A Lightweight Data Location Service for Nondeterministic Exascale Storage Systems

Zhiwei Sun[1], Anthony Skjellum[1], Lee Ward[2], Matthew L. Curry[2]

[1]Department of Computer and Information Sciences, The University of Alabama at Birmingham, Birmingham, AL;
[2]Sandia National Laboratories, Albuquerque, NM

Introduction

In this study, we present LWDLS, a lightweight data location service designed for Exascale storage systems (storage systems with order of 10^18 bytes) and geo-distributed storage systems (large storage systems with physically distributed locations). LWDLS provides a search-based data location solution, and enables free data placement, movement, and replication. In LWDLS, probe and prune protocols are introduced that reduce topology mismatch, and a heuristic flooding search algorithm (HFS) is presented that achieves higher search efficiency than pure flooding search while delivering comparable search speed and coverage to pure flooding search.

LWDLS is lightweight and scalable in terms of incorporating low overhead, high search efficiency, no global state, and avoiding periodic messages. LWDLS is fully distributed and can be used in nondeterministic storage systems and in deterministic storage systems to deal with cases where search is needed.

Extensive simulations modeling large-scale High Performance Computing (HPC) storage environments provide representative performance outcomes. Performance is evaluated by metrics including search scope, search efficiency, and average neighbor distance. Results show that LWDLS is able to locate data efficiently with low cost of state maintenance in arbitrary network environments. Through these simulations, we demonstrate the effectiveness of protocols and search algorithm of LWDLS.

Probe and Prune Protocols

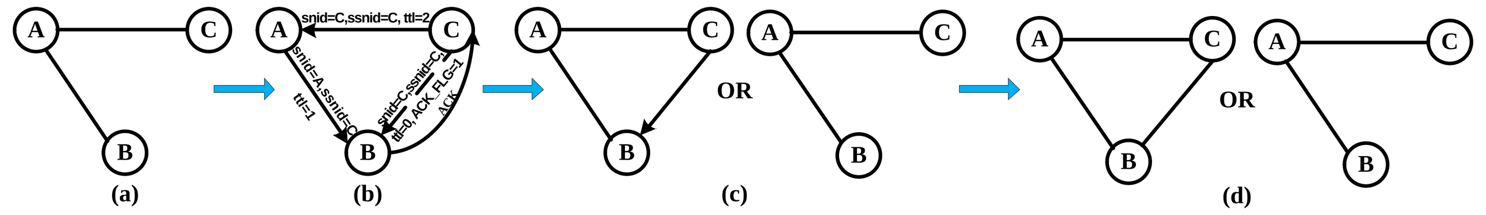


Figure 1. Example of a probing operation. (a) Initially, server B and C are not direct neighbors. (b) Server C starts to probe B, and B sends an ACK message back to C. (c) After probing, C may add B into its neighbor list or not. (d) If C adds B into its neighbor list, eventually B will also add C into B's neighbor list, since C will send to B message with the bit of IS_NEIGH being one.

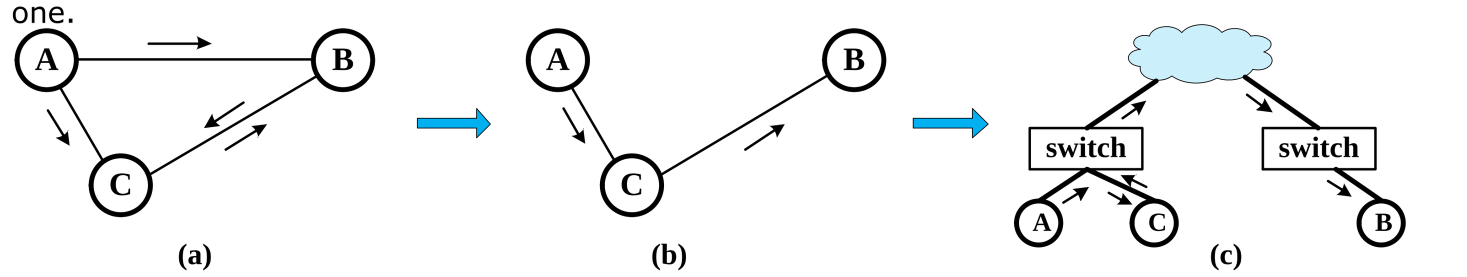


Figure 2. Pruning operation based on the ratio of the lengths of two adjacent edges of a triangle. (a) Server A initiates a flooding search, and two redundant messages are generated on edge BC. (b) After pruning edge AB, two redundant messages eliminated. (c) Example of an ideal case where A and C are more physically closer than to B, and topology mismatch is reduced by pruning operation of LWDLS.

