*Paper*

# Decision support tool for web cache management

Jarosław Pietrzykowski

**Abstract — Web caching is the subject of intense research and development since it seems to be very promising area. Web caching means storing copies of frequently used objects (documents) geographically close to users requesting them to reduce network load, servers load and user response times. Cache can be situated in different locations between user and servers with original content. It seems that the most significant improvement can be achieved by using the proxy server – a dedicated web server. Various parameters affect web cache performance: cache size, limitations on document sizes or documents removal policy. Several metrics are used to evaluate this performance: hit rate for example is the ratio of documents obtained by using the cache mechanism comparing to the total number of documents requested. Choosing adequate parameters for interesting traffic patterns to improve cache performance is not a trivial cache management problem. Prototype tool based on multicriteria model analysis and supporting such web cache management is presented. Simple case was examined where small number of sets of cache parameters (variants), miniature traffic representation and only two performance measures are considered.**

***Keywords — web cache management, multicriteria analysis.***

## 1. Introduction

In the era of immense grow of Internet advanced solutions for network overload reduction have special importance. One of such solutions is web cache acting as a proxy between users and web sites. This paper concentrates on some issues connected with web proxy cache management. Problem considered here concerns choosing configuration parameters for web cache for a given requests stream representation in order to improve cache performance.

Suggestions for using cache parameters are general and does not necessarilly apply to individual web cache placed in specific network environment. Such environment consists of population of computer users and computer infrastructure providing connectiviness between users and access to external network, like Internet. Users activity induces specific traffic for their network. This traffic commonly does not bear constant characteristic. Some patterns emerge repeatadly and some new can be observed. Experienced web cache managers are able to tune parameters mechanism accordinlgy to changes in traffic directed to the cache. Nevertheless assesment of the influence of cache parameters on its perfomance becomes harder with each new version of the software because number and complexity of parameters increases. Necessity of choosing from only dozen of configuration alternatives can be very confusing and human intuition can fail.

This paper presents concept and its realization of decision support tool that could help web cache managers with choosing adequate configuration for better cache performance. Such tool should conform to several requirements. It should allow for examination many configuration alternatives for interesting patterns of requests stream flowing into the cache. During evaluation of cache performance for these alternatives several criteria should be regarded. Important feature of software supporting decision making is to enable clear and easily understandable expression of decision maker preferences concerning used criteria. It is crucial for the correct measurement to separate examined cache from unstability of external network and ensuring that experiment is repeatable. Analysis of cache configuration alternatives should not disturb regular work of cache software. Such analysis should also allow for fast evaluating many alternatives. High level of automation and graphical, user-friendly interface are desirable.

In short perspective development of such tool complying with presented requirements was the main target of the work presented here. This includes inventing detailed concept of the tool, creating its components and testing them separately and assembled together and also examining the whole on some real example. This stage of work presented in this paper was intended also for gaining better acquaintance with web cache subject, capabilities of this category of computer software and for identification of specific problems. Some important results of this work are software component generating mathematical model for considered decision problem and two-phase design of performance measurument operation so that appropriate requirements are satisfied.

In the future more advanced system for automatic web cache configuration is planned. Such system will recognize traffic patterns in requests stream flowing into the cache and will apply corresponding set of parameters in order to obtain best performance. The role of the tool presented in next part of this paper will be to match configurations with new patterns for future use by this automated system.

Following chapter contains simple description of web cache concept and associated issues. Then details of the tool supporting tuning of web cache parameters and outline of its advancement to automated system are presented. Next, application of the tool is illustrated with some simple case. Finally short summary follows.

# 2. Web cache concept

### *2.1. Internet traffic explosion*

Tremendous increase in Internet traffic has been observed recently, especially with regard to World Wide Web area [1]. This results in servers overload, congested networks and delays in response times of document requests. This trend is augmented by fast development of new Internet applications such as electronic commerce, business-to-business exchange and multimedia communication. There is a huge demand for data, information and knowledge concerning almost all human activities.
Apparently, this demand cannot be satisfied by advances in Internet infrastructure development like high capacity backbone networks, cable modems or radio access. Other approaches are neccessary that apply more "soft" techniques like server load balancing, better traffic management or web caching.
Web caching basically means that copies of popular WWW documents (such as html pages, graphics or audio files – altogether called objects) are kept geographically close to users requesting them. When document request comes to web cache its mechanism tries to handle it by retrieving desired document from its own storage space. In the case of success document is delivered immediately and "hit" is recorded. Otherwise requested object has to be downloaded from external network, usually from the server where original document resides. In such situation web cache "miss" is recorded. Cache parameters and document's features determine if this document is stored in web cache for future use.
The idea of caching is not new and has been widely used in computer science area. Applications of this mechanism can be found both at hardware and software level. Usually it causes significant increase in performance. Recent incredible advance in microprocessor design partially results from sophisticated use of this concept.

### *2.2. Benefits of using web cache*

Advantages of using cache depend on its location within network structure. Basically there are three possible places on the way from user to server with desired content where cache is situated:

1) user's machine – **client cache**;

2) original server – **server cache**;

3) in the middle of network – **proxy cache**.

These three types of web cache are shown in Fig. 1.
In the first case user of client machine benefits most from applying cache. But still if many users need access to same document corresponding number of objects must be downloaded from the original server.

When web cache is situated on the side of server with original content this does not decrease network overload significantly. Nevertheless it reduces load on the server and shortens response time to the document request.

