

BRAHMA: Planning Tool for Providing Storage Management as a Service

Sandeep Uttamchandani, Kaladhar Voruganti, Ramani Routray, Li Yin+, Aameek Singh*, Benji Yolken**
 IBM Almaden Research, +UC Berkeley, *Georgia Institute Of Technology, **Stanford University

I. OVERVIEW

Storage management is becoming the largest component in the overall cost of storage ownership. Most organizations are trying to either consolidate their storage management operations or outsource them to a storage service provider (SSP) in order to contain the management costs. Currently, there do not exist any planning tools that help the clients and the SSPs in figuring out the best outsourcing option.

In this paper we present a planning tool, Brahma, that specifically addresses the above mentioned problem, as Brahma is capable of providing solutions where the management tasks are split between the client and SSP at a finer granularity. Our tool is unique because a) in addition to hardware/software resources, it also takes human skill set as an input b) it takes planning time window as input because plans that are optimal for a given time period (e.g. a month) might not necessarily be the most optimum for a different time period (e.g. a year) c) it can be used separately by both the client and the SSP to do their respective planning d) it allows the client and the SSP to propose alternative solutions if certain input service level agreements can be relaxed.

We have implemented BRAHMA, and our experiment results show that there definitely are cost benefits that one can attain by having a tool with the above mentioned functional properties.

II. INTRODUCTION

The cost of storage management is becoming the largest portion in the overall cost of storage ownership. Most organizations are finding that they need to hire more administrators as the amount of storage consumed by the organization increases. Storage administrators have to provide support for provisioning, disaster recovery, compliance, performance, security, and planning requirements for an application/business. Thus, there is a limit on the amount of storage that can be managed by a single administrator. It is increasingly becoming difficult for most organizations to find and retain experienced system administrators.

Most organizations are trying to address the above problem by a) carefully dividing up an administrator's responsibilities into high-skilled and low-skilled ones and hiring more low skilled administrators while sharing high skilled administrators across multiple locations b) outsourcing storage hosting and storage management tasks.

The tasks of a typical storage administrator can be divided and performed by a) *system planners* b) *system operators* and c) *system administrators*. System planners perform skill

intensive tasks such as capacity planning, disaster recovery planning, security infra-structure design and bottle-neck analysis. Their services are invoked on a per need basis. System operators install new hardware, software and firmware. They also typically assist with wiring, cooling and other installations related tasks. Their services are also utilized on a per need basis. Finally, system administrators perform daily storage management tasks such as resource monitoring, first level problem resolution, storage provisioning, and resource access control. Thus, most organizations are trying to share expensive system planner skill set across multiple sites, and hiring relatively inexpensive system operators and administrators on a per-site basis.

Organizations are also tackling the above mentioned problem by outsourcing their storage hosting and/or storage management operations to a storage service provider (SSP). The existing options for storage outsourcing can be categorized in the following manner:

- *Data out, Management out*: This is the most common outsourcing model where the data center is hosted and managed remotely by the storage service provider [7], [1].
- *Data in, Management out*: In scenarios where the customers may not be comfortable with having their data stored remotely, customers prefer the model where the data is hosted locally, but the management is done by the SSP (either remotely, or by having their personnel present at the customer site).
- *Data out, Management in*: In the this model, the data center is hosted remotely, but the management is done in-house. This is not a common scenario.
- *Hybrid Management*: This is a variation of the previous two approaches. In this approach the storage management tasks are split between the storage service provider and the customer.

Over time, as an organization's storage needs and in-house skill set evolve, different storage outsourcing models become more attractive. Therefore, there is not a single "one-size fits all" solution that can be advocated for all the customers. Similarly, the SSPs want to provide different classes of services to different customers for different price. Furthermore, both the customer and the storage service provider are trying to minimize their respective costs, and in some cases, their cost optimization goals might prompt them to advocate different storage outsourcing solutions. It is not easy for both customers and storage service providers to come up with their respective

plans as to what is the most optimum course of action due to the presence of numerous applications requirements (performance, availability, security), devices, business policies, and government regulations that one has to satisfy. Manual planning techniques that are being employed by customers and storage service providers are both time consuming and error prone. Furthermore, plans have to be constantly updated to cope with the constant changes due to resource consumption growth and new business and government requirements.

In order to address the above set of problems, in this paper we propose a web services based multi-site storage outsourcing planning tool that has the following key features:

- **Temporal Planning:** We present a planning infrastructure that can take different optimization time windows as input. For example, the planner output plan that is the most suitable for a one month time window might not be the most cost effective plan for a one year time window.
- **Integrated Human/System Planning:** When proposing storage outsourcing solutions, in addition to considering storage hardware and software resources, it is very important to consider the available human skill set. In this paper, we take human skills into account when proposing storage outsourcing solutions.
- **Generation of multiple plans:** The proposed planning infrastructure allows the SSP to propose solutions with different cost points if certain conditions are relaxed. For example, the SSP can indicate to the customer that if they are willing to downgrade their performance or availability requirements, the customers can get solutions that have better cost points. Thus, the tool outputs multiple solutions with different cost points.

III. SYSTEM OVERVIEW

This section provides an overview of BRAHMA. It discusses the inputs, outputs and the key components of the planning tool in the following subsections. The clients and SSPs can perform remote storage management tasks using a standard TCP/IP connection. The actual data flow between the clients and SSPs via either IP networks (using iSCSI) or via FCP (SCSI over Fibre Channel). BRAHMA can either reside on a separate physical server, or it can share the same physical server with a storage resource manager software packages like EMC control centre, IBM TPC or HP APPIQ.

