УДК 004.04, 004.6

V.A. Nikolaiev
VertaMedia Company
(224 West 35th St., Suite 1102-5, New York, NY10001, USA,
basil@vertamedia.com),
O.I. Konashevych, post-graduate
Pukhov Institute for Modeling in Energy Engineering
(15, General Naumov St., Kyiv, 03164, Ukraine,
a.konashevich@gmail.com)

Not only Structured Query Language Method of Ad Request Processing

VertaMedia Company's server provides operation of advertising exchange system between publishers (site's owners), advertisers and intermediaries (SSP[1] and DSP[2] platforms). The objective of the system server is to process a request from a Publisher's site as quickly as possible, choosing the most relevant advertisement campaign, to show it to a site's user. The system works real-time online and as faster it makes accurate choice, the more likely that a user will see an advertisement. The obvious solution was to use relational database to compare the parameters of queries with the parameters and settings of ad campaigns, stored in this database. However, it turned out to be unsuitable as such system showed high latency. VertaMedia™ programmers have found an original way to process data, in which comparison occurred in a flat table using the hash sum and a binary tree for matching ad campaigns and another part of the request, which contained a set of keywords/tags was processed by Sphinx Search as local software solution. A method incorporates the original decision to work with database management systemless non-relational tables and use of specialized software solutions for matching keywords. It showed remarkable results in performance of a resource-intensive process, as described in detail in the article.

Сервер компании VertaMedia обеспечивает работу рекламной системы обмена между издателями (владельцами сайтов), рекламодателей и посредников (SSP[1] и DSP[2]). Сервер системы должен обрабатывать запросы от сайтов издателей так быстро, насколько это возможно, выбирая наиболее подходящую рекламную кампанию, чтобы показать ее пользователю сайта. Система работает в реальном времени в интернете, и чем быстрее она делает точный выбор, тем больше вероятность того, что пользователь увидит рекламу. Очевидное решение состоит в использовании реляционных баз данных (БД) для сравнения параметров запросов с параметрами и настройками рекламных кампаний, которые хранятся в БД. Однако это оказалось недостаточно эффективным — система показала

[1] Supply side platform.
[2] Demand side platform.

высокую латентность. Программистами VertaMedia™ найден оригинальный способ обработки данных, когда сопоставления организованы в плоской таблице с помощью хеш-сумм и бинарного дерева, а также локального программного решения Sphinx Search, которым обрабатываются ключевые слова и метки рекламных кампаний. Метод представляет собой оригинальное решение проблемы работы с нереляционными таблицами без системы управления БД с использованием специализированного программного решения для согласования ключевых слов. Полученные результаты свидетельствуют о значительном увеличении скорости при выполнении ресурсоемких процессов.

K e y w o r d s: not only structured query language (noSQL), statistics, information technologies, big data, statistical process control.

The problem. VertaMedia™ company provides services of online advertising placement. Its main customers are so-called Publishers that manage their websites on the Internet, mobile applications, online games etc (let us call it 'content'). Publishers place advertisement (usually video clips) near their content [1] through specific protocols and algorithms within ad network [2]. Ad network provides interaction with ad operators, advertising owners and intermediaries.

When a user consumes Publisher's content, Publisher's software sends request to VertaMedia™ company. Then the company makes search in its available advertising campaigns database and finds the one or a few that matches best to Publisher's request parameters and provides a command to forward this request to the server of the chosen ad operator/owner. In case of a positive response from the operator's server, ad unit is playbacked to a user while he/she browses the site or application.

The main point on the technical side is query processing speed. A user of a site will not wait until a Publisher chooses ads. The problem is compounded when owners deny requests at their own discretion, including fraud traffic control services that Advertisers use to check requests. Advertisers do not announce most of their parameters of the selection. Therefore, in order not to waste time on requesting all ad campaigns one by one from a list, it is very important to find exact matching within the known parameters.

Body section. Publishers' requests consist of dataset transmitted by a given protocol. A request contains fields with parameters which are required by VertaMedia™ to make matching with the list of ad campaigns, as schematically shown in Fig.1. Generally, an amount of advertising campaigns is more than thousand. The task of the server is to produce the most accurate choice in the shortest possible time with minimal system resources.

Decisions are taken automatically with a given algorithm that has been developed by VertaMedia™, when it faced the fact that the speed of the relational database [3] processing using MySQL [4] was not high enough, and the need for resources was redundant. Therefore, it has been hypothesized to use database management system (DBMS) less approach to search and match data in a non-relational database [5], excluding matching keywords.