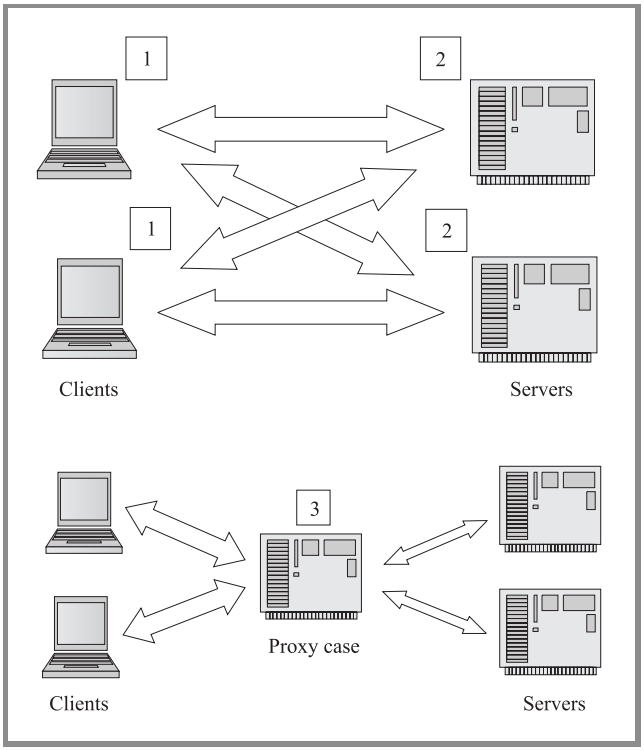


***Fig. 1.*** Proxy cache placement in the client-server network.

Proxy web cache is the most fruitful solution. Response time is reduced considerably because it is kept closer to the requester. Additionally, if document is popular among other users it results in substantial decrease in network congestion since number of requests is much smaller. This approach also causes reduction of original server load. This is especially important when Internet access is expensive. Proxy caches can make up hierarchies, where bigger ones handle smaller ones' requests or they can be organized as peer nodes network. There are some large national proxy cache networks which are used intensly by scientific society. SURFnet in The Netherlands or UNINETT in Norway can suit as an example (these two are connected to each other). Savings resulting from applying this solution are reported to be 30% to 50% of the traffic, depending on the traffic characteristic[1]. At all levels of such hierarchy advantages prevail drawbacks and significant decrease in response time is noticed. Although benefits from using web cache are the most apparent in large organizations it is recognized that such mechanism is useful also in small scale. Web proxy cache software finds also other applications within organization, such as:

- incoming traffic filtering;
- accelerating access to overloaded servers;

[1]http://www.desire.org/html/services/caching/

- protection against computer assaults from external network.

Web cache proxy mechanism is one of the most important solutions providing improvement in network utilization efficiency. It has been area of intense research for new algorithms, traffic models and other concepts leading to extending set of parameters designed for better performance. In the following parts of this paper term "web cache" means its proxy type since it is more concise form and can be used without confusion.

### *2.3. Performance measures*

Web cache performance can be measured in several ways. The most popular metrics are:

- **document hit ratio** – defined as ratio between number of documents delivered with the use of web cache mechanism and total number of requests;
- **byte hit ratio** or **weighted hit ratio** – amount of bytes retrived from network using cache compared to total amount of bytes requested;
- **average user response time**;
- **bandwidth utilization**.

There is a trade-off between first two metrics. Often web cache load analyses show that most of requests pertains to small documents. Therefore, in order to achieve high document hit ratio large number of smaller documents should be kept in cache. On the other hand these analyses demonstrate that transfers of huge documents increase network traffic significantly. So as to obtain high byte hit ratio several large documents should be stored in cache at the expense of smaller ones. Importance of these metrics depends on situation:

- when response time reduction is crucial – document hit ratio should affect cache configuration decisions mostly;
- when saving bandwidth is critical – cache configuration should result in high byte hit ratio values.

There are factors influencing web cache performance that are beyond managing person's control. Among these factors are some documents features determined by their authors that prevent them from caching or that set time limit for keeping such documents in cache with regard to their up-to-dateness. Usually authors or owners of web documents want to avoid caching because of possible decrease in their profits. Profitability of many web sites depends on count of accesses to presented web pages (documents). When pages are kept in proxy cache one cannot be sure if their download counts are properly reported to original server. Security concerns are also one of reasons against caching. Documents stored on unknown machine somewhere in Internet cannot be controlled sufficiently by their owners and danger of content abuse is real.

### *2.4. Configuration parameters*

Modern web cache software is complex. Squid - the most commonly used application has more than 150 configuration parameters. These parameters cover many aspects of web cache functionality including network setup, timeouts, access rights and other. Many parameters affect web cache performance significantly and allow controlling the way this mechanism works. Among them are:

- amount of computer memory dedicated for documents storage;
- disk space amount dedicated for caching;
- document replacement policy – determines how many and which documents are swept out from storage space when there is no room for freshly retrieved ones;
- maximal size of kept documents;
- minimal size of kept documents – both parameters allow traffic filtering;
- swap upper threshold (or **cache swap high**) – this parameter is algorithm-specific and means level when documents begin to be swept out more aggressively;
- swap lower threshold (or **cache swap low**) – when this level is approached process of removing documents from cache starts;
- method of stored content refreshment:
  - **passive caching**: documents are stored in cache only if it is requested by client machine;
  - **active caching**: freshness of documents is checked by cache itself which is useful option for popular but quickly outdateing documents;
- rules for document refreshing – specify when document is considered fresh.

Very important parameter is the cache removal policy. Software used in the research presented here implemented three types of algorithms for this task:

- LRU (last recently used) – documents not requested for the longest period are removed;
- GDSF (greedy dual size frequency) – removal is performed on the basis of sizes of documents and cost of their retrieval from the external network;
- LFUDA (least frequently used with dynamic ageing) – documents that have been requested less frequently and of certain age are removed.

Web cache performance is an area of many studies. This results in still extending set of cache parameters. For example the last two algorithms stated above were developed in Hewlett-Packard laboratories and were shortly after implemented in another version of web cache software.

