Analyzing the Resource Requirements of Usage Statistics Gathering on Online Newspapers

Gergely Kocsis* and Péter Ekler**
* *Origo Zrt, Budapest, Hungary*
** *Budapest University of Technology and Economics, Department of Automation and Applied Informatics, Budapest, Hungary*
kocsis.gergo@gmail.com, peter.ekler@aut.bme.hu

***Abstract*—Web analytics is a process through which website usage statistics and user behavior data are gathered. Such an analytics program can be used as a tool to get anonym information about users, for example who they are, from which site are they are coming, how often they are visiting the website, how much time do they spent on a site, etc. In this paper we focus on the performance of large websites from the perspective of web analytics and the required extra storage space.**

The number of page views can be millions per day on large websites, thus the stored user-data number (the amount of analytics data generated by a user visit event) is millions per day as well. Poor system design, even related to gathering user behavior statistics, can lead to system crash. Increasing the performance of the hardware exponentially is not possible in the practice, therefore only optimal data processing can lead to efficient system operation.

In this paper we show how to calculate the number of user-data records generated in the database per day and total from system startup in general. Based on that we propose a model to calculate the cost of article high score list which is usually generated at least daily in such systems. Finally, the affect of different search engines are examined.

Keywords: Web statistics, Web analysis, system and database design, Web search engines, high score lists

I. INTRODUCTION

Nowadays the Internet has an important part in our life, millions of people use it every day. The Internet is also a huge market for millions of small and large companies. If an e-commerce website wants to be successful on the market, it is necessary to know the behaviour of users. Figure 1 illustrates the domestic Internet access number from 2003 to 2011 [1] in Hungary Trade traffic of Hungarian online web shops were more than hundred billion HUF (Hungarian forint) in 2010. Figure 2 shows the trade traffic of online web shops in Hungary from 2001 to 2010 [2].

Web analytics applications are used to gather all available user information. These applications are able to indentify and track the behaviours of users and many times they serve different statistics (Top search engines, referring URLs, search keywords, etc.). There are many free web analyzer program like Google Analytics [3], Mystat [4], Addfreestat [5], but none of them is likely to meet all requirements (avoidance of sensitive data, general services, lack of integration with own systems). Therefore in many case companies are forced to develop a unique system.

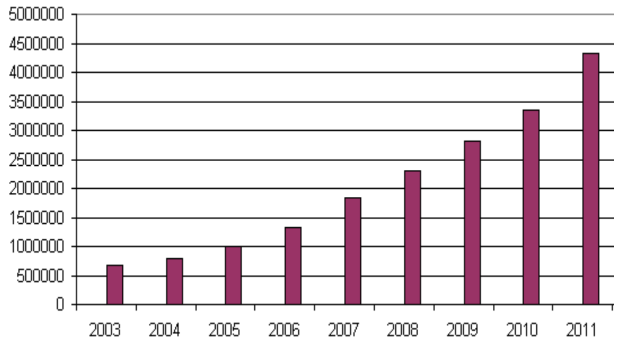

Figure 1. Internet access number 2003-2011

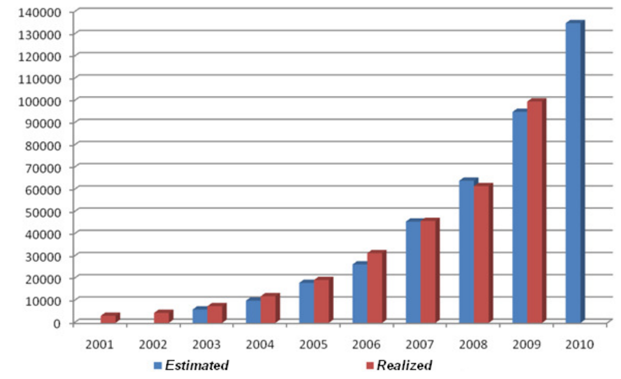

Figure 2. Online web shops trade traffic 2001-2010 (million HUF)

Basically, there are two main type of website from the perspective of usage analysis. One of them is the so-called „newspaper type" website where articles stay in the database in perpetuity. The system never deletes the users-data. The other type is called „advertising type" website where the system deletes expired contents and all related user-data. In this paper we are considering "newspaper type" websites, where the content stays in the system.

