Revisiting Why Kad Lookup Fails

Bingshuang Liu*†, Tao Wei*†‡, Jianyu Zhang*†, Jun Li§, Wei Zou*† and Mo Zhou*†
*Beijing Key Laboratory of Internet Security Technology, Peking University, Beijing 100871, China
†Institute of Computer Science and Technology, Peking University
Email: {liubingshuang, wei_tao, zhangjianyu, zou_wei, zhoum}@pku.edu.cn
‡University of California, Berkeley, CA 94720, USA
§Department of Computer and Information Science, University of Oregon, Eugene, OR 97403, USA
Email: lijun@cs.uoregon.edu

***Abstract*—Kad is one of the most popular peer-to-peer (P2P) networks deployed on today's Internet. Its reliability is dependent on not only to the usability of the file-sharing service, but also to the capability to support other Internet services. However, Kad can only attain around a 91% lookup success ratio today. We build a measurement system called Anthill to analyze Kad's performance quantitatively, and find that Kad's failures can be classified into four types: packet loss, selective Denial of Service (sDoS) nodes, search sequence miss, and publish/search space miss. The first two are due to environment changes, the third is caused by the detachment of routing and content operations in Kad, and the last one shows the limitations of the Kademlia DHT algorithm under Kad's current configuration. Based on the analysis, we propose corresponding approaches for Kad, which achieve a success ratio of 99.8%, with only moderate communication overhead.**

***Index Terms*—Kad; Lookup; Measurement; Optimization**

I. Introduction

In the past few years, Distributed Hash Table (DHT) has motivated a number of theoretical, modeling and empirical studies in the peer-to-peer (P2P) area. Kad is one of the most popular DHT networks [1], serving millions of users. Based on Kademlia DHT's powerful abilities, Kad should be capable of supporting more challenging services such as P2P-based IM, DNS [2], and Web browser [3]. However, its low reliability prevents Kad from achieving this goal in real-world environments.

"Why Kad lookup fails?" Kang et al. investigated the unsatisfactory reliability of Kad in 2009 [4]. They found that failures were mostly due to the high level of routing table similarity. Although Kad has been improved since then, we found that the current version is still not reliable.

In this paper, we study Kad's lookup performance, mainly focusing on reliability, i.e. lookup success ratio. Throughout our measurements, we found Kad's average lookup success ratio is only about 91% today. We built a measurement system called *Anthill*, in which instrumented clients interact with other normal clients in Kad to reveal causes of lookup failures and to verify the effectiveness of proposed solutions. Through numerous measurements and systematic analysis using Anthill, we classify Kad's lookup failures into four categories: packet loss, selective Denial of Service (sDoS) nodes, search sequence miss, and publish/search space miss. The first two are due to environment changes, including more congested Internet traffic and more diverse Kad participants; the third is caused by the detachment of routing and content operations in Kad; and the last one demonstrates the limitations of the Kademlia DHT algorithm in Kad's current configuration. To the best of our knowledge, the first three categories are discussed for the first time in this paper.

Among these causes, the sDoS nodes are more concealed and have a direct negative impact on lookup reliability. During the monitoring process, as much as 11% of the problem can be traced to attacks on Kad by some parties sharing an interest in disrupting the file sharing community, e.g., MPAA.

Based on the above analysis, we propose several targeted approaches to improve Kad's lookup performance. These approaches raise the lookup success ratio up to 99.8%, and significantly reduce the lookup latency to about 1 second, with only moderate communication overhead. Our results indicate that Kad—with our approaches integrated—can support more performance-sensitive P2P applications.

The rest of this paper is organized as follows: Section II describes the prerequisite knowledge of Kad, and related work. Section III presents the measurement system, Anthill. Section IV analyzes lookup failures in Kad. Section V briefly describes our approaches to improve Kad's reliability. Section VI discusses some potential questions, and Section VII concludes the paper.

II. Background and related work

A. Kad background

In Kad, every node has a unique identifier, referred to as the *node ID*. The node ID is a 128-bit random number. Kad has two kinds of objects: keyword and file. Every object is also assigned a 128-bit identifier using MD4 called the *key ID*. In the same 128-bit ID space for nodes and objects, the distance of two IDs is calculated using the bitwise XOR operation. Thus, Kad maps every object to those nodes closest to its key ID.