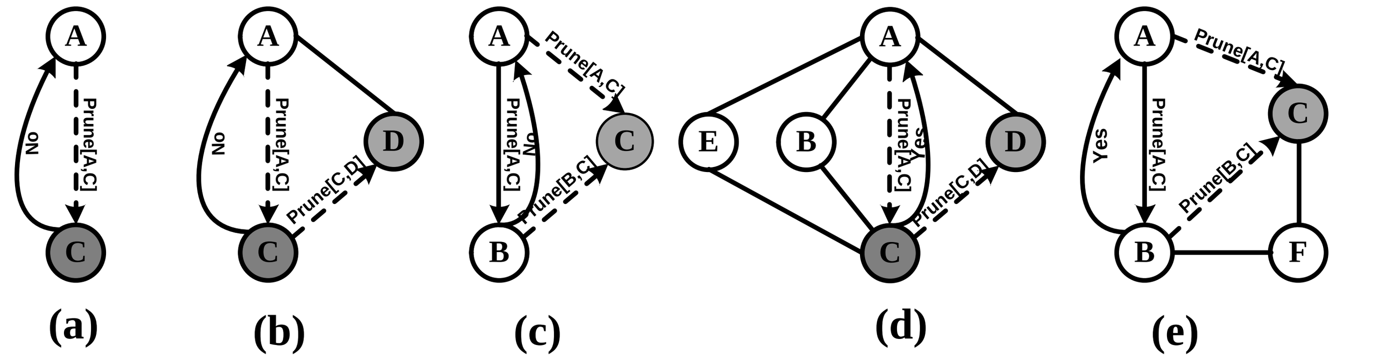


Figure 3. Cases of pruning operations for avoiding partitioning network graph, where server A tries to prune server C from its neighbor list in distributed environment. In cases (a) and (b), Server C will disagree on A's pruning request, since there is no alternative path to A. And in case (d), server C will agree on server A's pruning request. In case (c), one of server A's neighbors, say server B, will disagree on pruning C by A. And in case (e), B will agree on A's pruning operation.

