It's Not Easy Being Green

Peter Xiang Gao, Andrew R. Curtis, Bernard Wong, S. Keshav
Cheriton School of Computer Science
University of Waterloo

ABSTRACT

Large-scale Internet applications, such as content distribution networks, are deployed across multiple datacenters and consume massive amounts of electricity. To provide uniformly low access latencies, these datacenters are geographically distributed and the deployment size at each location reflects the regional demand for the application. Consequently, an application's environmental impact can vary significantly depending on the geographical distribution of end-users, as electricity cost and carbon footprint per watt is location specific. In this paper, we describe FORTE: Flow Optimization based framework for request-Routing and Traffic Engineering. FORTE dynamically controls the fraction of user traffic directed to each datacenter in response to changes in both request workload and carbon footprint. It allows an operator to navigate the three-way tradeoff between access latency, carbon footprint, and electricity costs and to determine an optimal datacenter upgrade plan in response to increases in traffic load. We use FORTE to show that carbon taxes or credits are impractical in incentivizing carbon output reduction by providers of large-scale Internet applications. However, they can reduce carbon emissions by 10% without increasing the mean latency nor the electricity bill.

Categories and Subject Descriptors.
C.2 [**Internetworking**]: Network Architecture and Design
General Terms. Design, Management, Performance
Keywords. Green computing, Energy

1. INTRODUCTION

Internet-scale applications, such as social networks, video distribution networks, and content distribution networks, provide service to hundreds of millions of end users. They achieve their enormous scale, while simultaneously reducing access latency, by routing service requests to a set of geographically distributed servers, typically located in datacenters. Such datacenters, which host up to 200,000 servers, are large-scale consumers of electricity, which is used not only for powering servers but also for keeping them cool. Currently, datacenters that power Internet-scale applications consume about 1.3% of the worldwide electricity supply and this fraction is expected to grow to 8% by 2020 [22]. Forrest, Kaplan, and Kindler in the McKinsey Quarterly, November 2008 showed data center carbon emissions were 0.6% of the global total, nearly those of the Netherlands. By 2020, the fraction is expected to reach 2.6%, exceeding the carbon emission of Germany. A single provider of Internet-scale services, Google, consumed 2.26×10^6MWh in 2010 [14]. In the United States, on average, generating a kWh of electricity emits about 500g of carbon [41], so Google's carbon emission in 2010 is equivalent to that emitted by 280,000 cars, assuming that each car runs 10,000 miles per year and emits 4 tons of carbon [1].

Given the rapid growth in the scale and usage of Internet-based applications, our primary goal is to help socially-aware companies reduce the carbon footprint of their infrastructure. We would also like to factor in the carbon cost of upgrades to this infrastructure. Importantly, we realize that carbon cost is only one factor in a complex decision process faced by application providers: what is needed is a way to navigate the three-way tradeoff between carbon footprint, electricity cost, and access latency.

This is a difficult problem. In addition to the inherent large scale, the carbon footprint of a datacenter varies both spatially and temporally, as does request load. Although requests can be easily routed to datacenters [33], optimal request-routing requires a complex joint optimization of both request-routing and data placement. Moreover, the continuing growth of request load implies that it is also necessary to periodically upgrade the infrastructure, keeping in mind both the available budget and the expected future workload. These inherent complexities have not been addressed in recent work, which focus essentially only on either the electricity cost [30,35,37] or the carbon footprint of datacenters [9, 26, 29].

In this paper, we introduce FORTE, a Flow Optimization based framework for Request-routing and Traffic Engineering and Fast-FORTE, a heuristic that closely approximates FORTE but runs about 20 times faster. FORTE takes a principled approach to the problem by using an objective function that balances the weighted sum of access latency, electricity costs, and carbon footprint. Our major contributions are:

- FORTE, a request-routing framework that provides a three-way tradeoff between access latency, electricity cost, and carbon footprint;
- Using FORTE to analyze the costs of carbon emission reduction for a large-scale Internet application; and,
- Using FORTE to determine a green datacenter upgrade and expansion plan for an Internet application.

Our approach is scalable and leads to three non-intuitive results. First, unless the price of carbon is set unrealistically high, application providers have no incentive to reduce their carbon footprint by more than a few percent. Second, by exploiting regional variations in carbon footprint and electricity costs, an application provider such as Akamai can reduce their carbon footprint by nearly 10% using FORTE, with no impact on their electricity cost or access latency. Third, FORTE can find green upgrade plans that can reduce the carbon footprint by over 25% over three years, compared to a carbon-oblivious upgrade plan, again, with no impact on electricity cost or a bounded increase in access latency.

SIGCOMM'12, August 13–17, 2012, Helsinki, Finland.