Kad supports two primary operations: *Put* and *Get*. Put is to publish $\langle key, value\rangle$ pairs (bind information) to nodes that are closest to key and Get is to retrieve $value$ for a specific key. In both operations, a Kad $lookup$ locates

‡ Corresponding author. This research was supported in part by National Natural Science Foundation of China (Grant No. 61003216) and Beijing Technological Innovation Project (Grant No. Z111101055311052).

nodes which are responsible for the target key. Kad lookup consists of two phases. In *Phase 1*, the initiator tries to locate nodes that are closest to the key by the routing process. The routing algorithm used in Kad is iterative and greedy. During routing process, the intermediate nodes return some closer nodes to the initiator; then the initiator picks up several nodes among them as next hop and repeats the process until no closer nodes are returned. To speed up routing process and cope with unreachable nodes, Kad applies parallel lookup (α) and requests more than one node (β) for next hop from an intermediate node. When the initiator cannot receive any routing response within the last τ seconds, *Phase 2* **starts**. In this phase, the initiator performs content operations (publish request or search request) on these candidate nodes, which are in the *tolerance zone* (the minimum length of common prefix is at least 8) of the key and have replied to a routing request. Taking churn into consideration, Kad also uses a data replication scheme, where a $\langle key, value\rangle$ pair is stored at γ nodes called $replica\ roots$.

B. Related work

Kad has motivated many research works including network measurement, performance evaluation and optimization. In [5], Steiner et al. presented an improvement approach to shorten Kad lookup latency, called Integrated Content Lookup (ICL), which couples content retrieval with the routing process in the tolerance zone during search process.

Stutzbach et al. [6] argued that the lookup inconsistency problem is caused by churn and slow routing table convergence. They improve lookup performance of eMule-Kad by performing parallel lookup and maintaining multiple replicas. However, through our measurement, just tuning such design parameters makes a limited contribution to the lookup performance for today's Kad.

Kang et al. [4] analyzed Kad's poor lookup performance from the perspective of inconsistency between the publish and search processes. They think that the lookup inconsistency is due to the high level of routing table similarity. Two fixes were proposed to improve lookup consistency up to 90% and achieve a success ratio of about 99%. However, it has been verified in Sec. IV-A that nowadays these fixes are not as effective as three years ago.

More recently, Cholez et al. [7] presented an approach to detect suspicious nodes in Kad. This approach is able to detect eclipse attacks, but it cannot detect those misbehaving nodes discovered in this paper.

III. Measurement system: Anthill

A. Anthill

We develop Anthill to flexibly and efficiently measure Kad's performance in the real world.

In Anthill, one Kad lookup task is executed as follows. First, given a specific key, one node (called publisher) takes charge of publishing its binding information in Kad. Then after the publish process finishes, another node (called searcher) immediately starts to search it by the key. While publishing and searching, Anthill records all the traffic of Kad to produce lookup traces. Based on these traces, we can analyze Kad's lookup process in detail.

Table I lists Kad parameters mentioned earlier and those to be used in Anthill. Unless otherwise specified, all the parameters adopt the default values during following experiments.

	Default	Description
α	3	Parallelism degree of lookup
β	4(publish),2(search)	Number of nodes required for next hop from an intermediate node
γ	10	Number of replica roots
o	Y	Whether *Fix1* will be used
ω	N	Whether *Fix2* will be used
τ	3s	Timeout for each routing request
κ	1.5s	Timeout for packet-retransmission scheme
λ	0	Retransmission times of a request packet
θ	128	The minimum length of common prefix for neighborhood lookup (ref. to Sec. V)

TABLE I: The description and default values of Kad and Anthill parameters. α, β, γ and o are Kad's parameters, and the remainder are for Anthill.

In order to quantitatively measure the performance, we give some profiling metrics as follows.

Search (Lookup) Success Ratio: The rate of successful searches which retrieves at least one $value$ for a key from any replica root is calculated as:

$$\frac{number\_of\_successful\_searches}{number\_of\_total\_searches}.$$

Lookup Latency: The time interval from starting routing process to retrieving a $value$ for the first time in one search process.

Lookup Cost: The total bytes sent and received by one lookup process.