The rest of the paper is organized as follows. Section II discusses related work in the field of web site analysis and web statistics. In Section III the data structure of a Web analyzer application is defined and the base model is introduced, which is used later as the basis to the further models. In Section IV the base model is developed further to consider real-world effects. In Section V a model is proposed to calculate the cost of the article high score list. Finally, in Section VI we deal with aggregation effect on performance aspect. Section VII summarizes the paper and proposes future research objectives.

II. RELATED WORK

In [6] the authors present a web usage mining method which combines web usage mining and statistical analysis considering client side data. The method helps to reconstruct user session exactly as it has been and based on these data, it is possible to find web usage patterns with more accuracy.

On a typical website, most of the information is hidden behind the query interface and usually these sites are not indexed by the Search Engines. Another paper [7] discusses the various reasons due of which they are not indexed by the search engines and the possible solutions for these reasons.

In [8] the authors present results of an extensive long-term click-stream study of Web browser usage, focusing on character and challenges of page revisitation. Individual navigation strategies differ dramatically and are strongly influenced by personal habits and type of site visited. Based on user action logs and interviews, they distinguished short-term revisits (backtrack or undo) from medium-term (re-utilize or observe) and long-term revisits (rediscover). They have analyzed current problems and provided suggestions for improving support for different revisitation types.

Efficient management of large databases is different for each database management system, each has its own "soul" and each has its own "Query Execution Plan". In case of large website the user-generated data is over billions over years whose size can be several terabyte. So the inner works of different database management system must be considered in any case. The key difference between previous works and our work is that this paper mainly focuses on the data generated by applying usage statistics gathering mechanism. The MySQL database manager is one of the most widely used database software which is available for free. According to official MySQL documents the performance of query execution depends on the number of disk seeks, the amount of data and data can be found in the cache or not. Cache behaviour has the greatest effect on the performance of the three factors [9], [10], therefore a key priority that the data fit into the cache on.

III. BASE MODEL

In order to analyze website statistics we have implemented an anonym data collecting module that is running on a large "newspaper type" website. The component collects only the most important user-data, which is be sufficient for research. Let's assume that a company has multiple domain names and each domain belongs to the same web analyzer application. In addition, the page structure of a website can be divided further. For example, online newspaper articles can be categorized as domestic, sports, and abroad. Considering the above line of thought, the stored user-data fields in database are as follows:

- page unique identifier,
- date of visit,
- domain,
- category,
- referrer URL,
- user unique identifier,
- user agent data.

Following a model is proposed to calculate the number of user-data records in database per day and total in the above described environment. This means that during every page visit a user-data record is generated in the statistics table. With the help of this model it becomes easier to scale database and "newspaper type" website as well.

In a "newspaper type" website the articles will never be erased from the system, they are available at any time. T_{maxday} denotes the number of days as long the system is in operation. The system stores also user-data for T_{maxday} days. The average number of new article per day is ART_{new}. The average number of the article page views on the first day is ART_{vis}. ART_{vis} is a general parameter in any similar system and can be queried easily.

The exponential distribution is often used to model the lifespan in general therefore we will use that in our model to calculate the number of article readings value, to determine how the article "goes away" over time [11]. We suppose that the number of article page views is decreasing over time. Therefore an exponential distribution model can be applied with λ parameter. The number of different articles in the system (ART_{num}):

$$ART_{num} = ART_{new} * T_{\max day} \tag{1}$$

In addition to the number of articles, R_n specifies the number of page views per day for all new articles:

$$R_n = ART_{new} * ART_{vis} \tag{2}$$

We calculate the amount of user-data records, created in the database per day (R_d) and the amount of total number of records (R_a).