A. Planner Input

Brahma obtains the following different types of input:

Client Demand: Client demand consists of SLOs, penalty agreements, and the client's willingness to relax certain SLOs. Clients specify capacity, performance, availability (includes disaster recovery), future growth, and security related SLOs. To assist the clients in defining their requirements, BRAHMA provides pre-defined application templates that vary in performance and availability requirements such as OLTP, scientific, and email applications to capture a diverse set of requirements.

Client and SSP Resources: In this paper we are dealing with both hardware resources as well as human resources. BRAHMA can either directly query the resources in concern or indirectly obtain the data from storage resource manager databases (which, in turn, directly query the resources using SMI-S [17] and SNMP protocols). This discovered hardware resource information is used as input to BRAHMA. Currently, we do not have an automatic way of discovering and classifying human resources, therefore, we provide templates which classify human skill level into different buckets. In this paper, we have classified human skill level into high, medium and low based on the administrative tasks an administrator can perform, and the number of man-hours he takes to complete a particular task.

Business Policies: Clients can specify many types of organizational, government regulations and plan optimization input criteria. For example, clients can specify requirements such as the exclusive use of a storage controller for a particular user/department, or a government regulation that requires that there should be no duplicate copies of data, or that the data needs to be physically removed from devices on a particular date etc. Clients can also specify optimization time window such as the number of months or year as an input.

B. Planner Components

The framework for BRAHMA, as shown in Figure 1, consists of three key modules:

SLO Parser: Typically clients describe their demands and available resources in plain English. These are mapped into our *resource description* and *resource requirement* formats by the SLO Parser. The output of the parser feeds into the Policy Manager as well as constraints for the Optimizer namely the penalty function and optimization window.

Policy Manager: This module maps customer SLO requirements to the list of candidate hardware and human resources that can satisfy those requirements. This module will be developed by domain experts and pre-packaged with the BRAHMA as a customizable *Policy Manager*. This step generates a list of candidate resource locations (hardware and human) for the customer SLOs; it embeds the following domain expertise:

- Mapping of an SLO attribute to hardware resource requirements. For example, a recovery window SLO of 5 minutes will require a network bandwidth proportional to the size of the dataset between the customer site and the backup device
- Mapping of an SLO attribute to human tasks, skill-level and manhours.

This step also helps prune the search space for the optimizer by eliminating storage devices and human resources that cannot be used to service an SLO.

Constraint-based Optimizer : This module takes the candidate resource list and generates an optimal allocation of

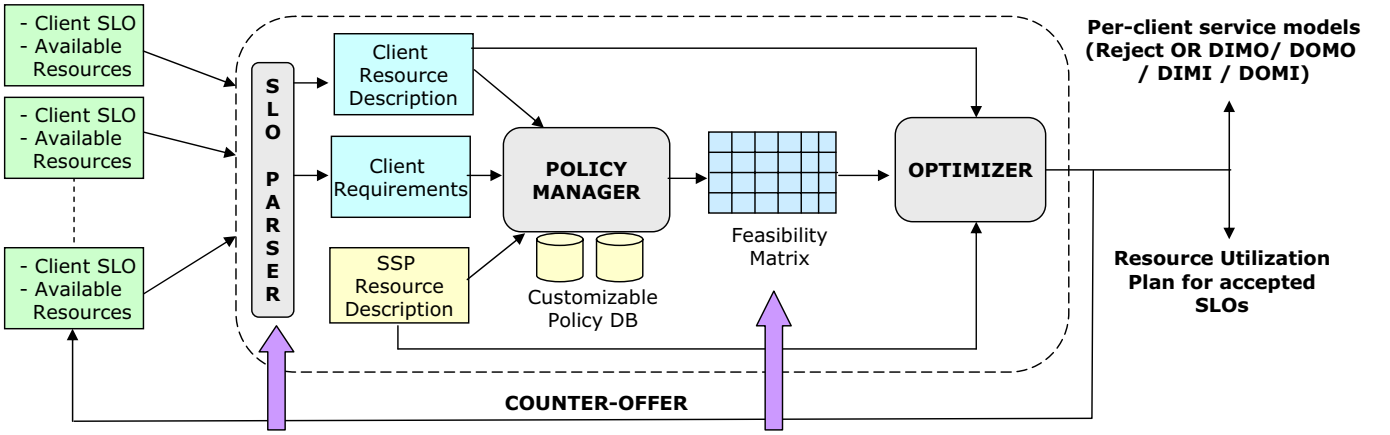


Fig. 1. Design of BRAHMA

resources for a given customer SLO. We formulate the optimization as a 0/1 multi-knapsack problem [16], which is a well-known NP-hard problem. Figure 1 shows the overall design of BRAHMA.

C. Planner Output

The output of BRAHMA can contain either a single plan or multiple plans. When the planner outputs multiple plans, it typically uses the additional plans to inform the user the cost benefit if one or more SLO requirements were relaxed.

- **Data Placement Information:** This information describes on what site and storage location a particular type of user data (like log data or temporary data) should be placed.
- **Administrator Placement Information:** The administrator placement information describes what type of system administrator (with the appropriate skill level) should be placed at which location.
- **Solution Cost:** The solution cost indicates the cost in dollars with respect to the overall setup cost. In this paper we use a simple cost model where we assign cost for both different types of storage as well as cost for different administrator tasks. We obtained these costs after surveying the cost models of various storage service providers.

Next, we delve deeper into the different components of BRAHMA.