	Coal	Oil	Gas	Nuclear	Hydro	Other	Total
Elec.(TWh)	8,263	1,111	4,301	2,731	3,288	568	20,261
Proportion	41%	5%	21%	13%	16%	3%	100%

Table 1: Source of Electricity (World total, year 2008)

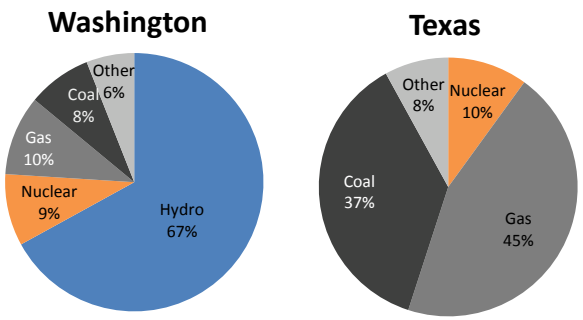

Figure 2: Generator fuel type in Washington and Texas

2. BACKGROUND

This section describes large-scale Internet services, the datacenters that host them, and the nature of electricity generation.

2.1 Datacenters and Request-Routing

We focus on large-scale Internet applications whose components run at multiple geographically distributed locations to provide scalability and reliability, and to reduce access latency (high access latency has been shown to have a negative economic impact [39]). For example, Google has more than 30 datacenters in at least 15 countries with an estimated 900K servers [31] and Akamai has more than 95,000 servers in nearly 1,900 networks in 71 countries [2].

Application providers usually place copies of data items such as video files or entire websites at one or more datacenters. Then, a request-routing system uses a set of metrics to direct end-hosts to datacenters that can best serve their requests. Current best practices for request-routing are *carbon oblivious*, that is, they ignore the amount of carbon emissions they generate. For example, a datacenter may be selected based on content availability, latency to the end-host, and load at that datacenter [33]. Our work modifies the request-routing algorithm to additionally take both the electricity cost and the carbon footprint of the datacenter into account.

2.2 Spatial and Temporal Variability in Electricity Carbon Footprint

Because of their scale, large-scale Internet services consume considerable amounts of electricity, which is delivered by the electricity grid. This grid inter-connects generator plants with consumers, including datacenters, in a geographical region. A region is usually served by several generators, which use different fuel types, such as gas, oil, coal, nuclear, and wind. Generation by burning fossil fuels emits much more carbon than generation with renewable energy, such as hydroelectricity, or nuclear plants. Two thirds of electricity is generated by fossil fuel today, as can be seen in Table 1, which shows fuel sources of electricity worldwide in 2008. However, there is significant difference among the fuel mix in different regions. A datacenter in Washington State uses cleaner electricity fuel sources than one in Texas. This is shown in Figure 2, which shows the fuel mix for these two states.

The carbon output of electricity generation varies temporally as well. The electricity grid has no storage, so the supply of electricity must match demand. Some generators are turned off in periods of low demand, and turned back on during peak hours. This is illustrated in Figure 3, which shows both the peak loads and the generation fuel mix in California on 23 January, 2012. The generators turned on during peak demand are usually thermal generators that emit more carbon. It is difficult to adjust the output of a nuclear plant, so they typically serve the base load.

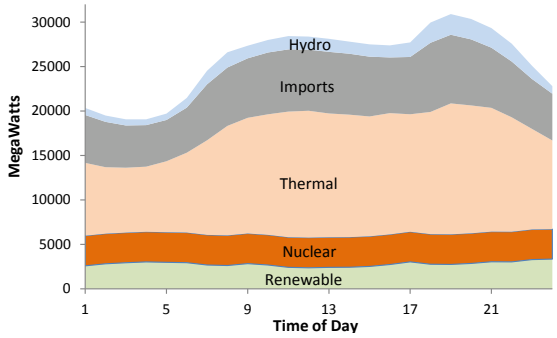

Figure 3: Hourly breakdown of total production by resource type, California, Jan 23rd 2012 [5]

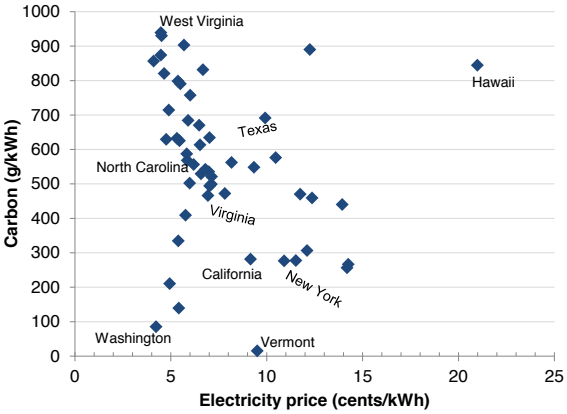

Figure 4: Grams of carbon per kWh vs. the cost of electricity in the 50 U.S. states.

