УДК 004.75

O. V. Hordiichuk, postgraduate

Location oriented peer-to-peer networks

Taras Shevchenko National University of Kyiv, 83000, Kyiv, Glushkova st., 4d,

e-mail: oleg.gordiychuck@gmail.com

Гордійчук О. В., аспірант

Одноранговi мережі з урахуванням місцезнаходження

Київський національний університет імені Тараса Шевченка, 83000, м. Київ, пр-т. Глушкова 4д,

e-mail: oleg.gordiychuck@gmail.com

Додатки побудовані на основі одноранговых мереж домогають розв'язати проблему доставки інформації багатьом абонентам, шляхом розподілення мережих навантажень серед усіх учасників. Проте із ростом мережі, також зростає і затримка доставки даних, яка виникає внаслідок неоптимального з географічної точки зору розташування вузлів в топології. В цій статті запропонований новий алгоритм пошуку найближчих сусідів, а також топологія, яка дозволяє робити подібні запити, в основі яких лежить принцип пошуку найближчих в мережевому розумінні вузлів за допомогою розподіленої системи мережевих координат. Даний підхід дозволяє більш ефективно використовувати мережеві ресурси, звести до мінімуму надлишкову затримку, яка виникає при передачі даних, а також знизити навантаження мереж інтернет-провайдерів. В свою чергу використання алгоритму побудови топології на основі 7-вимірного простору надає змогу вузлам знайти найближчих сусідів за кількість запитів, яка логарифмічно залежить від кількості вузлів в мережі, що робить систему швидшою, порівняно з існуючими розв'язками.

Ключові слова: одноранговi мережі, пошук найближчих сусідів, мережеві координати.

Applications that use peer-to-peer networks help to solve a data delivery problem to many endpoints by distributing network load among all participants. However with network growing also a delivery data delay increased, which occurs due to unoptimal location in geographical meaning of nodes in a topology. In this paper proposed a new algorithm of the nearest neighbor search as well as a topology that enables these queries that uses a principle of searching the nearest nodes in the network by using a distributed coordinate system. This approach helps to use network resource more efficiently, decrease to a minimum unwanted delay that appears during data dissemination, and at the same time decrease a network load for Internet providers. Also usage of this algorithm, which is based on the 7-dimensional space, makes possible to find nearest neighbors with a count of queries that is equal to a logarithm of the total amount of nodes in the network and makes this system faster compared to existing solutions.

Key Words: peer-to-peer networks, nearest neighbors search, network coordinates.

Статтю представив д.т.н., проф. Погорілий С.Д.

Introduction

Peer-to-peer network is a promising technology for solving data dissemination problems. Its distributed feature makes possible to unload a network bandwidth of a central server. While for file transmission there are a lot of efficient existing solutions like BitTorrent, for video streaming and other latency dependant data this approach is still too far away from meeting quality of experience (QoE) criteria. Main reasons are unpredictable behaviour of users that join and leave the network and a network delay that appears during data transmission. Moreover this approach is not scalable in general, because the bigger size of the network, the bigger quality impact for the end users. With a network growth participants of the network also notice sudden interruptions and a jitter until new data will be available. Another possible side effect is a time lag of the executed query in the network like accessing key-value resource or searching another nodes. In case of real-time peer-to-peer streaming event like football or basketball match, users in adjacent rooms will observe a synchronization

problem. While in one room viewers will see a score, in another room they will wait up to several minutes until this event occur. These reasons make peer-to-peer networks for real-time applications rather proof of concepts then production ready solutions.

In this paper proposed a new distributed location service that has a goal for the decreasing end-user delays, which in a result improves quality of experience. The key idea is to make things completely distributed and thus scalable to networks of any size. At the same time proposed schema significantly decreases query latency by avoiding inefficient transmitting between peers that are located far a way from each other. It is mainly based on an approach of finding nearest neighbours in network meaning that is achieved by using a special topology structure, which enables this query for every node.