IV. SLO PARSER

The input to the SLO Parser is a combination of customer requirements, resource information, and business policies. The SLOs are typically in plain English or some human-readable format. The hardware resource information is typically a collection of data-sheets or from management tools. This section gives details of the internal representation of this information as generated by the SLO Parser.

A. Client Requirements

The SLO requirements are translated into corresponding measurable parameters. Table I shows the parameters used by the BRAHMA prototype implementation.

category	SLO parameters
Capacity	Gigabytes
Performance per gigabyte	throughput, latency
Availability	9s, MTTF, MTTR
Disaster recovery	RTO, RPO, Distance, Application Impact, Backup Window
Compliance	HIPAA, SOX
Future Growth Per Month	Capacity Growth, Performance Growth, Port Count Growth
Time to provision	YY:MM:DD:HH:MM:SS
Provisioning granularity	GBytes
Security	access control, on disk, authorization, on wire, physical security, separate setup,
others	customer support (24), Problem turnaround time
Reliability	CRC, Checksum, LRC

TABLE I
EXAMPLE SLO PARAMETERS

In BRAHMA, the given customer SLO is internally represented as three sub-SLOs corresponding to the level of service required for the customer's regular data, archival/backup data, and compliance data.

For each of the SLO attributes in Table I, it is possible for the customer to specify whether the attribute is *flexible* – this allows BRAHMA to relax the corresponding constraint and generate multiple allocation plans.

B. Hardware resource description

The information about the hardware attributes can either be generated manually or using SAN management tools such

as EMC Control Center [2] and IBM Total Productivity Center [4]. In the BRAHMA prototype, a storage device is configured with attributes, such as `hot code upgrade`, `Maximum Capacity` and `Encryption Support`.

C. Human resource description

The human resource model consists of the following parameters:

- Skill Level: This is categorized into high, medium, and low
- Expertise: This is based on the *task groups* supported by the administrator. In BRAHMA, the task groups are *planning* (performance, disaster recovery, security), *monitoring* (performance and system status), and *operations* (firmware and system installation, provisioning, compliance).
- Yearly man-hours: The number of man-hours for which the administrator will be available
- Hourly rate: The dollar cost associated with the administrator

V. POLICY MANAGER

As shown in figure 1, the Policy Manager takes as input the customer requirements, the hardware and human resources of the customer and the SSP to generate the following:

- A feasibility matrix of the candidate hardware resources that can satisfy the customer SLO. The rows in the matrix are the data types (regular, backup, compliance) of all customer SLOs and the columns are the hardware devices.
- A feasibility matrix of the candidate human resources that can satisfy the administrative tasks associated with the customer SLO. The rows of the matrix represent the six task groups for each SLO (performance, capacity, availability, disaster recovery, compliance, security) while the columns represent the available human administrators on the customer as well as SSP sites
- An enumeration of human manhours required for each administrative task group associated with the SLO

In data-centers today, the matrix generation is done informally and manually – automating this process with the Policy Manager reduces errors and planning time, and combined with optimization (described later) it will significantly improve resource usage and SSP costs.

Internally, the Policy Manager uses two sets of configurable policies to generate the output:

Policies for mapping SLOs to hardware attribute constraints: These are rules that establish whether a particular SLO can be fulfilled by a specific resource. It can be thought of as a boolean function $feasible(SLO, Stg - Resource)$ which indicates whether $Stg - Resource$ is *feasible* for the SLO . For example, for compliance data only WORM storage devices are candidates and thus will return *true*. Additionally, the customer may specify the SLO as a level of service such as bronze, gold, platinum. For example, a policy rule that maps the SLO requirement of platinum for availability to a storage controller

which has all the following attributes: 1) Fault Tolerant & Dual redundant; 2) Hotswap RAID Controller Cards; 3) Battery Backup Units; 4) Non-disruptive hardware and software code load updates (Hot Code Upgrade); 5) Multi-pathing device driver

Policies for mapping SLOs to human tasks, skill level and manhours: These rules map the administrative tasks required to meet an SLO. For example, servicing a 10 TB SLO with security-enabled and full disaster recovery capability would require planning administrators, performance and security analysts and DR experts. This identification of tasks is based on a *customizable* set of best practices and rules-of-thumb that are commonly used in data centers. Next, based on task identification all administrator who have the necessary skills to service an SLO are considered *feasible* for the specific SLO and added to the candidate list.

Generating the acceptability matrix is intuitively a *matching* problem, where the set of SLO requirements need to be matched to appropriate hardware and human resources. Here, we briefly illustrate an example work flow of the Policy Manager from starting with a customer SLO and generating a feasibility matrix. Consider a SSP with two sites A and B with high-end storage controllers of 1 PB capacity in site A and low-end storage devices with 500 TB capacity and backup tapes of 100 TBs in site B . Additionally, site A has high skilled performance and availability analysts and medium skill monitoring administrators, while site B has high skill planning and security analysts. Now, assume a client SLO which requires 50 TB of regular data, out of which 10 TB needs to be backed up on tapes. The client requires high throughput requirement and full security with at-rest and on-wire encryption.

The policy manager first looks at the storage resource requirements for the client. Based on the capacity requirements, storage controllers at both site A and site B are candidates. However, using the storage resource policies ($feasible()$), the high throughput requirements of the client might eliminate devices at site B as they are low end devices. For the backup 10 TB data, only backup devices available at site B are candidates. Next, the policy manager uses the task identification rules to break down the management requirement into individual tasks and number of man hours required for that task. Then, it selects candidates for fulfilling those tasks. For example, using the thumb rule of 1 high-skill man hour of planning for provisioning every 10 TB capacity, the policy manager chooses the administrators at site B as candidates and assigns 5 man hours requirement. Similarly for other task groups.

