Symantec – RIPS 2015 Project Proposal

Incremental Set Cover Computation

Introduction

BigData is an important keyword today mainly because of the great data deluge that we are witnessing. The rate and amount at which data that is being generated today is simply staggering. Nowadays, in order to efficiently handle the problem of extreme data volumes and velocity, data is spread out on to large number of machines and jobs are executed in a distributed fashion. These machines also contain replica of the data items to guarantee certain data availability and fault tolerance in the presence of machine failures.

In this distributed data setup, when a job or a query arrives, it is sent to all the machines that contain the data items needed for its execution. But to minimize the overall distributed overheads, considering that data is replicated across machines, it makes sense to dispatch a query to minimum number of machines that contains all the data items needed for its execution.

Modeling

We can model each machine m_i as a set s_i and data items stored in machine m_i as elements in the set s_i. Then given a query q_i accessing a set of data items, finding the minimum number of machines containing the data items needed for the query can be modeled as a set cover problem.

Set Cover problem:

The Set Cover problem is: Given a set of elements $E = \{e_1, e_2, ..., e_n\}$ and set of m subsets of E, $S=\{S_1, S_2, ..., S_n\}$, find a collection C of minimum number of sets from S such that C covers all elements in E.

Set Cover problem is NP-Hard problem. Popular way of solving Set Cover problem is to follow a greedy approach. In this approach at each stage, choose the set that contains the largest number of uncovered elements until all the required elements are covered.

Problem:

In the context of queries and machines, given several millions of queries, running a greedy set cover algorithm for each query to find the minimum number of machines for query routing can be very expensive. Question is, in order to speed up the routing of queries, can we reuse the Set Cover computations across the queries while subjecting to certain constraints. Are there any guarantees that can be provided regarding the effectiveness and optimality of the solution?

Constraint: One of the constraints can be, each machine has a load capacity on it. Load can be defined as number of queries touching a machine. Once the machine exceeds its load limit, it cannot be accessed.

Expected Outcomes:

- Any reasonable assumptions about the query workloads or sets that would help in arriving at certain guarantees
- Algorithm for incremental Set Cover computation
- Proofs, optimality discussions
- If possible, experiments comparing new approach with traditional Set Cover approach.

Related Readings:

http://en.wikipedia.org/wiki/Set_cover_problem

Vazirani, Vijay V. (2001), *Approximation Algorithms*, Springer-Verlag, ISBN 3-540-65367-8

K. Ashwin Kumar, Abdul Quamar, Amol Deshpande, and Samir Khuller. 2014. SWORD: workload-aware data placement and replica selection for cloud data management systems. *The VLDB Journal* 23, 6 (December 2014), 845-870. DOI=10.1007/s00778-014-0362-1 http://dx.doi.org/10.1007/s00778-014-0362-1

Abdul Quamar, K. Ashwin Kumar, and Amol Deshpande. 2013. SWORD: scalable workload-aware data placement for transactional workloads. In *Proceedings of the 16th International Conference on Extending Database Technology* (EDBT '13). ACM, New York, NY, USA, 430-441. DOI=10.1145/2452376.2452427 http://doi.acm.org/10.1145/2452376.2452427

Michael R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.