Resource Overbooking: Using Aggregation Profiling in large scale Resource Discovery

[1]Dr.M.Helda Mercy, [2]*C.Anand,* [3] *T.S. Suganya*
[1]*Faculty, M.C.A, Panimalar Engineering College, Chennai*
[2]*PG Scholar, M.C.A, Panimalar Engineering College, Chennai*
[3]*Faculty, M.C.A, Panimalar Engineering College, Chennai*

Abstract

Resource discovery is an important process for finding suitable nodes that satisfy application requirements in large loosely coupled distributed systems. Besides inter node heterogeneity, many of these systems also show a high degree of intra node dynamism, so that selecting nodes based only on their recently observed resource capacities can lead to poor deployment decisions resulting in application failures or migration overheads. However, most existing resource discovery mechanisms rely mainly on recent observations to achieve scalability in large systems. In this paper, we propose the notion of a resource bundle—a representative resource usage distribution for a group of nodes with similar resource usage patterns—that employs two complementary techniques to overcome the limitations of existing techniques: resource usage histograms to provide statistical guarantees for resource capacities and clustering-based resource aggregation to achieve scalability. Using trace-driven simulations and data analysis of a month-long Planet Lab trace, we show that resource bundles are able to provide high accuracy for statistical resource discovery, while achieving high scalability. We also show that resource bundles are ideally suited for identifying group-level characteristics (e.g., hot spots, total group capacity).

I. INTRODUCTION

RECENT years have seen increasing use of loosely coupled distributed platforms for scientific computation data sharing and dissemination and experimental test beds. Examples of such large-scale platforms include volunteer computation grids such as SETI@home (over 3.6 million participant machines), P2P systems such as Kazaa (over 30 million users) and Kad networks (over 2 million users). While such platforms are highly attractive due to their low deployment cost and inherent scalability, they are also highly heterogeneous and dynamic. The nodes participating in such platforms differ widely in their resource capabilities such as CPU speeds, bandwidth, and memory capacity. As a result, resource discovery is often used in such large-scale systems to find suitable nodes that satisfy application requirements.

Many existing resource discovery systems rely on the recently observed resource capacities of individual nodes to make their deployment decisions. However, resource allocation decisions based on current status of nodes have severe limitations in these systems, because of the presence of intranode dynamism in addition to the internode heterogeneity. Individual nodes can have widely varying resource capabilities due to varying loads, network connectivity, churn, or user behavior. For instance, a resource usage study of PlanetLab has shown that node resource capabilities fluctuate on the order of about 30 minutes.

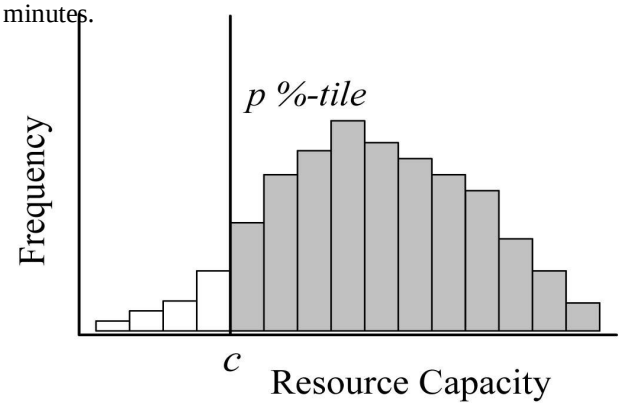

Fig. 1. Mapping a statistical requirement to a resource capacity distribution.

II. EXISTING

RECENT years have seen increasing use of loosely coupled distributed platforms for scientific computation, data sharing and dissemination, and experimental test beds. Examples of such large-scale platforms include volunteer computation grids such as SETI@home (over 3.6 million participant machines), P2P systems such as Kazaa (over 30 million users) and Kad networks (over 2 million users). While such platforms are highly attractive due to their low deployment cost and inherent scalability, they are also highly heterogeneous and dynamic. The nodes participating in such platforms differ widely in their resource capabilities such as CPU speeds, bandwidth, and memory capacity. As a result,

resource discovery is often used in such large-scale systems to find suitable nodes that satisfy application requirements. Many existing resource discovery systems rely on the recently observed resource capacities of individual nodes to make their deployment decisions.

