eTransform: Transforming Enterprise Data Centers by Automated Consolidation

Rahul Singh, Prashant Shenoy
Department of Computer Science
University of Massachusetts
Amherst, MA

K. K. Ramakrishnan
AT&T Labs Research
Florham Park, NJ

Rahul Kelkar and Harrick Vin
Tata Research Development and Design Center
Pune, India

***Abstract*—Modern day enterprises have a large IT infrastructure comprising thousands of applications running on servers housed in tens of data centers geographically spread out. These enterprises periodically perform a transformation of their entire IT infrastructure to simplify, decrease operational costs and enable easier management. However, the large number of different kinds of applications and data centers involved and the variety of constraints make the task of data center transformation challenging. The state-of-the-art technique for performing this transformation is simplistic, often unable to account for all but the simplest of constraints. We present eTransform, a system for generating a transformation and consolidation plan for the IT infrastructure of large scale enterprises. We devise a linear programming based approach that simultaneously optimizes all the costs involved in enterprise data centers taking into account the constraints of applications groups. Our algorithm handles the various idiosyncrasies of enterprise data centers like volume discounts in pricing, wide-area network costs, traffic matrices, latency constraints, distribution of users accessing the data etc. We include a disaster recovery (DR) plan, so that eTransform, thus provides an integrated disaster recovery and consolidation plan to transform the enterprise IT infrastructure.**

We use eTransform to perform case studies based on real data from three different large scale enterprises. In our experiments, eTransform is able to suggest a plan to reduce the operational costs by more than 50% from the "as-is" state of these enterprise to the consolidated enterprise IT environment. Even including the DR capability, eTransform is still able to reduce the operational costs by more than 25% from the simple "as-is" state. In our experiments, eTransform is able to simultaneously optimize multiple parameters and constraints and discover solutions that are 7x cheaper than other solutions.

I. Introduction

Large enterprises depend heavily on their IT systems and applications to run their business spread out in multiple geographic locations. The IT infrastructure of such large enterprises is complex; it is common for these enterprises to have tens of data centers of varying sizes running hundreds of business applications running on an even larger number of servers, each handling users spread out in multiple geographically distant locations.

Since IT costs are a significant component of an enterprise's overall operational costs, there is constant pressure on large enterprises to become more efficient and still cut costs. To achieve this goal enterprise periodically undertake large transformation projects where the entire IT infrastructure is simplified by consolidating it into a smaller number of data center locations. Doing so, enables the IT departments to optimize costs by exploiting economies of scale, eliminating redundancies, simplifying hardware and network configuration and optimizing the application performance as perceived by the users. For example, the largest such transformation project is being undertaken by the US Federal government [1] which is attempting to consolidate, simplify and transform nearly 2100 of its data center locations down to less than 1000. Similarly, the UK government [2] is planning to undergo a major consolidation of its 120 data centers down to 10. As another example, Hewlett Packard recently transformed 85 of its data centers worldwide into 8 larger data centers.

While enterprise IT transformation (through consolidation and adoption of new, more efficient hardware) is extremely beneficial, with enterprise and governments potentially saving tens or hundreds of millions of dollars, there has been little research on how to systematically undertake such a transformation. In addition, such transformations also help in saving considerable energy because of the optimal use of data center resources. The state-of-the-art used in practice is to manually determine an IT transformation plan by using simple rules of thumb and via data captured in spreadsheets. For instance, a common practice to perform such a transformation is to a priori choose a few data center locations and then assign existing applications and servers to one of these locations using ad-hoc metrics such as the new location closest to the customers, or minimum real-estate cost, etc.

Transformation and consolidation of an IT infrastructure that is distributed across the world is no different from traditional distributed system design; several costs and factors must be optimized:

- Reduced cost: this includes reducing the cost of space, power and communications (especially wide area networking).
- Dependency between applications: applications have associativity constraints, and application groups should not be split across data centers (because of communication costs, shared data, etc.)
- Latency: applications running on data centers far away from where users are located may be undesirable.
- Shared Risk: application groups should not be co-

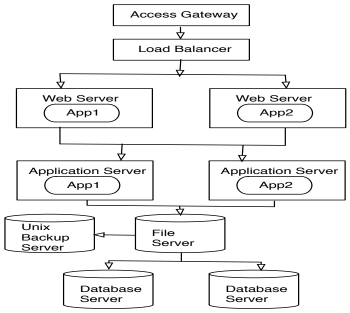


Figure 1. Example of a complex application group from a large enterprise

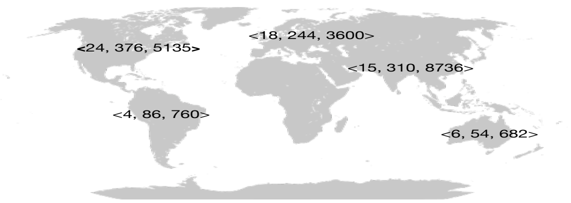


Figure 2. Number of data center locations, servers and users spread all over the world for a large enterprise. Data for each continent is depicted as $\langle X, Y, Z \rangle$ where X is the number of data center locations, Y is the number of servers and Z is the number of users.

No. of application groups	196
No. of applications	1122
No. of physical servers	1070
No. of data centers	67
No. of server models	95
No. of server OSes	107

Figure 3. Summary statistics about the IT infrastructure of a large enterprise

located in the same data center.

- Disaster Recovery: failure of a data center requires applications and data to be replicated at another data center
- Environmental considerations: site-specific or business-specific constraints of a location e.g., a certain application can only reside in a data center with a specified level of redundancy.

Data center transformation must systematically take into account all the above considerations. The current practice of manual consolidation planning is typically suboptimal and unable to fully consider the optimization accounting for all these constraints. In fact, for complex scenarios, such manual practice of performing transformation may be unable to even find a feasible plan, requiring considerable ad-hoc manual intervention to address all these conflicting constraints.