At the end of the first day (n=1):

$$R_{d1} = R_{a1} = R_n \tag{3}$$

At the end of the second day:

$$R_{d2} = R_n + R_n \lambda e^{-\lambda}$$

$$R_{a2} = R_{a1} + R_{d2}$$

At the end of the third day:

$$R_{d3} = R_n + R_n \lambda e^{-\lambda} + R_n \lambda e^{-2\lambda}$$

$$R_{a3} = R_{a2} + R_{d3}$$

At the end of the n-th day:

$$R_{dn} = R_n + R_n \lambda e^{-\lambda} + R_n \lambda e^{-2\lambda} + \ldots + R_n \lambda e^{-(n-1)\lambda}$$

$$R_{an} = R_{a(n-1)} + R_{dn}$$

The general formula can be written for n>1:

$$R_d(n) = R_n + R_n * \sum_{n-1}^{x=1} \lambda * e^{-\lambda x} \qquad (4)$$

$$R_a(n) = n * R_n + R_n \sum_{n-1}^{x=1} (n-x) * \lambda * e^{-\lambda x} \qquad (5)$$

If you know the average number of new articles per day (ART_{new}), the average number of article page views on the first day (ART_{vis}) and how it decreases over time (λ), then the user-data generated per day and the total user-data can be estimated on the n-th day. How many is the number of page views ($R_d\,(n)$) and the total number of records in the database ($R_a\,(n)$) on the n-th day.

Example 1. In order to demonstrate the behaviour of the model, suppose that the number of article page views follow an exponential distribution with parameter 0.1. The average number of new articles per day is 10. The average number of article page views on the first day is 100. This will be our base example in the future.

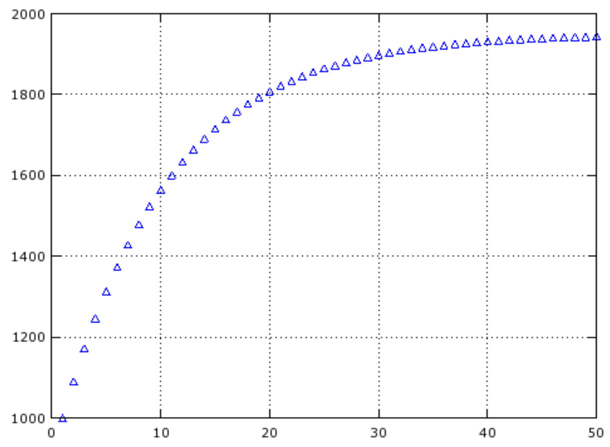

Figure 3. Sum of the number of page views over days (R_d(n))

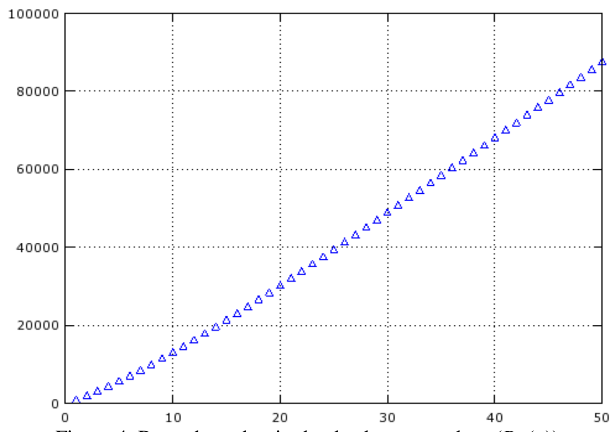

Figure 4. Record number in the database over days (R_a (n))

Figure 3 shows how increases the sum of the number of page views over days ($R_d\,(n)$) from the beginning of system startup. The total number of page views number is growing more slowly, and finally takes a constant value.

Figure 4 shows the total amount of records in the database ($R_a\,(n)$) over days. The linear curve corresponds to the expectations, because every day can be considered as a constant number of page views ($R_d\,(n)$) after initial transient. When this constant is multiplied by the number of measured days (T_{maxday}), we get approximately the number of total records in the database.