There is no correlation between the "cleanness" of a region's electricity fuel mix and the price of electricity in the region. This is shown in Figure 4, which plots the cost and carbon output of electricity in each of the 50 U.S. states. The electricity in states like Washington is both clean and cheap, in contrast to states like West Virginia, where electricity is cheap but has a large carbon footprint. Consequently, schemes to reduce the electricity cost of powering an Internet service using request-routing (such as [35]) do not also reduce the carbon emissions of their operations.

2.3 Power Proportionality and PUE

Another way to reduce the carbon output of an Internet-scale application is to reduce the amount of energy its datacenters consume. There are two main ways to do this: power proportionality and lowering energy overheads. A *power-proportional* device uses power directly proportional to its utilization, e.g., if it is 10% utilized, then its power use will be 10% of its maximum power use [4]. A power-proportional datacenter can be built using power-proportional servers or by dynamically shutting off unnecessary equipment. Another approach to reducing datacenter power is to improve the effectiveness with which it uses electricity. This notion is captured by the *power usage effectiveness* (PUE) metric, which

is defined as:

$$\text{PUE} = \frac{\text{Total IT equipment power}}{\text{Total facility power}} \quad (1)$$

The most efficient datacenters have a PUE of around 1.07–1.2 [15, 34], while the industry average is around 2.0 [10]. That means the average datacenter uses nearly 50% of its power on cooling and power transformation.

A power-proportional datacenter with a low PUE is clearly desirable. However, a datacenter's carbon emissions still depend on its regional electricity fuel mix. Therefore, a service operator that wants to minimize its carbon output should build and use datacenters in regions with green sources of electricity. In the remainder of this paper, we design algorithms to navigate the three-way tradeoff between carbon footprint, access latency, and electricity cost.

3. FORTE

The location and time-specific nature of carbon emission rates offers a currently untapped opportunity for Internet applications, deployed across multiple datacenters, to reduce their carbon footprint by directing traffic to cleaner locations. In this section, we describe FORTE, a Flow Optimization based framework for request Routing and Traffic Engineering, that offers a principled approach to assigning users and data objects to datacenters. FORTE performs user assignment by weighting each assignment's effect on three metrics: access latency, electricity cost, and carbon footprint. By making this three-way tradeoff explicit, FORTE enables Internet application providers to determine their optimal operating point that balances performance with cost and carbon footprint. Any application that uses dynamic request-routing can be easily modified to use FORTE to assign users to datacenters.

3.1 Model

Determining the impact of different user assignment decisions requires data on the location of users and datacenters, regional electricity costs, and the carbon emission per watt at each datacenter location. Additionally, one must also consider what data users are interested in and the location of the data. To encapsulate this information, we use a simple abstract model representing the relationship between the following three entities:

1. **User Group**: A user group is a collection of users who are geographically close or in the same autonomous system, for example: users from the same city or customers of a regional ISP.
2. **Datacenter**: A datacenter is a collection of servers that can serve user requests frequently. We use the term datacenter generically, and use it also to represent a cluster in a content delivery network (CDN).
3. **Data**: A data is an abstract object that users request, such as a video in an online video website. It can be a webpage, including all associated images and text files. Data can also be associated with a cloud service such as search or email for a cloud service provider. Note that we only model popular data, as the transfer of popular data accounts for most of Internet traffic.

The relationship between these three entities is illustrated in Figure 5. User groups indicate their desire for data through data requests, and these requests are served by any datacenter with a copy of the data. Note that content distribution systems are complicated, so our model necessarily makes some simplifying assumptions. However, we believe that this model largely captures the behavior of content distribution systems, and therefore our conclusions are robust to changes in the model's parameters.

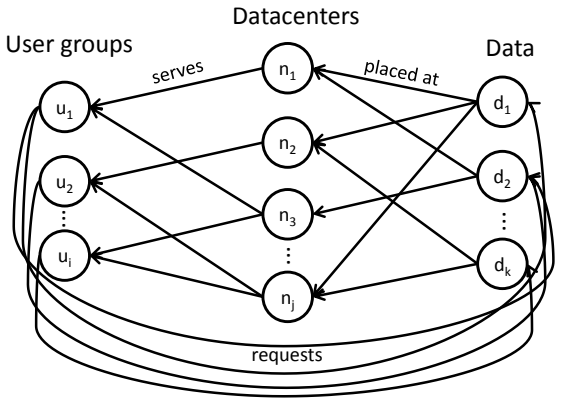

Figure 5: The relationship between user groups, datacenters, and data

We model the relationship between users and data as a minimum cost flow circulation problem. Here, the user group to datacenter assignment is re-evaluated periodically to adapt to changing request patterns. In addition to mapping user groups to datacenters, FORTE can also migrate or copy data to different datacenters with changes in the workload. The following is a complete list of the model parameters:

- $\mathbf{c}(\mathbf{n_j})$ is the average server carbon emission rate of the datacenter n_j.
- $\mathbf{e}(\mathbf{n_j})$ is the average server electricity cost of datacenter n_j.
- $\mathbf{f}(\mathbf{u_i}, \mathbf{n_j}, \mathbf{d_k})$ is the size of the flow from user group u_i, served by datacenter n_j, and requesting data d_k.
- $\mathbf{l}(\mathbf{u_i}, \mathbf{n_j}, \mathbf{d_k})$ is the average latency cost of the link between user group u_i and datacenter n_j for data d_k.
- $\mathbf{m}(\mathbf{n_j})$ is the number of servers running in datacenter n_j
- $\mathbf{p}(\mathbf{n_j})$ is the average processing power of servers in datacenter n_j. It is normalized to the number of requests it can process concurrently.
- $\mathbf{r}(\mathbf{d_k}, \mathbf{u_i})$ is the number of users in user group u_i requesting data d_k.
- $\mathbf{rep}(\mathbf{d_k}, \mathbf{n_j})$ equals $true$ if data d_k has a replica at data center n_j.
- $\mathbf{u}(\mathbf{n_j})$ is the number of servers in datacenter n_j.

Note that we model latency cost as a two part function $l(u_i, n_j, d_k)$. When latency is less than $l_{\max}$, the latency cost is linearly proportional to latency. Beyond $l_{\max}$, latency cost grows quadratically in order to model the user tendency to abandon Internet applications with perceptible interaction delays [39].

The FORTE model also differentiates between different data types. Specifically, it categorizes data as either throughput-sensitive or latency-sensitive, and enables different assignment decisions to be made based on the data category. For a data d_k that is throughput-sensitive, the latency cost will be 0 when access latency is less than $l_{\max}$. This significantly relaxes the constraint on feasible assignment decisions and enables throughput-sensitive traffic to be directed to any datacenter within the cut-off latency without paying latency costs.

3.2 Design

Our model captures the relationship between user groups, datacenters, and data, which enables FORTE to reason about the optimal mapping of users to datacenters *and* data to datacenters. The following sections describe FORTE's algorithms for optimally performing these two mappings. We also describe Fast-FORTE, an ap-

proximation algorithm that significantly improves upon the performance and scalability of FORTE while maintaining near-optimal assignments.

3.2.1 Assigning Users to Datacenters

The user to datacenter assignment problem, as we had previously mentioned in Section 3.1, can be modeled as a flow circulation problem. There are two main objectives when performing this assignment: all user requests must be satisfied and the weighted summation of access latency, carbon emission, and electricity cost must be minimized. We formulate this as a linear program:

$$\underset{f(u_i,n_j,d_k)}{\text{minimize:}} \quad (2)$$

$$\sum_{u_i,n_j,d_k} f(u_i,n_j,d_k)l(u_i,n_j,d_k)$$

$$+ \lambda_1 \sum_{n_j} m(n_j)c(n_j) + \lambda_2 \sum_{n_j} m(n_j)e(n_j)$$

subject to:

$$\forall n_j, \sum_{u_i,d_k} f(u_i,n_j,d_k) \leq m(n_j)p(n_j) \quad (3)$$

$$\forall u_i, d_k, \sum_{n_j} f(u_i,n_j,d_k) = r(d_k,u_i) \quad (4)$$

$$\forall n_j, 0 \leq m(n_j) \leq u(n_j) \quad (5)$$

$$\forall u_i, n_j, d_k, f(u_i,n_j,d_k) \geq 0 \quad (6)$$

$$\forall u_i, n_j, d_k, \text{ s.t. } rep(n_j,d_k) = false, f(u_i,n_j,d_k) = 0 \quad (7)$$

This linear program minimizes its objective function, Equation 2, which represents the sum of access latency, electricity costs, and carbon emissions. The relative weight between the three metrics can be adjusted by changing the weight parameters λ_1 and λ_2. Equation 4 represents the constraint that all flows are served by some datacenters.

Additionally, we can easily incorporate several other constraints in this linear program to address pragmatic, deployment-related requirements. This includes: ensuring that the total load assigned to a datacenter is less than the datacenter's service capacity (Eq. 3), limiting the number of active servers in each datacenter (Eq. 5), and restricting users from requesting data d_k from datacenter n_j if a replica of d_k does not reside in n_j (Eq. 7). To ensure that the linear program return feasible results, we also restrict the flow and load balancing cost variables to be positive values (Eq. 6).

3.2.2 Assigning Data to Datacenters

In the previous section, we assumed that the data assignment is fixed and pre-determined, and use linear programming to solve the user assignment problem only. However, the data assignment is actually a parameter under our control, and in this section, we show how FORTE determines the optimal data assignment while assuming that each user has already been assigned to a datacenter.