In this paper, we address the problem of enterprise IT transformation by designing and implementing eTransform, a system for the IT transformation and consolidation planning for enterprise data centers. By designing and implementing eTransform we make the following research contributions:

- *Transformation and Consolidation Algorithm:* We design a linear programming (LP) based algorithm for transformation and consolidation planning. Our algorithm captures the various aspects of enterprise data centers like economies of scale, space costs, WAN costs and latency seen by the users of the applications. We enhance our basic algorithm to incorporate disaster recovery (DR) so that our algorithm simultaneously plans for both DR and consolidation.
- *Experimental Evaluation using Real Datasets:* We perform a thorough experimental evaluation using real datasets from three large enterprises: 1) a large private multinational enterprise, 2) the Florida State government IT infrastructure and 3) the US Federal government IT infrastructure., eTransform spells out a data center consolidation plan. We compare the performance of our LP-based algorithm with a simple greedy algorithm and the state-of-the-art manual heuristic for performing consolidation. Our results show that eTransform is able to reduce the "as-is" operational costs by more than 50%. eTransform is also able to generate a disaster recovery plan on top of the consolidation plan at a 25% lesser cost than adding DR to the "as-is" state. We perform experiments to study the influence of the various data center and application parameters on the quality of the solution suggested by eTransform. We explore the sensitivity of the cost of the solution on the value of these parameters and the various tradeoffs between the parameters. In our experiments, eTransform is able to simultaneously optimize multiple parameters and find solutions 7x cheaper than others solutions.

The rest of the paper is organized as follows: In Section 2, we give a background and problem formulation of the data center transformation and consolidation problem. Section 3 describes the design of eTransform and our LP-based transformation and consolidation algorithm. Section 4 describes the enhancements to eTransform to plan for disaster recovery as well. Implementation details are discussed in Section 5. Section 6 describes the experimental results. We present the related work in Section 7 and the conclusion in Section 8.

II. Background and Problem Formulation

We assume an enterprise with multiple data centers at different locations. Each data center consists of a cluster of servers that provide compute and storage resources to the enterprise applications. Each data center also has a wide-area-network (WAN) connection to connect to other data center locations or other enterprise branch office locations, possibly over a virtual-private-network (VPN).

Each data center runs one or more distributed server applications. Depending on the nature of the business, this may include applications for enterprise resource planning (ERP), business intelligence, planning and forecasting, logistic tracking, payroll, project management, online ordering etc. Each application is distributed, consisting of multiple software components and may be tiered. Each application is accessed by users that are distributed across multiple geographic locations.

Multiple applications may have loose dependencies between each other as dictated by a business process. For example, in case of the IT infrastructure of a university, the student course registration application and the tuition and billing application are two different applications. While these two applications are ostensibly independent, they have dependencies between them since the tuition amount is determined by the courses that the student has registered for. We assume that such dependent applications would be grouped together into a clustered application group. We define an application group to be a group of applications that either interact closely with one another to perform a business process or have common data that they access. Figure 1 shows an example of a complex application group comprising multiple applications that extensively interact with each other.

As an example, consider the IT infrastructure of a multinational corporation that has its operations spread out in multiple countries worldwide. Figure 2 shows the statistics about the number of data center locations, number of servers and number of users present in each region of the world. Figure 3 shows the total number of applications, application groups, servers, data centers and users present in the IT infrastructure of this enterprise. To perform the transformation and consolidation planning of such an enterprise we require the *"as-is"* state information about the enterprise's IT infrastructure. We assume knowledge about the current data center locations and the number of servers at each location. We are given the application group to server mapping i.e., for each server we know which application group it belongs to. We also know the traffic matrix for each application group i.e., the number of users, the location of the users and the amount of data transferred between the application group and the users.

We also assume that the constraints associated with every application group are known. For example, an application group can have a constraint that the latency perceived by its users should be less than a threshold or that it cannot be placed outside a certain geographical area because of legal issues. We are given the associativity constraint with all application groups, which dictates that the applications comprising each application group should be placed within the same data center, since splitting the application group will introduce large amount of data traffic across the WAN between communicating applications leading to higher latencies seen by the users. Additionally, splitting an application group across data centers will lead to higher bandwidth costs because intra-application group traffic will now become wide-area-network (WAN) traffic instead of local-area-network (LAN) traffic.

In the extreme case where one application group is too large to be placed in any single datacenter, techniques like [3] can be used first to automatically determine how to split the application group amongst two data centers and then etransform can be used to pack the smaller application groups.

Symbol	Meaning
M	Number of application groups
N	Number of target data centers
R	Number of user locations
C_{ir}	Number of users of the i^{th} application group in the r^{th} location
S_i	Number of servers in the i^{th} application group
D_i	Monthly Data transferred in megabits by the i^{th} application group
O_j	Capacity of the j^{th} data center
Q_j	Space cost per server at the j^{th} target data center
W_j	WAN cost per megabit at the j^{th} target data center
E_j	Power cost per kilowatt at the j^{th} target data center
T_j	Labor cost per administrator at the j^{th} target data center
L_{ij}	Latency penalty of placing i^{th} application group in j^{th} target data center

Table I
SPECIFICATION OF THE "AS-IS" STATE OF THE ENTERPRISE

We assume that the enterprise has a new set of target data center locations. We know the capacity of each of these data centers and the costs associated with each data center location: for space, WAN, power and labor. We also are given the latencies and the distances between the locations of the users and the locations of the target data centers.

Given the above, the consolidation and transformation planning problem is to generate a future, *"to-be"*, state of the enterprise IT infrastructure by determining a subset of the target data center locations where the application groups can be placed in order to 1) reduce the total number of data center locations thus leading to simplification of the IT infrastructure, 2) to reduce the operational costs from the "as-is" state by jointly optimizing the cost for space, WAN, labor and power, by exploiting economies of scale and 3) to place application groups while meeting their constraints.

III. eTRANFORM: DATA CENTER TRANSFORMATION AND CONSOLIDATION

In this section, we discuss the design of eTransform, a system for data center transformation and consolidation planning, and present our LP-based transformation and consolidation algorithm.

