes:

> "We have as much chance of agreeing on a consensus definition for the word 'agent' as AI researchers have on arriving at one for 'artificial intelligence' itself - nil!"

In fact there are nearly as many definitions on what agents are and what their properties should be, as there are active researchers in the area.

The distinction between agents and non agents has been made even harder by software companies that have started to use the term agent for almost any kind of software. If it's new and hot, lets call it an agent.

The disagreements on agent properties might lead to people having expectations on agents, which will not be fulfilled in some systems.

3 What agents can do for TV viewers and how

One application for a TV-agent is to propose interesting programs to the viewer. To be able to do that the agent must learn about the viewer and the outside world. How much information can the agent retrieve from the user? Is it reasonable that the agent should ask the viewer about his opinion on an ongoing program every 5th minutes or so? Or should the agent collect data about the habits of the viewer only by counting the amount of time spent on a program? The information gathered might be crucial to the value of the proposed programs and the usefulness of the agent.

An agent can also be used as a guide to the TV-set as well as to itself. The agent can be taught the viewer's skills and adopt menus and the amount of assistance the viewer requires. If, for example, the viewer moves around in lots of menues without issuing any commands, the agent might give the user a first introductory lesson about how the TV-set works.

4 Agent environment

The agent environment is the so called world that the agent inhabits and is best described divided into two parts, an inaccessible part and an accessible part (Russel & Norvig 1995); the viewer and the TV-programs respectively.

4.1 The viewer

The viewer is inaccessible for an agent in the sense that an agent does not have access to the viewer's internal states and feelings. An agent has to build its own model of the viewer by observation and by asking the viewer questions.

Collection of information from the viewer can be passive or active. Passive registration of behaviour means that the agent only observes what the viewer is doing; in reality, which buttons the viewer presses on the remote control. In its most basic way the agent keeps track of what programs the viewer actually watches and how much time is spent on each program. The agent might get a deeper understanding of the viewer's emotional feelings by studying how often buttons are pressed and in what order. For example, a viewer that does a lot of button wear an tear might be either upset or lively or perhaps in another mode.

A problem with passive registration is that the agent does not know if the viewer is present or not. The viewer might also be present but not attentive.

Having a viewer actively participating in a reasoning or debate about what programs are to be preferred is an enormous step forward compared to having just a passive viewer. A simple way of getting feedback is to have some sort of opinion poll after each program. The agent could ask questions like, did you enjoy the program? and, would you like to see more programs like this one in the future?

Having access to the viewer's mind and internal psychological processes is a most appealing property of a TV agent. This would imply that we could always make a perfect prediction of what TV programs the viewer would choose. It would in fact let us build an agent that not only could pick out the most interesting programs, but also do the swapping between channels at the wish of the viewer. Unfortunately (or fortunately) this will not be possible until we have a deeper understanding of how the human brain works and how to probe it in order for an agent to get inside information about the viewer's desires and preferences. The viewer's internal states will however remain inaccessible for a long time. The best an agent can do in the meantime is to register behaviour and to communicate with the viewer to get a deeper understanding of her thoughts and desires about TV programs.

4.2 TV-program information

The agent has access to the TV-program information at all times. This means that the agent will not have to keep track of the current states in the TV-program environment for other reasons than to improve access speed. The program information is supplied by the broadcasting companies and is, for digital television, described in the DVB-SI (1996), Digital Video Broadcasting-Service Information, standard.

The standard includes information about:

- program title;
- description of contents;
- starting time;
- program length

In addition to this information there is also information about parent rating, cost, subtitling, languages, display formats, etc.

5 Personal TV program schedules

As the number of TV channels increase there is an urgent need for a tool to help viewers to find programs that they find interesting. The rest of this report will focus on personal TV program schedules. These can be implemented either as a part of a TV-agent, or as a stand alone application.

The first thing which comes to mind is perhaps a complex search engine with numerous controls and gadgets. However, anyone who has used an search engine for the World Wide Web to find information on a topic has most certainly experienced that whatever the question formulated, there are always too many pages that fit the query. It always takes more than one effort to get a good answer. An exhausted viewer that has just sat down in her favourite TV chair probably does not want to type long search sentences on a keyboard. Of course, saying that a viewer never wants to do this might be a little ignorant, but in general most viewers just want to watch TV and relax. I expect that a TV-set with lots of exclusive features will only be used by a minority of people; primarily people with some computer experiences.

Closely related to the problem of finding interesting TV programs is News and e-mail filtering. Kilander (1996) finds three different ways in which typical News filters work:

- *The filter is programmed by the user.*
- *The user creates a query which the filter attempts to answer.*
- *The filter is trained from examples given by the user.*

A programmable filter might be powerful but this is not applicable for the average TV user; mainly because most people do not have any programming skills.

With a query form the viewer can search for specific programs. A query form is a valuable tool when viewers have widely changing demands. An agent that tries to learn the behaviour of a viewer with irregular viewing habits will probably recommend all programs or none.

A filter that is trained by examples is an appealing thing. By counting the amount of time spent on a program or asking the viewer for an opinion, the agent can probably gather enough information to be able to make good program proposals.

5.1 What functions does the viewer need and how is she about to use them?

How do viewers get information on what TV programs are available?

Will viewers be prepared to rely on a TV agent?

What criteria are important for the choice of TV programs?

A small opinion poll was made to reveal the answers to these questions.

Eighteen people were asked questions about how they find interesting programs and what they would expect to get from a personal program schedule. This is far too few people to draw any conclusions from, but serves as an indication of how a viewer might reason.

- 17% (compare with: 13%, Wigren 1990) use text-TV as a source for program information and 78% uses printed schedules. This might be an indication that printed schedules are more easy to use and thus preferred. In order to be used, a TV agent must provide something more valuable than ordinary text-TV program schedules do.
- There is a strong positive correlation between the number of hours spent watching a TV channel and a viewers opinion about that channel. This is a strong reason to give the agent information about which channels the viewer watches.
- 50% thought that it would be worth the extra work to give a judgement about each program watched if that could help the agent to give them better suggestions.
- 77% thought that they would use a search function often or sometimes. The high percentage might be due to the fact that most of the persons asked are students who are familiar with AltaVista and other search functions.
- A question which received many different answers was whether the viewer thought that he or she would have confidence in the agent or still browse through all schedules. This will, of course, depend on whether the viewer likes the proposed programs or not, but even with a good personal program schedule some people still want to read about all programs.

5.2 Mutations, the source of evolution

An agent that has achieved a broad knowledge about a viewers interests and habits will be able to present an excellent personal program schedule, day after day. However, the schedule will probably look almost the same, day after day. The agent will be very pleased with the result of providing the best schedule ever achieved. This is not necessarily a bad thing, the viewer will most certainly be pleased. But is this really what we want? Isn’t there a need to widen our horizons, to see new unexplored areas of television?

Naturally, the viewer could select another channel. But will the viewer do that? Won’t the viewer simply stick to a couple of well known channels? What about channel 354 or so, will the viewer ever look at that channel?

One solution might be a mechanism that at random reorders the rules used to pick programs.

An attempting idea is to let a personal TV agent match its viewer’s watching profile against other viewers to find similarities in their viewing habits. When the agent finds a viewer which seams to have the same interests as its owner, it can recommend programs that the other viewer watches.

6 Testing a personal TV program schedule

The main problem with user tests of personal TV program schedules is the large amount of test time needed. To be able to do experiments with personal program schedules I realized that I needed either test persons willing to spend a couple of hundred of hours with their TV-set or test data from an external source. I was fortunate to get in contact with Mediamätning i skandinavien AB, MMS, who perform market and viewer opinion polls in media and TV. MMS provided me with useful data about the swedish viewers and their TV habits.

The data from MMS contains information about the seven most widespread Swedish channels and viewer logs from approximately 1400 people from about 650 households.

The viewers' TV habits were registered with an equipment called People Meter. The People Meter is an equipment that is connected to the TV-set and registers which channel is on. Who is watching TV is also registered. This is done with a special remote control on which each household member has their own button. When a viewer starts or stop watching, he or she presses the appropriate button to inform the People Meter who is watching TV.

The People Meter keeps a detailed log for each household member. The log contains information about:

- the viewer's: gender and age;
- the viewing: TV-set, channel, starting time and duration with minutes resolution.

See Appendix B for further details about MMS data.

7 Design proposals for a personal TV program schedule algorithm

The core function for a personal TV program schedule algorithm is to select a subset of the broadcasted programs that are likely to be accepted as good by the viewer. This can be done in a number of different ways. I considered three different ways, namely:

1. Statistical methods

   Statistical methods are useful for representing information about TV viewers' habits. It is an easy task to count the number of times a viewer has watched a specific type of program at a certain time of day, might well be used. Numerous parameters can be thought of to be included in a statistical representation.

