Decentralized Approximation Algorithm for Data Placement Problem in Content Delivery Networks

Maciej Drwal, Jerzy Józefczyk

▶ **To cite this version:**

Maciej Drwal, Jerzy Józefczyk. Decentralized Approximation Algorithm for Data Placement Problem in Content Delivery Networks. 3rd Doctoral Conference on Computing, Electrical and Industrial Systems (DoCEIS), Feb 2012, Costa de Caparica, Portugal. pp.85-92, 10.1007/978-3-642-28255-3_10 . hal-01365572

HAL Id: hal-01365572
https://inria.hal.science/hal-01365572

Submitted on 13 Sep 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Decentralized Approximation Algorithm for Data Placement Problem in Content Delivery Networks

Maciej Drwal[1], Jerzy Józefczyk[1]

[1] Institute of Informatics,
Wroclaw University of Technology, Poland
{maciej.drwal, jerzy.jozefczyk}@pwr.wroc.pl

Abstract. Recent advancements in Internet technology research, as well as the widespread of commercial content delivery networks, motivates the need for optimization algorithms designed to work in decentralized manner. In this paper we formulate data placement problem, a special case of universal facility location problem with quadratic terms in objective function. The considered combinatorial optimization problem is NP-hard. A randomized algorithm is presented that approximates the solution within factor $O(\log n)$ in decentralized environment, assuming asynchronous message passing of bounded sizes.

Keywords: computer networks, facility location, randomized rounding.

1 Introduction

Content delivery networks (CDN) are systems used to efficiently distribute the Internet traffic to the users by replicating data objects (media files, applications, database queries, etc.) and caching them at multiple locations in the network. This allows not only to reduce the processing load on the server hardware, but also helps eliminating transmission network congestion. Currently all major content providers entrust their offered services to such systems. The optimal data placement problem is one of the most fundamental theoretical challenges arising from the design of such systems.

The emergence of CDNs has triggered a rebirth of algorithmic research in the locational theory, a long established branch of operations research [1]. One of the new aspects of locational problems, which have not yet been investigated thoroughly, is the decentralized solution approach. Such characteristic is especially important for applications in the area of computer network systems design. In the decentralized environment we assume that input data to the optimization problem is scattered among different network nodes, and it is impossible to collect them at one location in order to apply a traditional centralized algorithm. Similarly, decision variables are bound to different nodes, and feasible solutions are constrained by the interactions between them.

In this paper we investigate the problem of optimal placement of single data object in network, as formalized in Section 3. In Section 4 we give a decentralized approximation algorithm, based on randomized rounding, which decides at which

nodes to cache data object and finds an assignment of users, resulting in a solution of a value bounded by $O(\log n)$ factor of the optimal.

2 Contribution to Value Creation

The research on new Internet technologies is of a priority importance for the information age society. Given the observed growth in the amount of content provided in the Internet and the limitations of existing IP transfer technology, new networking paradigms are sought. One of such novel concepts is the "content-aware" networking [4]. Across the years of Internet history, the stage, at which the main performance bottlenecks in the Web access were located, changed gradually. Initially it was the *last mile* (slow user's access connection, e.g. dial-up modems), which later turned to the *first mile* (insufficient server infrastructure for handling thousands of concurrent requests) in mid 1990s. Nowadays, as these problems have been largely averted, it is the *middle mile* (Internet backbone network and edge router devices) that is considered to be the most performance critical area [7].

As for now, middle mile content delivery methods have already contributed to the generation of value by such Internet enterprises as Akamai Technologies (pioneers in CDN technology research), which provides services to numerous content supplier companies (Amazon, Apple, Facebook, Netflix, Yahoo!, and others). On the top of that, the shift towards the cloud-computing paradigm, observed in recent years, shows the prevalence of content location problems.

The theoretical underpinnings of considered methods would allow to further increase the performance of server systems. This creates the added value by allowing for transfers of much more data in short time without the expensive replacement of the existing telecommunication infrastructure.

3 Problem Formulation

The data placement problem (or cache location problem) is formulated as follows. Given is a set of clients, each described by the demand $w_i \geq 0$, denoting the expected (or exact) number of data access requests issued by client i. Given is a set of cache servers, where the application's content can be stored. Each client or server is located at a vertex of a connected and directed graph with vertex set V, $|V| = n$. A matrix $\boldsymbol{D} = [d_{ij}]$ of nonnegative distances between nodes $i, j \in V$ is given. Each cache server j is characterized by the cost of caching the object $b_j \geq 0$ (e.g. time needed to download it from the origin server and/or time spent on performing updates), as well as the cost of processing a single client's request $h_j \geq 0$. In general, a client and server can coincide at the same node in V.