A. eTransform Design

Our system needs a specification of the "as-is" state as input. Consider an enterprise whose IT infrastructure comprises M application groups where the i^{th} application group runs on S_i physical servers. The users of each application group are spread out in R locations and we denote the number of users of the i^{th} application group that are present in the r^{th} user location by C_{ir}. Let D_i denote the data exchanged between the i^{th} application group and its users in Megabits.

The enterprise has N target data center locations. Let O_j denote the capacity of the j^{th} data center and Q_i, W_j, E_j and T_j denote the cost at the j^{th} data center for space, WAN communication, power and labor respectively.

Given this "as-is" state, the basic transformation and consolidation planning algorithm works as follows. The algorithm first finds the cheapest data center locations by simultaneously considering all the different cost components and finding a global minimum. We then pack the application groups into as few data centers as possible in order to take maximum advantage of economies of scale. Under a volume pricing structure, the price per unit decreases as the quantity purchased increases, so the algorithm tries to optimize cost by concentrating servers in as few data centers as is feasible.

eTransform tries to perform the repacking of application groups into data centers in a manner such that the physical resources available to the applications remain the same before and after the repacking. The goal is to preserve the performance, and we believe it is a critical goal of our work to not degrade performance by the repacking.

But the algorithm also needs to choose the data center locations in a manner such that the constraints of the application groups are met. To meet the associativity constraint, the algorithm places all the servers belonging to each application group in the same data center without splitting them. The algorithm also takes into account the user traffic distributions and latency requirement of the application groups in order to determine the future "to-be" state. For instance, consider an application group that is latency sensitive; if the users of an application group are spread out in multiple locations, then the algorithm favors placing the application group in a data center location that balances the latency from all user locations. On the other hand, if the users of the application group are skewed to a single location, the algorithm may choose to place the application group in a data center close to the user location to satisfy the latency constraint of the application group.

Thus, the algorithm seeks a solution that collectively captures all the costs and constraints of the enterprise. This global multi-criteria optimization is performed by an LP-based algorithm that we discuss next.

B. LP-Based Transformation and Consolidation Algorithm

Our LP-based algorithm takes the "as-is" state and computes a "to-be" plan. The algorithm proceeds by representing the total cost of a "to-be" state as a linear objective function and representing the constraints of the applications as linear constraints, thus transforming the transformation and consolidating planning problem into a linear programming (LP) problem.

Using the variables described in Table I, the algorithm creates the following LP from the "as-is" state information:

Minimize: $\sum_{j=1}^{j=N} \sum_{i=1}^{i=M} X_{ij} \left(S_i(Q_j + \alpha E_j + \frac{T_j}{\beta}) + D_i W_j + L_{ij} \right)$
Subject To: 1)$\sum_{j=1}^{j=N} X_{ij} = 1, \forall 1 \leq i \leq M,$
2) $\sum_{i=1}^{i=M} X_{ij} S_i \leq O_j, \forall 1 \leq i \leq N, \quad 3) X_{ij} \in \{0,1\}$
where α is power consumption of a server in kilowatts and β is the number of servers an administrator can typically handle, in order to size labor costs.

The algorithm needs to decide a target data center for each application group; this decision is captured using the binary decision variable X_{ij} which is 1 if the i^{th} application group is placed in the j^{th} data center and 0 otherwise. The total cost of the "to-be" is expressed as a function of these decision variables to form the objective function that the algorithm minimizes. We add constraints and ensure that the linear program has feasible solutions. Constraint 1) ensures that a data center location is found for every application group; constraint 2) ensures that we do not exceed the capacity of any givendata center.

The algorithm handles latency constraints by adding L_{ij}, a latency penalty, in the objective function that needs to be minimized. The latency penalty, L_{ij} is the penalty added to the cost based on the latency perceived by the users of the i^{th} application group if the i^{th} application group is placed in the j^{th} data center. To calculate this we use the latency between the locations of the users and the data center locations that are given as part of the input, and determine the average latency. Each application group specifies its latency constraint in the form of a latency penalty function. The latency penalty function is a step function, expressing the cost added to the objective function per user based on the range for the average latency. For example, the latency penalty function of an application can specify that a penalty of $10 per user be added if the average latency $> 10ms$. We do not focus on the how to determine the latency penalty of the application and assume that the administrator can use one of the many techniques available [4], [5] for deciding such penalty values.

To capture economies of scale, the algorithm represents each of the costs at the data center, Q_j, W_j, E_j and R_j, as a function of the quantity purchased. For example, Q_j, the space cost at the j^{th} data center can be a step function such that the space cost per server is Qb_j if the total number of servers that the algorithm places in the j^{th} data center is less than b_j. Similarly, the space cost decreases by H_j per server every time the algorithm places b_j more servers in this data center. The algorithm employs the technique described in [6] to incorporate such step functions into the objective function and thus handle economies of scale in the LP.

If the enterprise uses dedicated VPN links between data centers and the users, the WAN cost is captured differently since the WAN cost of VPN links may typically depend on the distance between the two endpoints of the link. To account for the WAN cost in this case the algorithm first determines the number of VPN links required between each application group and its users. Assuming an equal amount of data is exchanged with each user, the number of dedicated links required to connect this application group to its users in the r^{th} location is given by $(C_{ir}D_i)/(\gamma * \sum_{r=1}^{r=R} C_{ir})$ where γ is the capacity of a single link. Thus, the WAN cost of the i^{th} application group if placed in the j^{th} data center is given by $\sum_{r=1}^{r=R}\left((C_{ir}D_i)/(\gamma * \sum_{r=1}^{r=R} C_{ir})\right)(F_{jr})$, where F_{jr} is the monthly cost of leasing a VPN link between the j^{th} data center and the r^{th} user location.

After converting the transformation and consolidation planning problem into a linear program, the algorithm invokes a linear program solver to solve this optimization and thus find an optimal plan that minimizes cost and satisfies all the constraints.