This process is repeated for all client SLOs. As a result of this analysis, a comprehensive feasibility matrix can be created which has all storage devices and human resources as columns and client SLOs as rows. The cell (i, j) is marked *yes* based on whether a resource j is *feasible* for the SLO i . Based on this feasibility matrix, the optimizer will try and find the best allocation of storage and administrator resources to satisfy client SLOs. The optimizer is discussed in the next

section.

VI. OPTIMIZER

BRAHMA formulates the decision-making as a constraint optimization problem, where the *objective function* is to maximize the summation of the cost difference that the SSP can deliver to all customers; the *variables* are the allocation decision for all data (e.g., where to store the data) and the management tasks (where the management tasks will be performed); the *constraints* are based on the input information. In addition, to account for the future growth potential, BRAHMA introduces a lookahead time window and estimates the possible clients' requirement using the specified *growth probability* and *growth percentage*. The total cost savings are the sum of the cost savings over the entire lookahead window. The run-time provisioning is performed by appropriately balancing the benefit (saved penalty cost) and the cost (hardware purchasing and maintenance cost).

For a given SLO, the allocation of hardware resources is done in units of the data types (i.e., regular data, backup data, and compliance data). One straight-forward solution is to plan all three together in a single optimization problem (e.g., determine the location for $3N$ data sets, where N is the number of customer). However, this formulation fails to capture many real world constraints that exist in such environments, e.g. the backup data often can not stay together with its regular data. Such constraints introduce correlations between planning decisions and grow the solution space exponentially. BRAHMA assumes that the regular, backup and compliance data need to be placed on different devices and breaks the resource optimization problem into three independent sub-problems.

The allocation of human resources is in terms of available manhours for each administrator and the number of manhours required for different administrative tasks associated with the SLO. Finally, the optimization for allocating hardware and human resource in BRAHMA can be done independently. This is possible because the management of hardware can be done remotely and is not required to be local. In our prototype, we choose to plan for management tasks independently. *The details of the optimizer are presented in the appendix.*

A. Generating Multiple Offers

The primary design objective of BRAHMA is to create an extensible framework that can capture most of the intricacies of today's human driven SLO pricing process. An important component of this process is *counter-offers* where after considering a client's SLO, the SSP generates additional plans with different costs for certain relaxations to the client SLO. For example, the SSP might offer the client a better price if it was willing to have a lower security requirement (such as at-rest encryption instead of on-wire encryption). Such counter-offers are motivated by lack of availability of resources to handle the task (e.g. no available high-skill security administrators with the SSP at that time). Offering such alternatives to the client is extremely common, and in fact expected.

While SLOs relaxations introduce significant complexity, the performance of our approximation algorithm (see Section

VII) makes it a little easier to account for these counter-offers. We allow the client to specify SLO parameters that are *flexible* – internally, the optimizer is run several times, changing one or more parameters at a time. This allows us to retain the core optimization algorithm while creating counter-offers. Clearly our solution has the limitation on the number of parameters that can be relaxed. Additionally, we have tried several techniques that preserve state during multiple optimization cycles which provide marginal run-time benefits.

VII. EXPERIMENTAL EVALUATION

The experimental evaluation of BRAHMA consists of the following tests:

- First, **Sanity Check**. The goal of this test is to examine BRAHMA's ability to adapt to simple changes in system parameters such as cost functions, penalty functions, optimization time window and SLO requirements.
- Second, **Counter offers**. In this test, we examine the BRAHMA's ability to generate flexible allocation plans by relaxing SLO requirements.
- Third, **Performance Comparison**. In this test, we compare the performance of the BRAHMA with two commonly used, simplistic heuristics- all in-house and all outsourcing.

In the rest of this section, we describe our configuration for the BRAHMA prototype and present experimental results for the tests listed above.

Configuration of the BRAHMA Prototype

The BRAHMA prototype is implemented in Java. The SLO Parser implements XML-based parsers for the client requirements, business policies, and human resource model. The hardware resource information is imported using JDBC calls to the monitoring database of a commercial storage management suite. The Policy Manager implements the *feasible* function for the hardware and human resources as described in Section V. The optimizer implements the approximation algorithm described in appendix and outputs a final data and management allocation plan. To generate counter offers, the optimizer then modifies the SLO requirements of the clients (according to clients' willingness to relax them) and feeds them to the policy manager to start the process of finding alternative allocation plans.

Because of the large number of parameters required to specify a testing scenario, we implemented a **configuration generator** that assists in the creation of realistic testing scenarios. The configuration generator maintains the standard configuration for a number of different client applications, SSP storage devices and SSP administrators. Each application type, for instance, is associated with a number of measurable SLO parameters (Table I) as well as a capacity and throughput range. Due to space limitations and the large number of parameters involved, we omit most of the details. As an example, Table II shows the capacity and throughput ranges for regular data of each application type as well as the feasible device tiers that result when the associated application parameters are passed through the policy manager.