Two decision variables are used. First one is a binary vector $\mathbf{z} = [z_j]$, j = 1, …, n, which assumes value $z_j = 1$ if server j caches the data object, and $z_j = 0$ otherwise. Second one is a binary n-by-n matrix $\mathbf{x} = [x_{ij}]$, where $x_{ij} = 1$ if client i is assigned to server j for requesting the data.

The goal is to minimize the total cost of placement and assignment decisions, which is expressed as the sum of connection costs, server processing costs and object caching cost. Each node $i \in V$ generates the following fraction of total cost:

$$v_i(\mathbf{x}, z_i) = \sum_j x_{ij} d_{ij} w_i + \sum_j \left(x_{ij} h_j \sum_k x_{kj} w_k \right) + z_i b_i. \quad (1)$$

It is assumed that the more clients access the same server, the higher is the response latency perceived by each of the clients. Since all the clients accessing the same server perceive the same latency, the middle sum in (1) contains a quadratic term in $x_{ij}x_{kj}$. In this paper we consider only the min-sum formulation of the problem, i.e. the objective is to minimize the expression:

$$V(\mathbf{x}, \mathbf{z}) = \sum_i v_i(\mathbf{x}, z_i), \quad (2)$$

subject to the following constraints:

$$\forall i \quad \sum_j x_{ij} = 1, \quad (3)$$

$$\forall i, j \quad x_{ij} \le z_j. \quad (4)$$

The set defined by the constraints (3)-(4) and $x_{ij}, z_j \in \{0,1\}$ is the same as in the well-known uncapacitated facility location problem (UFLP) [5]. In particular, it is assumed that the storage capacities of servers are unlimited (which is reasonable in most practical applications, as the mass storage memory is the cheapest resource among the considered ones). Note that UFLP can be reduced to the considered problem by setting $w_i = 1$ and $h_i = 0$ for all i. Thus the presented problem is NP-hard and at least as hard to approximate as UFLP. In particular, there is no ε-approximation algorithm for the considered problem unless P=NP.

4 Decentralized Rounding Algorithm

In this section the description of a decentralized method of minimizing function (2) in binary variables subject to constraints (3)-(4) is given. It is assumed that each node i in the network controls a row $\mathbf{x}_i$ of the decision matrix $\mathbf{x}$, and the decision variable z_i. The algorithm works in asynchronous communication rounds. In each round of communication any two nodes can exchange up to $O(\log n)$ bits of information. They use two asynchronous operations: *send* and *receive*. The *receive* operation triggers waiting for other nodes to *send* information. Thus it is required to assure that algorithm will not cause a node to be blocked indefinitely. Any node may send information even to all $n-1$ remaining nodes (in a form of broadcast), but the message sizes are bounded proportionally to the network size. For example, it is not possible to send whole vector $\mathbf{D}_i$ to a neighbor, as that requires sending $\Omega(n \log n)$ bits.

The method is based on the filtering scheme given in [8], which was originally developed for approximating *k*-median and uncapacitated facility location problems (the latter is also known as fixed-cost median problem). A decentralized version of similar scheme for UFLP was used in [9]. The basic idea is to solve the linear programming relaxation of the original problem and use the obtained solution to construct a provably good solution of an associated 0-1 programming problem of minimizing the packing constraint violation, as described below.

Unfortunately, since the considered problem has quadratic terms in objective function, it does not admit straightforward linear programming relaxation. There are, however, many ways to linearize a quadratic binary problem to obtain equivalent linear binary problem. Then a lower bound on the optimal binary value can be computed by solving linear programming relaxation. An example of such method is the reformulation-linearization technique [11]. Another way to get the relaxation of quadratic problem is to use semidefinite programming [13], [3], which results in polynomial-time solvable convex optimization problem. Such problems can be also solved in decentralized environment (within a bounded accuracy), see for example algorithms based on Gaussian belief propagation [2]. Both approaches increase the number of decision variables and the number of constraints. Moreover, different approaches vary in the tightness of obtained relaxation and consequently in the resulting precision of approximation. Nevertheless, for the purpose of this paper, we assume that a feasible fractional solution of (2)-(4), denoted $(\hat{\mathbf{x}}, \hat{\mathbf{z}})$, can be computed in decentralized way, such that $0 \le \hat{x}_{ij}, \hat{z}_i \le 1$, for all $i, j = 1, \ldots, n$, and is a lower bound on the optimal binary solution, i.e. $V(\hat{\mathbf{x}}, \hat{\mathbf{z}}) \le V(\mathbf{x}^*, \mathbf{z}^*)$. For further details on the decentralized linear and convex programming we refer to [9].