IV. Designing for Disaster Recovery

Data centers are sometimes exposed to potential disasters, both natural threats like floods, fire, earthquakes, as well as electrical grid failures, that can take down an entire data center. All the applications running in the data center become inaccessible to its users. Large enterprises need to plan for such failures and ensure that their mission-critical applications continue to function despite these disasters.

A. Basic Intuition

Currently, for most enterprises, disaster recovery is often an afterthought. It is likely not a part of the IT transformation and consolidation plan. eTransform takes an integrated approach by generating a disaster recovery plan as part of the transformation and consolidation plan. To plan for disaster recovery, the transformation and consolidation algorithm is modified to select two different target data center locations for every application group: one of the data center location acts as the primary location of the application group while the other acts as the secondary location. In the event of a disaster striking the primary location, the application group can failover and operate from the secondary location. Every data center has separate backup servers in the secondary data center that act as the backup for the application groups. Typically enterprises plan for a single failure. Thus, the same backup servers can be shared among multiple application groups placed in different data centers. On the other hand, to plan for more than one concurrent failure, each application group needs to have its own set of dedicated backup servers.

Planning for both disaster recovery and data center consolidation needs a deeper analysis since disaster recovery and data center consolidation may be viewed as conflicting requirements. While disaster recovery wants to spread application groups across multiple data centers so that fewer application groups are impacted by a disaster striking a data center, data center consolidation wants to concentrate servers into fewer data centers to exploit economies of scale and optimize cost. eTransform achieves a balance between these two objectives by optimizing a global function comprising both these objectives.

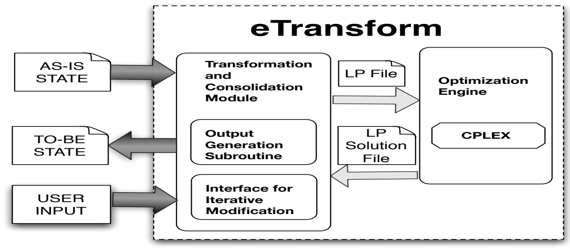


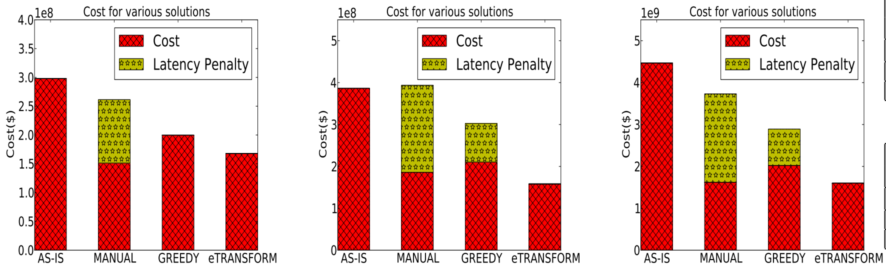
Figure 5. eTransform System Architecture

B. eTransform LP-Based Algorithm to Handle DR

Our LP-based algorithm generates a single-failure disaster recovery plan in addition to the transformation and consolidation plan. To achieve this, it introduces a new set of binary decision variable, Y_{ij}, into the LP. These denote if the j^{th} data center is chosen as the secondary data center of the i^{th} application group. The algorithm adds the constraint $X_{ij} + Y_{ij} < 2, \forall 1 \leq i \leq M, 1 \leq j \leq N$ to ensure that the primary and secondary data centers for every application group are different.

The algorithm also adds the variables G_a to denote the total number of backup servers in the a^{th} data center. Since the algorithm is planning for single failure disaster recovery, the same backup server can be shared between more than one application group. For example, if two application groups have been placed in two different data centers, say A and B, then we can use the same data center, say C, to act as the secondary data center for both these applications, since a disaster can only strike one of A or B. If C acts as the secondary data center for both A and B the number of backup servers needed at C is the larger of the number of servers needed for application groups A and B rather than the sum of the number of servers.

Our algorithm captures this sharing of backup servers by introducing the binary variable $J_{a\_b\_c}$ into the linear program; $J_{a\_b\_c}$ is 1 if the c^{th} application group has the a^{th} data center as its primary data center and the b^{th} data center as its secondary data center. This variable can be expressed as a function of the other variables as $J_{a\_b\_c} >= X_{ca} + Y_{cb} - 1$. Using these variables, the right number of backup servers needed at the b^{th} data center, G_b, can be expressed as $G_b = \max_{a=1}^{a=N}\left(\sum_{c=1}^{c=M} J_{a\_b\_c} S_c\right)$. To capture the "maximum" operator, the algorithm converts it into N linear constraints $G_b >= \left(\sum_{c=1}^{c=M} J_{i\_b\_c} S_c\right), 1 \leq i \leq N$ and adds it into the linear program. To enable enterprise administrators to mitigate the business impact of a disaster by spreading servers across multiple data centers, the algorithm introduces a business impact parameter, ω, and adds a constraint, $\sum_{i=1}^{i=M} X_{ij} \leq \omega M, \forall 1 \leq j \leq N$, that ensures



Enterprise1
(a)

Florida
(b)

Federal
(c)

Dataset	Cost Reduction		
	Manual	Greedy	eTransform
Enterprise1	-19%	-32%	-43%
Florida	-5%	-21%	-58%
Federal	-28%	-35%	-59%

(d)

Dataset	Latency Violations		
	Manual	Greedy	eTransform
Enterprise1	74	0	0
Florida	84	71	1
Federal	878	576	1

(e)

Figure 4. Comparison of Greedy, Manual and eTransform without Disaster Recovery

that the fraction of total application groups placed in any single data center does not exceed ω.

The algorithm accounts for the extra space, WAN, power and labor cost incurred at the data center locations for the backup servers by adding the appropriate costs into the objective function. The cost of buying backup servers i.e., $\sum_{j=1}^{j=N} G_j \zeta$ where we denote the cost of a backup server by ζ, which is added into the objective function as well.

The algorithm thus creates an extended linear program by adding new variables and constraints and solves the linear program using a LP-solver to generate a DR plan along with a consolidation plan.