Application Types	Capacity	Throughput	Feasible Devices
Archival	100GB-100TB	10MB/s - 10GB/s	Tier 1-5
Compliance	3TB-1PB	100MB/s-32GB/s	WORM
OLTP Standard	100GB-100TB	10MB/s-10GB/s	Tier 1
OLTP High	100GB-100TB	25MB/s-25GB/s	Tier 1
Scientific Computing	100TB - 10PB	1GB/s - 100GB/s	Tier 1-4
SMB	100GB-100TB	100MB/s-10GB/s	Tier 1-3

TABLE II
APPLICATION SLO REQUIREMENTS

The physical characteristics of the storage devices created by the generator are derived using the data sheets provided by hardware vendors. In addition, the device cost functions are set based on a cost/benefit analysis from a major storage service provider, which estimates that the *burdened* cost (including the hardware and software cost, floor space, etc.) of Tier 1 storage is roughly \$4.20/GB/month. The cost functions of the other tiers are estimated by scaling this number (e.g., Tier 2 0.8X, Tier 3 0.6X, Tier 4 0.2X, Tier 5 0.1X and WORM 0.5X). Table III lists the types of storage devices and their cost ranges as implemented by the BRAHMA prototype.

Tier	Products	Cost [\$/GB/Month]
Tier 1	EMC Symmetrix, Hitachi Lightning, IBM DS8000	4.2±0.4
Tier 2	3PAR InServ S800, Hitachi Thunder, IBM DS6000	3.3±0.3
Tier 3	3PAR InServ S400, IBM DS4000	2.5±0.3
Tier 4	Adaptec JBOD Systems, IBM DS400	0.8±0.2
Tier 5	ADIC Scalar, HP StorageWorks VLS, IBM TS1120 Rewritable	0.4±0.1
WORM	EMC CENTERA, HP StorageWorks Ultrium, IBM TS1120 WORM Tape	2.1±0.2

TABLE III
STORAGE DEVICES CONFIGURATIONS

For the cost of administrators, we assume that a high skilled administrator in the United States has an hourly rate of \$55-\$70, medium skilled administrators are in the range of \$40-\$55 per hour and low skilled administrators cost \$30 to \$40 per hour. For administrators in developing countries (e.g., China and India), we apply a scaling factor of 15% to account for the lower pay scales in those locations.

Using the default configuration as the base, testing scenarios are easily generated by specifying different levels of detail for the device types, application types, the number of SSP sites, administrators, and clients.

A. Sanity Check

In this test, we evaluate BRAHMA’s ability to adapt to changes to various input parameters. For this test, the configuration generator is to first create a “default” configuration,

and then modify this configuration one parameter at a time to observe the results generated by BRAHMA.

The details of the default configuration are as follows:

- **SSP resources and sites.** There are four SSP sites (China, India, US1, US2) in the system. In total, we have 500TB Tier 1, 255TB Tier 2, 380TB Tier 3, 10TB Tier 4, 45TB Tier 5 and 250TB WORM storage, and 18 high skilled, 36 medium skilled and 70 low skilled administrators with different skill sets. The locations and exact cost functions (e.g., administrators’ hourly rates) of these resources are randomly determined according to the ranges specified in the generator. We skip a more detailed discussion of the SSP resources at each site because of space constraints.
- **Client Configuration.** We have six clients in the system. Their application types and specifications are given in Table IV.
- **Other Constraints.** The optimization time window is set to be six months unless otherwise specified. Additionally, a “penalty” represents the dollar amount that the SSP will be charged for every GB of capacity it has promised to the client but cannot provide. The optimizer is set to run for 20 iterations and the best plan is selected.

For the default configuration, BRAHMA generates the following data allocation plan as shown in Table V.

Client ID	Regular Data	Backup Data	Compliance Data
Client 1	N/A	Tier 3 (US1)	N/A
Client 2	Tier 3 (China)	Tier 3 (China)	N/A
Client 3	Tier 1 (US1)	Tier 2 (India)	WORM (China)
Client 4	Tier 2 (India)	Tier 3 (US1)	WORM (US2)
Client 5	Tier 1 (India)	Tier 1 (US2)	N/A
Client 6	Tier 1 (India)	Tier 2 (US1)	WORM (China)

TABLE V
Data Allocation Plan For the Default Configuration.

Compared to complete in-house solution with an estimated cost of \$7,670,420, the above allocation decision saves \$4,827,968. The latter cost savings includes a \$166,550 initial provisioning cost on the SSP side. On the human administration side, BRAHMA saves 49.1% of cost by outsourcing a part of the management tasks. The details of the recommended management plan are omitted for brevity.

In the remainder of this section, we modify parameters in the above default configuration and discuss the changes that result in the BRAHMA’s plan.

1) *Test 1: Impact of cost functions:* In this test, the cost of the Tier 1 device in India is increased from \$3.9/GB/month to \$4.2/GB/month. As a result, the jobs originally on these devices (namely the regular data of Clients 5 and 6) are now placed on the US1 site. The total cost savings drops to \$4,783,253 (from \$4,827,968 in the default setup). Similar tests done on the human costs yield similar results.

2) *Test 2: Impact of penalty functions:* In this test, the penalty functions for the regular and backup data for Client 2 are increased from \$3/GB to \$10/GB. In both the default and modified cases, these jobs are placed on Tier 3 storage in China. As a result of increasing the penalty, however,

Client ID	Application Type	Capacity	Throughput	Variability	Future Growth [per month]	Penalty [\$/GB]
Client 1	Archival	50TB	2GB/s	10%	5%	2
Client 2	SMB	15TB	1GB/s	30%	20%	3
Client 3	OLTP Standard	50TB	5GB/s	20%	10%	4
Client 4	Scientific	100TB	10GB/s	10%	10%	3
Client 5	OLTP High	40TB	10GB/s	10%	5%	4
Client 6	OLTP Standard	20TB	10GB/s	30%	15%	4

TABLE IV
CLIENT CONFIGURATIONS

the amount of provisioning at that location increases from 47.52TB to 53.44TB.