As part of our data to datacenter problem formulation, we first define a temporary variable:

$$f(n_j,d_k) = \sum_{u_i} f(u_i,n_j,d_k) \quad (8)$$

which is the flow size of edge (n_j, d_k). Given this definition of edge flows size, we further define f_{th} as a percentile threshold across a set of flows.

From our analysis of Akamai traffic data (the dataset is described in Sec. 4.1), we found that flow sizes follow the Pareto Principle, that is, a small percent of large flows account for a large percent of total bytes transferred. We see in Table 6 that 10% of the largest flows account for 90.998% of the total bytes transferred. Therefore, the size of a flow is the primary factor in determining data placement; we decide whether we need a replica of data d_k at datacenter n_j by the size of $f(n_j, d_k)$. More precisely, in our problem formulation, data d_k is needed by datacenter n_j if $f(n_j, d_k)$ is among the f_{th} percentile of large flows.

Flow size percentile	Percentage of total bytes
10%	90.998%
20%	97.941%
30%	99.520%

Table 6: Flow sizes follow the Pareto Principle for our Akamai workload.

Algorithm 1 Algorithm for determining the initial data placement.

Input: User Data Request Matrix $R(d_k, u_i)$
Input: Network Latency Cost Matrix $L(u_i, n_j)$
Input: Percentile Threshold f_{th}
Output: Need of Data Matrix $NEED(d_k, n_j)$
  Assume $rep(d_k, n_j) = true$ for all (d_k, n_j)
  Find out $F(u_i, n_j, d_k)$ by linear programming
  for each (n_j, d_k) **do**
    if $f(n_j, d_k)$ is a f_{th} percentile large flow **then**
      $need(n_j, d_k) = true$
    else
      $need(n_j, d_k) = false$
    end if
  end for

To solve the data assignment problem without regarding the user assignment as a priori, we use the following algorithm.

First, we perform user assignment using the linear programming approach we introduced in Section 3.2.1 with the added assumption that all data is available at every datacenter. Using this user assignment, we then determine whether data d_k is needed at datacenter n_j. Algorithm 1 shows the details of this approach[1].

However, this algorithm does not account for changes in user request patterns. Accommodating user assignment changes in response to request pattern changes is relatively simple and straightforward; users would be redirected to a different datacenter by changing request-routing. Unfortunately, changes in user assignment will lead to changes in data assignment, which potentially requires migrating a large volume of data. Such data migration is unreasonable if the data is only needed a small number of times before being migrated to another location. Therefore, we limit data migration frequency in our data assignment algorithm by first defining two additional matrices and parameters:

- $rep\_ttl(d_k, n_j)$ indicates the time to live of data replica d_k at data center n_j. $rep\_ttl(d_k, n_j) = 0$ if the data d_k has no replica at data center n_j. $rep\_ttl(d_k, n_j) > 0$ if there is a replica and the value of $rep\_ttl(d_k, n_j)$ indicates the time to live.
- $rep\_counter(d_k, n_j)$ is a counter that tracks how frequently $need(d_k, n_j) = true$ recently.
- $rep\_initial\_ttl$ is the initial time to live.
- $rep\_threshold$ is a parameter deciding when to copy data. If $rep\_counter(d_k, n_j) > rep\_threshold$, data d_k will be actually copied to data center n_j.

[1]Please note that we use upper case letters to represent matrices and lower case letters to represent entries in a matrix.

Algorithm 2 Update Data Replication

Input: Need of Data Matrix $NEED(d_k, n_j)$
Input: Original TTL Matrix $REP\_TTL(d_k, n_j)$
Input: Original Replication Counter Matrix $REP\_COUNTER(d_k, n_j)$
Output: Updated Replication Status Matrix $REP\_TTL(d_k, n_j)$
Output: Updated Replication Counter Matrix $REP\_COUNTER(d_k, n_j)$
for each (d_k, n_j) **do**
  if $need(d_k, n_j) = true$ **then**
    if $rep\_ttl(d_k, n_j) = 0$ **then**
      $rep\_counter(d_k, n_j) + +$
      if $rep\_counter(d_k, n_j) = rep\_threshold$ **then**
        Copy data d_k to data center n_j
        $rep\_ttl(d_k, n_j) = rep\_initial\_ttl$
      end if
    else
      $rep\_ttl(d_k, n_j) = rep\_initial\_ttl$
    end if
  else
    $rep\_ttl(d_k, n_j) - -$
    if $rep\_counter(d_k, n_j) > 0$ **then**
      $rep\_counter(d_k, n_j) - -$
    end if
    if $rep\_ttl(d_k, n_j) = 0$ **then**
      Remove data d_k from data center n_j
      $rep\_counter(d_k, n_j) = 0$
    end if
  end if
end for

At every time step, we obtain the updated user data request matrix $R(d_k, u_i)$ and determine the $NEED(n_j, d_k)$ matrix by Algorithm 1. We increment $rep\_counter(d_k, n_j)$ by 1 if data d_k has no replica at data center n_j and $need(n_j, d_k)$ = $true$. If the counter reaches $rep\_threshold$, we replicate the data d_k to data center n_j. The $rep\_ttl(d_k, n_j)$ will decrease one if $need(n_j, d_k) = false$ at this time step. This process is described in Algorithm 2. By using Algorithm 2, we guarantee that only data replicas that tend to be used frequently will be replicated and only data that is infrequently used for a long time will be removed. This reduces unnecessary data migration across the datacenters.