State Transition among the Types of Server Lists in LWDLS

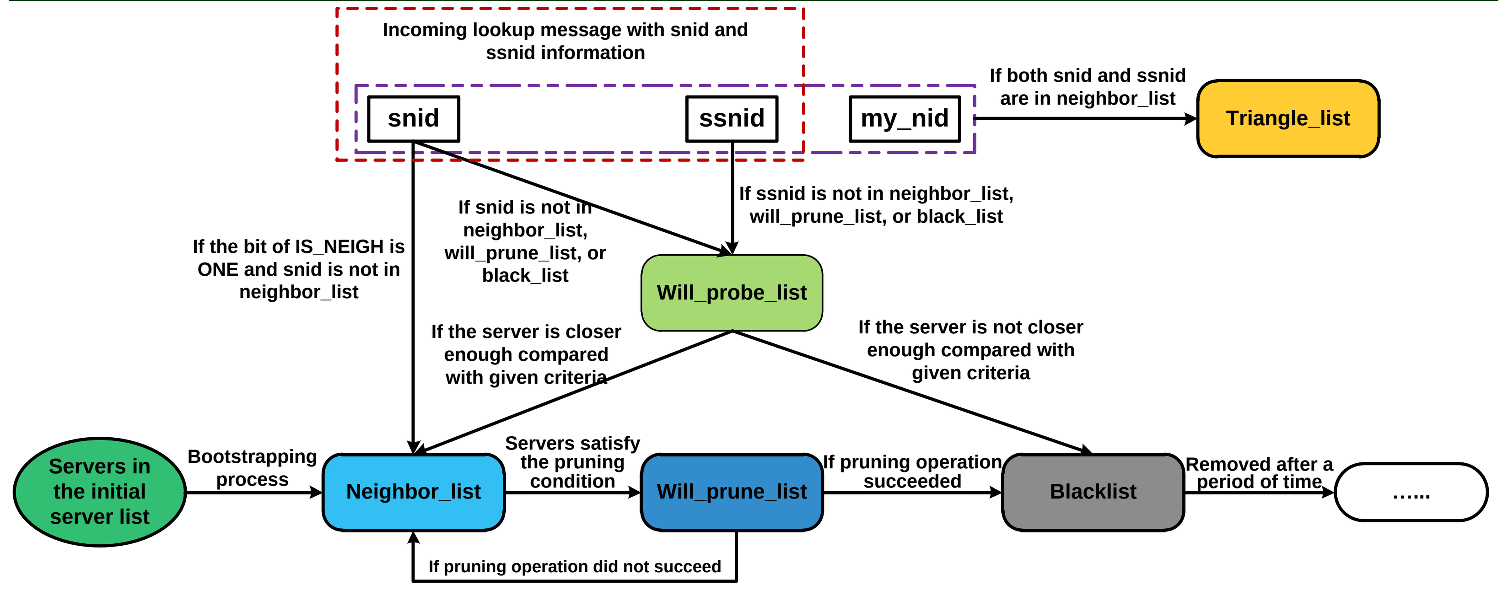


Figure 4. Diagram of state transitions for servers among the types of server lists used in LWDLS

A Heuristic Flooding Search (HFS) Algorithm

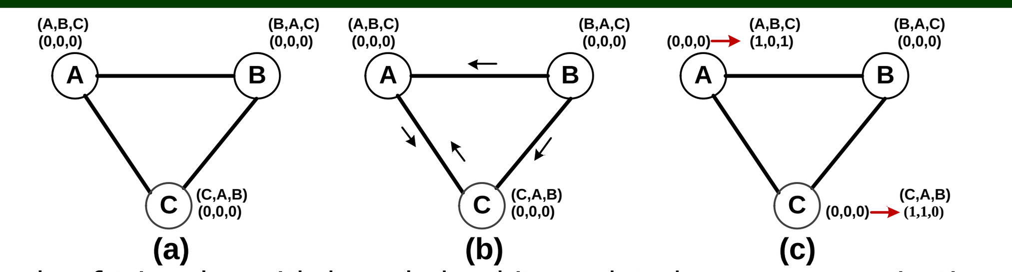
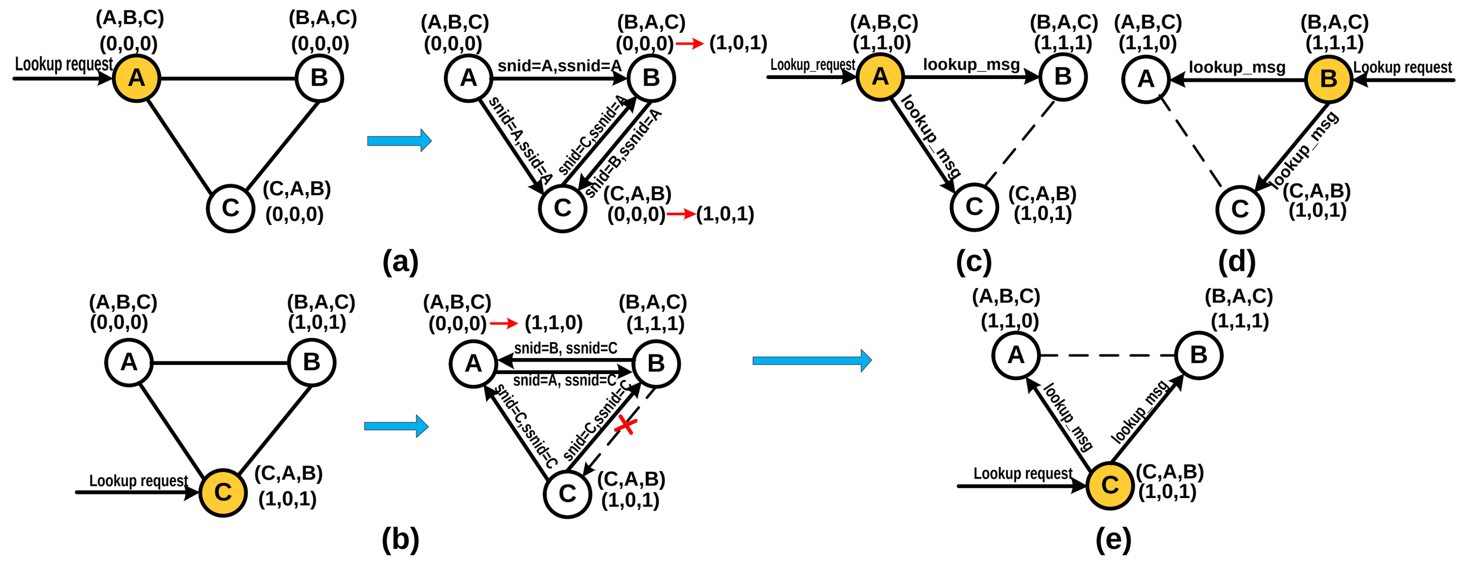