   A problem with statistical methods is the large number of properties for TV programs and their mutual dependencies. For example: The viewer prefers football to news with a probability of 75%. But if a football game is broadcasted on a different channel than the viewer's favourite sport channel it might not be considered interesting. Then footballs priority over news is dependent on the channel. A viewer's preferences can also depend on the time of the day, the programs' participants etc.

   The agent will have to keep a record of all dependencies, strong or weak.

2. Neural network

A neural network has an implicit representation of knowledge. It is not possible to have a brief look at the network and extract some of its knowledge. A neural network is self organizing in the meaning that important relations are given a large amount of the networks capacity while unimportant relations are weakened and in the end forgotten.

The strongest reason for using neural networks is their ability to generalise when confronted with new situations.

3. Knowledge based artificial intelligence

   Though neural networks is a form of artificial intelligence, AI, it is different from classical AI due to of its low level representation of knowledge. AI usually refers to high level reasoning with explicitly formulated knowledge and reasoning capability.

   A most difficult matter in AI is to purge knowledge. When is a rule valid and when should it be removed? A purging utility with poor performance can remove useful rules and may let the database grow out of bounds. Another difficulty with AI is that the system needs a core set of rules and relations to describe the problem at hand. When dealing with human being it is very difficult to define such a set.

7.1 Chosen design

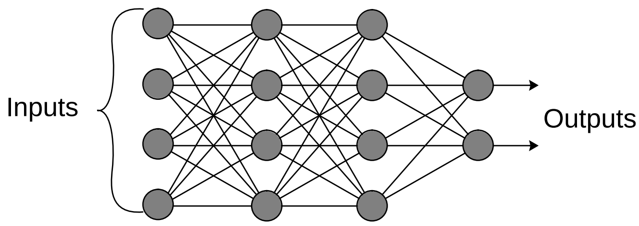
I decided to focus on neural networks mainly for three reasons:

- A neural network is self organizing. No prior knowledge has to be built in to the network. It requires little prior knowledge about the underlying phenomena at hand. This is especially valuable when dealing with human beings.
- A neural network has the ability to generalize from learned examples and solve new problems.
- The human brain is a gigantic neural network. Even if a computer can not simulate the whole human brain, it is a tempting thought that an artificial neural network will have a knowledge representation that lies close the human brain.

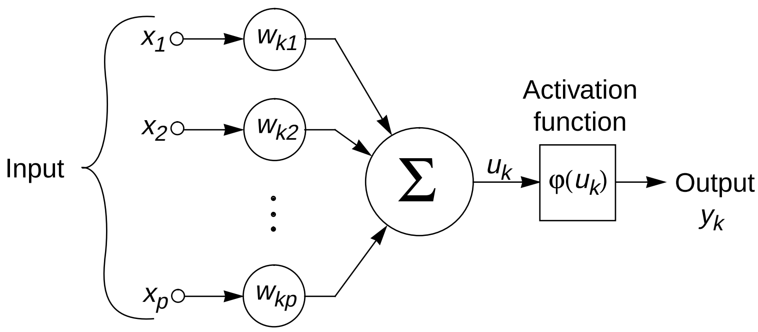
Part II Experiments with neural networks

8 Introduction to neural networks

A neural network is basically described as number of computational nodes organised in layers. These nodes are called neurons. Every neuron is connected to all neurons in its neighbouring layers through weighted links. Neurons at the input layer are called input neurons and neurons at the output layer are called output neurons. The intermediate neurons are called hidden neurons.



The input to one neuron consists of outputs from the neurons in the preceding layer multiplied with a link weight specific to each connection.

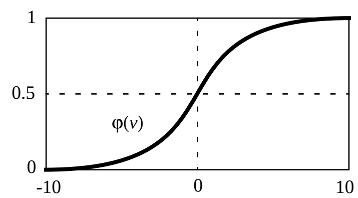


The output value for a neuron is calculated as:

$$y = \varphi\left(\sum_{j=1}^{p} w_j \cdot x_j\right)$$

Where φ(x) is a limiting function. A commonly used function is the sigmoid function:

$$\varphi(v) = \frac{1}{1 + \exp(-av)}$$



This function limits the output to be between zero and one. The variable a is a parameter that controls the slope of the curve.

By changing the weights we can get any desired values on the outputs for a certain input pattern. Weights can be both positive and negative.

Training neural networks is done by changing the link weights. The back-propagation algorithm is the most popular and widespread learning algorithm for multilayer neural networks. The algorithm can shortly be described as a method to update the weights in a network by starting at the output; adjusting the weights connected to the output to minimize the error at the output and then move to the preceding layer and repeat the task for every neuron in that layer and so on until the input layer is reached.

9 Available test data

As mentioned in Section 6 on page 6, MMS provided the data about viewers and their viewing. MMS also provided a log from the broadcasting companies which contains information about program name, date, starting time, and length for every program broadcasted.

9.1 Program data

Program data consist of:

- Channel name.
- Program name, 64 letters.
- Date.
- Starting time as number of minutes after midnight.
- Length in minutes.

In addition to the original log file the following have been added:

- Content descriptor. The programs are classified according to the DVB-SI standard (1996). In DVB-SI the program content is categorised in areas such as movies, news, sports etc.
- Short information, 512 letters. The short information contains a short program description in natural language.

The starting time and length are taken from the log provided by the broadcasters. This means that these are the actual timings. Small differences between program schedule and program log are otherwise common to occur, mainly due to program announcements and commercials.

9.2 Viewer data

Viewer data consist of:

- Age

- Gender
- Per minute information about which channel the viewer is watching.

10 Rating program information and program name

The short information field, as it is named in DVB-SI (1996), contains free text information about each program it belongs to. The information field often contains a short description of the program and a list of its participants. An attractive thought is that one can learn a lot about a viewer if one read about the programs he or she has seen.

There has been a lot of research carried out in the area of text classification. The main applications are as information filters. In a growing information society people are likely to get overloaded with information. These filters serve as a help to extract information that is suitable for a reader (Karlgren et al, 1994). The purpose of this experiment was however to investigate whether a classification of the programs could act as a source of information at all. That's why a relatively unsophisticated classification tool was chosen.

Also the program name can contain information. The program name is included as a part of the program information to simplify the experiments. In a real implementation the program name could have its own inputs in the network.

10.1 The algorithm

The idea is quite simple. By assuming that programs seen by the viewer contain words of interest in the short information field, these words are stored in a database. Together with each word, information is stored about how many minutes the viewer has watched programs with this word in the short information field and the total possibly viewing time of programs that contain this word in the short information field.

In order to sort out the majority of unwanted words such as coordinators and prepositions from the texts, a limit of four letters was set. Shorter words are not included in the database.

IN order to classify every word in the short information field, the quotient of minutes spent by the viewer and total number of minutes are calculated.To reduce the influence of common words, non keywords, the root mean square is calculated. Common words are likely to appear in many program descriptions and therefore they will get a smaller quote.

$$rating = \sqrt{\frac{1}{n}\sum_{n}\left(\frac{minutes_{watched}}{minutes_{total}}\right)^2}$$

Where n is the number of words in the program description.

11 Generation of patterns

Every time the viewer changes channel or a channel changes program there is a new situation for the agent to learn; a new pattern.

A If the viewer has changed channel or turned the TV on, the new program watched by the viewer shall be preferred to all other programs available. Therefore n-1 patterns are generated (n = total number of broadcasting channels).

B If the channel watched by the viewer changes program the situation is similar to a channel change initiated by the viewer. The new program shall be preferred to all other programs.

C If a channel, other than the one seen by the viewer, changes program only one new pattern will be necessary.

Figure 1 shows an example situation where new patterns are generated.