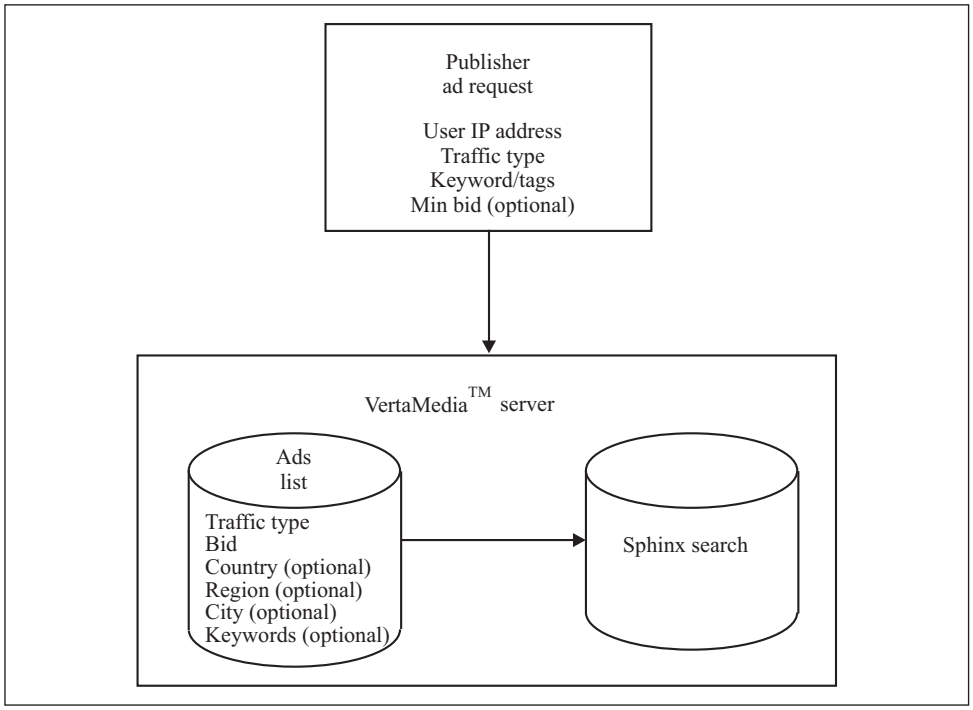

Fig. 1. Publisher request processing

Keywords comparison in various combinations appeared, as well, rather resource-intensive task for such ordinary solution as MySQL and unreasonably complex to create its own solution. It required a customized application. Full-text search engine "Sphinx Search" [6], was found as a the most suitable solution. That will be discussed hereinafter.

Let us consider first of all what solutions turned out unsuitable and why. It is known several types of data models [7]. MySQL belongs to relational database data model [8]. Structured query language (SQL) together with Document-oriented [9], Key-value [10], and Information Retrieval Systems [11] could not provide high performance for VertaMedia™ system tasks, as it is empirically reasoned by experience of company's researchers. Latency was high and hindered reaching the required 0.02-0.90 m per request with reasonable resources. Finally Row-based [12] model was approached with originally developed solution. But firstly, it was tested ready-for-use software application Sphinx Search for processing the entire ad request data.

Sphinx Search appeared unsuitable as a standalone solution for all issues in ad request, still it could work standalone [13]. It was good enough to process full text search. However, it is not required often, because the share of ad campaign with tags/keywords fields is not more than 10 percent among all campaigns. The

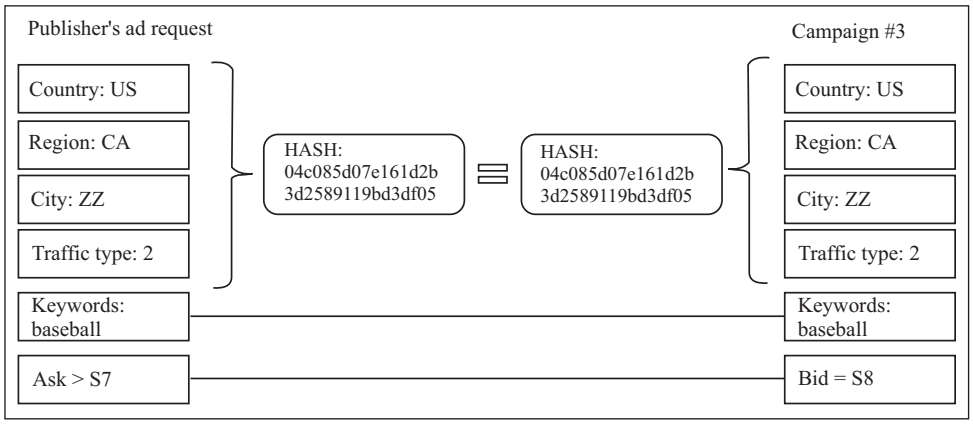

Fig. 2. Request matching

application appeared to be unable to keep the burden of the entire flow of requests at a desired speed.