Figure 5. Example of triangles with knowledge bits updated over communications. (a) Initially, A, B, and C don't know that they have form a triangle. (b) Server B broadcasts lookup messages to its neighbors, and server A and C forward lookup messages. (c) After A and C receive redundant messages they update their knowledge bits accordingly.



Phase-I: Accumulating states from duplicated messages received.

Phase-II: HFS reduced sending redundant messages by using the states accumulated.

Figure 6. Accumulating states for reducing redundant messages with triangles and knowledge bits encoded. (a) After B and C receive redundant messages from each other, both of them detect the triangle ABC, and update their knowledge bits. (b) When B receives a redundant message from A, B is allowed to send a duplicated message to A on purpose, so that A can detect this triangle and updated its knowledge bits. In (c),(d), and (e), on each triangle only two messages are needed among the three nodes without sending redundant message by HFS.

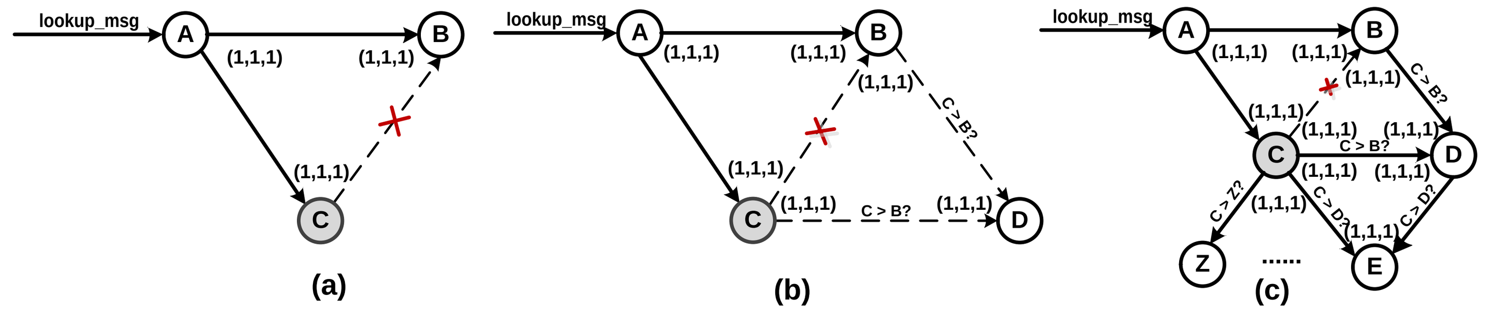


Figure 7. Examples of selectively forwarding messages by HFS. (a) Only one triangle is involved. C does not need to send message to B, since A has sent. (b) Two triangles are involved. For D, the one having larger ID among B and C should send to D. (c) A number of triangles are involved.).