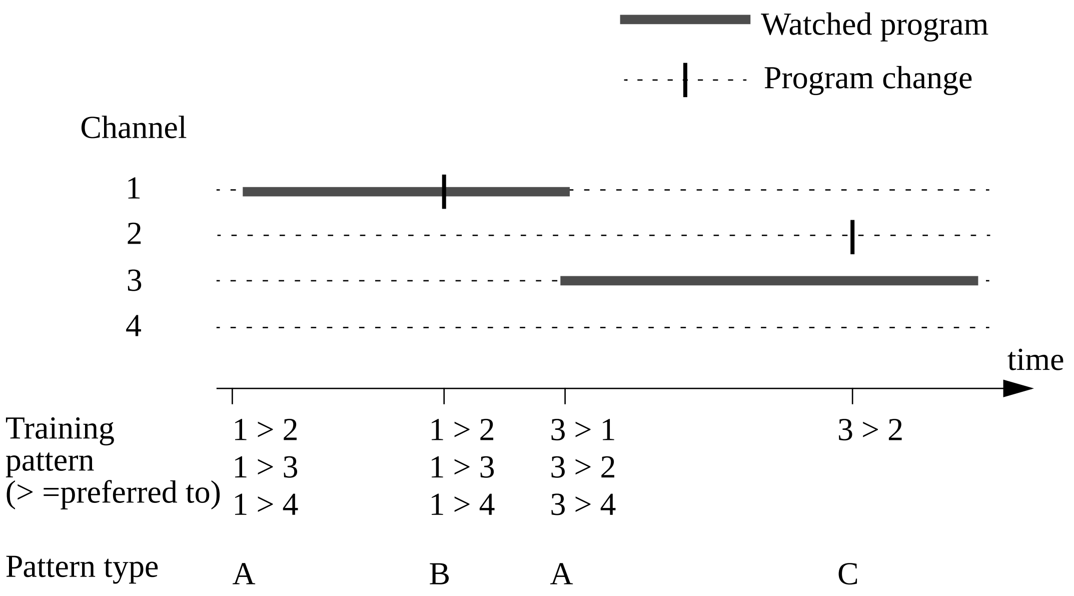


FIGURE 1. Example of training pattern generation

12 Network training.

There is a close relation between the number of training patterns, the number of links and the network's ability to generalize. When trying to predict the future, network generalization is the most important factor; it is the network's ability to do something good with a pattern that it has never seen before. By having dozens of hidden neurons and links the network could easily learn the training set to perfection. But the ability to generalize will be poor. This is because each training pattern will have its own path through the network, and this will reduce the network's ability to draw conclusions from a larger number of patterns. The knowledge is spread over a wider area with less interconnections. With fewer neurons the network is forced to have a more compact and general representation of its knowledge.

The ability to generalize is measured by testing the network with a validation set. The validation set consists of patterns that are not part of the training set. After each training cycle the network's performance is tested against the validation set. In a network with too many links, the generalisation error will at first fall as the network gets its first brief knowledge about the problem. But then, when the network learns more specific cases, the error will rise.

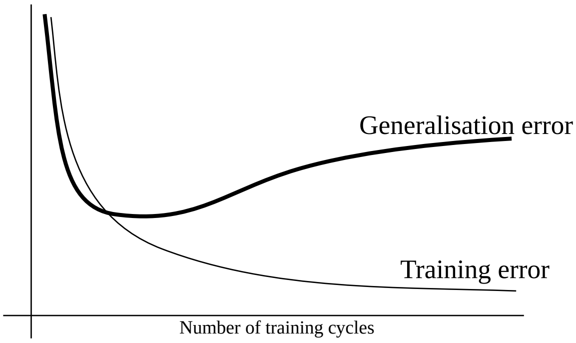


FIGURE 2. Example of Training and generalisation errors

12.1 How to limit the generalization error

A common technique is to stop the training when the generalization error increases. Another way is to reduce the number of hidden neurons and links so that perfect training is impossible. Generalization error is thus kept low. This will of course limit the possibilities to take advantage of an increased number of training patterns made available, but for experiments and testing it will help to keep computation time to a minimum.

13 Network architecture

Two different designs for arranging input were thought of:

- A network with two programs as its input. The network has as its main input information about two different programs. The task is then for the network to choose between these two programs. This information could then be used by a sorting algorithm.
- An all program input network. Program information from all available channels are fed into the network. The network then gives a relative score for the programs on each channel. The program with the highest score is the one expected to be the most wanted by the viewer.

I decided to choose the design that compares two programs. The reason for this was that having more inputs requires more test data, and that comparing two programs is more easily extended with an increasing number of channels. Only one new node for channel information will be needed for each new channel.

13.0.1 Input to the network

Input to the neural network consists of mainly two different parts:

- Time. The time of day is encoded as a twelve bit pattern. Each bit corresponds to two hours. The reason for not using more than twelve bits is the limited number of training patterns available.
- Program information. The program information is divided in the program currently watched by the viewer, a program in a neighbouring channel that is to be compared with the program seen by the viewer and the last program seen by the viewer (when there was a change in patterns).

Each of the three program information parts consist of:

- channel number, encoded as an seven bit pattern, where each channel has its own bit;
- short information rating, encoded as an eight bit pattern;
- content descriptor, encoded as a sixty four bit pattern[1].

Common to these kinds of pattern encoding is the fact that only one bit is active at a time. This was made to reveal the network from any overhead needed to decode more compact codes.

Output from the network comes from a single neuron. In training mode this neuron is used to inform the net, with a value of one or zero, which of the two programs being compared is preferred by the viewer.

13.0.2 Hidden neurons

Theoretically the number of links can be estimated for a back propagation neural network that works as a binary classifier. An approximation can be made from the formula (Haykin 1994):

$$N > \frac{W}{\varepsilon}$$

Where N is the number of training patterns, W the number of weights (links) and ε the fraction of errors permitted. With an accepted error of 10 percent and about 5000 training patterns the number of weights should be less than 500. This number of links would justify only two hidden neurons for a network with 249 inputs, a small network indeed. For these experiments I tested a number of configurations with up to a hundred neurons in one, two, or three layers. For networks with more than two hidden layers and more than 32 nodes in each, there was no significant improvement in network performance.

1. DVB-SI (1996) allows for a total of 256 different codes. However less than two hundred of these are in use. The test data used contains programs from only 64 of these categorises. Because of this the number of neurons used to represent the content descriptor was reduced to 64.

The final network with two hidden layers and 32 neurons in each, is depicted in Figure 3. The links between the layers have been removed for clarity's sake.

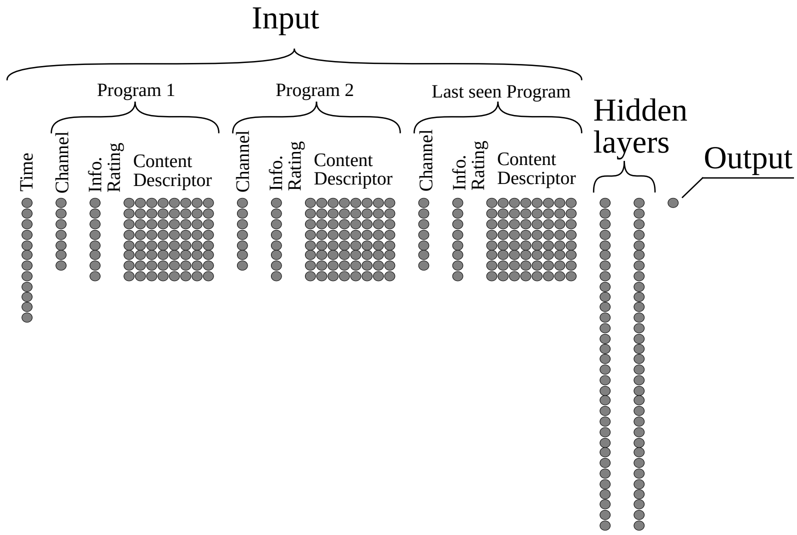


FIGURE 3. Final network design. Each dot represents a neuron.

All input neurons have a link to all neurons in the first hidden layer. All neurons in the first hidden layer are connected to all neurons in the second hidden layer. Finally all neurons in the second hidden layer are connected to the output neuron.

13.0.3 Output

The output is limited to be between zero and one, this is due to the sigmoid function at the output of the neuron (described under section 8 on page 9). Output is interpreted as a probability measure for the likelihood that the viewer prefers one program in front of the other.

14 Measuring network performance

The goal of the agent is to provide the viewer with a personal program schedule. Performance is therefore best measured as the viewer's satisfaction with the proposed programs. One way to measure viewer satisfaction is to count the number of minutes spent by the viewer on the proposed programs. However, these tests are based on historical data only; viewers have not seen the proposed schedules. Performance could be expected to increase if the viewer is influenced by the proposals made by the network. Therefore the performance is instead measured as the network's ability to predict a viewers viewing habits.

14.1 Calculating the value of a program

For every pair of programs the network calculates the likelihood that the viewer chose one or the other. Due to the finite number of training patterns, the network has a limited knowledge about pairs of programs never seen before. Because of this a circular relationship among a group of programs might be possible. For example: if program A is calculated to be more probable than program B and program B is more probable than program C, we can not be sure that A will be more probable than C. This could be the case if the network makes a misjudgement between A and C.

A circular relationship will cause a sorting algorithm to be trapped in an infinite loop or to produce an incorrect result.

If all pairs of programs are fed trough the network we get a matrix with output values. Table 1 shows an example matrix of probabilities for a viewers preference between pairs of programs. The values in the matrix is interpreted as the probability to choose the column programs in favour of the row programs. For example: the value in row 1 column 3 is 0.6, this means that program 3 is more interesting than program 1 with a probability of 0.6. A value of 0.5 means that the network is unable to distinguish between a pair of programs.

TABLE 1. Matrix of viewer preference probabilities

program	1	2	3
1	.5	.3	.6
2	.7	.5	.1
3	.4	.9	.5

In the example shown in Table 1 we see that program 1 > program 2 > program 3 > program 1 (> means more probable), which will yield no specific order for these programs.