Disadvantage

However, resource allocation decisions based on current status of nodes have severe limitations in these systems, because of the presence of intra node dynamism in addition to the inter node heterogeneity. Individual nodes can have widely varying resource capabilities due to varying loads, network connectivity, churn, or user behavior. Dynamism in node-level resource capacities makes it difficult to deploy long-running services and applications that need consistent resource availability to ensure desired performance and avoid disruptions or migration overheads.

III. PROPOSED

We propose the notion of a resource bundle—a representative resource usage distribution for a group of nodes with similar resource usage patterns. A resource bundle employs two complementary techniques to capture the long-term resource usage behavior of a set of nodes: 1) resource usage histograms to provide statistical guarantees for resource capacities, and 2) clustering-based resource aggregation to achieve compact representation of a set of similarly behaving nodes for scalability. To handle the parameterization of the clustering algorithm, we present an adaptive algorithm to detect the amount of heterogeneity in the system and show its ability to adjust to fluctuations in an online fashion. Besides providing a scalable resource discovery mechanism to achieve stable application deployment, resource bundles can also be used for several other purposes in a large distributed system. Resource bundles can be used to easily find a group of nodes satisfying a common requirement. Resource bundles can also be used to find load hot spots: geographical regions in the distributed system with several nodes experiencing overloads due to reasons such as heavy demand for a popular resource in that region or locality-based application stresses. The identification of such hot spots can be used to inform decisions about application deployment or load balancing. Finally, resource bundles can also be used for auditing and accounting purposes, e.g., to determine the resource assignment of a distributed application running on multiple nodes, or to determine the spare capacity in an administrative domain.

Advantage

Resource bundles are able to provide high accuracy for resource discovery through the use of resource usage histograms, while achieving high scalability through aggregation.

IV. ALGORITHM

- Bundling algorithm
- Adaptive algorithm

(a) Bundling Algorithm

Once the input profiles have been used to create the graph, the bundling algorithm uses the graph to put the files into bundles. The algorithm works as follows:

1. Initially, each file is put in a separate bundle.
2. Edges whose weight is less than the *minimum edge weight* are discarded.
3. The edges are sorted according to the *edge comparator*.
4. The edges are processed one at a time, in the order determined in step 3. For each edge, if the files connected by the edge are not already in the same bundle, and if the number of files in the resulting bundle would not exceed the *maximum bundle size*, and if the resulting bundle would not exceed the *maximum bundle spread*, then the bundles containing the files are combined into a single bundle.
5. After all the edges have been processed, the files within each resulting bundle are sorted according to the *bundle sort comparator*.

Once the contents of the bundles have been determined, they are combined and compressed cumulatively. We evaluated both zlib (at the highest compression level) and Pack as compressed formats for the bundles.

(b) Adaptive Algorithm

An **adaptive algorithm** is an algorithm that changes its behavior based on the resources available. For example, stable partition, using no additional memory is $O(n \lg n)$ but given $O(n)$ memory, it can be $O(n)$ in time. As implemented by the C++ Standard Library, stable partition is adaptive and so it acquires as much memory as it can get (up to what it would need at most) and applies the algorithm using that available memory. Another example is adaptive sort, whose behavior changes upon the presorted ness of its input.

V. MODULE DESCRIPTION

- Main Module
- Node Module

MAIN MODULE

Main module contains a login page for authentication purpose. This module provides facilities for the following:

- ✓ Emptying clusters table
- ✓ Inserting cluster details into clusters table
- ✓ Search functionality to identify the node to identify prospective nodes and clusters they present in.
- ✓ Introspection functionality.

When search functionality is performed for a resource request is send to nodes of all clusters for that resource. Clusters, of which, the node(s), having the said resource is added in the search table along with the resource detail. When the search functionality is performed for a resource for a consecutive time, the request is sent to the clusters against the said resource in the search table to confirm the status of the resource in the clusters under scope. In both the cases, the resource details (content) are displayed in the system that is running the main module. The introspection functionality checks the status of a resource in the clusters in the search table and also additional clusters in the clusters table and accordingly updates the search table for the resource under inspection.

NODE MODULE

The node module represents node(s) of a cluster. The node module receives request from the main module for a resource and updates the status of the resource that is whether the resource is available in the node or not and if available it returns the detail of the resource to the Main module. To minimize the number of systems used two separate node module projects are installed in a system and the system itself made to represent as a cluster that contains those nodes. Each node module installed in the system is made to look into / work with separate folders in order to simulate the functionality of different nodes of a cluster.