IV. DISTORSION OF SEARCH ENGINES

In this section the base model is improved further considering search engines. Let us~~,~~ consider the case if the number of article page views is not decreasing further over time beyond a certain point but also will be a constant (A_c) value. In the real word an article is never lost over time because via different links the various Web search engines periodically index (visit) them. These “non-user” visits will appear in the database as well. By supposing that Web search engines visit an article with the same frequency, then A_c is a constant number and remains the same value over time. Suppose that SE_{num} is the number of Web search engine visit on an article on average T_{se} times per days. The traffic, generated by Web search engines can be calculated with the formula:

$$A_c = SE_{num} * 1 / T_{se} \qquad (6)$$

According to the new model A_c is the smallest number of articles page views that they can reach on any given day. To calculate the new model we can use the former formula (4) and (5), specified in the previous section with a slight modification. Only one condition must be supplemented. In case one day the number of article page views is below the number of A_c (according to (7)), then we use (6) to calculate A_c further. Easy to see that the degree of distortion can be described with the following formula:

$$DT = ART_{vis} * \lambda * e^{-\lambda x} \qquad (7)$$

If value of ART_{vis} is higher (average number of article page views on the first day) or value of λ is lower (slow decline in readership), then the distortion effect of A_c occurs later. We must add the following condition to formula (4) and (5):

$$DT < Ac \rightarrow DT = A_c \qquad (8)$$

However if the A_c=0 then the total number of page views per article is a constants value but if A_c>0 then we can calculate with a linearly increasing value. The slope of the line depends on the value of A_c. In addition to that, the growth of page views is not “real” because of not human visitors.

Solution. In order to avoid distortion and unnecessary record growth, it is necessary to filter incoming requests (discard Web search engines visits) before inserting the visit data in the database. Search engines filtering can be a significant effect on storage of data over years. The filtering can be based on user agent monitoring of incoming requests because major Web search engines use custom user agents (Googlebot, YahooSeeker, etc.).

Example 2. According to (7) the parameters λ and ART_{vis} determine the degree of distortion. Based on the first example, let us calculate two additional cases. In the first case, change the value of parameter ART_{vis} from 100 to 500. Secondly, change the value of parameter λ from 0.1 to 0.01. These three cases are calculated with parameter $A_c=0$ and $A_c=0.4$. The total number of calculations is 6 in example 2.

Figure 5 shows the result of calculations in example 2. In case the value of A_c=0.4 (after initial transient behavior), then in all three cases the result is a straight line defined by a linear, and each line has the same slope (dashed line shown). The middle dashed line belongs to the original example (ART_{vis}=100, λ=0.1). The upper dashed line belongs to the ART_{vis}=500 parameter value, and the lower dashed line belongs to λ=0.01.

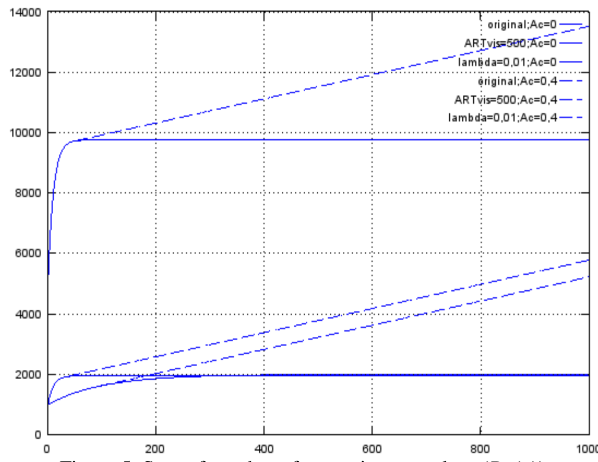

Figure 5. Sum of number of page views per days (R_d (n))

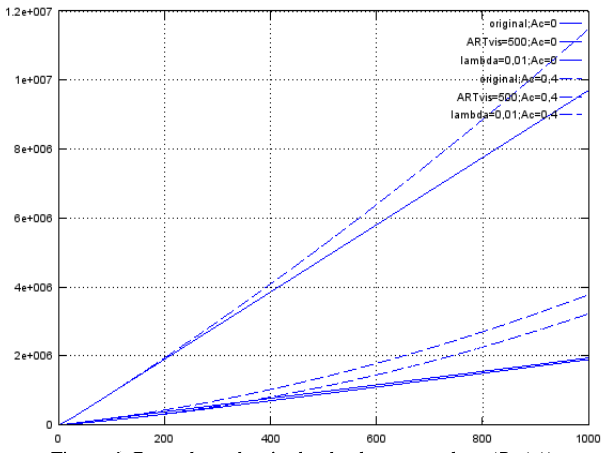

Figure 6. Record number in the database over days (R_a (n))