The probability matrix can be visualized as a multi dimensional vector space. As shown in Figure 4, programs 1 to 3 are ordered in different ways depending on which dimension is considered.

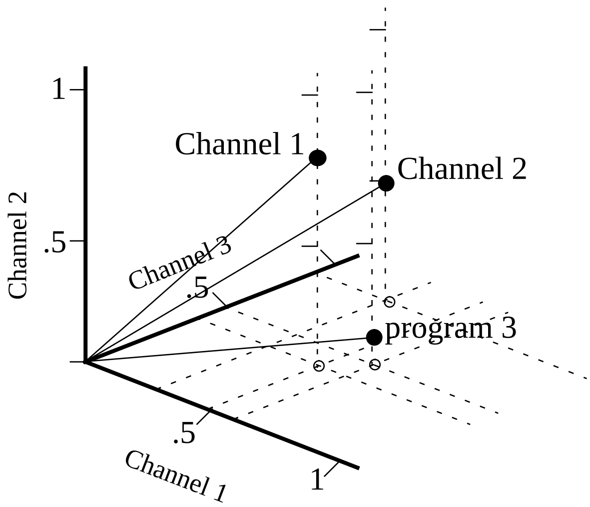


FIGURE 4. A vector space for three programs

To solve this contradiction the Euclidean distance is used. The heuristic for using the Euclidean distance is that program types often seen by the viewer are likely to get large probabilities in most dimensions. Misjudged programs will hopefully only have overestimated probabilities in a few dimensions.

The Euclidean distance for program k is calculated as:

$$\|r_k\| = \sqrt{\sum_{j=1}^{n} Ch_j^{\,2}}$$ Where n is the number of channels.

Using the example values in Table 1 we get the Euclidean distance in Table 2.

TABLE 2.

program	**|| *r* ||**
1	1.24
2	1.40
3	0.84

The programs will then be proposed in order of descending distances.

14.2 Measuring the overall performance

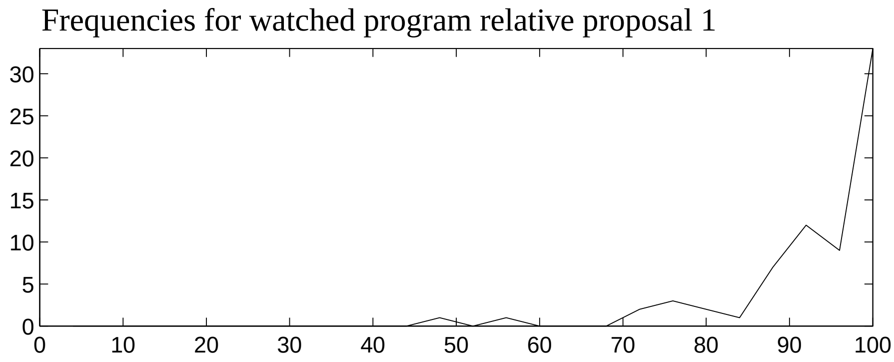
As the intention of the network is to propose programs that are interesting to a viewer, overall performance is best measured as the number of times the viewer has seen the programs proposed by the network. Table 3 shows how many times an example viewer has watched the proposals made by the network.

TABLE 3.

Proposal	Frequency	Relative frequency
1	207	0.38
2	156	0.28
3	120	0.22
4	34	0.06
5	18	0.03
6	10	0.02
7	5	0.01
Total	550	1

In a real TV situation with hundreds of channels a viewer might want to have a dozen or more proposals to choose from. However, with only seven channels, accepting a dozen programs as good proposals would yield a performance over a hundred percent! It is more realistic to only accept the number of times the viewer has watched the programs proposed as number one by the network as a performance measure.

However, before we reject the other proposals as uninteresting, we must first take a closer look at their Euclidean distances. If, for example the difference in distance between proposal two and one is minimal it would be a good idea to also accept proposal two as a good proposal.



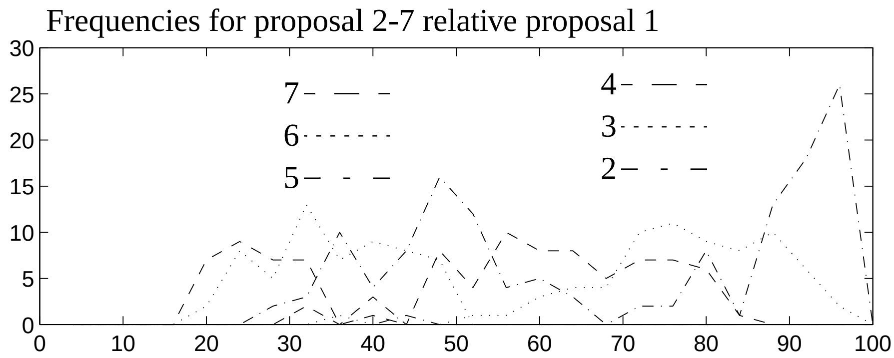


FIGURE 5. Relation between Euclidean distances

The first diagram in Figure 5 shows the relation between the Euclidean distance of the program seen by the viewer and the Euclidean distance of the program proposed as choice number one by the network. If the network had always proposed the program seen by the viewer there would be a single pulse at 100 percent.

The second diagram shows how the other proposals (7 to 2, left to right) relate to the first proposal. Here we can see that programs proposed as second choice often are close to choice one in its relative Euclidean distance. One can therefore say that even if the viewer has chosen proposal 2, the network has probably made a good judgement anyway.

Counting the first and second proposals as good proposals, the network made a good prediction in 207+156 cases of 550; about 66%.

15 Results

The group of 117 test persons primarily selected was further reduced for two reasons: Firstly many of the viewers did not have access to all seven channels. Secondly several viewers had spent very few hours in front of their TV. After removing these people, 20 people of varying ages were left. The test results for these 20 persons are presented in Appendix D and commented on here.

Most of the test persons had a top to the left for proposal seven and six in their second diagram. This is because when a channel is not broadcasting, its program is rated as 0.5 (maximum euclidean distance is the square root of 7 = 2.65). Channels that are not broadcasting are included to give an idea of how often they are closed.

Viewers 3,5,7-9 and 13-17 all have a high rated and relative narrow second proposal. This can be seen as an indication that it might be correct to include the frequency for both the first and the second proposals when calculating the performance for these fourteen persons.

The result for number 6 is very poor. This might be due to the relative few days of TV watching for this person.

TABLE 4.

Person	Rel. Freq for proposal 1	Rel. Freq for proposal 2	Rel. Freq for proposal 1+2
1	0.504	0.286	Not Applicable
2	0.234	0.231	Not Applicable
3	0.465	0.282	0.747
4	0.221	0.291	Not Applicable
5	0.433	0.191	0.624
6	0.219	0.125	Not Applicable
7	0.376	0.284	0.660
8	0.422	0.206	0.628
9	0.624	0.179	0.803
10	0.306	0.310	Not Applicable
11	0.505	0.296	Not Applicable
12	0.590	0.188	Not Applicable
13	0.320	0.306	0.626
14	0.488	0.245	0.733
15	0.382	0.268	0.650
16	0.315	0.321	0.636
17	0.859	0.122	0.981
18	0.717	0.150	Not Applicable
19	0.481	0.264	Not Applicable
20	0.508	0.221	Not Applicable

16 Discussion

If a viewer changes channel for a short moment, the new program will be accepted as the one preferred by the viewer for the moment and a pattern will be generated. The result is that the network learns erroneous patterns as well as correct patterns. A correct thing to do would be to take into account the amount of the program watched by the viewer.

One way to implement this idea could be by an using adjustable learning rate parameter that depends on how large part of a program the viewer has seen, i.e. the weights in the network will have smaller adjustments for smaller amounts of viewing.This would prevent the network from adapting too much to irrelevant swapping. The reason why this has not been tested is the difficulties in implementing this in the simulator used, SNNS.

The model used relies on the heuristic that the viewers spend most of their time watching interesting programmes; and that more patterns are generated during these periods than the others.

These tests were made on single people only. In a situation with more than one viewer, a TV agent must have some way to identify the viewers; otherwise it will perform poorly when the group is changed. In fact, most people watch TV in a group of two or more (Gahlin 1989).

The limited number of channels is definitely debatable. The main criticism is that seven channels is far too few compared to the hundreds of channels offered by digital TV. The advantage of having only seven channels is that the viewers can be expected to have choose the most interesting program available for every moment of time. The likelihood of a viewer watching a program just because he or she has not read through the whole program schedule, should be smaller than for a situation with more channels.

The main reason for having only seven channels was, however, that only the seven channels chosen could provide a detailed per minute log of what actually was broadcasted.

The program name and short info are stored in the same database to reduce the number of input nodes to the network. The network is expected to achieve increased performance if these are separated into two databases. The reason for this is that some serial programs have different program descriptions for each episode whereas others have a more general description used to describe all episodes. The result is that serial programs with the same description for each episode get a higher rating because more words are the same from time to time. With a separate database for program names the network can learn the difference between a recognized program name and its contents.