Simulations

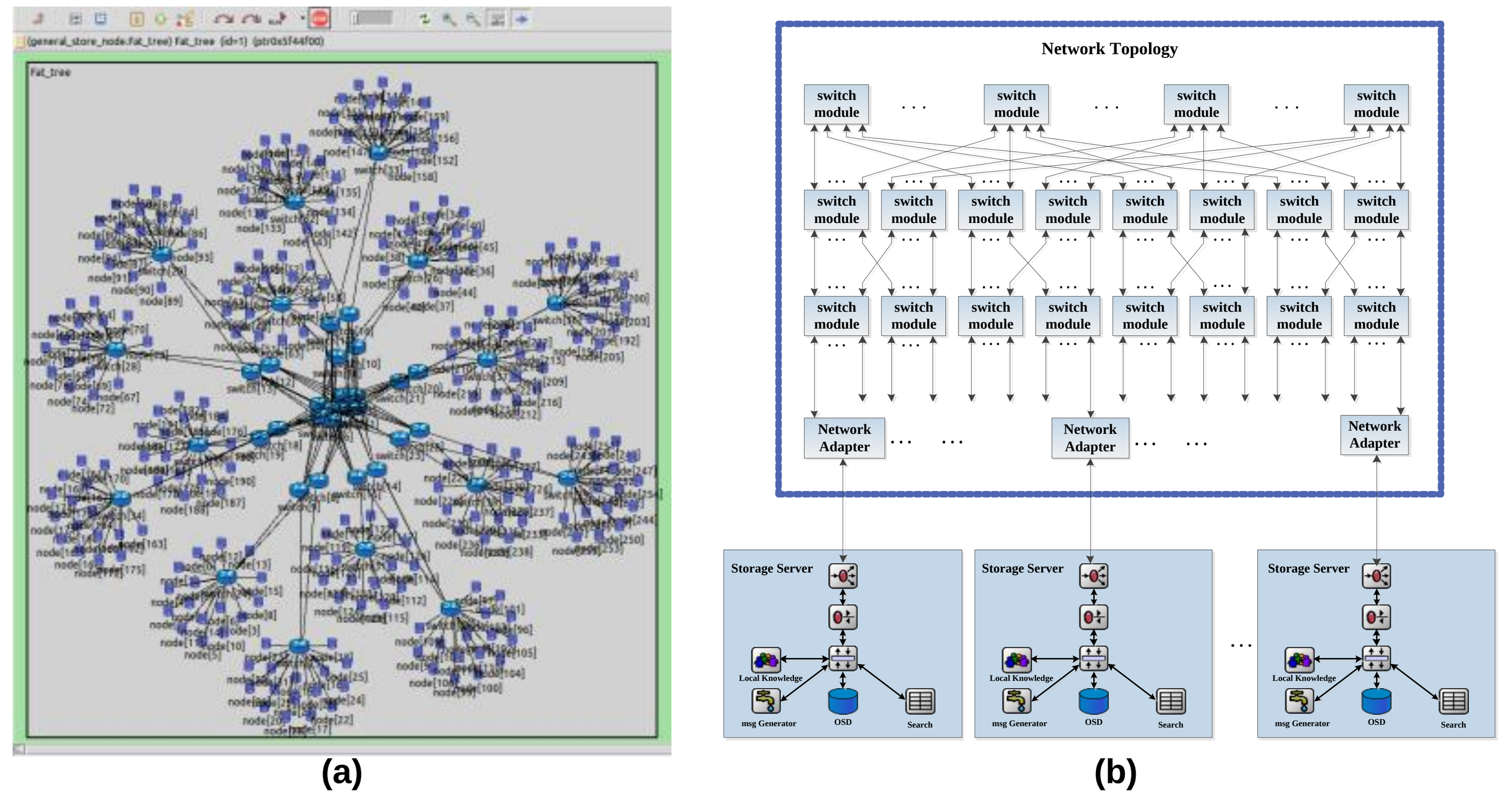


Figure 8. Simulator. (a) Simulated HPC storage system with three-level fat-tree network topology. (b) Architecture of simulator.