In case when A_c=0 then there is no difference between the cases of the same ART_{vis} value, but different λ (0.1 and 0.01). Except for the initial transient behavior both move together (lower solid line). In contrast, when A_c=0.4 and calculated with different λ value, the line shift in the vertical axis (lower two dashed lines). ART_{vis} and λ not affect the slope of the line. The parameter ART_{vis} affects only the position of the srtaight line on the vertical axis.

Figure 6 shows the total amount of records in the database (R_a *(n)*) in different cases over the days. In contrast to the former linear growth (solid lines) in all cases the new curves are exponential (dashed lines). Like on the previous figure, the middle dashed line belongs to the original example ($ART_{vis}=100$, $\lambda=0.1$). The upper dashed line belongs to the $ART_{vis}=500$ parameter value, and the lower dashed line belongs to $\lambda=0,01$. If parameter value is $A_c=0$, then there is no difference between the two cases, where ART_{vis} value is the same (lower solid line), but λ parameter is different (0.1 and 0.01). In contrast if $A_c=0.4$, then the higher λ causes slightly rising exponential growth curve.

V. HIGH SCORE LIST

High-score lists are used almost in every “newspaper type” websites to determine which the most popular articles are. In this section a model is presented to calculate the cost of producing such high score lists. Consider the worst case, when calculations of high score lists include the full content of the database. In this situation, all existing article in the database appear in the high score lists in a maximum possible period of time. If the system measures user-data from T_{maxday}, then the number of records in the database is R_a (5) and number of various articles are ART_{num} (1).

Following T_r denotes the average processing time per database record. T_r includes three different operation: retrieving database records (T_{rdata}), searching the number of page views that belong to the article (T_{fnum}) and increasing this number by one (T_{inc}).

$$T_r = T_{rdata} + T_{fnum} + T_{inc} \tag{9}$$

The record retrieval (T_{rdata}) can be considered as a constant-time, because a predetermined memory address will be retrieved all the time. We suppose that the structure of processed data is stored in a hash table and hash table’s load factor is optimal, so hash lookup operations can also be considered as a constant time operation (T_{fnum}). Finally, increasing the value of a variable is also a constant (T_{inc}). The total processing time is (database contains R_a record):

$$T_{db} = T_r * R_a \tag{10}$$

If all page views data is in the hash, then it is only needed to sort them in series. The fundamental theorem of comparative sorts [12] says that there is no asymptotically faster resolution than *N*LOG10(N)*. If a sort operation costs is T_s, then the total time cost of production high score list:

$$T_{all} = T_{db} + T_s * ART_{num} * \log(ART_{num}) \tag{11}$$

The formula shows that if a sorting algorithm is optimal (eg, heap sort), then there is a linear relationship between cost of high scores list production and the size of the database. Moreover, mainly the database time dominates compared to sorting the elements of hash. But in case of a bad sorting algorithm (*N*N*), the situation will be reversed and dominate the running time of the sorting algorithm:

$$T_{all} = T_{db} + T_s * ART_{num} * ART_{num} \qquad (12)$$

Solution 1. To reduce the cost of producing the high score list, the most effective way is to try minimizing the number of processed records in the database. This can be done, for example, by aggregating the data by certain time intervals. For this, it is necessary to summarize the number of page views per article at the end of each day and store this value. Therefore in the future it will not be needed to process all the records, only to retrieve the aggregated data and process it properly.

Solution 2. There is a linear relationship between cost of high scores list production and the average processing time per database record (T_r). Thus, a significant increase in speed can be achieved also by trying to reduce the T_r time parameter. An effective way to minimize the amount of data transfer is trying to process them on their original location. If the data are stored in a database, then the database server might process all data, thus it won't be necessary to transfer them in another application.