17 Conclusion

Neural network is a new and promising technology well suited to extract personal program schedules. This master's project has showed that with limited knowledge about TV viewers and an unrefined neural network, performance achieved is quite good.

The next step to take with personal TV program schedules is the user interface. How should the schedule be presented for maximum usability? Should the proposed programs be extracted and presented on their own or should they be kept in context with all other programs and supported with some sort of navigational aid?

This and other questions that will arise in the continued work will most certainly best be answered by the users.

18 References

DVB-SI, Digital broadcasting systems for television, sound and data services; Specification for Service Information (SI) in Digital Video Broadcasting (DVB) systems. SI-DAT 201 Rev. 4. pr ETS 300 468. ETSI 1996

Gahlin, A. T*ittarsituationen - om sällskap, bredvidsysslor och uppmärksamhet framför tv:n.* Sveriges radio publik och programforskning, Nr 16-1989.

Haykin, S. *Neural Networks: A Comprehensive Foundation*. Macmillan Publishing Company 1994

Jennings, N. R, and Wooldridge, M. *Intelligent Agents: Theory and Practice.* Knowledge Engineering Review, October 1994.

Karlgren, J., Höök, K., Lantz, A., Palme, J. and Pargman D. *The glass box user model for filtering.* Swedish Institute of Computer Science. S-164 28 Kista, Sweden 1994

Kilander, F. A *Brief Comparison of News Filtering Software*. http://www.dsv.su.se/~fk. 1996

Nwana, H. S. *Software Agents: An Overview.* Knowledge Enginering Review,Vol. 11,No 3, pp. 205-244, October/November 1996.

Russel, S. and Norvig, P. *Artificial Intelligence: A Modern Approach*. Prentice Hall Series in Artificial Intelligence 1995.

Wigren, G. *Hur informerar vi oss om tv-programmen?.* Sveriges radio publik och programforskning, PUB informerar 1990:III.

19 Recommended reading

Braspenning, P. J.,Thuijsman F. A.J.M.M. Weijters (eds.). *Artificial neural networks : an introduction to ANN theory and practice.* Lecture Notes in Compure Science 931, Springer.

Franklin, S. and Graesser, A. Is it an Agent, or just a Program?: *A Taxonomy for Autonomous Agents.* Proceedings of the Third International Workshop on Agent Theoris, Architectures and Languages, Springer-Verlag, 1996.

Jennings, N. R. and Wooldridge, M. *Intelligent Agents.* (LNAI Volume 890). Springer-Verlag 1994

Müller, J. P., Tambe M. and Wooldridge, M. *Intelligent Agents II: Agent Theories, Architectures and Languages,* (LNAI Volume 1037). Springer-Verlag 1995

20 WWW Home pages

Mediamätning i skandinavien AB, MMS
http://www.mms.se

Stuttgart Neural Network Simulator
http://www.informatik.uni-stuttgart.de/ipvr/bv/projekte/snns/snns.html

Appendices

Appendix A Tools

SNNS, Stuttgart Neural Network Simulator

To avoid spending to much time on implementing the network architecture I decided to use SNNS, an extensive neural network simulator developed at the University of Stuttgart. The simulator has a graphical user interface and numerous of functions. See Appendix C.

Viewer Program Log Data Base, VPLDB

I created a small database program in order to simplify conversion of input data from MMS and extraction of data for SNNS.

Output, the evaluate result function

The Evaluate result program works as a post-processor to SNNS. It takes as its arguments the test pattern file created by VPLDB and the result file from SNNS.

The resulting error is also weighted with the duration of the error. If the network fails to predict a short excursion to a different channel this should have less impact on the overall performance than if it misses a whole movie.

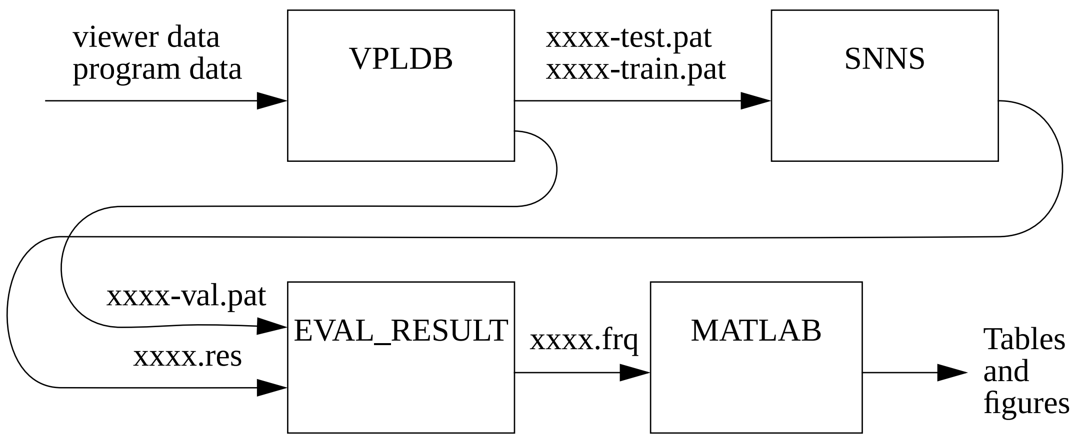


FIGURE 6. Data flow

Appendix B About the data from MMS

The data was collected by A.C Nielsen Company AB and is owned by Mediamätning i skandinavien AB, MMS.

The example data from MMS is available through KTH, NADA, CID for uncommercial use. The data is owned and marketed by MMS only.

Appendix C Stuttgart Neural Network Simulator

SNNS (Stuttgart Neural Network Simulator) is a software simulator for neural networks on Unix workstations developed at the Institute for Parallel and Distributed High Performance Systems (IPVR) at the University of Stuttgart. The goal of the SNNS project is to create an efficient and flexible simulation environment for research on and application of neural nets.

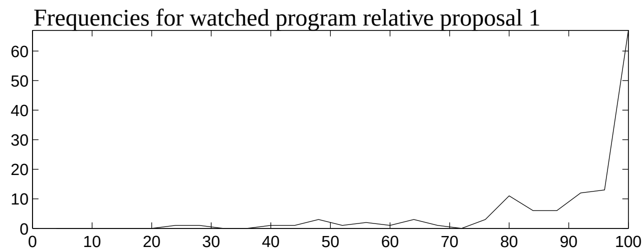
More information about SNNS can be retrieved from:
http://www.informatik.uni-stuttgart.de/ipvr/bv/projekte/snns/snns.html

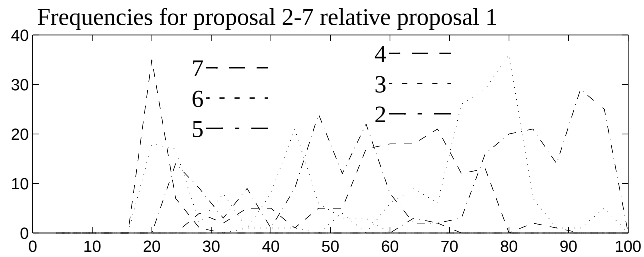
Appendix D Results and figures

- Frequency. This is the number of times the viewer has chosen a proposed program.
- Minutes. This is the total amount of minutes the viewer has watched a proposal.

Person 1 Age: 35, Male, Time: 18 days

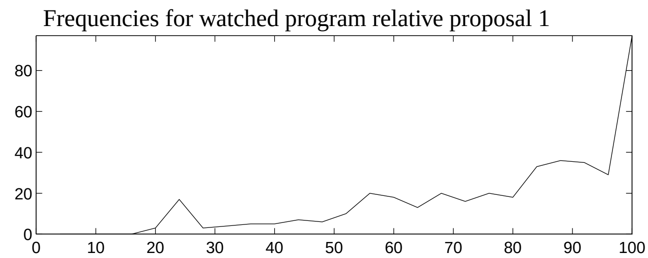
Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	67	0.504	208	0.399
2	38	0.286	202	0.388
3	13	0.098	40	0.077
4	11	0.083	53	0.102
5	3	0.023	12	0.023
6	1	0.008	6	0.012
7	0	0.000	0	0.000
Total	133		521	

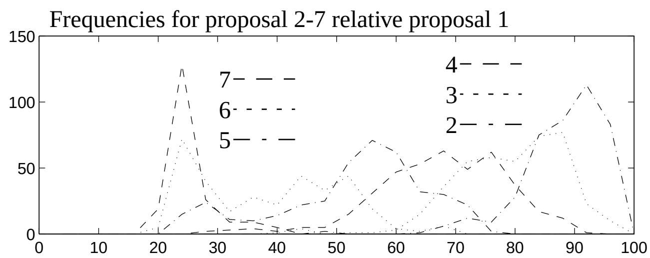




Person 2 Age: 33, Male, Time: 20 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	97	0.234	238	0.229
2	96	0.231	241	0.232
3	67	0.161	161	0.155
4	64	0.154	145	0.140
5	51	0.123	130	0.125
6	30	0.072	81	0.078
7	10	0.024	43	0.041
Total	415		1039	