Let $(\hat{\mathbf{x}}, \hat{\mathbf{z}})$ be a fractional solution of the linear programming relaxation of (2)-(4). Let us define the weighted cost of client-to-server assignment, obtained from the fractional solution $\hat{\mathbf{x}}$:

$$\hat{C}_i = \sum_j \hat{x}_{ij} \left(d_{ij} w_i + h_j \sum_k \hat{x}_{kj} w_k \right). \tag{5}$$

This includes a linear combination of connection and processing costs. Next, for appropriately high $\varepsilon > 0$ we define a neighborhood of node $i \in V$:

$$W_i = \{ j \in V : d_{ij} w_i + h_j \sum_{k: \hat{x}_{kj} > 0} w_k \le (1+\varepsilon) \hat{C}_i \}. \tag{6}$$

The value of ε must be chosen in such a way that each W_i is nonempty. It is easy to check that under this definition of W_i the following property holds [8]:

Proposition 1. Let $\varepsilon > 0$. For each node $i \in V$:

$$\sum_{j \in W_i} \hat{z}_j \ge \sum_{j \in W_i} \hat{x}_{ij} > \frac{\varepsilon}{1+\varepsilon}. \tag{7}$$

Consider the following 0-1 integer programming problem in variables (**x**, **z**) of minimizing real variable L subject to:

$$\forall i \in V \quad \sum_{j \in W_i} x_{ij} = 1, \tag{8}$$

$$\sum_{i \in V} b_i z_i \leq L, \tag{9}$$

$$\forall i, j \in V \quad x_{ij} \leq z_j, \tag{10}$$

$$\forall i \in V, j \in V - W_i \quad x_{ij} = 0. \tag{11}$$

Proposition 2. Let $\varepsilon > 0$ and let $\hat{\mathbf{x}}$ be a fractional solution of problem (2)-(4). Any feasible 0-1 solution **x** to the problem (8)-(11) satisfies:

$$\sum_i \left(\sum_j x_{ij} d_{ij} w_i + \sum_j \left(x_{ij} h_j \sum_k x_{kj} w_k \right) \right) \leq (1+\varepsilon) \sum_i \left(\sum_j \hat{x}_{ij} d_{ij} w_i + \sum_j \left(\hat{x}_{ij} h_j \sum_k \hat{x}_{kj} w_k \right) \right). \tag{12}$$

The proof follows from the definition (6) of W_i. If in the binary solution a client i is assigned to server j, and some subset $S \subset V$ of clients is also connected to the same server j, then the total connection cost of clients $S \cup \{i\}$ is upper-bounded by the corresponding value computed from fractional solution of (2)-(4).

The packing constraint (9) bounds the placement cost in the original problem. Observe that the Proposition 2 implies, that if we construct a feasible solution of the integer problem (8)-(11) we would also obtain an approximate solution of the considered data placement problem (2)-(4), which violates the optimal connection and processing costs at most by a factor $(1+\varepsilon)$. Additionally, such solution also gives the placement cost at most L times the optimal, where L is equal to the value of solution of binary problem (8)-(11).

We now show that it is enough to select only up to $O(\log n)$ server nodes to place the data object, in order to obtain a solution that is feasible with high probability. Consequently, the connection and processing costs will be $O(1)$ times higher than optimal and the placement cost would be up to $O(\log n)$ higher than optimal, which in turn gives $O(\log n)$-approximation algorithm for the data placement problem.