Related work

The task of finding nearest neighbours in the network is a known problem in peer-to-peer networks. A solution of this task is hard due to the fact that ISP networks are connected in a different manner with each other and even in case when users located in the same network they can have big value of RTT (round-trip time) compared to peers located in other networks. This situation is typical for cellular networks (2G-4G) as peers located in such network will be farther between each other then with another peer that uses a cable connection. It is also complicated due to a huge amount of possible distance pairs between peers and complexity of its measurement. However the last problem is partially solved by using King [1] tool that helps to evaluate distance between two hosts by using recursive DNS queries that return RTT values. This tool allowed measuring network distances between two hosts without direct querying them. However for answering the nearest neighbours query it still requires an efficient data structure and $n(n-1)$ "king" measurements, where n – is a number of nodes in the network. It means that using this approach for the problem solution requires a central server and a query complexity grows with every new node entrance and departure.

These problems are solved in Meridian [2]. It is a completely distributed system that helps peers to find nearest neighbors in a logarithm of network size amount of queries. Meridian has its own network topology that organizes neighbors into groups at different distances that are exponentially increased. Every group contains peers of a maximum possible observed square calculated every time new peer tries to join it. In the proposed paper we also use an idea of building topology, where every peer has a small set of neighbors at different distances that helps newcomers to find their nearest neighbors. But instead of requesting every subset of neighbors for retrieving distance information, here we use network coordinates that make possible to avoid additional operations and increases speed of the query.

Another approach is to use virtual network coordinate systems. A key idea here is assuming that all peers live in Euclidean space and in result every node has it's own coordinates in such space. There exist two different approaches: land-marking and distributed network coordinates. In the land-marking case every participant of the network measures distances to predefined nodes that are located as farther as possible between each other. Every land-marked node represents separate axis, thus naturally the more predefined nodes the more accurate results could be achieved. This idea is used in GNP [3], Lighthouse [4] and PCoord [5] systems. Mainly after landmarks measurements accomplished every peer applies some optimization method like Simplex Downhill that minimizes possible errors. However these systems require lot of landmarks over the world for the really accurate results, also this solution requires centralized server and thus is not scalable. That is why the distributed network coordinates approach was proposed. Here each node acts as a landmark and is used for coordinates calculations. Perhaps the most known system that implements this idea is Vivaldi [6]. This system consists of a 2-d coordinate system that is enriched by a height parameter that represents a jitter or error that occurs in the network space as the triangle inequality is often violated. The Vivaldi algorithm continuously updates coordinates of every node, however they use only 2-d coordinates system due to inability to decrease network overhead for greater dimensions. The approach proposed in this paper uses 7-d coordinate system that achieves at least same accuracy, but it doesn't require computing the height parameter and has constant network overhead. Also in the Vivaldi the nearest neighbor query is not implemented. Another existing implementation is PIC (Practical Internet Coordinates) [7]. Here every node is used for the network coordinates estimation like in the Vivaldi. But comparing to the previous approach it has an ability to request the system with the nearest neighbor query by using an active node discovery protocol. It combines 3 strategies: random nodes, closest nodes and a hybrid of both for minimizing errors that appears in the Simplex

Downhill method. This idea improves overall accuracy and makes system reliable for both – near and far peers. In the proposed paper we also provide an ability to answer the nearest neighbor query in a completely decentralized manner and for achieving better accuracy we use sets of neighbors at different distances that represent a better sampling than in PIC.