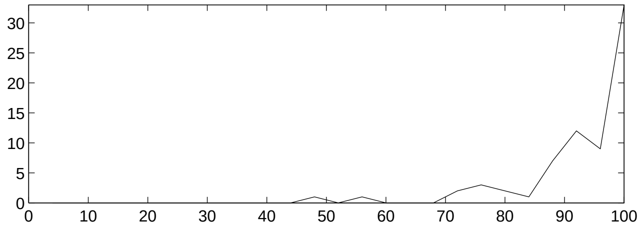




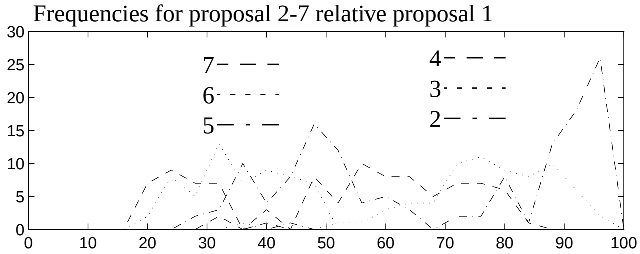
Person 3 Age: 28, Female, Time: 23 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	33	0.465	147	0.430
2	20	0.282	55	0.161
3	16	0.225	130	0.380
4	0	0.000	0	0.000
5	2	0.028	10	0.029
6	0	0.000	0	0.000
7	0	0.000	0	0.000
Total	71		342	

Frequencies for watched program relative proposal 1



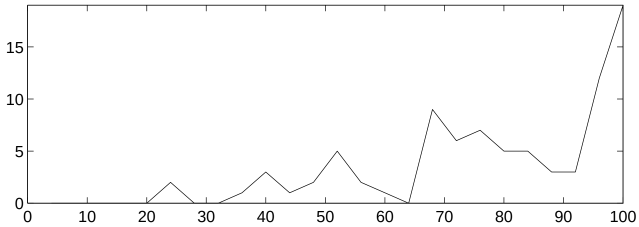
Frequencies for proposal 2-7 relative proposal 1



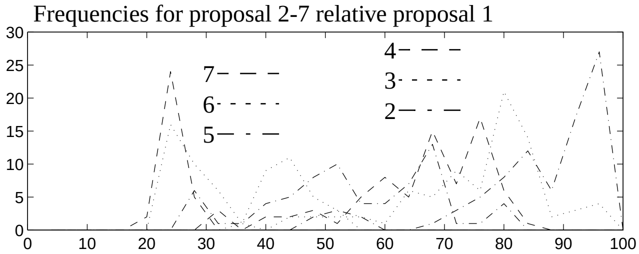
Person 4 Age: 64, Male, Time: 35 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	19	0.221	55	0.231
2	25	0.291	64	0.269
3	14	0.163	39	0.164
4	13	0.151	39	0.164
5	11	0.128	36	0.151
6	3	0.035	4	0.017
7	1	0.012	1	0.004
Total	86		238	

Frequencies for watched program relative proposal 1



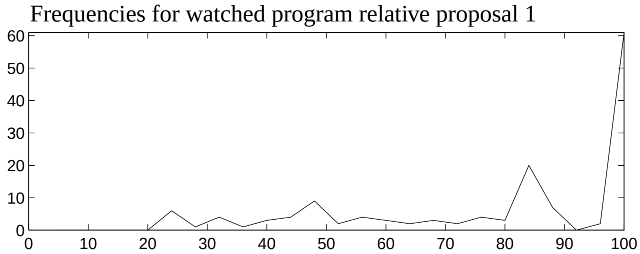
Frequencies for proposal 2-7 relative proposal 1



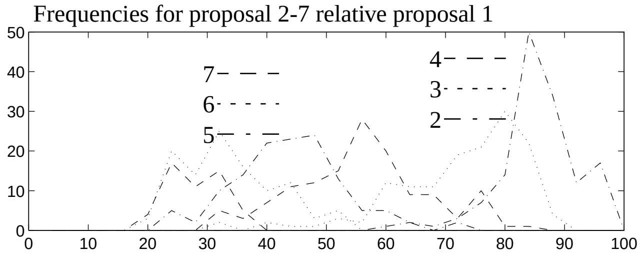
Person 5 Age: 30, Female, Time: 30 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	61	0.433	166	0.437
2	27	0.191	75	0.197
3	19	0.135	46	0.121
4	9	0.064	22	0.058
5	16	0.113	35	0.092
6	7	0.050	31	0.082
7	2	0.014	5	0.013
Total	141		380	

Frequencies for watched program relative proposal 1

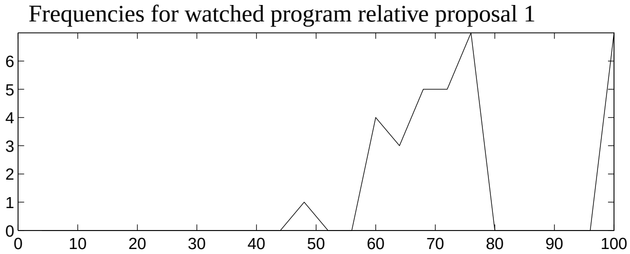


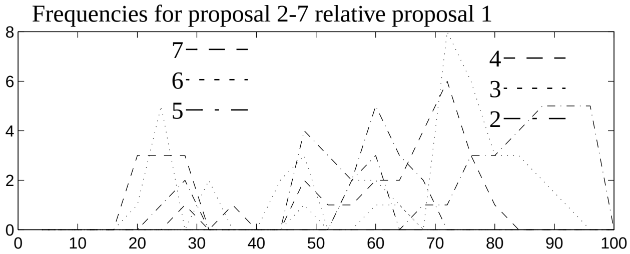
Frequencies for proposal 2-7 relative proposal 1



Person 6 Age: 53, Female, Time: 17 days

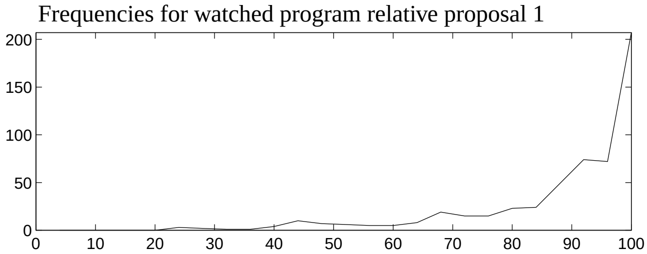
Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	7	0.219	62	0.408
2	4	0.125	7	0.046
3	8	0.250	17	0.112
4	8	0.250	32	0.211
5	3	0.094	24	0.158
6	2	0.062	10	0.066
7	0	0.000	0	0.000
Total	32		152	

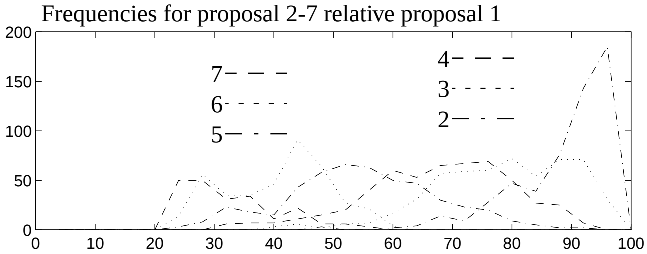




Person 7 Age: 35, Male, Time: 35 days

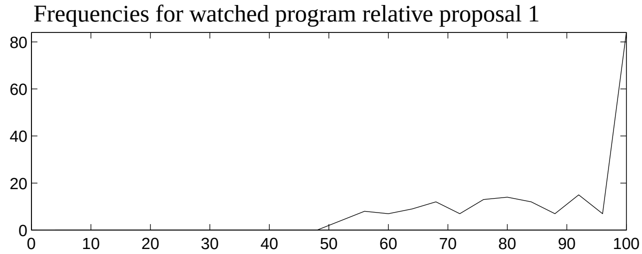
Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	207	0.376	687	0.387
2	156	0.284	463	0.261
3	120	0.218	438	0.247
4	34	0.062	89	0.050
5	18	0.033	65	0.037
6	10	0.018	23	0.013
7	5	0.009	9	0.005
Total	550		1774	

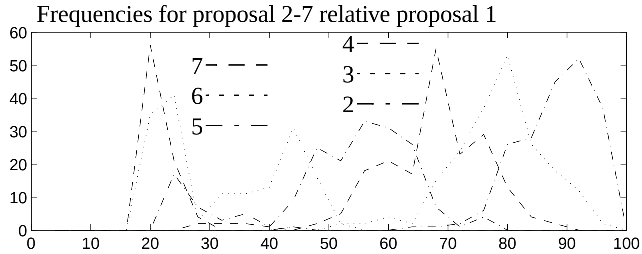




Person 8 Age: 58, Female, Time: 35 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	84	0.422	348	0.443
2	41	0.206	171	0.218
3	35	0.176	91	0.116
4	23	0.116	102	0.130
5	16	0.080	74	0.094
6	0	0.000	0	0.000
7	0	0.000	0	0.000
Total	199		786	