VI. CONCLUSION

In this paper, we addressed the problem of scalable resource discovery in large-scale systems. The presence of node-level dynamism means that selecting nodes based only on recently observed capacities can lead to poor deployments resulting in application failures or migrations. However, existing resource discovery techniques rely only on recent observations to achieve scalability. We proposed the notion of a resource bundle that employs two complementary techniques to overcome the limitations of existing techniques: resource usage histograms to provide statistical guarantees for resource capacities and clustering- based resource aggregation to achieve scalability. We presented an adaptive algorithm that detects fluctuations in heterogeneity in order to parameterize the clustering-based resource bundles algorithm. Using trace-driven simulations and data analysis of a PlanetLab trace, we showed that resource bundles are able to provide high accuracy for statistical resource discovery, while achieving high scalability. We also showed that resource bundles are ideally suited for identifying group-level characteristics such as finding load hot spots and estimating total group capacity.

VII. ACKNOWLEDGMENT

This work was supported in part by the US National Science Foundation (NSF) CAREER Award CNS-0643505.

VIII. REFERENCES

[1] D. Anderson, "BOINC: A System for Public-Resource Computing and Storage," Proc. IEEE/ACM Int'l Workshop Grid Computing (GRID), 2004.

[2] V. Lo, D. Zappala, D. Zhou, Y. Liu, and S. Zhao, "Cluster Computing on the Fly: P2P Scheduling of Idle Cycles in the Internet," Proc. IEEE Fourth Int'l Conf. Peer-to-Peer Systems, 2004.

[3] Grid2: Blueprint for a New Computing Infrastructure,I. Foster and
C. Kesselman, eds. M. Kauffman, 2004.

[4] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S.Shenker, "Making Gnutella Like P2P Systems Scalable," Proc. ACM SIGCOMM, Aug. 2003.

[5] B. Cohen, "Incentives Build Robustness in Bittorrent," Proc. First Workshop the Economics of P2P Systems, June 2003.

[6] S. Guha, N. Daswani, and R. Jain, "An Experimental Study of the Skype Peer-to-Peer Voip System," Proc. Int'l Workshop Peer-to-Peer Systems (IPTPS), 2006.

[7] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "PlanetLab: An Overlay Testbed for Broad-Coverage Services," ACM SIGCOMM Computer Comm. Rev., vol. 33, no. 3, pp. 3-12, July 2003.

[8] A. Iamnitchi and I. Foster, "On Fully Decentralized Resource Discovery in Grid Environments," Proc. IEEE/ACM Int'l Workshop Grid Computing (GRID), 2001.

[9] P. Yalagandula and M. Dahlin, "A Scalable Distributed Information Management System," Proc. ACM SIGCOMM, 2004.

[10] B. Chun, J.M. Hellerstein, R. Huebsch, P. Maniatis, and T. Roscoe, "Design Considerations for Information Planes," Proc. Workshop Real, Large Distributed Systems (WORLDS '04), Dec. 2004.

[11] J.M. Schopf, "A Practical Methodology for Defining Histograms for Predictions and Scheduling," NU technical report, 1999.

[12] S.Zhong http://www.cse.fau.edu/~zhong/software/index.htm, 2009.

[13] A. Gupta, D. Agrawal, and A.E. Abbadi, "Distributed Resource Discovery in Large Scale Computing Systems," Proc. Int'l Symp. Applications and the Internet (SAINT), 2005.

[14] R.V. Renesse, K.P. Birman, and W. Vogels, "Astrolabe: A Robust and Scalable Technology for Distributed System Monitoring, Management, and Data Mining," ACM Trans. Computer Systems, vol. 21, no. 2, pp. 164-206, 2003.

[15] J. Cappos and J.H. Hartman, "San Fermı´n: Aggregating Large Data Sets Using a Binomial Swap Forest," Proc. USENIX Symp. Networked Systems Design and mplementation (NSDI), 2008.

[16] J. Mickens and B. Noble, "Exploiting Availability Prediction in Distributed Systems," Proc. USENIX Symp. Networked Systems Design and Implementation (NSDI '06), May 2006.