3) *Test 3: Impact of client’s capacity and throughput:* In this test, we change a client’s SLO parameters and examine the BRAHMA’s ability to adapt. Specifically, we reduce the capacity requirement of Client 4 from 100TB to 10TB and its throughput from 10GB/s to 1GB/s. Due to the reduced capacity and throughput requirements, Client 4 can now be satisfied at lower tier (Tier 5) storage devices. The BRAHMA correctly captures this change and puts the client’s regular and backup data on tier 5 devices in China and the US, respectively.

4) *Test 4: Impact of the optimization time window:* The goal of this test is to check the BRAHMA’s ability to account for temporal behavior. This is done by changing the optimization time window from six months to one month only, while keeping all other settings the same. As a result, the regular data of Client 4 are placed on the Tier 3 storage in China, as opposed to the original Tier 2 storage in India.

B. Multiple Offers

In this test, we demonstrate the BRAHMA’s ability to generate multiple allocation plans when a client is willing to be flexible on one or more of its SLO requirements. Specifically, we randomly generate a testing scenario where Client 7 (a SMB application) is willing to relax the reliability requirement on its regular data (e.g., they are non-profit with a low budget, and short outages are an annoyance but will not fundamentally hurt their operations). In this situation, BRAHMA gradually moves the reliability requirements from “PLATINUM” to “SILVER” and to “BRONZE”. As a result, the regular data of Client 7 is moved from Tier 1 to Tier 2, and then finally to Tier 3 storage. At the same time, the cost savings grow from \$121,520 to \$283,549 to \$445,578. BRAHMA returns all of these options to the client so it can study the trade-offs between reliability and the corresponding cost to support.

C. Comparison of BRAHMA output with Cookie Cutter techniques

In this test, we compare the BRAHMA with two simple, commonly used strategies: storing and managing all operation in-house (“all in”) and outsourcing everything (“all out”) on the other extreme. In the latter case, each job is placed on the least loaded device among those which are feasible. For each strategy, we measure the *final cost*, defined as the sum of the cost of jobs served by the client site and the cost of

jobs on the SSP sites, and the *decision time*. In addition, we focus on scenarios that do not require provisioning because the provisioning decision twists the final cost and makes the relative advantages of each allocation strategy unclear.

Scenario	Strategy	SSP Cost	Client Cost	Total Cost	Decision Time (s)
Scenario one	BRAHMA	0	483965	483965	0.31
	All In	0	483965	483965	0
	All Out	2723839	0	2723839	0.28
Scenario two	BRAHMA	4285205	0	4285205	0.31
	All In	0	6373392	6373392	0
	All Out	4285205	0	4285205	0.30
Scenario three	BRAHMA	1433615	568658	2002273	0.28
	All In	0	2803096	2803096	0
	All Out	30990033	0	30990033	0.33

TABLE VI
COMPARISON OF BRAHMA, ALL IN-HOUSE AND ALL OUTSOURCING APPROACHES

We start with three representative scenarios. In *scenario one*, the clients all have cheaper costs than the SSP (e.g., all clients are overseas). In this case, the optimal allocation plan should be all in-house. In *scenario two*, the hardware and operating costs on the clients are more expensive than the SSP costs (due to economies of scale) and the SSP sites have homogeneous resources. As such, the best result is clearly to outsource all jobs. In *scenario three*, both the clients and the SSP have a heterogeneous mixture of devices, applications, and costs. In this case, our tool should perform better than either of the simple heuristics discussed above because it allows for a hybrid of inhouse and outsourcing based on cost optimization. Table VI lists the total cost and the decision time for each option. Note that, for the all in-house strategy, the decision is fixed (always on the client site) and thus the decision making time is effectively zero.

VIII. RELATED WORK

BRAHMA handles both the outsourcing decision making and the capacity planning. In this section, we summarize the existing efforts for both problems.

A core component of our work is optimized utilization of resource for Storage Service Providers (SSPs) and is thus related to various capacity planning tools. Tools like Minerva [8], Hippodrome [9], Ergastulum [10] plan for storage environment setup and disaster recovery based on analysis of the workloads.

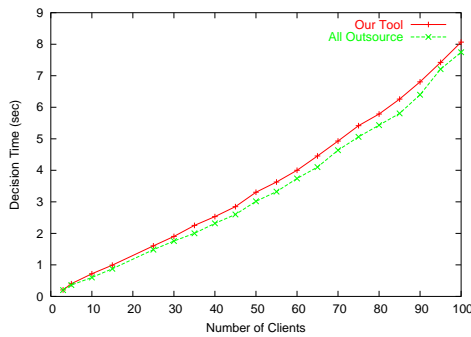


Fig. 2. Decision Time of BRAHMA and All outsourcing

However, their optimization is within the scope of a single SAN and they do not account for any management costs. In contrast, BRAHMA is a end-to-end framework mapping clients and their SLO requirements to SSP resources (both storage and human) optimized for maximum cost savings.

With the growing total cost of ownership, many organizations are outsourcing their storage hosting and management tasks. The first question to ask is, to outsource or not. Various studies articulate critical elements that need to be considered when answer this question. [13] studies 5000 firms and lists risks involved in outsourcing. They conclude that outsourcing can be dangerous for a company due to it's dependence on a service provider over time. To address this problem, they suggest alternatives such as having multiple service providers or selective outsourcing. Our tool can fulfill these requirements because it allows the clients to selectively outsource jobs and choose among multiple SSPs. Another empirical study [15] of application service providers find that the basic reason for companies to outsource is to save cost and maximize ROI. They identify three main reasons for clients to adopt the application service provider concept—core competence, lack of skilled personnel and the organization's overall strategy. Our tool provides a framework to account for these factors. It eases the decision process and helps clients perform the *what-if* analysis.