3.2.3 Fast-FORTE

FORTE optimally assigns users to datacenters to minimize the cost function. However, its reliance on linear programming limits its scalability. In our experiments, FORTE's optimizer takes 2 minutes to run on a standard personal computer using a partial trace of Akamai traffic data as input. Because measurements are taken every 5 minutes, this allows us to essentially solve the assignment problem in real time. However, this is unlikely to be fast enough for a complete CDN network, given the rapid growth rates of Internet services. Hence, the number of data objects and user groups in the user-datacenter-data model (as shown in Figure 5) may be much larger than those in our simulations. For these scenarios, FORTE can be very slow as the complexity of the Simplex method grows quadratically. We therefore designed an alternative solution, called Fast-FORTE, that uses the same model as FORTE but replaces the linear programming flow solver with a heuristic algorithm based on entropy. Before describing the algorithm in detail, we first define the aggregate cost of a link between user group u_i and datacenter n_j as:

$$a\_cost(u_i, n_j) = l(u_i, n_j) + \lambda_1 \frac{c(n_j)}{p(n_j)} + \lambda_2 \frac{e(n_j)}{p(n_j)} \qquad (9)$$

which is the cost to route user u_i to datacenter n_j. We also define

Algorithm 3 Fast-FORTE User Assignment

Input: User Data Request Matrix $R(d_k, u_i)$
Input: Network Latency Cost Matrix $L(u_i, n_j)$
Input: Data Replication Matrix $REP(d_k, n_j)$
Output: Flow Assignment Matrix $F(u_i, n_j, d_k)$
  Set all $f(u_i, n_j, d_k) = 0$
  Calculate $A\_COST(u_i, n_j, d_k)$
  Calculate $E(d_k)$
  Sort $A\_COST(u_i, n_j, d_k)$ in ascending order and put them in queue
  Sort $E(d_k)$ in descending order
  while Queue of $A\_COST(u_i, n_j, d_k)$ not empty **do**
    pop out link (u_i, n_j)
    put $E(d_k)$ in queue
    while Queue of $E(d_k)$ not empty **do**
      Pop out data d_k
      if $rep(n_j, d_k) == true$ **then**
        if $remaining(n_j) \geq r(d_k, u_i)$ **then**
          $f(u_i, n_j, d_k) = r(d_k, u_i)$
        else
          $f(u_i, n_j, d_k) = remaining(n_j)$
        end if
      end if
    end while
  end while
  Output flow assignment

the weighted entropy of data d_k

$$\begin{aligned} & E(d_k) \\ =& P(d_k) H(U|D = d_k) \\ =& -P(d_k) \sum_{u_i \in U} P(u_i|d_k) \log P(u_i|d_k) \end{aligned} \qquad (10)$$

where $P(u_i|d_k)$ is the probability that, given a request is asking for data d_k, it comes from user group u_i. The weight $P(d_k)$ is the probability that a request is asking for data d_k. The entropy value $H(U|D = d_k)$ measures the uncertainty of the requests. When the entropy reaches its maximum, we are uncertain about the location of requests as the location probability of each request is evenly distributed.

The Fast-FORTE algorithm is as follows. First, links between u_i and n_j are sorted by their aggregate cost $a\_cost(u_i, n_j)$. Data objects are also sorted by their weighted entropy $E(d_k)$. At each step, we select the link with the minimum aggregate cost and remove it from the graph. We then attempt to fill this link with data requests until we have exhausted all data requests. As we fill the link with data requests, we also attempt to fulfill requests asking for data with larger weighted entropy. The details of the algorithm is described in Algorithm 3. We use $remaining(n_j)$ to keep track the remaining capacity of datacenter n_j.

We evaluated Fast-FORTE by using the same experimental data we used for evaluating FORTE (described in Sec. 4.1). Fast-FORTE finds a solution in approximately 6 seconds compared to 2 minutes for FORTE. To estimate the quality of approximation, we calculate the approximation ratio:

$$Ratio = \frac{\text{Objective in Equation 2 by Approximation}}{\text{Objective in Equation 2 by LP}}$$

The average of approximation ratio is 1.00320 with a standard deviation 0.00558. Among the data we tested, the approximation ratio is less than 1.03 for 99.5% of the data points.