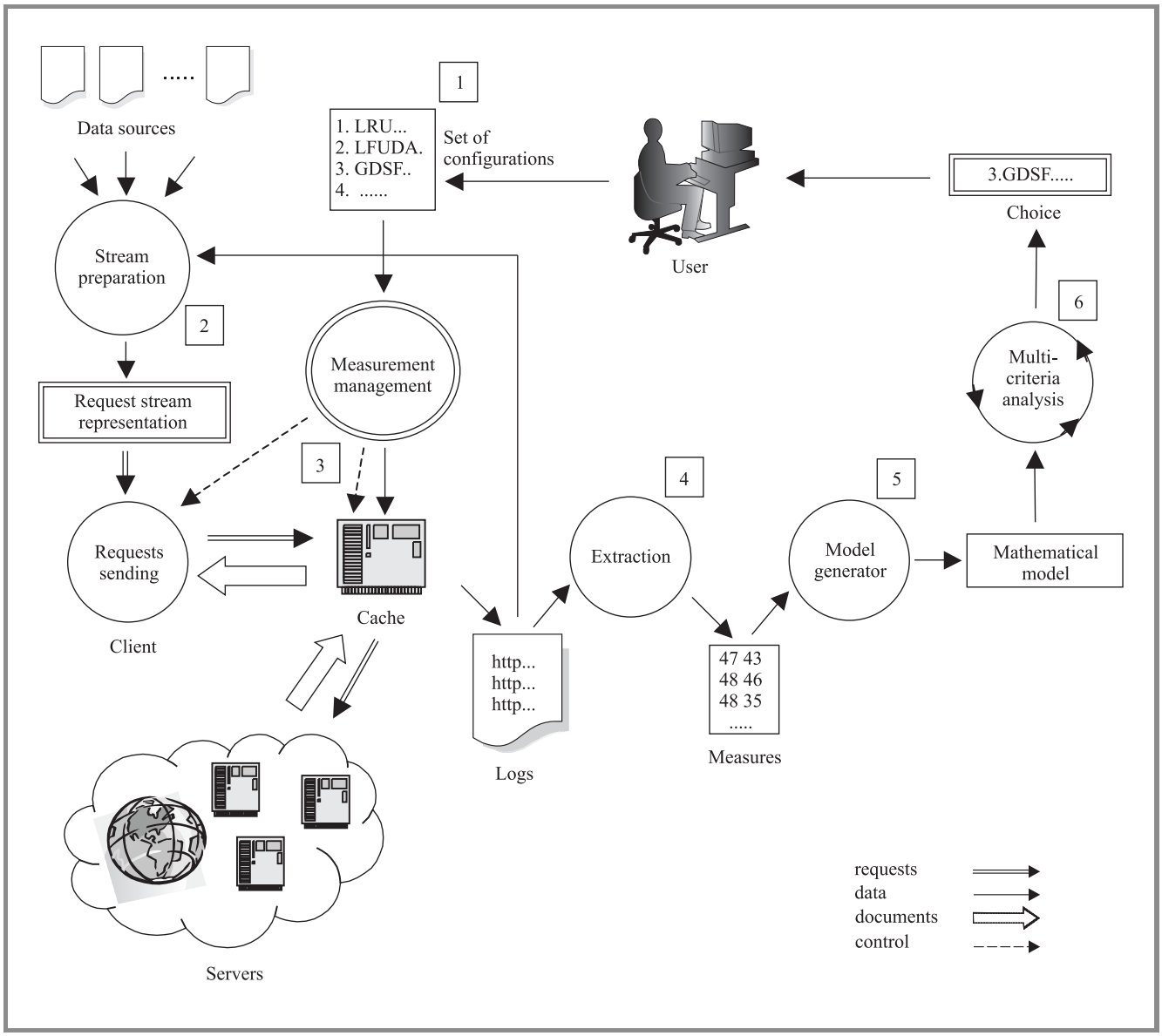


***Fig. 2.*** Process of analysis of decision alternatives for proxy cache configuration.

## 3. Decision support tool concept

In order to meet requirements presented earlier the tool supporting web cache management should be able to perform several tasks. These tasks should be organized in a process that leads to reasonable choice about better configuration for examined cache. Figure 2 presents such process. In this process, for a given representation of stream of documents requests a number of possible sets of proxy cache parameters are examined. Each set of parameters (configuration) can be seen as a decision alternative. Collected performance results are used to evaluate these alternatives. Multicriteria analysis methodology based on mathematical modeling is used for the purpose of analyzing decision alternatives. The effect of such analysis is the best – according to preferences of person performing analysis – configuration, that can be matched with considered requests stream representation.

The process illustrated in Fig. 2 consists of following stages:

1) preparation of configuration alternatives;

2) preparation of requests stream representation;

3) performance measurement;

4) extraction of output variables;

5) mathematical model generation;

6) multicriteria problem analysis.

Stages 2 and 3 are coupled because of requirements presented earlier that performance measurement should meet.

The details of this process are presented in the following parts of this chapter.

At present the elements of the process constitute rather some analytical environment than fully integrated tool. Nevertheless valuable analyses of web cache configuration alternatives for interesting requests patterns are possible. Author hopes it will be helpful for web cache managers to improve web cache performance.

The process being described here has been implemented as several connected components responsible for performing individual stages mentioned above. Some of the components are UNIX shell scripts, some are realized as programs written in C++ language and some of them are ready to use pieces of software. The latter enclose ISAAP (interactive specification and analysis of aspiration-based preferences) modular tool and MOMIP (modular optimizer for mixed integer programming) solver, which are used for multicriteria model analysis, and WGET program used for sending prepared requests to web cache.

Although some of the stages – like performance measurement and extraction of output variables – were automated, there are still some tasks (namely: preparation of configuration alternatives and preparation of requests stream representation) that has to be done manually or with use of other tools. These parts need more automation so that they could be seamlessly integrated with other elements into one tool. In the future such tool should coordinate performed tasks and provide user-friendly graphical interface. X Windows or Java environments seem to be quite suitable for this purpose.

The software used in this research is distributed either freely as an open-source code (web cache) or with the GNU license (request sending client) or free of charge for non-commercial and educational purposes (multicriteria analysis tool and solver).

All experiments were planned for and conducted with appliance of Squid[2] software (version 2.3.STABLE4) running on Sun Solaris platform. Although there are WWW servers with built-in cache functionality (i.e. Apache) Squid is dedicated for caching. Thus its code is less complicated and more reliable. Squid is also most widely used software in its category and has been awarded several times.

### *3.1. Preparation of configuration alternatives*

This task is performed manually. During this stage a file describing the plan for the experiment has to be prepared in a special format (Fig. 3). In this file there are two sections:

# para – this section contains in each row names of the cache parameters and number of their values that are to be used in the experiment;

# data – includes combinations of values of examined parameters.

```
#para
replacement_policy 3
cache_swap_low 3
cache_swap_high 2
#data
LRU 80 95
LFUDA 80 95
GDSF 80 95
LRU 85 95
LFUDA 85 95
GDSF 85 95
LRU 90 95
LFUDA 90 95
GDSF 90 95
LRU 80 100
LFUDA 80 100
GDSF 80 100
LRU 85 100
LFUDA 85 100
GDSF 85 100
LRU 90 100
LFUDA 90 100
GDSF 90 100
```

***Fig. 3.*** Example of configuration alternatives specification.

Preparing only several alteratives of cache configurations can be done manually but there can be demand for dozens to be examined. This can happen quite often since when dealing with only few parameters hundreds combinations of their values make up possible alternatives. It is also possible that some of this combinations are not interesting for cache manager and should be excluded. This leads to the conclusion that this stage must be automated in order to make the presented web cache management support tool truly useful.

### *3.2. Preparation of requests stream representation*

Requests stream representations can be prepared from several sources:

- server's logs;
- browser's logs;
- web cache's logs;
- traffic simulators;
- files prepared by hand.