The idea is to perform appropriate randomized rounding of the fractional solution $(\hat{\mathbf{x}}, \hat{\mathbf{z}})$. Given $\varepsilon > 0$ and $0 < \delta < 1$, the algorithm selects $(1+1/\varepsilon) m \log(n/\delta)$ nodes with replacement and the set of selected nodes will hold the data object, i.e. their decision variables z_i are rounded to 1. Each node is selected with probability $p(i) = \hat{z}_i / m$, where m is normalization constant. Thus the total expected cost of all selected nodes is no greater than $O(\log n)$ times optimal placement decisions:

$$(1+1/\varepsilon)m\log(n/\delta)\sum_j \frac{\hat{z}_j}{m} b_j \le (1+1/\varepsilon)\log(n/\delta)\sum_j z_j^* b_j. \quad (13)$$

In the second phase, each client connects to the "cheapest" selected server. The parameter δ controls the probability that the random solution constructed by the algorithm is feasible for (8)-(11). It is assumed that each node i knows only the values: i (its own index), n (network size), b_i, h_i, w_i, a row $\mathbf{D}_i$ of matrix $\mathbf{D}$, and fractional solution $(\hat{\mathbf{x}}_i, \hat{z}_i)$ of the relaxation of problem (2)-(4).

The following procedure is performed concurrently by all server machines $j \in V$:

1. *Send* $\hat{z}_j$ to all other server nodes.
2. *Receive* values $\hat{z}_k$, $k \neq j$, from other nodes.
3. Compute $m = \sum_i \hat{z}_i$ and put $\hat{z}_k := \hat{z}_k / m$ for all k.
4. Determine the minimal and maximal $\hat{z}_k$, denoted $\hat{z}_{\min}$ and $\hat{z}_{\max}$, respectively.
5. If $\hat{z}_j = \hat{z}_{\min}$ then let $R_j = n\hat{z}_j / m$ and $Y_j = k_{\max}$, where $k_{\max}$ is the index of node with $\hat{z}_{\max}$. Go to step 8.
6. If $\hat{z}_j = \hat{z}_{\max}$ then substitute $\hat{z}_j = \frac{\hat{z}_{\max}}{m} - (\frac{1}{n} - \frac{\hat{z}_{\min}}{m})$.
7. Go to step 1 (skipping step 3, as the normalization constant is already known).
8. Wait for all nodes to complete the steps 1-7 (e.g. by *sending*/*receiving* status messages to all servers).
9. Let $t = 0$.
10. Generate random bit and *send* it to all other server nodes.
11. *Receive* random bits from all other server nodes. Construct random variable $0 \le R \le 1$ by concatenating all n bits.
12. Let $q = \lfloor nR \rfloor$ and $p = (nR) \bmod 1$.
13. If $q = j$ and $p < R_j$ then *send* "mark" to node j+1.
14. If $q = j$ and $p \ge R_j$ then *send* "mark" to node Y_j.
15. If $q \neq j$ then *receive* messages from other nodes. If a "mark" message is received then server j is selected for holding data object.
16. Increment t by 1 and go to step 10, until $t \ge (1+1/\varepsilon)m\log(n/\delta)$.

The above algorithm can be seen as fully decentralized version of Walker's method for sampling from discrete probability distribution [6]. Observe that one server node may be selected multiple times in step 15. When the above algorithm terminates the object placement $\mathbf{z}$ is determined. Then the second algorithm is executed to obtain connection matrix $\mathbf{x}$. Each client i queries all servers j for which $\hat{x}_{ij} > 0$, obtaining information whether an object is placed at that node. In the response, along with the value z_j, client also receives the worst-case processing cost, i.e. the value of $h_j \sum_{k:\hat{x}_{kj}>0} w_k$. For j such that $z_j = 1$ each client computes $d_{ij} w_i + h_j \sum_{k:\hat{x}_{kj}>0} w_k$, and connects to server j for which this value is the lowest. If for some client there is no

such server that $z_j = 1$ and $\hat{x}_{ij} > 0$ at the same time, then algorithm chooses among all servers caching the object.

The following theorem is based on approach described in [8].

Theorem 1. The randomized rounding algorithm constructs a feasible solution of problem (8)-(12) with probability at least $1 - \delta$.