3.3 Using FORTE for Upgrading Datacenters

Determining a user and data assignment strategy that strikes a

good balance between latency, electricity cost, and carbon footprint only addresses half of the assignment problem. To datacenter operators, determining how to upgrade their datacenters in response to growth is equally important, as Internet traffic has maintained an explosive growth rate of 40 percent per year in the past decade (see, e.g., [25]). Specifically, datacenter operators need to determine which datacenters should be upgraded, and how many servers within each datacenter should be upgraded. Simply upgrading the greenest datacenters first is not optimal, as it does not properly weigh the three-way tradeoff. For example, upgrading a green datacenter in a remote area will provide minimal benefits as, due to the high latency to end-users, most traffic will be directed away from it. In this section, we extend FORTE to help decision makers determine the optimal upgrade plan. The objective of upgrading is to ensure that an application service provider can meet user traffic demands while minimizing electricity cost and carbon emissions. This is made easier by assuming the continued adherence to Moore's law, which allows us to assume that servers in datacenter can be replaced with new models that consume the same amount of energy but provide twice the computing power. In this modified model, we must introduce three new variables:

- $\mathbf{a(n_j)}$ is the amortized purchasing cost of upgrading a server at datacenter n_j. The one time upgrading investment on the server is divided by its service period. e.g.: 5 years.
- $\mathbf{budget}$ is the maximum number of server that are allowed to upgrade by the budget.
- $\mathbf{up(n_j)}$ is the percentage of server that are upgraded.

The new model includes the cost and benefit of upgrading a datacenter. If the benefit of upgrading, such as reduction in latency, offsets the amortized purchasing cost $a(n_j)$, the datacenter will be upgraded. The following is the linear program formulation for solving the upgrade problem. The parts that differ from our previous user assignment linear program are marked in bold font.

$$\underset{f(u_i,n_j,d_k),up(n_j)}{\text{minimize:}} \tag{11}$$

$$\sum_{u_i,n_j,d_k} f(u_i,n_j,d_k)l(u_i,n_j,d_k)$$

$$+\lambda_1 \sum_{n_j} m(n_j)c(n_j) + \lambda_2 \sum_{n_j} m(n_j)e(n_j)$$

$$-\boldsymbol{\lambda_1} \sum_{\mathbf{n_j}} \mathbf{u(n_j)up(n_j)c(n_j)} - \boldsymbol{\lambda_2} \sum_{\mathbf{n_j}} \mathbf{u(n_j)up(n_j)e(n_j)}$$

$$+\boldsymbol{\lambda_3} \sum_{\mathbf{n_j}} \mathbf{u(n_j)up(n_j)a(n_j)}$$

subject to:

$$\forall n_j, \sum_{u_i,d_k} f(u_i,n_j,d_k) \leq m(n_j)p(n_j) \tag{12}$$

$$\forall u_i,d_k, \sum_{n_j} f(u_i,n_j,d_k) = r(d_k,u_i) \tag{13}$$

$$\forall \mathbf{n_j}, \mathbf{0} \leq \mathbf{m(n_j)} \leq \mathbf{u(n_j)(1+up(n_j))} \tag{14}$$

$$\forall u_i,n_j,d_k, f(u_i,n_j,d_k) \geq 0 \tag{15}$$

$$\forall u_i,n_j,d_k, \text{ s.t. } rep(n_j,d_k) = false, f(u_i,n_j,d_k) = 0 \tag{16}$$

$$\forall \mathbf{n_j}, \mathbf{0} \leq \mathbf{up(n_j)} \leq \mathbf{1} \tag{17}$$

$$\sum_{\mathbf{n_j}} \mathbf{u(n_j)up(n_j)} \leq \mathbf{budget} \tag{18}$$

We highlight four equations that are changed in this model:

Equation 11: The new expression adds the total amortization cost

$$\sum_{n_j} u(n_j)up(n_j)a(n_j)$$

into the objective. If the amortized purchasing cost of adding a server is smaller than the extra latency cost to direct further away traffic to this datacenter, it will then be worthwhile to add a new server. Since the new model consumes the same amount of energy as the original one, upgrading a server will not lead to extra electricity cost nor carbon emission. The term

$$-\lambda_1 \sum_{n_j} u(n_j)up(n_j)c(n_j) - \lambda_2 \sum_{n_j} u(n_j)up(n_j)e(n_j)$$

removes the extra carbon counted by

$$+\lambda_1 \sum_{n_j} m(n_j)c(n_j) + \lambda_2 \sum_{n_j} m(n_j)e(n_j)$$

when $up(n_j) > 0$.