Initial Overlay Network Graphs

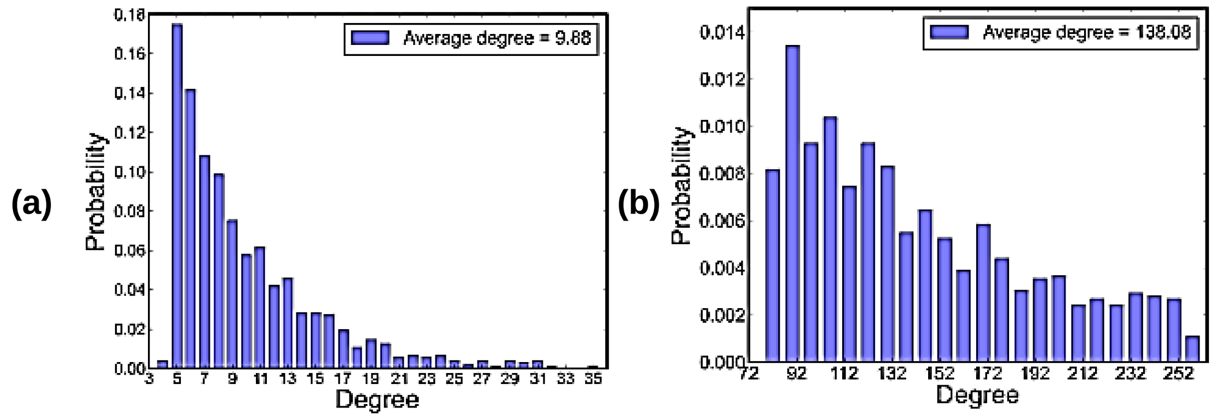


Figure 9. Two overlay network graphs used for testing LWDLS. (a) Initial degree distribution of network graph with M=5. (b) Initial degree distribution of overlay network graph with M=80.