Proof sketch: In steps 1-7 the algorithm constructs lookup tables for Walker's sampling method [6]. In steps 9-16 the distributed sampling is performed. A feasible solution of the 0-1 integer program (8)-(11) consists of such selection of server nodes **z**, that every client node i has at least one server with $z_j=1$ in its neighborhood W_i. From the Proposition 2, the probability that there would be at least one selected server in the neighborhood W_i is:

$$\sum_{j \in W_i} p(j) = \sum_{j \in W_i} \frac{\hat{z}_j}{m} > \frac{\varepsilon}{m(1+\varepsilon)}. \quad (14)$$

Since the algorithm selects (with replacement) exactly $(1+1/\varepsilon)m\log(n/\delta)$ nodes, the probability that no server from ith client's neighborhood is selected is:

$$\left(1 - \frac{\varepsilon}{m(1+\varepsilon)}\right)^{(1+1/\varepsilon)m\log(n/\delta)} < \frac{\delta}{n}. \quad (15)$$

Since there are up to n client nodes, then with probability at least $1 - \delta$ all clients will neighbor a selected server.

5 Experimental Results

The presented algorithm was implemented in simulation environment. A short summary of experimental results for different input parameters is presented in Table 1. Optimal solutions for problem instances of size $n \leq 15$ were obtained by exhaustive search, while problem of size $n=100$ was taken from plant location benchmarks library [12]. The results confirm proven bounds.

Table 1. Comparison of approximate and optimal values for random problem instances

n	ε	δ	$(1+1/\varepsilon)m\log(n/\delta)$	approx. value	optimal
5	2	0.1	10	36.0	32.5
10	0.1	0.2	70	740.9	427.0
10	1	0.01	70	938.4	427.0
15	1	0.1	48	723.3	383.8
15	2	0.2	43	519.0	383.8
100	2	0.2	84	66246.4	36154

6 Conclusions and Further Work

In this paper the problem of optimal data placement in content delivery network was formulated in a variant, which combines connection, processing and storage costs. A decentralized algorithm was given, based on randomized rounding, which achieves an asymptotically logarithmic performance bounds with high probability. The presented algorithm combines the general filtering technique given by [8] with a novel decentralized sampling method applied for randomized rounding. A major drawback of the presented algorithm is that it requires on the input a fractional solution of the relaxation of the original problem (lower bound). Unfortunately, obtaining such solution efficiently for a quadratic problem in decentralized environment is a non-trivial task, which will be subject to further work.

Acknowledgments. This research is partially supported by the scholarship co-financed by European Union within European Social Fund.

References

1. Bektas T., Cordeau J., Erkut E., Laporte G.: Exact Algorithms for the Joint Object Placement and Request Routing Problem in Content Distribution Networks. Computers & Operations Research, Vol. 35, pp. 3861—3884 (2008)
2. Bickson, D. and Dolev, D. and Shental, O. and Siegel, P.H. and Wolf, J.K.: Gaussian Belief Propagation Based Multiuser Detection. In: IEEE International Symposium on Information Theory, pp. 1878—1882 (2008)
3. Drwal M., Jozefczyk J.: Load Balanced Location-Routing Problem in Content Distribution Networks. In: 23rd International Conference on Systems Research, Informatics and Cybernetics, pp. 34—38 (2011)
4. Jacobson V. et al: Networking Named Content. In: 5th International Conference on Emerging Networking Experiments and Technologies, pp. 1—12 (2009)
5. Jain K., Vazirani V.: Approximation Algorithms for Metric Facility Location and k-median Problems Using the Primal-Dual Schema and Lagrangian Relaxation. Journal of the ACM, vol. 48(2), pp. 274–296 (2001)
6. Knuth, D.E.: The Art of Computer Programming, Volume 2: Seminumerical Algorithms. Adison Wesley, Reading, MA (1997)
7. Leighton, T.: Improving Performance on the Internet. Communications of the ACM, vol. 52(2), pp. 44—51 (2009)
8. Lin J.H, Vitter J.S.: e-Approximations with Minimum Packing Constraint Violation. In: 24th ACM Symposium on Theory of Computing, pp. 771—782 (1992)
9. Moscibroda, T. and Wattenhofer, R.: Facility Location: Distributed Approximation. In: 24th ACM Symposium on Principles of Distribued Computing, pp. 108—117 (2005)
10. Mosk-Aoyama, D. and Roughgarden, T. and Shah, D.: Fully Distributed Algorithms for Convex Optimization Problems. SIAM Journal on Optimization, vol. 20(6) (2010)
11. Sherali, H.D. and Adams, W.P.: A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems. Kluwer Academic Publishers (1999)
12. UFLP Benchmark Library, http://math.nsc.ru/AP/benchmarks/english.html
13. Vandenberghe, L. and Boyd, S.: Semidefinite Programming. SIAM Review, vol. 38(1), pp. 49—95 (1996)