Example 3. In order to demonstrate the results, based on the first example, two additional cases are calculated. In the first case, change the value of parameter ART_{new} from 10 to 20 (smallest slope line). In the second, change it from 10 to 100 (largest slope line). Suppose that T_r=T_s=*1* unit of time, A_c=*0* and apply (11) for this situation.

Figure 7 shows the result of how the cost of high score list calculation increases depending on record number of the database (solid lines). The dashed lines indicate T_{db} values, while disregarding the running time of the sorting algorithm. The difference is minimal between T_{all} and T_{db} if the algorithm running time is optimal (*N*LOG10(N)*).

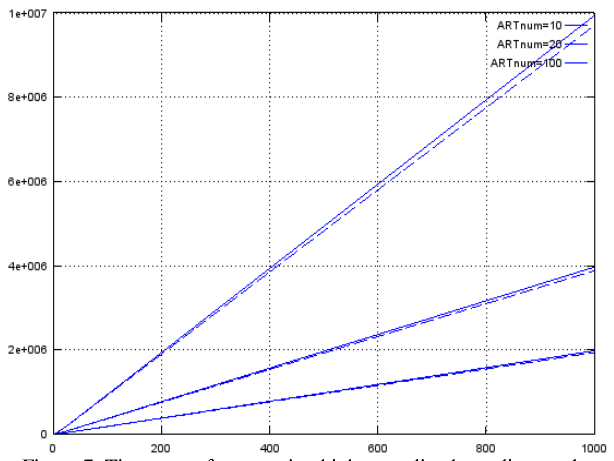

Figure 7. Time cost of preparation high score list depending on the amount of data

VI. DATA AGGREGATION

In the previous section, the preparation time of high scores list was calculated considering the amount of user-data. To reduce the data that is needed to be processed, we propose an aggregation model, where the aggregation is based on the database records per article at the end of each day. Let's examine three different cases. Suppose that in all cases the number of article page views follow an exponential distribution with parameter λ, but the behaviour over time will be treated differently.

1. In the first case (A_c=0), so, the distortion of Web search engines is ignored.
2. In the second case (A_c>0), so the impact of Web search engines are considered in a way, that we count the value of A_c if the number or article page views falls below the average. As the number of aggregate data per article is maximum one per day, therefore if A_c>*1*, then the model should use the A_c=*1* parameter value. For simplicity, if the daily average number of article page views falls below 1, then we count the value of A_c.
3. In the third case, the number of article page views are considered as 0 if the calculated value is smaller than 1.

N is the number of days when the average number of article page views is more than one (ART_{vis}>=*1*) without Web search engines. So the articles of last *N* days generate AG_n record. The articles from *N+1* days generate AG_r amount of record in the database. The sum of two multiplied by the processed days (T_{maxday}) approximates the total number of aggregates. (AG_s).

$$AG_n = N * (N+1)/2 \qquad (12)$$

$$AG_s = (AG_n + AG_r) * T_{\max day} \qquad (13)$$

In the first case (A_c=0):

$$AG_r = ART_{new} ART_{vis} \sum_{x=T \max day}^{x=N+1} \lambda e^{-\lambda x} \qquad (14)$$

In the second case (A_c>0):

$$AG_r = ART_{new} \sum_{x=T \max day}^{x=N+1} (x-N) A_c \qquad (15)$$

In the third case:

$$AG_r = 0 \qquad (16)$$

The difference is negligible between the first and third case. The amount of data is near the same and it is growing linearly over time. In the second case, the amount of data is growing exponentially over time.

Solution. The aggregate amount of data can be minimized if the articles are ignored, which number of page views are one per day or if they are just visited by

Web search engines. These articles should not be taken into the daily aggregation.

Example 4. Based on the first example, let us calculate the above three cases in a 2000 days interval of (Figure 8). The difference is not visible between the first and third case. (bottom line). However in the second case (A_c>0), the curve is clearly an exponential (top line).