To resolve the issue the researchers built their own code using Java. Row-oriented flat table was created that included ad campaign list with all possible combinations of parameters initially stored and received from relational database. An example is illustrated in Table, where null values are replaced with ZZ, and the column Campaign ID shows advertising campaigns named by numbers which are repeated in proposed example with different options. In accordance with Table, Campaign 2 contains two requirements with traffic of the second type:

1) show advertising in the United States in any city of California;

2) show advertising in the United States in the UK in any region, in any city.

The above mentioned example shows that each campaign can have multiple options.

Then, HashMap [14] uses composed keys as concatenation of fields: TrafficType, Country, Region and City. After that a tree has been put which consists of Bid and advertisements list:

HashMap [
Hash(TrafficType+Country+Region+City)
↓
TreeMap[bid → List[Ad]
]

Campaign ID	Country	Region	City	Traffic type
1	US	ZZ	ZZ	1
2	US	CA	ZZ	2
2	GB	ZZ	ZZ	2

The set of input parameters of hash function does not contain a campaign number. This approach allows matching hashes in database and ad request eliminated campaign number.

Hash and bid's options allow one to get an advertisement list of relevant matches in two iterations:

1) hashes matching;

2) bids matching. When equal hashes and fee conformity is found, VertaMedia^{TM} server redirects request to the server of Advertiser who is the owner of the best matched ad. If the request has keywords (tags), then after matching appropriate ads, the request is processed by Sphinx Search. Finally application creates a list of best matched ad campaigns as schematically shown in Fig. 2.

Designed and implemented VertaMedia^{TM} solution has been tested and compared to popular alternatives. The testing results are as follows:

```
benchmark
=============== Verta Media ===============
wrk-t4-c400-d180s'http://localhost:9090/handmade?group=260&ip=216.58.214.206&n= 4&min_
bid= 0.002'
Running 3m test @ 'http://localhost9090/handmade?group=260&ip=216.58.214.206&n=4&min_
bid= 0.002
  4 threads and 400 connections
  Thread Stats Avg  Stdev  Max ± Stdev
    Latency 614.23us 1.54ms 189.30ms 92.66%
    Req/Sec 3.24k 1.83k     11.52k    67.52%
  580257 requests in 3.00m, 324.83MB read
  Socket errors: connect 0, read 4844645, write 0, timeout 0
Requests/sec: 3222.88
Transfer/sec:       1.60MB
================= Lucene =================
wrk -t4 -c400 -d 180s 'http://localhost:9090/lucene?group=260&ip=216.58.214.206&n= 4&min_
bid= 0.002'
Running 3m test @ http://localhost:9090/lucene?group=260&ip=216.58.214.206&n=4&min
_bid=0.002
4 threads and 400 connections
  Thread Stats  Avg     Stdev    Max     ± Stdev
    Latency 1.90ms            3.00ms 287.51ms  91.85%
    Req/Sec 1.05k 537.28   2.96k     68.19%
  188082 requests in 3.00m, 105,29MB read
  Socket errors: connect 0, read 4423595, write 0, timeout 0
Requests/sec: 1044.43
Transferee: 593.71KB
```

```
================= Sphinx =================
wrk -t4 -c400 -d180s'http//localhost:9090/sphinx?qroup=260&ip=216.58.214.206&n=4& min
bid=0.002‘
Running 3m test @'http://localhost:9090/sphinx?group=260&ip=216.58.214.206&n=4&min
bid= 0.002
  4 threads and 400 connections
  Thread Stats Avg Stdev Max ± Stdev
    Latency  5.94ms7.53ms 762.50ms 93.55%
    Req/Sec 338.46 117.75 820.00      73.92%
  60274 requests in 3.00m, 33.74MB read
  Socket errors: connect 0, read 4616561, write 0, timeout 0
Requests/sec:  334.68
Transfer/sec:  191.85KB
```

The results of testing showed high performance of the proposed and implemented method compared to Lucene and Sphinx. VertaMedia^{TM} solution is almost ten times faster than Sphinx and three times more efficient than Lucene: 3222.88 to 335.68 and to 1044.43 request per second, respectively. The average latency is reduced to 0.61 ms compared to 1.90 ms (Lucene) and 5.94 (Sphinx).