B. Experiment setup

Anthill sets up 64 instrumented aMule nodes on a PC server with two Intel Xeon CPUs (E5645, 2.40GHz, 24GB RAM). These nodes are equally divided into two groups: one for publishing and the other for searching. Each node uses a different IP address to connect to the Internet directly, and maintains an independent routing table. All nodes distribute evenly across the whole ID space.

Each experiment contains 3200 lookup tasks, which are selected from 64 6-bit sub-zones of the entire ID space. 50 key IDs are chosen randomly in each sub-zone. For simplicity, Anthill only uses file binding information, denoted as "⟨key ID, file information⟩". In order to distinguish Anthill lookup targets from other existing ones, the $file\ information$ follows this format "*randomfilename*.anthill". The *randomfilename* is a 15-byte string generated randomly. Furthermore the file size is a particular number for each experiment. Anthill controls the speed of processing tasks according to the flooding control mechanism of Kad. It takes Anthill about two hours to accomplish the 3200 tasks during each experiment. All the experiments were done from January to March 2012.

IV. Analysis of lookup failures

In this section, we first examine the reliability of the current version of Kad. Then we will try to analyze the root causes of lookup failures. Through systematic measurements and analysis, four main issues are identified: packet loss, sDoS nodes, search sequence miss and publish/search space miss.

A. Reliability of current Kad

In 2009, two fixes were proposed to improve Kad lookup reliability in [4]. At present, the *Fix1* has been adopted by the latest eMule and aMule. We implement *Fix2* in Anthill according to the descriptions in [4] (our implementation has been sent and checked by the author of [4]). We conduct experiments to test their performance. The results are presented in Table II, and indicate that today the improvement of these existing fixes is limited for Kad.

Version	Succ. ratio(%)	Latency(s)	Cost(KB)
Standard	91.19	7.38	10.497
Standard+Fix2	93.34	8.13	11.704

TABLE II: The lookup reliability of Kad.

In the following sections, we conduct systematic analysis of lookup failures and put identified causes into four categories: packet loss, sDoS nodes, search sequence miss, and publish/search space miss. Fig. 1 presents that the proportions of failures in each category.

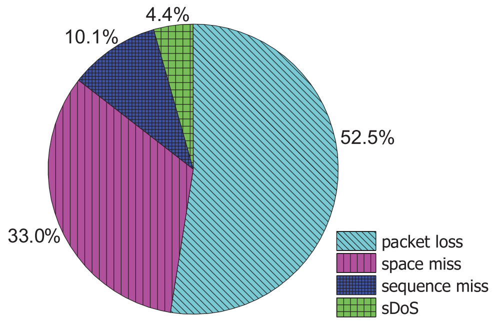

Fig. 1: The distribution of lookup failures.

B. Packet loss

At present, the Internet has become more and more congested and packets may be dropped on the fly due to various problems. Furthermore, Kad uses UDP as the transport protocol in its underlying network, which cannot provide any guarantee of reliable transport. Unfortunately, all the existing implementations of Kad have not considered this problem. Therefore we infer that the reliability of the underlying Internet has critical impacts on Kad.

We first measure the proportion of packet loss (P_{loss}) in the real world from Kad's perspective. Here, we give the definition of P_{loss}:

$$P_{loss} = 1 - \frac{number\_of\_response}{number\_of\_request}. \tag{IV.1}$$

When computing P_{loss}, unreachable nodes should be excluded. According to previous study [8], nearly 35.1% of Kad nodes are unreachable, due to the maintenance schemes of routing tables, firewalls or NAT. In this paper, the identification of these nodes is achieved by the packet-retransmission scheme (refer to Sec. V). Nodes who keep silent for any request are thought to be unreachable. In the experiment setup, retransmission times "λ" is set to 3, which means one node will have at most 4 chances to declare its activeness. Then only the first request and the corresponding response linked to reachable nodes are considered in Equation IV.1.

Network Type	BW. (Mbps)	$P_{loss}(\%)$				
		Avg.	Max.	Min.	SD.	(0.95)CI.
ADSL(fj_cu)	4	13.61	27.84	4.24	3.65	(13.41,13.80)
ADSL(bj_cu)	4	15.87	38.37	0.00	4.04	(15.72,16.02)
ADSL(fj_ct)	4	15.03	33.33	3.36	4.42	(14.79,15.26)
ADSL(cq_ct)	4	21.45	46.00	1.30	4.93	(21.19,21.71)
WCDMA(bj_cu)	↑5.76;↓7.2	17.09	53.33	3.45	6.53	(15.99,18.19)

TABLE III: Packet loss in different networks. "bj","fj" and "cq" are three places. "cu" and "ct" are the two ISPs.