In effect such representation contains URLs (uniform resource locators) of interesting documents as shown in Fig. 4.

It is important issue which source choose because the better representation of interesting request traffic we have the more applicable are the results of web cache configuration

[2]Software and documentation are available at http://www.squid-cache.org/

```
.....
http://hel.cee.hw.ac.uk/Vortices/pages/QueryForm.html
http://www8.org/w8-papers/3a-search-query/semantic/semantic.html
http://sun3.ms.mff.cuni.cz/%7Edingle/webcoherence.html
http://www.arrowpoint.com/solutions/white_papers/renaissance.html
http://www.mnot.net/cache_docs/
http://www.eecs.harvard.edu/%7Evino/web/usenix.196/
http://www.surfnet.nl/innovatie/desire1/deliver/WP4/sumreport.html
http://www.imimic.com/WPfeatures.html
http://www.ariadne.ac.uk/issue21/web-cache/
http://mows.rz.uni-mannheim.de/mows/pub/paper/www6.html
http://www.scope.gmd.de/info/www6/technical/paper151/paper151.html
.....
```

***Fig. 4.*** Request stream representation sample.

analysis for our network. The natural choice is the third source: logs of the web cache which performance we want to improve. But there are situations when other sources are also valuable. For example if we try to predict some requests stream that can be specially troublesome for our network in the future we can use other sources where such traffic patters already occurred.

The use of each source has its advantages and disadvantages. Apart from requests simulating software the data obtained from presented sources needs some processing. This should be done very carefully since high quality of requests stream representation is crucial for further analysis.

Collecting requests data from web browser caches of individual users can be troublesome since such caches are unpredictably cleaned and their content cannot usually be obtained without permission.

Logs of WWW servers normally represent wider range of network users than browser caches that we can use. But this data is limited to only one place within network and access to such logs is often restricted.

On the contrary to those two sources web cache logs contains data with very adequate characteristic. Such stream contains requests from many users workstations directed to many servers with original content. But acquireing access to web cache logs with specially interesting requests patterns from outside our organization may also be difficult.

There are several problems with pre-processing log's data. For example often the result (stored in cache) of fulfilling a document request is not only the desired document but also some additional objects like accompanying graphics. Of course such objects does not belong to the original requests stream and should be removed from examined representation. Otherwise these objects are reported as cache hits and thus detoriate cache performance outcomes. There are some web sites that – when accessed – cause generation of new requests by browser (usually new browser windows appear) that are also stored in cache log. Very often these automatically generated requests refer to web pages that exist very shortly. In order to make results of web cache performance measurement repeatable such requests should be removed as well. Dynamically generated content of a web page also cause problems. Accessing such page results in different records for the same request for different times of access. This problem can be solved either by removing such requests from the request stream being prepared for examination or by proper counting different records for such request while computing performance measures.

Another important issue are documents specially marked to prevent them from caching because of the reasons presented in the previous chapter. Since such objects affect web cache performance they should be appropriately treated during the preparatory stage or while computing performance measures. It is worth to remove uncachable documents from the examined representation when speed of running experiments is important.

In order to address some of these problems a concept of two-phase measurement of web cache performance has been developed. This concept is more precisely presented in the next part of this chapter.

The disadvantage of simulated request stream is that such representation can be "too artificial" comparing to representations based on real traces, like computer logs. On the other hand simulation is more flexible. There is software[3] for simulating different request stream patterns. Such stream is generated according to given distribution of requests and number and characteristics of "virtual" clients and servers. The content of the servers are profiled. The space needed for requests storage is also reduced because instead of "real" some simplified request are used (they refer not to URLs but to some short identifiers). Thus processing such streams is also faster.

### *3.3. Proxy cache performance measurement*

Metrics used for measuring web cache performance and factors influencing it have been introduced in the previous

[3]Web Polygraph software and documentation is available at http://polygraph.ircache.net

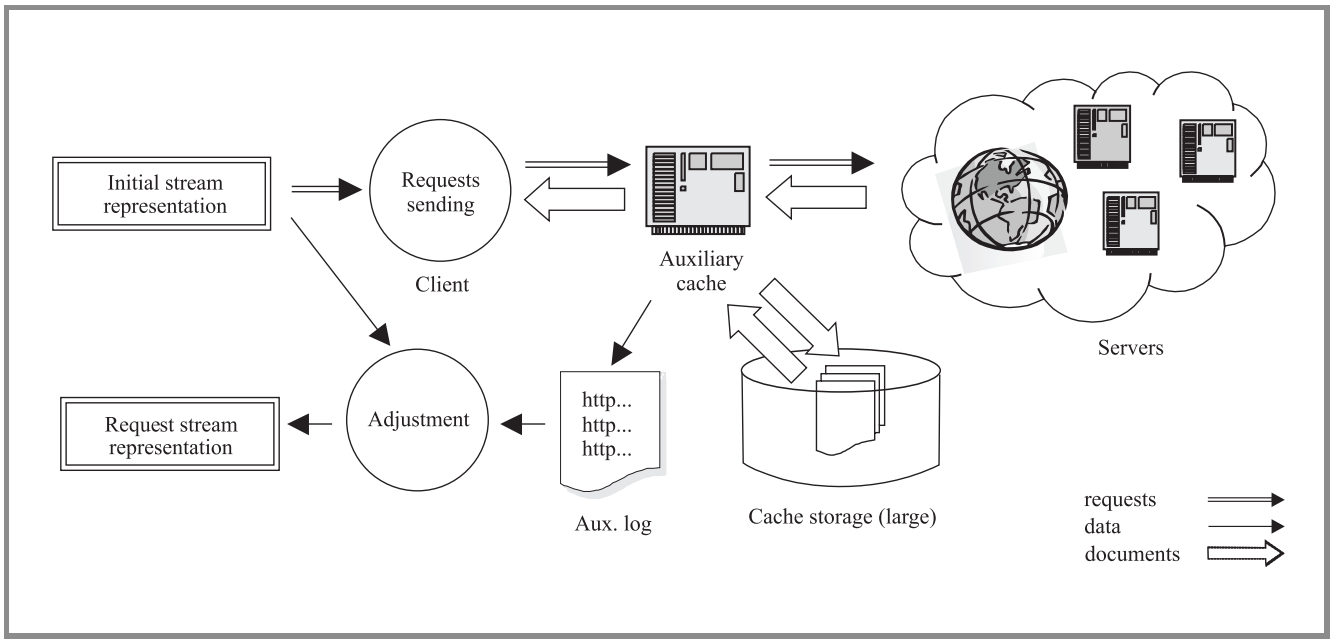


*Fig. 5.* Measurement phase 1 – preliminary data collection with use of auxiliary proxy cache.

chapter. The tool presented in this article has been designed to have capability to compute following measures:

- document hit ratio;
- byte hit ratio;
- average response time.