An algorithm for buidling network coordinates

We assume that a network space could be represented as an n-dimensional Euclidian space, where distance function $r(n_1, n_2) = ||n_1 - n_2||$ represents RTT (round-trip time) between a pair of n_1 and n_2 nodes in the swarm. While it's known [6] that a real network space is not the Euclidian one due to a triangle inequality violations, it's still possible to accurately predict network distances with a high probability (evaluation of this method is described in the section below). The network coordinate of node n denoted as a tuple $(x_1, x_2, \dots, x_p)$, where every x_i represents a value of a dimension an p is a cardinality of x. Initially, when the peer joins the network, it assigns to itself a coordinate where all dimensions are equal to 0: $(0, 0, \dots, 0)$ and starts communicating with other nodes. Periodically (every k seconds) every peer requests the neighborhood current coordinates and updates information about RTT using response time. It triggers updating of its own coordinates and rebuilding local searching topology. As only coordinates values need to be sent, a network overhead of this algorithm is constant and configurable via k and p parameters. In this case for a valid operating of the algorithm every peer needs to additionally allocate b bytes for the upload and download bandwidth that may be evaluated as following equation:

$$b = cpb \tag{1}$$

Here b is an amount of bytes required to encode one dimension and c is a cardinality of the neighborhood set. Naturally the more neighbors every peer has and the greater cardinality of coordinates is, the more accurate results may be obtained. However, increasing of these values also increase the network overhead. That is why it's important to choose a good tradeoff. In this paper we use c=10, b=2, p=7 and k=5 that means it's only need 140 bytes of the bandwidth for every 5 seconds to accomplish next stage of the network coordinates updating. These values provide reasonable accuracy and overhead (comparison with other parameters described in the next section).

Whenever peer receives new information about coordinates, it's possible to evaluate current error e that is defined as following equation:

$$e(n_i, n_j) = |r(n_i, n_j) - d(n_i, n_j)|, \quad n_i, n_j \in N, i \neq j. \tag{2}$$

Where N – is a set of all nodes, r function that returns the round-trip time between two nodes and d is a distance function on the defined Euclidean space that calculates distance by using network coordinates. Ideal case when all error values are equal to 0. However, as it was mentioned before, generally it's known to be an impossible task due to the triangle inequality viloations in the Internet network. So in this case network coordinate will always contain error information and therefore a key idea of the algorithm is updating all coordinates by minimizing possible errors. It could be represented as a task of minimizing a square sum of observable errors for every node n_i:

$$\min \sum_{n_j \in N_i} e^2(n_i, n_j), \quad N_i \subseteq N. \tag{3}$$

Where N_i is a set that represents the neighborhood of a peer n_i. So the defined task is a common non-linear least squares problem. It could be solved by using Levenberg-Marquardt algorithm (LMA) [7]. In the input the current network coordinates of every neighbor, their values of RTT and the current node's coordinate are provided. And the algorithm will modify values of the current peer's coordinates to satisfy condition described in (3). In a result network coordinate of the node updated and availiable for requesting by other nodes and possible errors are minimized. Next time, when neighbors of the peer will request new coordinates they also update their ones, which will produce more accurate result as errors are minimized. It means that this algorithm is iterative and at every step accuracy is improved. Also in a case when peer churn appeared or new participants entered the network coordinate topology will be rebuilt in a couple of updates that makes it robust to dynamic changes in the network. These approach enables possibility to use network coordinates for answering the nearest neighbor query for both: server based and distributed implementations. While both of implementations have their advantages and disadvantages, we

recommend to use network coordinates obtained from the described algorithm and use them in the Meridian topology. In this case it's possible to avoid direct measurements for the nearest neighbor query and make things completely in a distributed manner.

Evaluation of the algorithm

For evaluating the algorithm that described in the previous section we used a statistical table data for 168 cities that contains ping measurements for every pair in the table (28056 records totally) [9]. Cities are located on all continents of the Earth and mostly represent captials and big cities. It's not hard to notice that main parameters of the algorithm, which will affect accuracy are number of dimensions and a size of the neighborhood. The more dimensions and the greater size are, the better accuracy and higher performance penalty. While the neighborhood size may vary in different peer-to-peer applications, we assume that in most cases the minimum value equal to 10 neighbors. This value doesn't mean that it must be used in implementations of the proposed method, rather it helps to compare how number of dimensions will impact in a typical case. Every neighbor in every test has chosen randomly, all tests repeated for 10 times and in the end obtained values are averaged. Such approach shows results in different environments and thus eliminates possibility of "well-tuned" effect. Below represented a column chart that compares average error of every dimension type after 20 cycles of the updating coordinates:

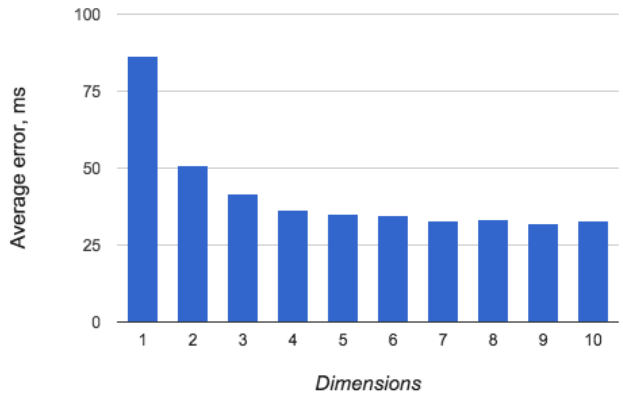

Figure 1. Average error compared to dimensions.

As it's shown in the figure above starting from the 7th dimension overall accuracy doesn't significantly change. On the average it's equal to 33.07 ms for and in the case with 10 dimensions it's equal to 32.86 ms. Therefore it's recommended to use p parameter (which represents amount of dimensions) equal to 7. As in this case it's possible to achieve reasonable accuracy without overloading CPU for applying the Levenberg-Marquardt algorithm. Separately for the case with 7 dimensions presented charts, which show how many (in percentage) nodes are covered with relative and absolute errors:

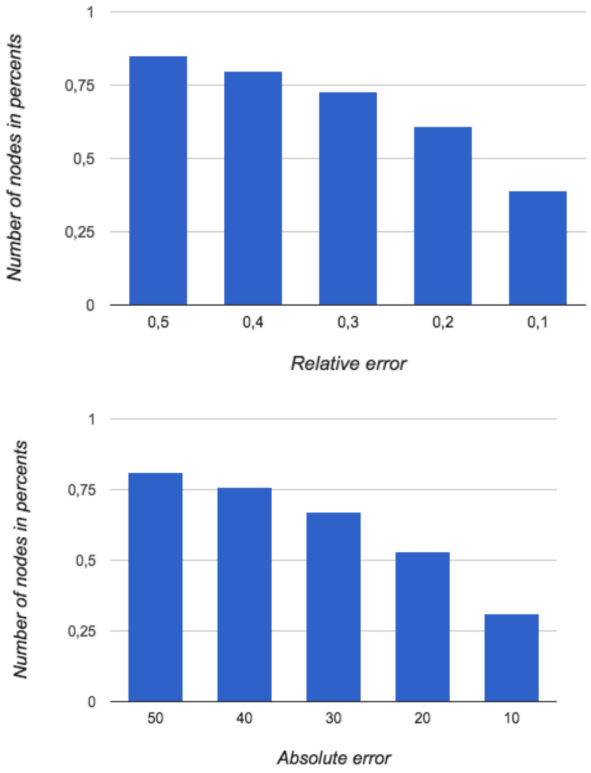

Figure 2. Relative and absolute errors compared to number of nodes.

Moreover these results may be improved if apply following heuristic. It's known that some of the nodes are affected by network jitter that may appear in mobile carrier networks, DSL or due to the network congestion. If assume that these nodes don't have any neighbors closer than 100 ms, than it's possible to create a second layer of network coordinates, where such nodes will be removed. In this case they still can use the first layer, while others use both layers for more precise calculations. In this case the second layer of our dataset contains 130 cities from 168. Applying the proposed algorithm results to an average relative error being equal to 17.12 milliseconds and nearly in 50% of cases peers have relative error less than 10 milliseconds.