To make results more comprehensive, we did the experiments across networks in different locations and during different hours of the day. During each experiment, more than 150 thousand reachable nodes are encountered. We present these results in Table III, which shows P_{loss} is 16.5% on average. In the worst case, 53.33% of packets in WCDMA are lost during one lookup process.

It is obvious that packet loss on the Internet depresses Kad's current lookup reliability. According to our measurements under the packet-retransmission scheme described in Sec. V, the search success ratio would improve to 95.97% from 91.19% (standard Kad) after the impact of packet loss is eliminated as much as possible. In other words,packet loss contributes 4.78% to the whole search failure ratio.

C. sDoS failure

When analyzing Kad lookup trace files, we found that some searches fail in an odd way. Though the searcher can locate at least one "replica root" (which gives the publisher a normal publish response) and send search requests to them, it receives no search responses. We call such a replica root an **sDoS node** and this kind of failure an **sDoS failure**.

In our work, we design an experiment to reveal and measure sDoS nodes in Kad. To eliminate the impact of packet loss, λ is set to 3. And the publisher performs search requests on replica roots for identifying sDoS nodes. Other parameters not mentioned are the same as current Kad's. In addition, this experiment has been conducted five times at different times to minimize the influence of accidental factors.

Next, sDoS nodes will be comprehensively analyzed, including the behavioral characteristic, impacts on lookup reliability and the origin of such nodes.

Behavioral characteristic of sDoS nodes: By comparing nodes' behavior to Kad protocol in a meticulous way, we capture the principal behavioral characteristic of sDoS nodes:

Country or region	%
CN	38.7
TW	32.3
US	13.9
HK	2.3
IT	1.9
FR	1.4
JP	1.2
ES	1.1
MY	1.0
SG	0.8

(a) Geographic-Level

ISP	Type	%
CHTD, Chunghwa Telecom Co., Ltd.	Commercial ISP	27.2
ChinaNet	Commercial ISP	20.7
China Unicom	Commercial ISP	14.6
*Dino Solutions(Quick Connect Hosting)	Hosting Provider	10.9
City Telecom(H.K.) Ltd.	Commercial ISP	1.3
Hoshin MultiMedia Center Inc.	Hosting Provider	1.2
Telekom Malaysia Berhad	Commercial ISP	0.9
Telecom Italia	Commercial ISP	0.8
TUNG HO MULTIMEDIA CO. Ltd.	Hosting Provider	0.8
Free SAS	Commercial ISP	0.8

(b) ISP-Level

IP	%
208.86.198.78	8.24
208.86.198.79	2.68
x.x.x.x	0.18
x.x.x.x	0.16
x.x.x.x	0.12
x.x.x.x	0.12
x.x.x.x	0.12
x.x.x.x	0.12
x.x.x.x	0.12
x.x.x.x	0.10

(c) IP-Level

TABLE V: The top 10 distribution of sDoS nodes.

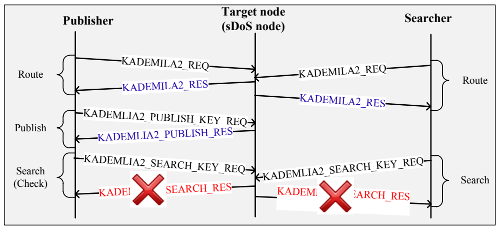

Fig. 2: The interactive diagram of an sDoS example.