Comparison of Search Performance

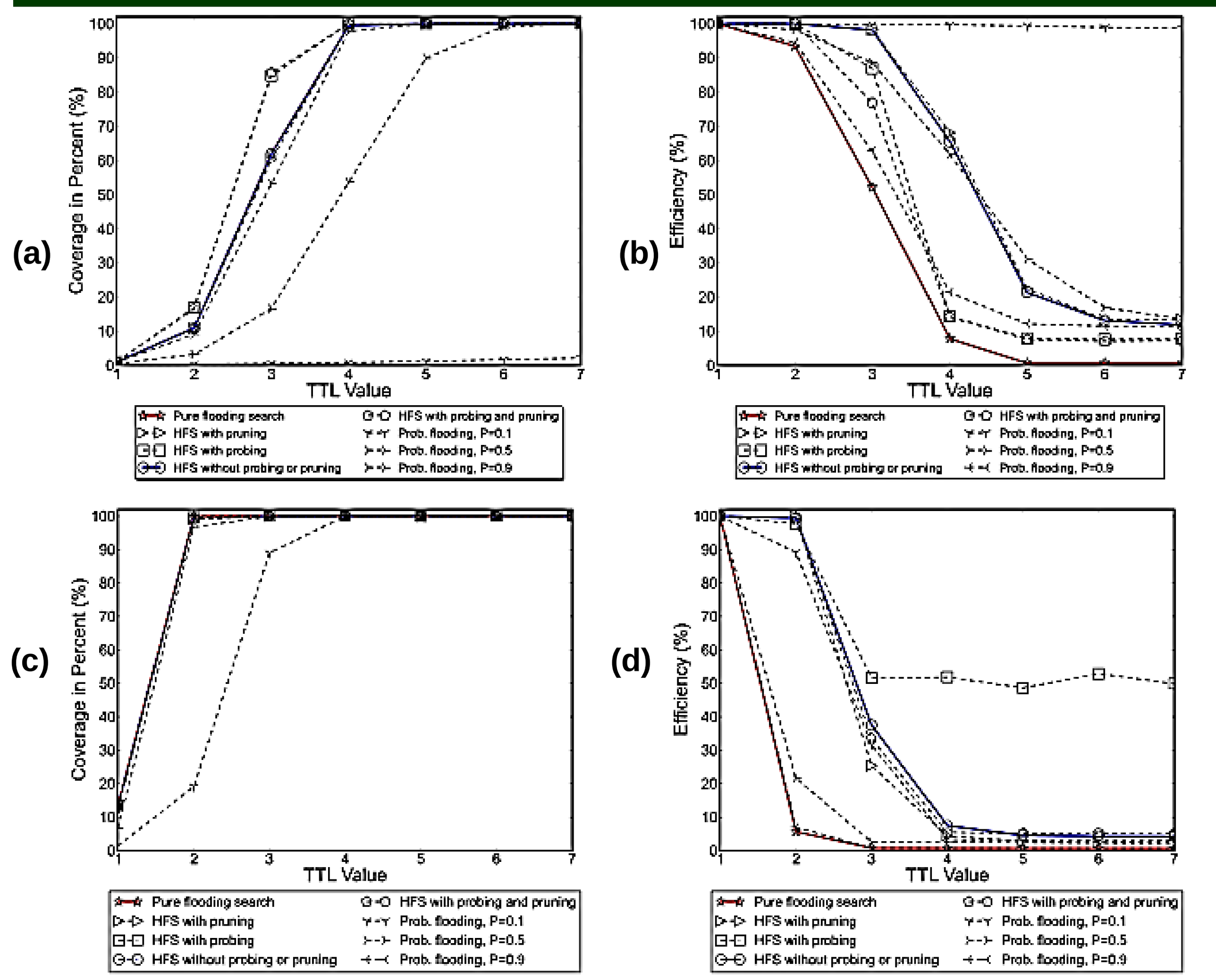


Figure 10. Comparison of search performance on the two network graphs (a) Search scope with M=5. (b) Search efficiency on the graph with M=5. (c) Search scope on the graph with M=80. (d) Search efficiency on the graph with M=80.

Effectiveness of Probing and Pruning Operations

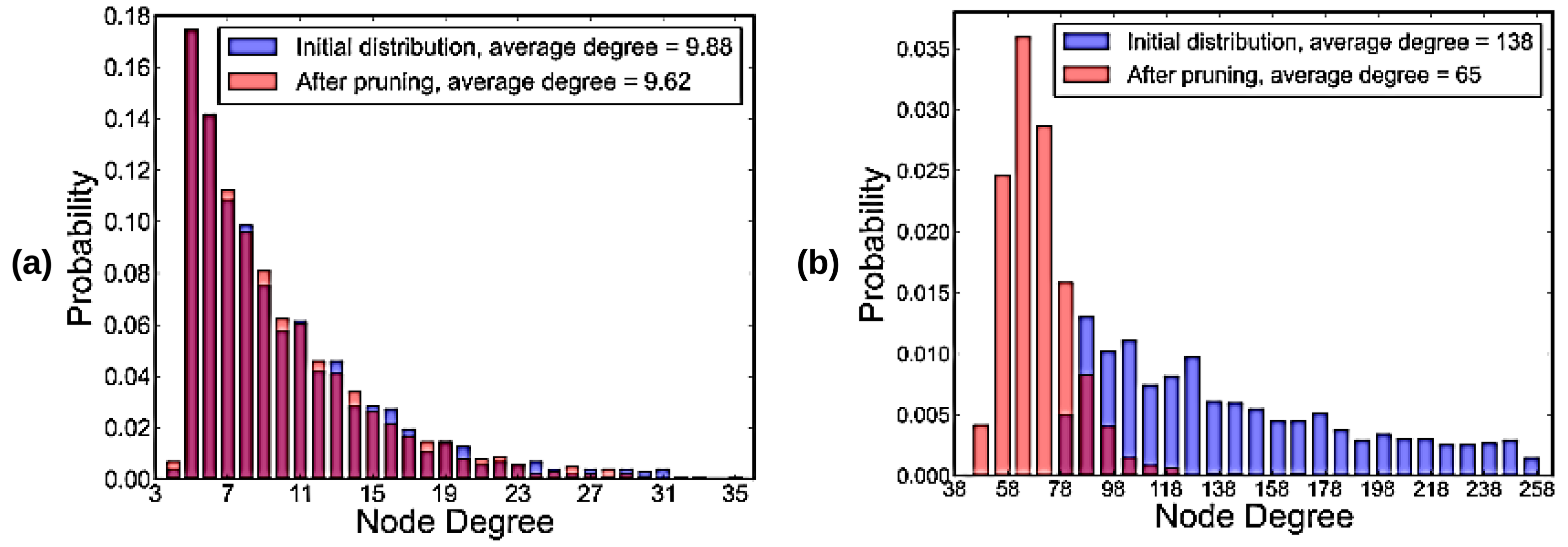


Figure 11. Effects of probing and pruning operations on the node degree distribution, tested on two overlay network topologies. (a) Change of node degree distribution on overlay network with M=5. (b) Change of node degree distribution on overlay network graph with M=80.