V. eTRANSFORM IMPLEMENTATION

We have implemented a prototype of eTransform, our transformation and consolidation planning system. In this section we present the implementation details of eTransform.

The eTransform system comprises of four primary modules: i) the transformation and consolidation module, ii) the optimization engine, iii) the output generation module and iv) the admin interface for iterative modification. Figure 5 shows these components and their interactions.

A user invokes eTransform by providing the "as-is" state of the enterprise by specifying the values of the parameters specified in Table I. The "as-is" state specification is taken as input by the transformation and consolidation module that implements the logic of the transformation and consolidation algorithm. This module converts the transformation problem into the suitable linear program, creates a file in the LP file format and then invokes the optimization engine with this file as input. The optimization engine uses a LP-solver to solve this linear program; we use the CPLEX LP-solver [7] to solve the linear program. The output generated by the LP-solver is given as input to the output generation subroutine that converts the solution of the linear program into the "to-be" state of the enterprise. eTransform also allows the user to iteratively interact and change the initial solution by adding more constraints. This functionality is implemented by the admin interface for iterative modification. All these components have been implemented using the Python programming language.

Dataset	As-Is State # of Data Centers	# of Target Data Centers	# of Servers	# of App Groups
Enterprise1	67	10	1070	190
Florida	43	10	3907	190
Federal	2094	100	42800	1900

Table II
SUMMARIZED DATA FOR THE THREE DATASETS USED

VI. EXPERIMENTAL EVALUATION

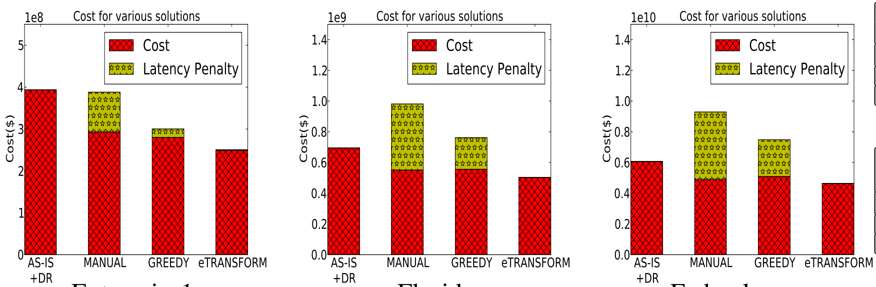
In this section we evaluate our LP-based transformation and consolidation algorithm by using eTransform on various scenarios. First, we perform case studies, where we use eTransform to transform and consolidate the IT infrastructure of three large enterprises. Then we explore the influence of various parameters on the transformation plan generated by our algorithm by using eTransform under various parameter values. We start by describing the datasets we use for performing our experiments.

A. Dataset Description

We use data from three different enterprises to perform our experiments.

The first enterprise is the multinational corporation that we described in Section II. The details of this dataset are described in Figure 3. We call this dataset, the *enterprise1* dataset.

The second enterprise is the state department of the state of Florida, USA. This dataset is publicly available as part of a study conducted by Gartner to evaluate the feasibility of performing a data center consolidation of the current data center infrastructure of the departments and agencies of the Florida State government [8]. Since this dataset does not contain information about the number of application groups, the number of servers in each application group and the number of servers in each data center, we assume the same number of application groups as found in the *enterprise1* dataset and also the same distribution of servers in application groups and data centers.



Enterprise1 (a) | Florida (b) | Federal (c)

Dataset	Cost Reduction		
	Manual	Greedy	eTransform
Enterprise1	-1%	-18%	-36%
Florida	+37%	+9%	-27%
Federal	+51%	+23%	-23%

(d)

Dataset	Latency Violations		
	Manual	Greedy	eTransform
Enterprise1	178	57	4
Florida	176	121	0
Federal	1940	1175	1

(e)

Figure 6. Comparison of Greedy, Manual and eTransform with Disaster Recovery

The third enterprise is the federal government of USA. The federal government has 2094 data centers which it is planning to consolidate. Since this dataset does not contain the number of servers and the number of application groups, we assume that the federal dataset has ten times as many application groups as present in the *enterprise1* dataset. We then use the same distribution as present in the *enterprise1* dataset to assume a distribution of servers into application groups and data centers.

Table II summarizes the sizes of these three datasets including the number of application groups, the number of data centers in the "as-is" state and the number of target data centers.

B. Enterprise Transformation and Consolidation Case Studies: Non-DR Case

We evaluate the performance of our LP-based algorithm in performing data center consolidation by using eTransform to perform consolidation of the three datasets described above. The datasets however only contain information about the applications groups; we obtain data about the target data center costs from other sources.

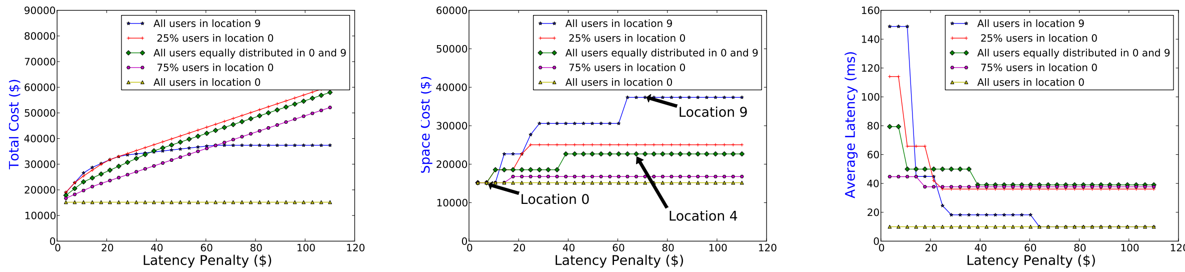
The space costs of the data centers assumed in our experiments are based on a study [9] conducted of 2213 colocation providers. We used a report [10] containing the average salary of IT administrators in different U.S states to calculate the labor costs at the data centers. We assumed that each administrator can handle 130 servers. The price of power in terms of cents per kilowatt hour for the different data center locations was obtained from a report published by the U.S. Energy Information Administration [11]. We assumed that each server consumes between 300 and 400 watts power on average to calculate power costs. The WAN prices are based on the pricing shown in Amazon's cloud report [12]. We assume that the data centers have different capacities ranging from 100 servers to 1000 servers.