Exp.	Average total failures(%): 4.31±0.57			
	Space miss failures(%)	Seq. miss failures(%)	sDoS failures(%)	
			1-failures	(⩾2)-failures
No.1	2.34	1.28	0.38	0.03
No.2	2.66	0.88	0.19	0.06
No.3	3.19	0.56	0.25	0.06
No.4	3.09	0.75	0.47	0.09
No.5	3.69	1.13	0.38	0.06
Avg.&SD.	3.00±0.52	0.92±0.29	0.34±0.11	0.06±0.02

TABLE IV: The distribution of lookup failures after eliminating the impact of packet loss. The "i-failures" means the lookup fails when the searcher has located i replica roots.

selective Denial of Service. Here the service refers to providing Kad functions to other nodes, such as routing and store service, and the selectivity means that these nodes do not refuse all kinds of requests, except for search requests. To put it more clearly, an interactive diagram in Fig. 2 shows the strange behavior of such nodes. Though the *target node* seems to be a replica root, both *publisher* and *searcher* receive no search response. Therefore this kind of nodes has a vital impact on lookup reliability.

Impact of sDoS nodes: The experiment results show the average ratio of sDoS nodes among all replica roots is around 3.5%. Referring to the column "sDoS failures" in Table IV, this directly makes 0.40% of searches fail, though the searchers have located at least one replica roots. Furthermore, the distribution of sDoS failures are listed in columns "i-failures". Obviously, the impact of sDoS nodes is mainly reflected in the cases of "1-failures". When only one replica root is located by the searcher, the lookup fails at a 3.5% probability due to the existence of sDoS nodes. While the searcher can locate at least two replica roots, the probability of failure (refer to "(⩾2)-failures") radically decreases.

The origin of sDoS nodes: In order to investigate them further, we study the distribution characteristics of these nodes on geographic-level, ISP-level and IP-level. Nodes' IPs are mapped to above information using the MaxMind Database. Table V gives the top 10 distribution.

The results show such misbehaving nodes have been widespread in Kad and show very strong aggregation, especially in IP-level. In Table V(c), the two most active IPs (nearly 11% of sDoS attacks), 208.86.198.78 and 208.86.198.79, both come from the organization "Quick Connect Hosting" under the ISP "Dino Solutions" in Table V(b). After further investigation, we found that the two IPs were rented by Media Defender [9] at that time. Hired by the MPAA and RIAA, Media Defender has conducted several attacks on P2P file-sharing systems for copyright protection, such as the DDoS attack on Revision3 in 2008.

In summary, there are four types of sDoS nodes: Anti-P2P (copyright protection) organizations, malicious nodes, free rides and network measurement nodes.

D. Sequence miss failure

The **search sequence miss failure** means that several replica roots are indeed discovered by the searcher, but the searcher has no time to perform search operations on these nodes in *Phase 2*. In the publish process, some replica roots are actually not very close to the target due to the limitations of the publisher's local view. According to Kad lookup mechanism, the closer the candidate node is, the earlier the searcher sends a search request to it. Yet the timeout of the entire search process is only 45s, so the searcher perhaps tries to get wanted information from some closer but non-replica roots incorrectly, and misses these replica roots in the end.

During the experiments in Table IV, the phenomenon of sequence miss occurs in more than 27% of searches, and makes 0.92% of searches fail.

E. Space miss failure

In both cases of sDoS failures and sequence miss failures, the searcher is still able to encounter some replica roots. However, there is another kind of failure, **publish/search space miss failure**, i.e. a searcher cannot meet any replica root during routing process. Once space miss happens, the search must fail. The column "Space miss failures" of Table IV shows that about 3.00% of searches fail due to this reason.

Space miss shows the limit of the Kademlia DHT algorithm under current Kad's configuration. In Kad, the routing capability of publisher is much stronger than searcher: The timeout of publish process is 140s, while only 45s for search; β, the number of nodes required for next hop, is 4 and 2, respectively.

V. Solutions

Based on the above analysis, we propose these approaches to improve Kad's performance:

Packet-retransmission: Retransmit a request packet if the response packet is not received before a pre-set timeout, κ. Experiment results show that retransmitting is a simple but effective for applications on top of the underlying network, and the search success ratio is raised up to 95.97% (when λ=3, κ=1.5s) from the initial 91.19%.

Neighborhood lookup: The publisher/searcher directly sends publish/search requests to the nodes in the target's neighborhood, without waiting for routing termination. The θ-neighborhood of a specific key is defined as follows: the region of these nodes whose node IDs have at least θ common prefix bits with key. It is an intuitive approach to eliminate search sequence miss caused by detachment between content operations and routing process. In additions, the impact of sDoS nodes will be also mitigated, because the searcher gets more opportunity to encounter replica roots.