Computing the last metric has not yet been implemented. There is number of possible other metrics that can be derived from these basic ones.
Important issue connected with examining web cache performance is reducing the influence of unstability of external network – like Internet – for the results. This must be accomplished in order to make the experiments repeatable. It is crucial for valuable analyses to ensure that response times for the same requests are equal and web cache parameters values do not differ significantly. The design of experiments should also guarantee availability of responses for consecutive runs. In order to accomplish this aim web cache is examined in two phases using two joined web caches:

1) preliminary phase (connected with the second stage of the process);
2) examination phase.

### 3.3.1. Phase 1 – preliminary data collection

During this phase auxiliary web cache is used (Fig. 5). The requests from the examined stream representation are sent to this cache. In response auxiliary cache tries to retrieve requested documents from external network and deliver them to requester. Its storage space is large enough to store all the documents requested and its configuration is set up so that no documents are sweeped out because of their features like size. This phase is finished when all requests has been processed and auxiliary cache storage space is filled with all obtainable objects. To make results of the experiment more insensitive to external network failures this phase should be repeated when any undesirable network event occur. Log of the auxiliary cache contains values of response times for sent requests that can be further used for computing performance metrics. This phase is coupled with the data preparation stage. Requests stream representation is adjusted depending on the outcomes of this phase by removing some requests from the initial representation. Thus results of the next phase are not detoriated and the processing time is shorter.

### 3.3.2. Phase 2 – proxy cache performance examination

During this phase adjusted requests stream representation is used (Fig. 6). This time requests are sent to the actual web cache assigned for examination. This cache is connected to auxiliary cache so that requests sent to examined cache are fulfilled by retrieving documents from the auxiliary cache. Refering to external network is not needed.
In the course of experiment number of configuration alternatives are tested, accordingly to the plan saved in a special file introduced before. In each iteration one alternative is examined and performance data is stored in another log of actual web cache. Examining cache performance requires

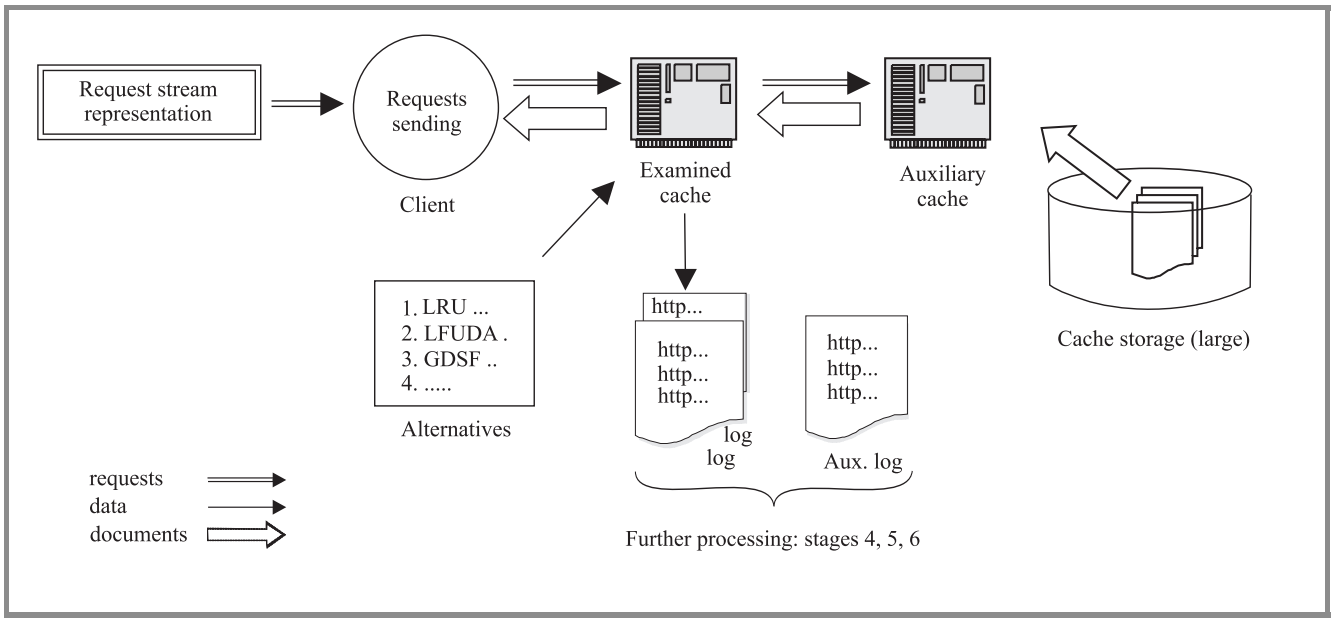


***Fig. 6.*** Measurement phase 2 – proxy cache performance examination.

several technical tasks (they are automated by UNIX shell scripts):

1) resetting examined web cache to some initial state;

2) retrieving next set of parameters and setting them for examined web cache;

3) starting program sending request from examined representation to web cache;

4) log recycling.

### *3.4. Extraction of output variables*

During this step desired performance measures are computed using the output data extracted from the examined and auxiliary web cache logs. This metrics are saved in a file where one row represents one configuration alternative and values of measures are kept in columns.
For the presented research purposes only three metrics were considered but number of other are possible to define depending on analytical needs. Thus number of criteria applied during multicriteria model analysis can be increased by computing new metrics here. In order to fulfill this task UNIX shell script has been created and used.

### *3.5. Generation of mathematical model*

The problem of choosing the best one from a set of discrete alternative can be represented by the model shown in Table 1.
Because during the multicriteria analysis step criteria are chosen from the set of output variables condition $i <= m$, must be satisfied, where $i$ is the number of criteria. Following equations complete formulating of the mathematical model for this problem is shown in Table 2.