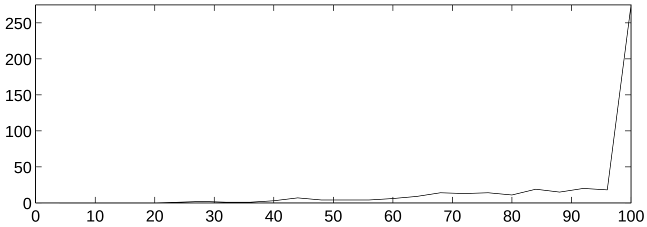




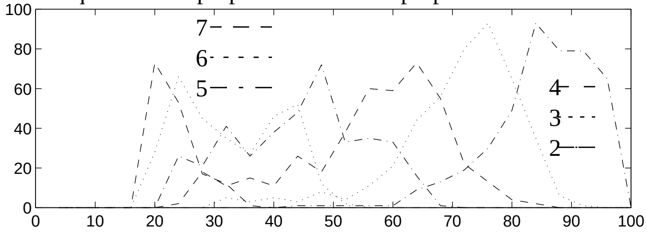
Person 9 Age: 35, Male, Time: 27 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	275	0.624	1077	0.643
2	79	0.179	299	0.178
3	40	0.091	128	0.076
4	24	0.054	83	0.050
5	11	0.025	29	0.017
6	12	0.027	60	0.036
7	0	0.000	0	0.000
Total	441		1676	

Frequencies for watched program relative proposal 1



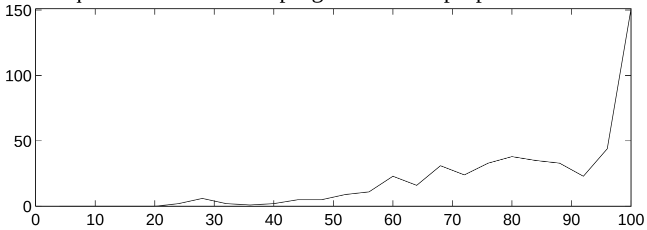
Frequencies for proposal 2-7 relative proposal 1



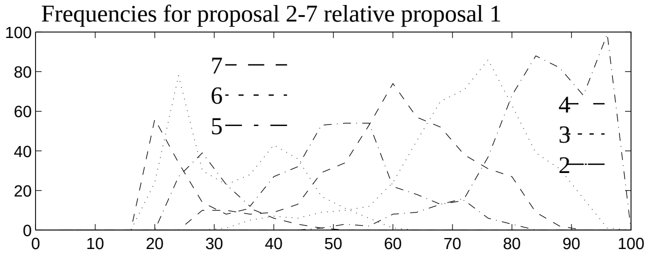
Person 10 Age: 54, Female, Time: 35 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	151	0.306	526	0.293
2	153	0.310	482	0.268
3	89	0.180	298	0.166
4	66	0.134	360	0.200
5	24	0.049	83	0.046
6	9	0.018	40	0.022
7	2	0.004	7	0.004
Total	494		1796	

Frequencies for watched program relative proposal 1



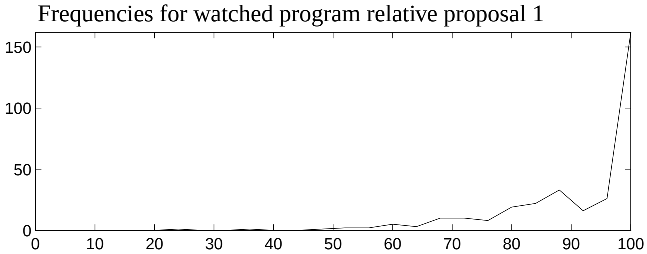
Frequencies for proposal 2-7 relative proposal 1



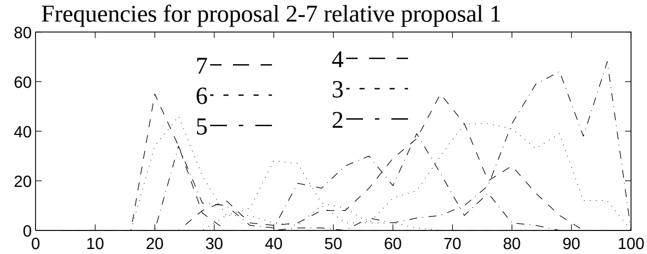
Person 11 Age: 50, Male, Time: 35 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	162	0.505	501	0.462
2	95	0.296	339	0.313
3	26	0.081	105	0.097
4	22	0.069	97	0.089
5	15	0.047	41	0.038
6	1	0.003	1	0.001
7	0	0.000	0	0.000
Total	321		1084	

Frequencies for watched program relative proposal 1



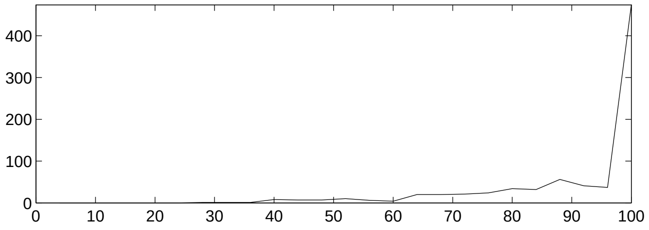
Frequencies for proposal 2-7 relative proposal 1



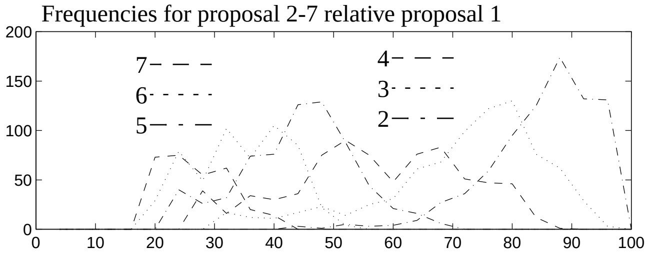
Person 12 Age: 68, Female, Time: 33 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	474	0.590	1416	0.608
2	151	0.188	408	0.175
3	101	0.126	289	0.124
4	46	0.057	132	0.057
5	23	0.029	58	0.025
6	8	0.010	24	0.010
7	1	0.001	3	0.001
Total	804		2330	

Frequencies for watched program relative proposal 1



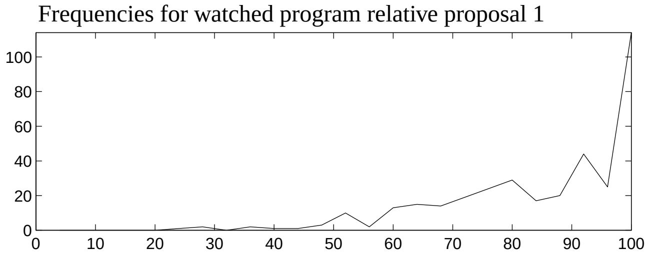
Frequencies for proposal 2-7 relative proposal 1



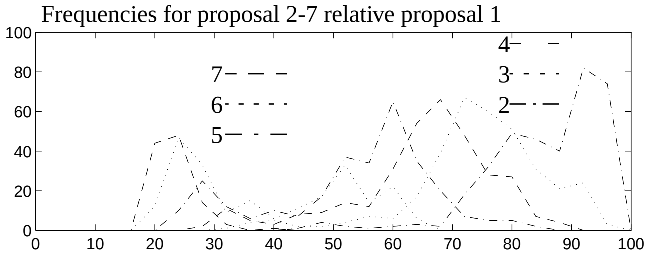
Person 13 Age: 57, Female, Time: 32 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	114	0.320	389	0.281
2	109	0.306	473	0.342
3	56	0.157	233	0.168
4	36	0.101	117	0.085
5	36	0.101	156	0.113
6	4	0.011	13	0.009
7	1	0.003	2	0.001
Total	356		1383	

Frequencies for watched program relative proposal 1



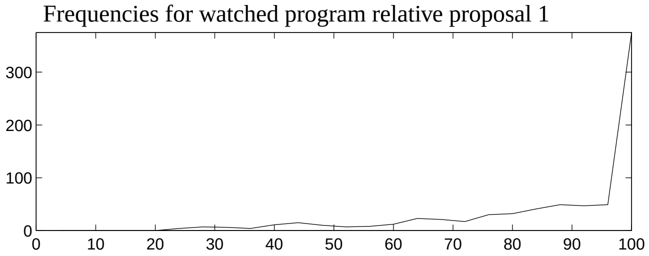
Frequencies for proposal 2-7 relative proposal 1