Currently, many organizations hire a consulting firm specializing in outsourcing IT operations [3] to make the outsourcing decision. Such firms rely heavily on experienced consultants who make decisions based on industry best practices and their experiences. There also exist several automatic planning techniques for making the outsourcing decisions. The Return-On-Investment (ROI) calculators [6], [5] use some static rules of thumb to estimate whether outsourcing will yield significant benefits or not. Other works are analytical model based solutions. One interesting work applies an Analytic Hierarchy Process (AHP) [11] to develop a model to make outsourcing decisions. Their model identifies activities to outsource to maximize ROI and they also help to choose appropriate outsourcing methodology (e.g. vendors and contracts to adopt). However, this work is not specific to storage services and thus, cannot leverage on domain expertise—a critical element in our design. In addition, they cannot account for human and administrative resources while planning.

IX. CONCLUSION

In this paper, we propose a multi-site storage outsourcing planning tool, BRAHMA. It takes into account the clients SLOs, available hardware/software resources, management resources and the optimization time window, and outputs the data and management allocation plan that leads to maximum cost savings for the clients and SSPs. BRAHMA has a policy manager that prunes the candidate search space and formulates the problem as a constraint optimization problem. BRAHMA is implemented and evaluated using a Java-based implementation. Our evaluation results show that BRAHMA's plan can account for system parameters such as cost functions, penalty functions, optimization time window and clients SLOs and it also allows for the clients and SSPs to explore alternate options. Finally, by comparing the plans generated by this planning tool with commonly used outsourcing strategies we show that BRAHMA can adapt to system changes and always find the optimal plan with a decision overhead similar to the simple outsourcing strategies.

REFERENCES

- [1] Amazon Web Storage Services. <http://aws.amazon.com/s3>.
- [2] EMC ControlCenter family of storage resource management (SRM). http://www.emc.com/products/storage_management/controlcenter.jsp.
- [3] Enterprise portfolio analysis and collaborative decision support. <http://www.expertchoice.com>.
- [4] IBM TotalStorage. <http://www-1.ibm.com/servers/storage>.
- [5] Insight builder. <http://www.insightbuilder.net/whaticcm/roicalculator/index.shtml>.
- [6] Outsourcing calculator. <http://www.info-sourcing.com>.
- [7] Sun Storage Services. <http://www.sun.com/service/storage/index.html>.
- [8] G.A. Alvarez, J. Wilkes, E. Borowsky, S. Go, T.H. Romer, R. Becker-Szendy, R. Golding, A. Merchant, M. Spasojevic, and A. Veitch. Minerva: An automated resource provisioning tool for large-scale storage systems. *ACM Transactions on Computer Systems*, 19(4):483–518, November 2001.
- [9] E. Anderson, M. Hobbs, K. Keeton, S. Spence, M. Uysal, and A. Veitch. Hippodrome: Running circles around storage administration. In *Proceedings of USENIX Conference on File and Storage Technologies*, pages 175–188, 2002.
- [10] E. Anderson, M. Kallahalla, S. Spence, R. Swaminathan, and Q. Wang. Ergastulum: quickly finding near-optimal storage system designs. *HP Laboratories SSP technical report HPL-SSP-2001-05*, June 2002.
- [11] Veena Bansal and Vivek Pandey. A decision-making framework for it outsourcing using the analytic hierarchy process, 2006.
- [12] C. Chekuri and S. Khanna. A PTAS for the multiple knapsack problem. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, pages 213–222, 2000.
- [13] Reyes Gonzalez, Jose Gasco, and Juan Llopis. A study of information systems outsourcing risks. <http://csrc.lse.ac.uk/asp/aspecis/20040054.pdf>, 2004.
- [14] O.H. Ibarra and C.E. Kim. Fast Approximation Algorithms for the Knapsack and Sum of Subset Problems. *Journal of the ACM*, 22(4):463–468, 1975.
- [15] Bjorn Johansson. Exploring application service provision: adoption of the asp concept for provision of icts in smes, 2004.
- [16] D. Pisinger. A minimal algorithm for the 0-1 knapsack problem. *Journal of Operations Research*, 45:758–767, 1997.
- [17] Storage Networking Industry Association. SMI Specification version 1.0. <http://www.snia.org>, 2003.

X. APPENDIX

A. Optimizer Formulation

The unit of allocation in the optimization formulation is defined as a *job*. A job be in terms of data type namely regular data, backup data, compliance data, or in terms of administrative tasks such as provisioning, security planning, disaster recovery.

For each resource (i.e., hardware devices or human administrators) j , the cost of satisfying job i ($f_{s_{ij}}$) can be calculated using its associated cost function (defined in Section III), which takes the amount of capacity required as input and outputs the estimated dollar amount to provide the service (e.g., purchasing cost, cost of power, administrators hiring cost). Similarly, using the cost functions associated with the clients' resources, we can calculate the cost of satisfying the same jobs on the client site (f_{c_i}). In addition, considering the temporal behavior of client's, the clients job requests can differ at different time intervals. To capture the costs of satisfying the jobs at time t , $f_{s_{ij}}$ and f_{c_i} are both represented as a function of time $f_{s_{ij}}(t)$ and $f_{c_i}(t)$.