Table 1
Discrete alternatives choice model

| Alternatives | Output variables | | | |
|---|---|---|---|---|
| | $y_1$ | $y_2$ | ... | $y_m$ |
| $z_1$ | $a_{11}$ | $a_{12}$ | ... | $a_{1m}$ |
| $z_2$ | $a_{21}$ | $a_{22}$ | ... | $a_{2m}$ |
| ... | ... | ... | ... | ... |
| $z_n$ | $a_{n1}$ | $a_{n2}$ | ... | $a_{nm}$ |
| Explanations: $y$ – represents output variable (measure, metric); $z$ – represents decision alternative (one configuration); $m$ – is number of output variables; $n$ – is number of decision alternatives; $a_{11}, ..., a_{nm}$ – are output variables values. | | | | |

This mathematical model is the basis for generating "core" model – some representation of regarded problem expressed in a mathematical programming languge. Such model comprises all physical and logical relations between variables describing the problem. The core model defines implicitly a set of feasible solutions but it does not include information about decision maker preferences.
The core model generator has been developed as a computer program written in C++ language[4]. This program has been tested for use on Sun Solaris platform. The output of this program is a file in LP-DIT format used by the MOMIP solver.

[4]This software has been created with cooperation with dr Marek Makowski.

Table 2

| Bounds | Initial bounds | Criteria |
|---|---|---|
| $z_1 + z_2 + \ldots + z_n = 1$ | $y_1 = a_{11}z_1 + a_{21}z_2 + \ldots + a_{n1}z_n$ | $y_1 = a_{11}z_1 + a_{21}z_2 + \ldots + a_{n1}z_n$ |
| $z_j \ni \{0, 1\}$ | $y_2 = a_{12}z_1 + a_{22}z_2 + \ldots + a_{n2}z_n$ | $y_2 = a_{12}z_1 + a_{22}z_2 + \ldots + a_{n2}z_n$ |
| $j = 1, \ldots, n$ | ..... | ..... |
| $n = m + i$ | $y_m = a_{1m}z_1 + a_{2m}z_2 + \ldots + a_{nm}z_n$ | $y_i = a_{1i}z_1 + a_{2i}z_2 + \ldots + a_{ni}z_n$ |

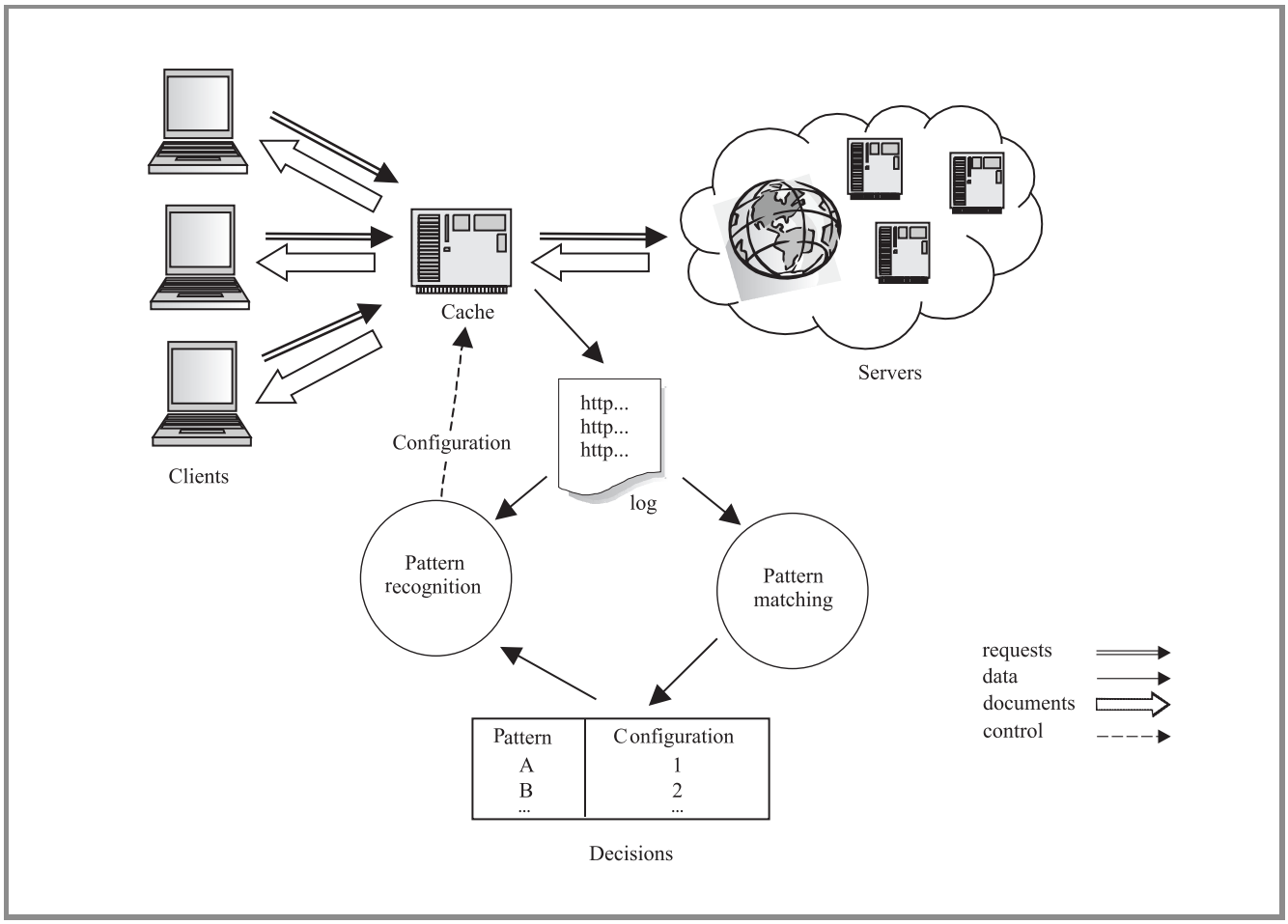


*Fig. 7.* Outline of the automated system for proxy cache configuration.

### *3.6. Multicriteria analysis of decision problem*

The last stage of the process of analysis of decision alternatives for proxy cache configuration is multicriteria model analysis. For this purpose ISAAP modular tool is used. This software implements aspiration-reservation based decision support (ARBDS) approach. This methodology proved to be very successful and has found applications in many areas [2]. It is based on interactive analysis of a set of efficient solutions[5] of mathematical model representing multicriteria problem. Each iteration of the analysis is composed of specification of user preferences with regard to interesting criteria and computing corresponding solution for the problem. This procedure lasts until the most satisfying result for decision maker is found.