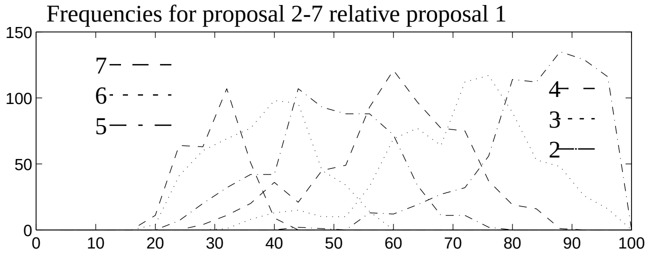
Person 14 Age: 71, Male, Time: 33 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	375	0.488	1250	0.532
2	188	0.245	571	0.243
3	100	0.130	316	0.135
4	48	0.062	110	0.047
5	28	0.036	55	0.023
6	26	0.034	44	0.019
7	3	0.004	3	0.001
Total	768		2349	

Frequencies for watched program relative proposal 1

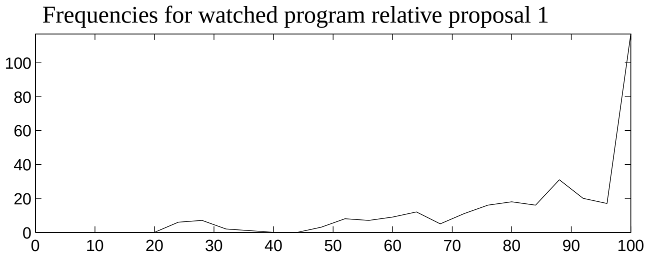


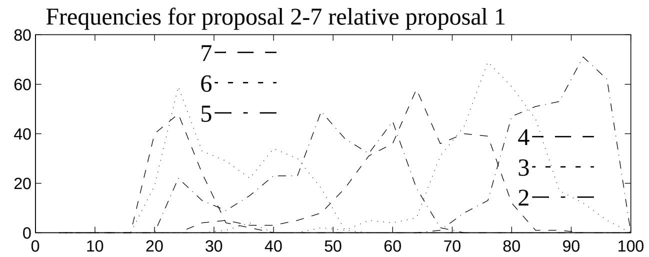
Frequencies for proposal 2-7 relative proposal 1



Person 15 Age: 61, Male, Time: 35 days

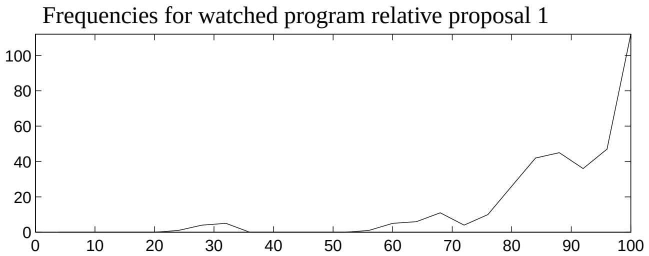
Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	117	0.382	367	0.380
2	82	0.268	304	0.315
3	34	0.111	83	0.086
4	39	0.127	110	0.114
5	20	0.065	70	0.073
6	7	0.023	17	0.018
7	7	0.023	14	0.015
Total	306		965	

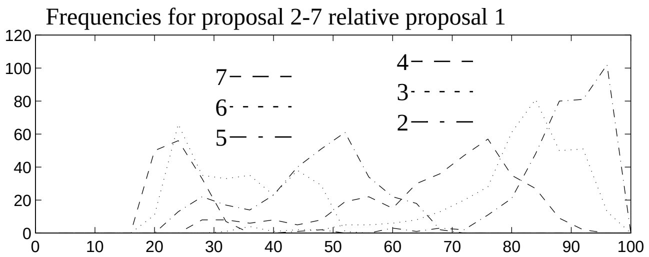




Person 16 Age: 26, Female, Time: 31 days

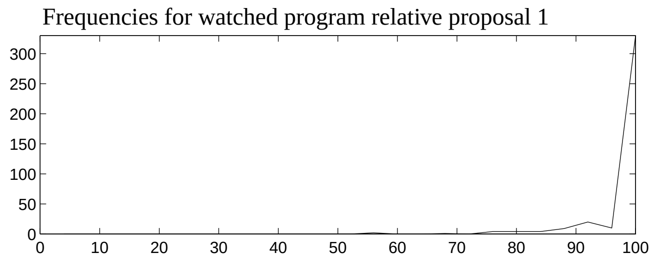
Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	112	0.315	429	0.334
2	114	0.321	376	0.293
3	84	0.237	349	0.272
4	26	0.073	81	0.063
5	12	0.034	36	0.028
6	4	0.011	6	0.005
7	3	0.008	7	0.005
Total	355		1284	

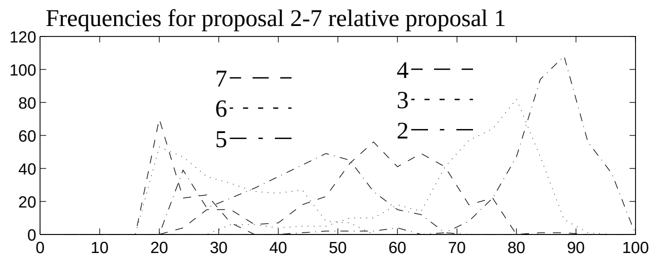




Person 17 Age: 79, Male, Time: 29 days

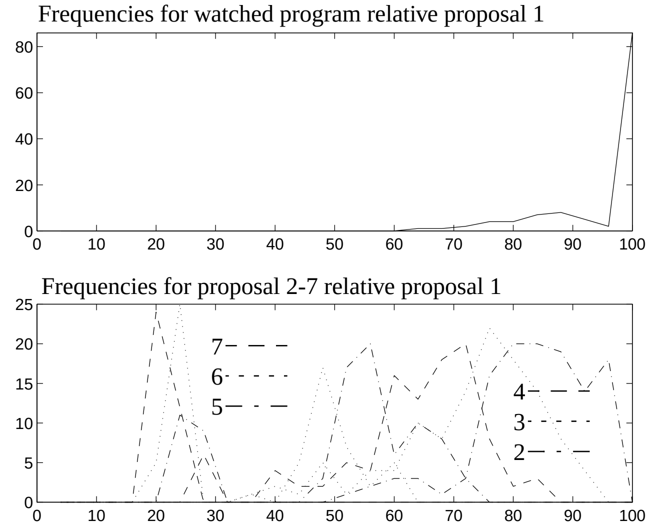
Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	330	0.859	1107	0.818
2	47	0.122	206	0.152
3	4	0.010	38	0.028
4	2	0.005	2	0.001
5	1	0.003	1	0.001
6	0	0.000	0	0.000
7	0	0.000	0	0.000
Total	384		1354	





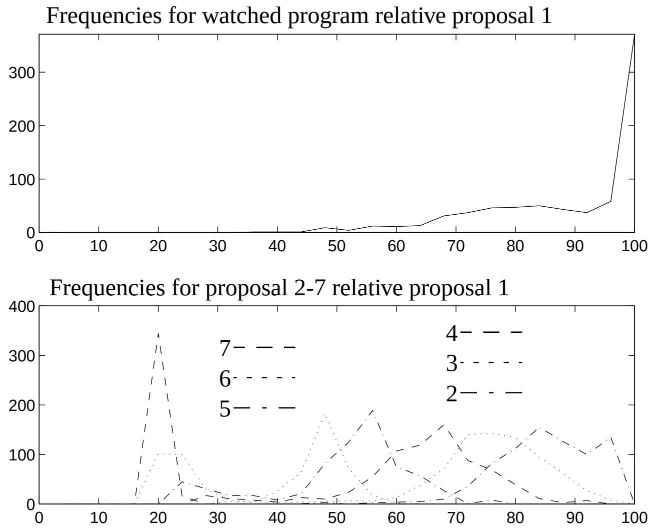
Person 18 Age: 26, Male, Time: 20 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	86	0.717	341	0.678
2	18	0.150	97	0.193
3	12	0.100	60	0.119
4	3	0.025	4	0.008
5	1	0.008	1	0.002
6	0	0.000	0	0.000
7	0	0.000	0	0.000
Total	120		503	



Person 19 Age: 55, Male, Time: 35 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	371	0.481	881	0.519
2	204	0.264	383	0.226
3	123	0.159	290	0.171
4	38	0.049	76	0.045
5	28	0.036	52	0.031
6	8	0.010	15	0.009
7	0	0.000	0	0.000
Total	772		1697	



Person 20 Age: 49, Male, Time: 35 days

Prop.	Freq.	Rel.frq.	Min.	Rel.min.
1	317	0.508	1132	0.510
2	138	0.221	447	0.201
3	88	0.141	330	0.149
4	44	0.071	159	0.072
5	30	0.048	118	0.053
6	7	0.011	35	0.016
7	0	0.000	0	0.000
Total	624		2221