objective function	$\max_{D \in \mathcal{D}} \sum_t \sum_i \sum_j [D_{ij} [f_{c_i}(t) - f_{s_{ij}}(t)]]$
variables	$D_{ij}=1$ if job i is placed on resource j . Otherwise, $D_{ij} = 0$
constraints	$\sum_i D_{ij} x_i(t) \leq a_j(t) \quad j = 1, \dots, S, t = 1, \dots, T$ $\sum_i D_{ij} y_i(t) \leq b_j(t) \quad j = 1, \dots, S, t = 1, \dots, T$ $\sum_j D_{ij} \leq 1 \quad i = 1, \dots, N$

Fig. 3. Formulation of the constraint optimization

Figure 3 gives the precise formulation of the problem. Where, N and S are the number of jobs and resources. T is the lookahead window. $x_i(t)$ and $y_i(t)$ are the capacity and throughput requirements of job i and $a_j(t)$ and $b_j(t)$ are the capacity and throughput available on resource j . In addition, the D represents the feasible assignment, which is determined based on the feasibility matrix generated by the policy manager. A notable feature of this objective function is that it captures the combined potential benefit to both the SSP and the clients. A larger objective value implies a larger potential benefit to all involved parties. Also, our constraints guarantee that there are no violations of the resource capacity and bandwidth (throughput) constraints and each job is assigned to no more than one SSP resource. Please note that when $\sum_j D_{ij} = 0$, the best option is to have the client's job in house, since the SSP is not equipped to handle that SLO.

B. Approximation

The problem of optimal task placement is equivalent to a 0/1 multi-knapsack problem [14], [12], [16], which is known to be NP-hard. The optimization in BRAHMA is even more complex

due to the temporal aspects of customer requirements, allowing on-demand provisioning where new hardware is added at runtime to meet the growth demands, and the ability to generate multiple offers (as discussed in the next section).

In theory, we can find the optimal allocation plan by calculating the objective function value for every $D \in \mathcal{D}$. In most "real" environments, however, the size of \mathcal{D} ($\mathcal{O}(S^N)$, ignoring the F constraints) makes this approach impractical for even medium sized environments (for example, in a toy environment with 5 customers and 4 SSP sites, the run-time is of the order of tens of hours). We are therefore forced to consider approximation techniques for solving the problem. Ideally, such methods should be fast, theoretically justified, and effective in the sense of leading to a "near optimal" solutions.

One commonly used heuristic in knapsack/bin packing problems is the randomized "rank-and-place" strategy. The basic idea is to pick a job randomly and place it at the location that maximizes the value of the objective function. This procedure is repeated until all jobs have been allocated. To improve the quality of the solution, this procedure can be repeated a large number of times and the solution leading to the maximum objective value is returned as the "optimal" solution. In addition, for physical resource allocation, to enable dynamic resource provisioning, BRAHMA estimates the expected penalty values after each job placement and determines the best provisioning amount leading to minimum provisioning and penalty cost.

The entire approximation algorithm works as follows:

- 1) Initialize: $\mathcal{J} = \{1, 2, \dots, N\}$, $D = 0$.
- 2) Pick a job from \mathcal{J} at random.
- 3) Place the current job (say, the k^{th}) into the resource maximizing the quantity $\frac{\sum_t [f_{c_k}(t) - f_{s_{kj}}(t)]}{\sum_i [a_j(t) - \sum_i D_{ij} x_{ij}(t)]}$ while maintaining the feasibility (say, the l^{th} resource). Set $D_{kl} = 1$ appropriately. If no such placement is feasible or leads to positive cost savings, then leave $D_k = 0$.
- 4) Let's first assume that the capacity requirement of each client follows a normal distribution $N(m_j(t), v_j(t))$, where m_j and v_j are the average capacity and the capacity variance at time t . For resource l (chosen in previous step), at any time t , the total capacity requirement (represented as $X_l(t)$) is also normally distributed with the mean as $\sum_j m_j(t)$ and the variance as $\sum_j v_j(t)$. To determine the best provisioning amount, we specify a minimum provisioning amount (e.g., zero) (lower bound $L_l(t)$) and a maximum provisioning amount (e.g., the maximum allowed capacity minus the existing capacity of the storage devices) (upper bound, $U_l(t)$), and perform a binary search as follows:

- Set the provisioning value $V_l(t) = \frac{U_l(t) + L_l(t)}{2}$.
- The expected cost of value $V_l(t)$ (sum of the provisioning cost and the penalty cost) is estimate by integrating the cost values along the distribution of the total capacity.
- Estimate the expected cost of $V_l(t) + \delta$, where δ is a random small positive number.

- If $Cost(V_i(t)) > Cost(V_i(t) + \delta)$, $V_i(t)$ is set as the new lower bound. Otherwise, $V_i(t)$ is set as the new upper bound.

The binary search continues until the upper bound and the lower bound converges or after a fixed number of iterations (in our simulation, we run 50 iterations). The final $V_i(t)$ is the provisioning amount at time t .

- 5) If $\mathcal{J} \neq \emptyset$ then goto step 2. Otherwise, compute cost savings for given D .

This algorithm is greedy in nature; it randomly picks jobs and places them into the system to maximize the total cost savings achieved per unit of remaining excess capacity. This metric appropriately captures the tradeoff between cost savings and excess capacity. For example, only trying to maximize the cost savings at each step could significantly reduce the quality of future decisions. For example, we need to be careful to leave room for the remaining jobs in \mathcal{J} . Placing a job into a machine which is barely big enough for it is better, with regards to this secondary objective, than placing it into a resource with a large amount of excess capacity.

Based on this approximation algorithm, BRAHMA determines an appropriate allocation of physical and administrator resources.