Conclusion. The method proposed by VetaMedia^{TM} showed better results in system performance, than it was expected [15]. Testing results confirmed that flat table allows reaching low latency which gave advantages to VertaMedia^{TM} in competitions with other ad operators of the market.

There is a HashMap generated on the upper level of the index. The system filters lists with the HashMap by matching the part of parameters. Then bids of matched campaigns and requests are cut with TreeMap. The last iteration is matching keywords with Sphinx which gave the final list of ad campaigns.

Searching via tree is more resource-intensive and slow task, unlike HashMap. SQL solutions practically perform lower results, as it was estimated by researchers of VertaMedia^{TM}. It is 3-10 times more in case of matching with SQL the whole bundle of ad request parameters. The third ‘layer’ of the request processing optimization was to use specialized solution to match keywords and tags contained in ad request by using Sphinx application, which suits better for full text search.

REFERENCES

1. ‘Ad inventory’ definition of the Interactive Advertising Bureau, available at: https://wiki.iab.com/index.php/Ad_inventory (accessed August 1, 2016).
2. ‘Ad network’ definition of the Interactive Advertising Bureau, available at: https://wiki.iab.com/index.php/Ad_network (accessed August 1, 2016).
3. Gray, J. (1981), “The transaction concept: Virtues and limitations”, *Proceedings of the 7th International Conference on Very Large Databases*, “Tandem Computers”, pp. 144-154, available at: http://research.microsoft.com/en-us/um/people/gray/papers/theTransactionConcept.pdf (accessed August 1, 2016).

4. https://www.mysql.com/ (accessed August 1, 2016).
5. http://nosql-database.org/ (accessed August 1, 2016).
6. http://sphinxsearch.com/ (accessed August 1, 2016).
7. Date, C.J. (2003), An introduction to database systems, 8th edition, available at: https://drive.google.com/a/verta.media/folderview?id=0B2Q8Nd2L-6PjZDI0NDk1ODktNGY4ZC00YTBlLWFmZjQtMzg1YzNiOWFlYjlj&ddrp=1# (accessed August 1, 2016).
8. Codd, E.F. (1970), "A relational model of data for large shared data banks", *Communications of the ACM* , Vol. 13, no. 6, pp. 377-387. doi:10.1145/362384.362685.
9. McCreary, D. and Kelly, A. (2013), Making sense of NoSQL: A guide for managers and the rest of us, Manning Publications, Greenwich, Connecticut, USA.
10. Redmond, E. and Wilson, J. (2012), Seven databases in seven weeks: A guide to modern databases and the NoSQL movement, 1st edition, ISBN-13:978-1934356920, ISBN-10: 1934356921.
11. Jansen, B.J. and Rieh, S. (2010), "The seventeen theoretical constructs of information searching and information retrieval", *Journal of the American Society for Information Sciences and Technology*, Vol. 61, no. 8, pp. 1517-1534, available at: https://faculty.ist.psu.edu/jjansen/academic/jansen_theoretical_constructs.pdf (accessed August 1, 2016).
12. http://searchdatamanagement.techtarget.com/definition/columnar-database (accessed August 1, 2016).
13. "AskMonty: About SphinxSE". Monty Program AB, available at: http://kb.askmonty.org/ (accessed August 1, 2016).
14. Konheim, A. (2010), 7. Hashing for storage: Data management. Hashing in computer science: Fifty years of slicing and dicing, Wiley-Interscience, ISBN 9780470344736.
15. Vaish, G. (2013), Getting started with NoSQL, Packt Publishing, ISBN 978-1-84969-498-8.

Received 02.08.16

NIKOLAIEV Vasyl' Anatoliyovych is a Chief Technical Officer, VertaMedia Company, USA, graduated from the National Aviation University, Computer Engineering, 2008. The field of research: systems design, systems performance optimization, high load systems.

KONASHEVYCH Oleksii Ihorovych is a post-graduate student of the Pukhov Institute for Modeling in Energy Engineering of NAS of Ukraine; graduated from the National Aviation University in 2005; in 2011 he graduated from Kyiv National Trade and Economic University, Advanced Training Institute. The field of research: blockchain technology.