We first evaluate the non-DR version of our LP-based transformation and consolidation algorithm. For each of the three datasets, we assume that the applications are divided equally into two classes: latency-sensitive applications and latency-insensitive applications. For latency-sensitive applications, there is a penalty of $100 per client if the average latency exceeds 10 milliseconds, while for latency-insensitive applications there is no latency penalty. We choose this value as an example for applications that are extremely sensitive to latency violations. In practice, this value can be chosen to be high or low depending on how critical latency is to the application. As an example, in the case of stock trading applications, latency is extremely important and a change of 1 millisecond in the end-to-end latency can lead to a loss or gain of $100 million dollars in yearly revenue [13]. Additionally, we study the impact of the latency penalty value chosen on the transformation plan suggested by eTransform in Section VI-D.

We assume that the clients are present in 4 different locations and we equally divide the latency-sensitive applications into 5 classes: applications that have all their clients in one of the 4 locations and applications that have their clients equally distributed in all 4 locations. Similarly, we divide the data centers into 5 classes: data centers that are close to one of the 4 client locations i.e. have a latency of 5 milliseconds from one location and 20 milliseconds from the other three and data centers that are close to all 4 locations i.e. have a latency of 10 milliseconds from all 4 locations.

To better assess the performance of our LP-based algorithm we compare it with two different algorithms. The first algorithm is the manual method of performing data center consolidation and resembles the way consolidation is currently performed in large enterprises. This method consolidates all applications into a fixed number of data centers, for instance, say only two data centers. It places applications into the data center closest to the current location of the application. The second algorithm is a greedy algorithm that sequentially looks at applications in decreasing order of number of servers, calculates the cost of placing each application in every data center and then selects the cheapest data center to place this application in.

Figure 4(a), (b) and (c) shows the cost of the solutions found out by the three algorithms and the cost of the "as-is" infrastructure for the three datasets. The figure shows the two components of the total cost: the actual cost and



(a) Total Cost

(b) Space Cost

(c) Average Latency

Figure 7. Influence of Latency Penalty on Solution found by eTransform

the penalties paid due to latency violations. Table 4(d) shows the percentage reduction in operational costs achieved by the three algorithms and Table 4(e) shows the number of latency violations suffered by each algorithm. The bar charts show that the cost of the solution found by our LP-based algorithm is significantly smaller compared to the cost of the other two algorithms and that while the manual approach is able to achieve costs almost equal to eTransform, it is unable to take latency constraints into account and pays huge latency penalties. The greedy approach tries to take latency constraints into account and is able to reduce the latency penalties but the cost of the solution goes up. eTransform, however is able to achieve lower costs and also maintain all latency constraints and is thus able to optimize both dimensions simultaneously. The table reveals that while the greedy algorithm is only 11% worse than the LP-based algorithm in the smaller *Enterprise1* dataset, on the larger Florida and Federal datasets the LP-based algorithm achieves 37% and 24% greater cost reductions over the greedy algorithm.

Result: On three different enterprise size datasets, eTransform is able to achieve much larger cost reductions ($> 50\%$) as compared to the manual and greedy approach while still being able to satisfy all latency constraints

C. Enterprise Consolidation Case Studies: DR Case

Next, we use our eTransform transformation and consolidation algorithm to perform consolidation and plan for disaster recovery simultaneously. We again compare our LP-based algorithm with DR variants of the two algorithms used in the previous section. In the DR variant of the manual method, we create two backup data centers to act as the backup for the two data centers used in the non-DR case. When an application is placed in one of the two data centers, its backup application is placed in the corresponding backup data center. We also create a DR variant of the greedy algorithm. This algorithm first places all the applications just like the non-DR greedy algorithm and then it places the backup application of each application similarly. The only difference being that while deciding on the cheapest data center for each backup application, the algorithm also adds the cost to buy new servers into the total cost. We compare these algorithm with the cost of adding DR to the as-is state by building a single backup data center that acts as the backup of all other data centers. We assume the cost of a DR server to be $1000.

Figure 6(a), (b) and (c) shows the total cost, the actual cost and penalties paid due to latency violations by the three algorithms on the three datasets. Table 6(d) shows the percentage reductions obtained by the three algorithms and Table 6(e) shows the latency violations suffered by the three algorithms. The bar chart shows that while both the manual and greedy approach achieve the same cost in their solutions, the greedy approach suffers lower latency violations. However, eTransform is able to provide a DR solution for a cheaper cost in all the three datasets without violating any latency constraints. The table shows that the while the greedy and manual algorithm exceed the cost of adding DR to the as-is state on the Florida and Federal datasets, eTransform finds a DR plan $> 25\%$ cheaper than adding DR to the as-is state in these datasets and is also able to satisfy all the latency constraints.

Result: On three different enterprise datasets, eTransform is able to simultaneously compute a consolidation and disaster recovery plan that can achieve significant cost savings ($> 25\%$) compared to the as-is state and also satisfy the latency constraints.

In the following sections, we study the influence of various parameters on the transformation and consolidation plan generated by eTransform.

D. Influence of Latency Penalty

In this section, we evaluate the influence of the latency constraints, user distributions and data center latencies from user locations on the transformation plan obtained by eTransform.