Since evaluating web cache performance usually demands several criteria this methodology seems very appropriate for this purpose. Results of simple experiment presented in the following chapter appear to confirm this choice.

ARBDS method can be summarized as a three-stage approach:

- specification and generation of a core model;
- preparatory stage;
- interactive procedure of analyzing efficient solutions.

The organization of process of analysis of web cache configuration presented in this paper is that the first stage of

[5]For efficient solutions no criterion can be improved without degrading a value of at least one other criterion.

ARBDS is handled by the program presented in the previous part of this chapter and the other two are managed by the ISAAP tool. Profound description of ISAAP functionality is beyond the scope of this paper and can be found in [2].
During the preparatory step of the method decision maker choses from the set of model variables criteria used for further problem analysis. For each criterion its type has to be specified:

- maximized – attaining maximum value for criterion is desired;
- minimized – attaining minimum value for criterion is desired;
- goal (stabilized) – achieving given goal (target) value of criterion is desired.

After some computations carried out by solver so-called **compromise solution** is found which corresponds to decision maker preferences set to outmost values (in case of MIP problem).
Aim of last step of ARBDS is to help decision maker find efficient solution corresponding the best to his/her preferences. During interactive procedure decision maker specifies his/her preferences and obtains another computed solution. Specification of preferences for each criterion is based on (see Fig. 12 for illustration):

- aspiration level – this value of criterion decision maker wants to achive;
- reservation level – this value of criterion decision maker wants to avoid.

This procedure is continued until decision maker is satisfied with solution or stops analysis.

### *3.7. Automated system for proxy cache configuration*

In the future analytical environment for web cache management presented here can advance to automated system supporting proxy managers in improving configuration cache in response to changing requests stream patterns. Such system should incorporate, apart from the components presented above, also some software responsible for recognizing requests stream patterns and applying corresponding set of parameters stored in a repository to the managed proxy cache.
The role of the decision support tool should be updating repository of traffic patterns matched with cache configurations based on the analytical process described in this chapter. This idea is outlined in Fig. 7.

## 4. Simple experiment

Experiment presented below is very simple and any important conclusion concerning examined requests stream representation or web cache configuration could not be drawn. The aim of this experiment was to demonstrate application of presented approach to some not paper-based example.

### *4.1. Configuration alternatives*

Three web cache parameters were used during this experiment (see chapter 2.3 for details):

- document replacement policy;
- **cache swap high** threshold;
- **cache swap low** threshold.

As shown in Fig. 3 three values of the first parameter were used during the test: `LRU`, `LFUDA` and `GDSF`. For second parameter also three values were applied: level of 80, 85 and 90 percent. Two values of the last used parameter were 95 and 100 percent. Altogether it makes up 18 combinations-alternatives of web cache configuration.

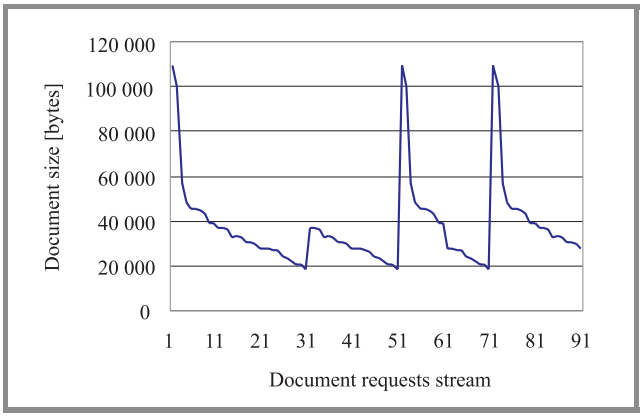


***Fig. 8.*** Illustration of examined requests stream.

### *4.2. Requests stream representation*

Representaion used in the experiment was prepared manually. Figure 8 shows characteristic of examined requests stream representation. This representation contains 100 requests for html documents concerning web cache issues. Document sizes vary from 20 to over 100 kilobytes. These documents were carefully selected from the larger set in order not to detoriate web cache performance outcomes. Only 30 of them were unique.

### *4.3. Experiment run*

Firstly, prepared representation has been sent using WGET software to auxiliary web cache. Since requests for troublesome documents has been removed from the representation beforehand and auxiliary cache storage space was sufficiently large all the documents were stored in this cache. This operations were performed manually.
Secondly, web cache assigned for examination was connected to the auxiliary cache. UNIX shell script was run

Table 3
Performance results – efficient solutions are bolded

| Alternative number | Parameters | | | **Measures** | |
|---|---|---|---|---|---|
| | replacement policy | swap lower threshold | swap upper threshold | **document hit ratio [%]** | **byte hit ratio [%]** |
| 1 | LRU | 80 | 95 | 47 | 43 |
| 2 | LFUDA | 80 | 95 | 48 | 46 |
| 3 | GDSF | 80 | 95 | 46 | 35 |
| 4 | LRU | 85 | 95 | 46 | 43 |
| 5 | LFUDA | 85 | 95 | 44 | 36 |
| 6 | GDSF | 85 | 95 | 46 | 35 |
| **7** | **LRU** | **90** | **95** | **50** | **46** |
| 8 | LFUDA | 90 | 95 | 41 | 34 |
| **9** | **GDSF** | **90** | **95** | **53** | **43** |
| 10 | LRU | 80 | 100 | 48 | 46 |
| 11 | LFUDA | 80 | 100 | 30 | 34 |
| 12 | GDSF | 80 | 100 | 46 | 35 |
| 13 | LRU | 85 | 100 | 47 | 44 |
| 14 | LFUDA | 85 | 100 | 47 | 43 |
| 15 | GDSF | 85 | 100 | 46 | 35 |
| 16 | LRU | 90 | 100 | 46 | 42 |
| 17 | LFUDA | 90 | 100 | 44 | 37 |
| 18 | GDSF | 90 | 100 | 52 | 42 |

that automated tasks presented in the previous chapter. Requests from prepared stream representation were fed into examined cache and performance data was collected in log for each configuration alternative accordingly to the sequence stored in experiment plan file.

### *4.4. Performance results*

After required data has been collected in cache logs values of two metrics were computed for each configuration alternative: document hit ratio and byte hit ratio. Results are shown in Table 3. Efficient solutions (alternatives 7 and 9) are bolded and further analysis concentrates on them. The table shows that for the given representation best results were achieved when applied document replacement politics was either LRU or GDSF. In both cases the values of upper and lower threshold for document sweeping were the same. These were also default values.