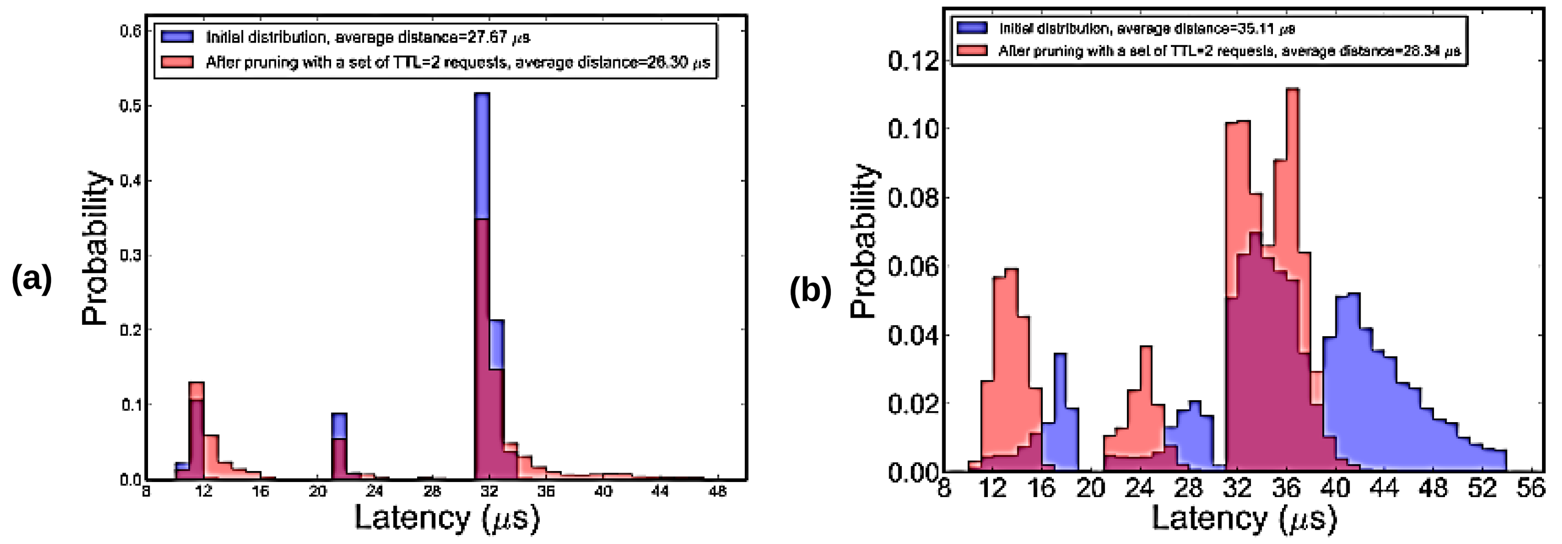


Figure 12. Reduction of neighbor distances. (a) Distribution of neighbor distances after pruning with the initial overlay network graph with M=5. (b) Distribution of neighbor distances after pruning with the initial overlay network graph with M=80.

Conclusions

- LWDLS is a search-based data location service, enables free data placement, movement, and replication.
- LWDLS is able to locate data efficiently in nondeterministic Exascale storage systems.
- LWDLS is lightweight, efficient, and scalable in terms of avoiding global state, periodic messages, or the limitations on locations of data.
- With the probe and prune protocols and HFS algorithm, LWDLS is able to address problems of topology mismatch and inefficient search performance of the pure flooding search algorithm.
- LWDLS provides higher search efficiency than pure flooding search while delivering comparable search speed and search coverage.
- The effectiveness of LWDLS has been tested through extensive simulations modeling large-scale HPC storage environments.

Acknowledgements

This work was supported in part by the United States Department of Energy under Contract DE-AC04-94AL85000. This work was supported in part by the National Science Foundation under grant OCI-1064247and CCF-1239962. We want to thank UAB CIS IT staff for maintaining the Linux clusters used and providing timely supports.