To perform this experiment we use the application groups from the *enterprise1* dataset and we use ten data centers, location 0 to location 9, with increasing latencies from, location 0, the first data center. The space cost also increases as we move from location 0 to location 9, location 0 being the cheapest location and 9 being the costliest. All other

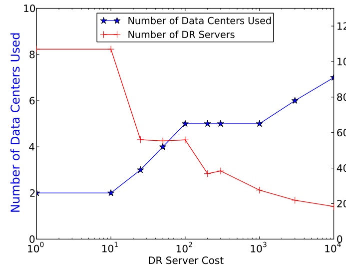


Figure 8. Influence of DR Server Cost

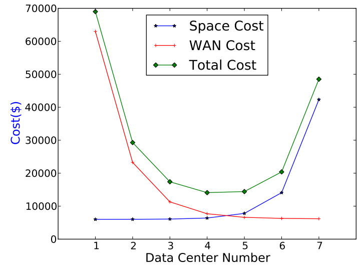


Figure 9. Tradeoff between Space Cost and WAN Cost

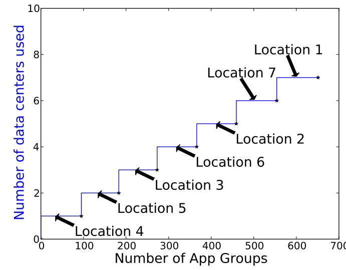


Figure 10. Placement by eTransform

costs are the same for all data centers. All application groups have their users distributed close to locations 0 and 9 only. We set the latency threshold to 10 milliseconds and impose a latency penalty per client if the latency exceeds this threshold.

In this experiment we vary 1) the latency penalty between \$0 and \$100 per client and 2) the relative user distribution in the two locations, location 0 and location 9. We then observe the placement of application groups into data centers as suggested by eTransform, the cost of the solution and the mean latency perceived by the clients.

Figure 7(a) shows the total cost of the solution found by eTransform for various values of the latency penalty and for different user distributions. Similarly Figure 7(b) shows the space cost of the solutions while Figure 7(c) shows the mean latency seen by the users as we vary the latency penalty and user distribution. Figure 7(a) shows that if the users are not totally concentrated in a single location the total cost increases as we increase the latency penalty. Figure 7(b) shows that as the proportion of users in the costliest location i.e. in location 9 increases, the space cost increases as we increase the latency penalty. When the latency penalty is low, eTransform places the application groups in the cheapest location i.e. location 0 but as we increase the latency penalty, eTransform places the application groups closer to where most of its users are. For example, when all the users are in location 9, at the highest latency penalty, eTransform places all application groups in location 9. Figure 7(c) shows that the mean latency perceived by end users decreases with increasing penalty.

Result: At low latency penalty eTransform optimizes cost, while at high latency penalties eTransform optimizes mean latency seen by the users of the application group and places application groups by taking into account user distribution.

E. Influence of Disaster Recovery Server Cost

In this section we explore the change in the disaster recovery plan generated by eTransform as we change the cost of the backup server.

To perform this experiment we assume the same setup as we assumed in the previous experiment. In this experiment we assume that the latency penalty is 0, therefore we only optimize the cost. In this experiment we want to generate a disaster recovery plan apart from a consolidation plan. We increase the cost of a DR server and observe the change in the transformation plan generated by eTransform.

Figure 8 shows the number of target data centers used by eTransform to place the application groups and also the total number of DR servers used as the cost of a DR server increases exponentially. The figure shows that when the cost of buying a DR server is low, eTransform tries to optimize space cost by consolidating all the application groups in the cheapest data center and all the DR servers in the next cheapest data center thus using only 2 data centers. In this case however, we need to buy larger number of DR servers since if the cheapest data center fails, all the application groups fail at the same time. When the cost of buying a DR server is extremely high, eTransform optimizes the cost of buying DR servers by spreading the application groups in 7 data centers. Spreading out the application groups in multiple data centers allows eTransform to buy smaller number of DR servers. Because the applications groups are spread out, any one data center only has a small fraction of the total application groups. So if one data center fails, we need a smaller number of servers for backup. Moreover, since only one data center can fail at a time, the same DR servers can be shared by application groups that are present in different data centers.

Result: At low DR server cost, eTransform optimizes data center cost, while at high DR server cost eTransform optimizes the cost of buying new DR servers by sharing backup servers with multiple application groups.

F. Influence of Tradeoff between Space Cost and WAN Cost

In this section we study the behavior of eTransform when the target data centers have multiple costs involving tradeoffs and the ability of eTransform to simultaneously optimize multiple costs.

To perform this experiment we assume the same setup as we assumed in the previous experiment. In this experiment we do not plan for disaster recovery. We assume 10 data centers with capacities 100 each. We also assume that all application groups have their users in, location 9, the costliest data center by space cost. We assume that all application groups use dedicated VPN links to connect to their users, so the WAN cost decreases when the application groups are closer to the users and increases when the application groups are farther from the users.

Figure 9 shows the space costs, WAN costs and the total cost of placing the application groups at the different data centers. The figure shows that there is a tradeoff between space cost and WAN cost; while space cost is cheapest at location 1, WAN cost is costliest here since it is farthest from its users and the WAN link is longest, on the other hand space cost is costliest at location 7, WAN cost is cheapest here. The total cost is minimum in between location 1 and 7 at location 4. Figure 10 shows the placement of the application groups performed by eTransform. This figure shows that eTransform starts by filling the location with the cheapest total cost and then fills location with increasing total costs.

Result: eTransform is able to optimize multiple costs simultaneously and find out locations with a globally minimum cost. In this experiment, eTransform is automatically able to find a suitable location that is 7x cheaper than the most expensive location by total cost.

VII. Related Work

To the best of our knowledge, our work is the first in literature aimed at transforming and consolidating IT infrastructure of large enterprises. A lot of research effort has been concentrated on the problem of server consolidation that deals with consolidating multiple virtual machines present inside a data center onto lesser physical machines. Most of these techniques find patterns in workload and resource utilization of these virtual machines to decide the virtual machines that can share the same physical machine [14], [15], [16], [17]. In [18], [19], [20], [21], the authors take an approach similar to ours and transform the server consolidation problem into an optimization problem to find out the optimal placement of virtual machines onto physical servers in a data center. The authors of [22] try to solve the related problem of optimally allocating virtual machines on the physical infrastructure of various cloud providers and frame it as a stochastic integer programming problem. Unlike, all these works, eTransfrom aims to perform a consolidation of applications across data centers and transform the IT infrastructure of an enterprise with multiple data centers into fewer data centers.