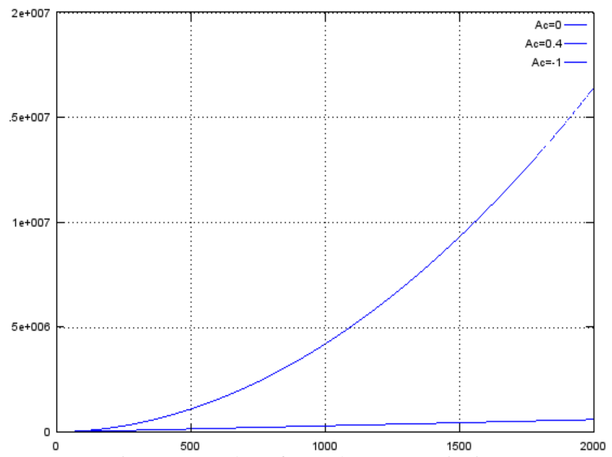

Figure 8. Number of record aggregates in time

VII. SUMMARY AND CONCLUSIONS

In this paper database and website design problems were introduced and some practical solutions were presented through a simple Web analytics situation. Each hypothesis were presented via examples and illustrated with graphs. The models apply parameters and they can be used in other "newspaper type" systems. In this paper we did not presented measurements with real data as they are very sensitive and they have huge market potential, but the proposed models can be applied in similar environments by using the proper parameters.

In section III, after the introduction of the data structure in the demo environment, an ideal model was proposed that was considered as the starting point for future calculations. In Section IV, the base model was developed further to consider the real-world effects, so it can be used in practice much better. In Section V, the cost of high score lists calculation was presented in different cases. Finally in Section VI, a practical solution was described for maintaining effective control over the high score lists production, even for long time as well. Taking everything into consideration, when designing a "newspaper-type" system, it is worthwhile to strive for that the cost of operations on the data remains approximately constant over time. If this is not possible for all operations, then in many case, it is worth to take effort for identifying this situation at the least for the most frequently performed operations. Otherwise, the computational costs will grow exponentially which may lead to the collapse of the system. All system and application is unique in most case, and general solutions must be customized. The best way to customize general solution is to analyze the behaviours of the system in production environment. Future work includes analyzing a large user-data database of a "newspaper type" websites and to develop the model further and verify it with measurements.

REFERENCES

[1] KSH. The number of internet access in Hungary http://www.ksh.hu/docs/hun/xstadat/xstadat_eves/i_oni001.html, Aug. 2012

[2] GKIeNET. There is no crisis in the Hungarian e-commerce http://gkienet.hu/hu/hirek/nincs-valsagban-a-magyarorszagi-e-kereskedelem-%E2%80%93-uj-forgalom-csucs-szuletett-2009-ben/**,** Aug. 2012

[3] Google Analytics. http://www.google.com/analytics/, Aug. 2012

[4] Mystat. http://mystat.hu/, Aug. 2012

[5] Addfreestat.. http://addfreestats.com/, Aug. 2012

[6] M. Heydari, R.A. Helal; K.I. Ghauth, "A graph-based web usage mining method considering client side data", Electrical Engineering and Informatics, 2009. ICEEI '09. International Conference on , vol.01, no., pp.147-153, 5-7 Aug. 2009

[7] B. Ahuja, N. Chaudhary, "Timestamp based Recrawling Technique (TSBCT)", International Journal of Computer Applications 45(22), pp.23-26, Published by Foundation of Computer Science, New York, USA, May. 2012

[8] H. Obendorf, H. Weinreich, E. Herder, M. Mayer, "Web page revisitation revisited: implications of a long-term click-stream study of browser usage", In Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '07), ACM, pp. 597-606. New York, USA, 2007[6] MySQL. http://dev.mysql.com/, Aug. 2012

[9] MySQL. 5.5. Estimating query performance. http://dev.mysql.com/doc/refman/5.5/en/execution-plan-information.html, Aug. 2012

[10] MySQL Performance Blog. Why MySQL could be slow with large tables. June 9, 2006 https://dev.mysql.com/doc/refman/5.5/en/estimating-performance.html, Aug. 2012

[11] Unideb. Definition of exponential distribution. http://www.inf.unideb.hu/valseg/JEGYZET/valseg/node139.htm, Aug. 2012

[12] The fundamental theorem of comparative sorts. http://ranagol.web.elte.hu/Algo/12%20-%20rendezesek_alaptetelei.pdf, Aug. 2012