Conclusion

In this paper proposed a new algorithm for building network coordinates. It's based on the idea of minimizing visible errors in a 7 dimensional space on the each side of the swarm. It requires a minimum of additional bandwidth allocation that will not be

noticed in most peer-to-peer applications. And its algorithm is simple in implementation and compared to analogues doesn't require to transmit or evaluate additional information except network coordinates. Also the proposed approach executed in a completely distributed manner and may be used in a existing engines for searching nearest neighbors. At the same time it demonstrates very accurate results: mean relative error is equal to 33.07 milliseconds on the proposed dataset. And in a case of using a second additional layer with the reducing heuristic the mean relative error is almost halved and equal to 17.12 milliseconds. Achieved results enable usage of the algorithm in all peer-to-peer locations, specially video-streaming ones, that improves quality of experience of such systems.

Список використаних джерел

1. *Gummadi K.* P. King: Estimating latency between arbitrary internet end hosts / Gummadi K.P., Saroiu S., Gribble S. D. Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment. – ACM, 2002. – P. 5-18.
2. *Wong B.* Meridian: A lightweight network location service without virtual coordinates / Wong B., Slivkins A., Sirer E. G. - ACM SIGCOMM Computer Communication Review. – ACM, 2005. – Vol. 35. – No. 4. – P. 85-96.
3. *Ng T. S. E.* Predicting Internet network distance with coordinates-based approaches / Ng T. S. E., Zhang H. - INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE. – IEEE, 2002. – Vol. 1. – P. 170-179.
4. *Pias M.* Lighthouses for scalable distributed location / Pias M. - Peer-to-Peer Systems II. – Springer Berlin Heidelberg, 2003. – P. 278-291.
5. *Lehman L.* Pcoord: Network position estimation using peer-to-peer measurements / Lehman L., Lerman S. - Network Computing and Applications, 2004. (NCA 2004). Proceedings. Third IEEE International Symposium on. – IEEE, 2004. – P. 15-24.
6. *Dabek F.* Vivaldi: A decentralized network coordinate system / Dabek F. et al. ACM SIGCOMM Computer Communication Review. – ACM, 2004. – Vol. 34. – No. 4. – P. 15-26.
7. *Cox R.* Practical, distributed network coordinates / Cox R. et al. ACM SIGCOMM Computer Communication Review. – 2004. – Vol. 34. – No. 1. – P. 113-118.
8. *Moré J. J.* The Levenberg-Marquardt algorithm: implementation and theory / More J. J. Numerical analysis. – Springer Berlin Heidelberg, 1978. – P. 105-116.
9. Wondernetwork statistical https://wondernetwork.com/pings/

References

1. GUMMADI K. P., SAROIU S., GRIBBLE S. D. (2002) King: Estimating latency between arbitrary internet end hosts. *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*. p. 5-18.
2. WONG B. SLIVKINGS A., SIRER E. G. (2005) Meridian: A lightweight network location service without virtual coordinates. Computer Communication Review 35 (4). p. 85-96.
3. NG T. S. E., ZHANG H. (2002). Predicting Internet network distance with coordinates-based approaches. *Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies 1*. p. 170-179.
4. PIAS M. (2003). Lighthouses for scalable distributed location. *Peer-to-Peer Systems II. – Springer Berlin Heidelberg*. p. 278-291.
5. LEHMAN L., LEHMAN S. (2004) Pcoord: Network position estimation using peer-to-peer measurements. *Network Computing and Applications. Proceedings. Third IEEE International Symposium on. – IEEE, 2004*. – p. 15-24.
6. DABEK F. (2004) Vivaldi: A decentralized network coordinate system. *ACM SIGCOMM Computer Communication Review 34 (4)*. p. 15-26.
7. COX R. (2004) Practical, distributed network coordinates. *ACM SIGCOMM Computer Communication Review 34 (1)*. p. 113-118.
8. MORE J. J. (1978) The Levenberg-Marquardt algorithm: implementation and theory. *Numerical analysis. – Springer Berlin Heidelberg, 1978*. – p. 105-116.
9. WONDERNETWORK STATISTICAL https://wondernetwork.com/pings/

Надійшла до редколегії 28.05.15