Experiments show that the proportion of failures caused by sDoS nodes under neighborhood lookup decreases from 0.4% to less than 0.1% on average when θ falls below 18. Sequence miss failures disappear entirely as θ falls below 14. At the same time, lookup latency decreases significantly from 7.6s in current Kad to about 1s. This is desirable for latency-sensitive P2P-based applications.

β-adjusting: After empirically evaluating the impact of Kad parameters, we found that increasing β is a good choice to balance the inconsistent routing capability between publishers and searchers. The experiment results show increasing β can effectively drop the percentage of space miss. When β is set to 4 (same for publisher and searcher), the space miss failure percentage decreases from initial 3.2% to 1.5%, with an additional 1.5KB cost.

Based above analysis and experiments, we integrate all proposed approaches into Kad to examine the overall performance. The recommended parameters in these solutions are $\{\kappa$:1.5s, λ:3, β:4, θ:14$\}$. In this configuration, Kad lookup reliability reaches a search success ratio of 99.8%, and latency is decreased to about 1.1s, while the communication overhead remains modest for both publishers and searchers: it generates about 14 replica roots, a little more than the current Kad's threshold (γ=10) in the publish process, and the increment of search cost is about 5KB. As a result, we believe that this lookup performance is capable of supporting more performance-sensitive applications.

VI. Discussion

In this section, we discuss potential questions.

Firstly, the meaningfulness of this work is further discussed. Nowadays, user experience of eMule-Kad does not seem so bad. Users usually can find some files by a certain keyword. It mainly benefits from the huge volumes of users and resources. In eMule-Kad, one binding information may be published by many users and be stored on hundreds of nodes. Therefore Kad currently provides good user experience. However, the situation will change if Kad is taken as an infrastructure for various applications. Though the large user community of Kad can be utilized, some applications such as IM and DNS usually require publishing or searching from a single user. As a result, the strong reliability studied in our work will be required.

Secondly, problems about sDoS nodes are considered. According to our measurements since Feb. 2011, the influence of such nodes on Kad has become more and more serious. If we do not take effective measures, it will further damage the entire Kad. Although we are able to identify such nodes, removing them from Kad is still an open problem.

Finally, we do not discuss long term availability of published resources [10] in this paper, which is left for our future work.

VII. Conclusion

In this paper, we conduct quantitative analysis of Kad's lookup failures (about 9%). We put Kad's failures into four categories: packet loss (52.5%), sDoS nodes (4.4%), search sequence miss (10.1%), and publish/search space miss (33.0%). Several countermeasures are proposed, and the evaluation results show that with these improvements, Kad can work much better and is capable of supporting more Internet services.

References

[1] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the xor metric," in *1st International Workshop on Peer-to-peer Systems (IPTPS'02)*.

[2] V. Ramasubramanian and E. Sirer, "The design and implementation of a next generation name service for the internet," in *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, 2004, pp. 331–342.

[3] P. Garcia Lopez, M. Sanchez Artigas, and J. Pujol Ahullo, "The p2pweb model: a glue for the web," in *16th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'07)*.

[4] H. Kang, E. Chan-Tin, N. Hopper, and Y. Kim, "Why kad lookup fails," in *IEEE Ninth International Conference on Peer-to-Peer Computing (P2P'09)*.

[5] M. Steiner, D. Carra, and E. Biersack, "Faster content access in kad," in *Eighth International Conference on Peer-to-Peer Computing (P2P'08)*.

[6] D. Stutzbach and R. Rejaie, "Improving lookup performance over a widely-deployed dht," in *Proc. Infocom*, vol. 6. Citeseer, 2006.

[7] T. Cholez, C. Henard, I. Chrisment, O. Festor, G. Doyen, and R. Khatoun, "A first approach to detect suspicious peers in the kad p2p network," in *6th IEEE Conference on Network and Information Systems Security (SAR-SSI'11)*.

[8] J. Yu and Z. Li, "Active measurement of routing table in kad," in *6th IEEE Consumer Communications and Networking Conference (C-CNC'09)*.

[9] "The famed anti-p2p company," http://en.wikipedia.org/wiki/MediaDefender, March 2012.

[10] D. Carra and E. Biersack, "Building a reliable p2p system out of unreliable p2p clients: The case of kad," in *Proceedings of the 2007 ACM CoNEXT conference*.