Figure 9 shows decision alternatives presented in criteria space. Number of points is less than number of alternatives since some alternatives correspond to same performance results. Black circles represent configuration alternatives for which solutions are not efficient. Efficient solutions are shown as rhombuses and these are most interesting to decision maker.

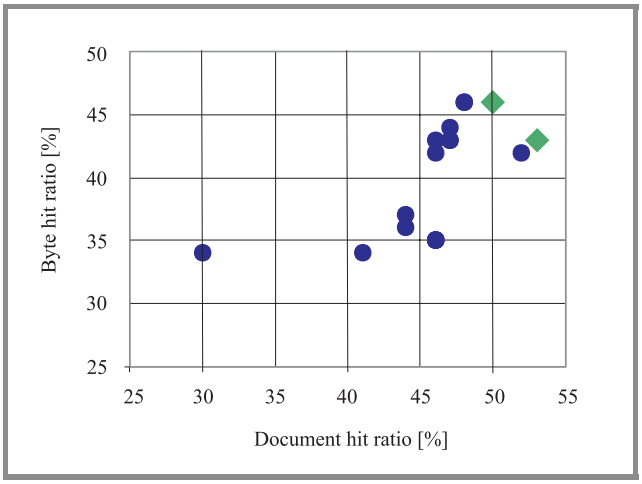


***Fig. 9.*** Decision alternatives presented in criteria space.

### *4.5. Multicriteria analysis*

Model generated using the software presented in previous chapter contained two metrics and 18 alternatives. These metrics were used as criteria in multicriteria analysis stage. It is natural to demand that document hit ratio and byte hit ratio were maximized. Therefore this type was selected for both criteria as shown in Fig. 10. Criteria names were chosen as doc_hits and byte_hits correspondingly.

First solution found was connected with alternative number 7. Corresponding criterion values are represented as white circles in Fig. 11. Decision maker preferences are set to criteria outmost values.

*Fig. 10.* Criteria type specification.

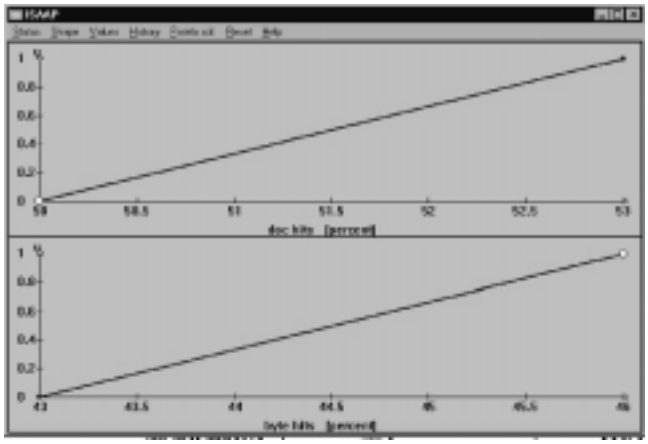

*Fig. 11.* Neutral solution (preferences set to criteria outmost values).

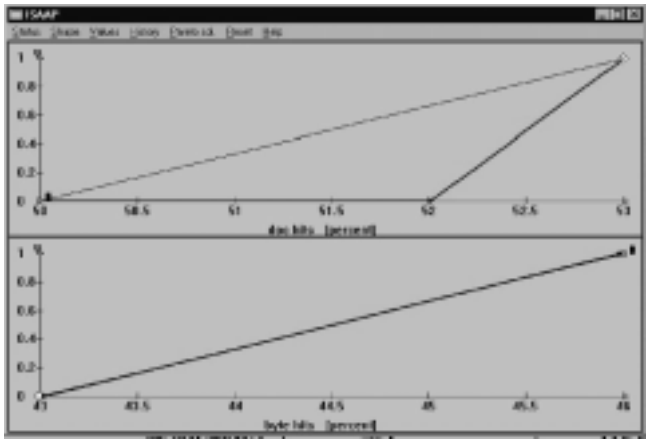

*Fig. 12.* Another solution – after preferences for the first criterion has changed.

In the next step preferences corresponding to doc_hits criterion were changed. Reservation level was set to 52%. Another efficient solution was computed. This time solution pertained to configuration alternative number 9. Value of the first criterion improved but at the expense of the value of second criterion (Fig. 12).
Because considered problem was much simplified there was no use to continue multicriteria analysis further. In this particular case it is hard to find justification for choosing one of these alternatives. Nevertheless in real situation when not miniature but significant requests stream representation and more criteria are involved, such analysis after several iterations may bring important results. Not only in terms of improved performance of web cache but also it could lead to better understanding by decision maker specific web cache behaviour and requests stream characteristic.

## 5. Summary

It has been shown that analysis of web cache performance can be presented as an analytical process. During this process several tasks have to be accomplished in order to transform input data to ready-to-analysis model. The concept introduced in this paper shows that this process can be backed by decision support tool helping in tuning web cache parameters.
Although results reported here are preliminary it seems that applying multicriteria model analysis in the field of web cache management can be fruitful. In order to advance presented tool to more mature status consultation of web cache expert is needed. Especially preparing requests stream representation and multicriteria analysis stages demand experienced user to achieve desired quality of such representation and to correctly interpret results of analysis.
The tool introduced here can be useful for persons dealing with web cache management in several ways. Firstly, it enables analysis of many configuration alternatives while number of interesting criteria are taken into account. Secondly, it can be used as a tool for testing cache functionality. This area is under fast development and new parameters are continously implemented. Their influence on web cache performance deserves scrutiny in specific network environment. Thirdly, when integrated with some additional software, it can be applied as an automated system for proxy cache configuration presented in chapter 3.

## Acknowledgements

Work presented here was initiated during 3-month Young Scientists Summer Program organized by International Institute of Applied System Analysis in Laxenburg (Austria) in the year 2000. Author would like to thank dr Marek Makowski, dr Janusz Granat and Jerzy Sobczyk for their invaluable cooperation.

## References

[1] J. Dilley, R. Friedrich, T. Jin, and J. Rolia, "Web server performance measuerement and modeling techniques", *Perform. Eval.*, vol. 33, pp. 5–26, 1998.

[2] J. Granat and M. Makowski, "ISAAP – interactive specification and analysis of preferences", *Interim Report*, IR-98-052, Nov., 1998.

**Jarosław Pietrzykowski**
e-mail: J.Pietrzykowski
National Institute of Telecommunications
Szachowa st 1, 04-894 Warsaw, Poland