The authors of [23] solve a problem complementary to ours where they try to decide the optimal location for an enterprise to build new datacenters in order to optimized multiple objectives. This technique can be used as a precursor to eTransform to first decide the target data center locations and then use eTransform to perform a transformation and consolidation of the IT infrastructure onto these target data center locations.

In [3] the authors solve a similar problem to ours, where they assume that each application can be partially hosted on the private local cluster and partially hosted on the public cloud. The authors formulate the problem as an optimization problem as well to decide the components of a given application that should be placed on the cloud given various constraints of the application. While this work assumes two data centers, eTransform is able to transform the entire IT infrastructure across multiple data centers.

In the area of disaster recovery planning, most of the current work in literature has focused on designing the disaster recovery plan for an application by choosing the appropriate DR mechanism [24], [25]. In these works, the authors assume that the location of the primary and secondary data center locations are already given for the application. eTransform fills this gap by automatically devising an DR plan for the enterprise by selecting the primary and secondary sites for every application group.

VIII. Conclusion

In this paper, we presented the design and implementation of eTransform a system that generates a transformation and consolidation plan for large enterprises. eTransform uses a linear programming based transformation and consolidation algorithm that takes the parameters of the "as-is" of the enterprise and generating a plan of the "to-be" state to pack the applications into a given set of target data center locations by simultaneously optimizing all the costs while still respecting all the constraints of the enterprise. We also devised a improved version of the our algorithm that also generates a disaster recovery plan for the enterprise on top of generating a transformation plan.

We illustrated the operation of eTransform by using it to generate the transformation plan for three different enterprises; we compared the cost reductions achieved by eTransform with those obtained by a manual and greedy approach. eTransform is able to generate plans that reduce the "as-is" operational costs by 25% and still provides disaster recovery. We also studied the impact of various parameters on the solution generated by eTransform.

Acknowledgment

This reseach was supported in part by NSF grants CNS-0855128, CNS-0916972, CNS-0720616, OCI-1032765, CNS-1117221 and a gift from AT&T.

References

[1] "Us federal govt. data center consolidation," 2010, online http://www.datacenterknowledge.com.
[2] "Uk govt. consolidation," 2011, online http://www.itpro.co.uk.

[3] M. Hajjat, X. Sun, Y.-W. E. Sung, D. Maltz, S. Rao, K. Sripanidkulchai, and M. Tawarmalani, "Cloudward bound: planning for beneficial migration of enterprise applications to the cloud," in *SIGCOMM*, 2010.

[4] E. Bouillet, D. Mitra, and K. Ramakrishnan, "The structure and management of service level agreements in networks," *Selected Areas in Communications, IEEE Journal on*, vol. 20, no. 4, may 2002.

[5] J. Kosinski, D. Radziszowski, K. Zielinski, S. Zielinski, G. Przybylski, and P. Niedziela, "Definition and evaluation of penalty functions in sla management framework," in *ICNS*, 2008.

[6] B. A. Schoomer, "The incorporation of step functions and ramp functions into a linear programming model," *Operations Research*, vol. 12, 1964.

[7] "Ibm ilog cplex optimizer," 2011, online http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/.

[8] "Florida state consolidation study," 2010, online http://www.cio.gov.

[9] "Space cost values," 2010, online http://www.telegeography.com.

[10] "2011 it skills and salary report," 2011, online http://www.globalknowledge.com/.

[11] "Average retail price of electricity by state," Website, 2010, http://www.eia.gov.

[12] "User guide: Amazon ec2 cost comparison calculator," 2011, online http://awsmedia.s3.amazonaws.com/.

[13] InformationWeek, "Wall street's quest to process data at the speed of light," Website, 2007, http://www.informationweek.com/news/199200297.

[14] A. Verma, G. Dasgupta, T. K. Nayak, P. De, and R. Kothari, "Server workload analysis for power minimization using consolidation," in *Proc. of USENIX ATC*, 2009.

[15] M. Cardosa, M. Korupolu, and A. Singh, "Shares and utilities based power consolidation in virtualized server environments," in *IM*, 2009.

[16] T. Wood, G. Tarasuk-Levin, P. Shenoy, P. Desnoyers, E. Cecchet, and M. D. Corner, "Memory buddies: exploiting page sharing for smart colocation in virtualized data centers," in *In Proc. of VEE*, 2009.

[17] A. Verma, P. Ahuja, and A. Neogi, "pmapper: power and migration cost aware application placement in virtualized systems," in *Middleware 2008*.

[18] B. Speitkamp and M. Bichler, "A mathematical programming approach for server consolidation problems in virtualized data centers," *IEEE Trans. Serv. Comput.*, vol. 3, October 2010.

[19] S. Mehta and A. Neogi, "Recon: A tool to recommend dynamic server consolidation in multi-cluster data centers," in *Proc. of NOMS*, 2008.

[20] R. Gupta, S. K. Bose, S. Sundarrajan, M. Chebiyam, and A. Chakrabarti, "A two stage heuristic algorithm for solving the server consolidation problem with item-item and bin-item incompatibility constraints," in *Proc. of ICSC*, 2008.

[21] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *INFOCOM 2010*.

[22] S. Chaisiri, B.-S. Lee, and D. Niyato, "Optimal virtual machine placement across multiple cloud providers," in *APSCC 2009*.

[23] I. Goiri, K. Le, J. Guitart, J. Torres, and R. Bianchini, "Intelligent placement of datacenters for internet services," in *Proc. of ICDCS*, 2011.

[24] S. Gaonkar, K. Keeton, A. Merchant, and W. Sanders, "Designing dependable storage solutions for shared application environments," *Dependable and Secure Comp., IEEE Trans.*, vol. 7, oct.-dec. 2010.

[25] K. Keeton, C. Santos, D. Beyer, J. Chase, and J. Wilkes, "Designing for disasters," in *Proc. of USENIX FAST*, 2004.