Equation 14: Note that in the new model, $m(n_j)$ is the effective computing power rather than the number of running servers. Therefore, it is possible that $m(n_j) > u(n_j)$. When 20% of servers in datacenter n_j are upgraded ($up(n_j) = 0.2$), the effective computing power $m(n_j) = 1.2 \times u(n_j)$, as 20% of the servers double their computing power. $up(n_j)$ can be viewed as a "slack variable" to $m(n_j)$. The value of the slack variable $up(n_j)$ determines the amount of growth $m(n_j)$ that is needed to reduce the total cost (Equation 11). In other words, the value of $up(n_j)$ is the optimal upgrading percentage to minimize cost.

Equation 17,18: These two constraints limit the range of upgrading percentages and the total number of servers to upgrade.

Linear programming will find the trade-off between the cost and benefit of adding new servers at each location. The output of $up(n_j)$ indicates the percentage of server that needs to be upgraded in datacenter n_j. The input traffic matrix should be the average value of a relatively long period, say a month, to give stable results.

4. RESULTS

We now evaluate the carbon emission reductions that are possible using FORTE. Throughout our evaluation, we use the Akamai content distribution network (CDN) as a case study. We begin by describing our simulation methodology and the Akamai dataset; we then find the maximum amount of carbon emission reductions possible before evaluating the three-way tradeoff between carbon emissions, latency, and electricity costs. Finally, we give the results of using FORTE to upgrade datacenters.

4.1 Methodology

We implemented a custom discrete time simulator to evaluate FORTE, using a 24-day workload from Akamai. In the following sections, we describe Akamai's infrastructure and this dataset, how we estimate the carbon emissions of each datacenter, and our approximation of latency between a user and datacenters.

4.1.1 Akamai CDN

We use Akamai as a case study to evaluate the carbon emission reduction potential of an Internet-scale system. Akamai is the largest content delivery network in the world. It delivers 15–20 percent of worldwide Internet traffic [33] and it has more than 95,000 servers in 1,900 networks across 71 countries [2].

The Akamai workload we use as a case study was collected from 18 December 2008 through 11 January 2009, and was col-

lected from a subset of their datacenters. The datacenters included are Akamai's public datacenters, which are generally located inside of co-location centers. As opposed to Akamai's private CDN nodes, these public datacenters can serve requests for any client. The dataset does not include logs for private Akamai datacenters. These are typically located within a university or enterprise network. The workload contains logs for users and CDN datacenters worldwide; however, the information given on users outside of North America is not detailed enough for our purposes, so we omit it from our simulations.

The dataset consists of measurements taken over a 5 minute interval. Each measurement contains the geographical origin of user requests, the datacenter that served the requests, and the aggregate size of the requests. We use FORTE to compute an updated user-to-datacenter assignment for each measurement. Then, this assignment is used for the duration of the 5 minute interval represented by the measurement. Because these logs contain Akamai's actual assignment of users to datacenters, we can compare FORTE to Akamai's proprietary request-routing scheme, even though we have no details about their scheme. In Section 4.2, we show that FORTE can closely approximate this scheme.

The logs include the load factor for each datacenter. This is estimated by the combination of CPU, memory, and network utilizations. We use the load factor together with the user request logs to estimate the capacity of each datacenter.

The user request logs contain only aggregate statistics, and do not provide us with the type or name of content requested by the user. Akamai handles interactive requests, which are latency-sensitive, as well as streaming requests, which are throughput-sensitive. To account for this, we assume that 90% of user requests are latency-sensitive, and that the remaining 10% of requests are not sensitive to latency. Recall that FORTE models user access latency costs for latency-sensitive traffic with a two-part function. For distances less than $l_{\max}$, the user access latency cost is linearly proportional to latency. Beyond $l_{\max}$, latency cost grows quadratically in order to discourage FORTE from routing users to far away datacenters. For our Akamai case study, we set $l_{\max}$ equal to 2,000 km which is half of the distance across the length of the U.S.. FORTE does not include access latency in its objective function for throughput-sensitive traffic, so these users are routed to datacenters with low carbon footprints, regardless of their distance.

4.1.2 Power proportionality and bandwidth

For all simulations, we assume that the datacenters are fully power-proportional . Most datacenters today are not power-proportional, though the industry is heading that direction [4]. One challenge here is that servers are not power-proportional. However, a datacenter can be made power-proportional by shutting off unused servers [28] and switches [18]. There are also proposals for power-proportional storage systems [3, 24]. By assuming that datacenters are fully power-proportional, we find an upper bound on the amount of carbon emission reductions possible with FORTE.

Further, we assume that bandwidth cost increases are insignificant. FORTE does not increase the number of user requests, and its optimization procedure minimizes the number of times a data item must be retrieved. However, because it routes users to minimize carbon emission, it is possible that it increases bandwidth costs. Currently, bandwidth costs are based on the 95/5 bandwidth pricing scheme, which divides traffic into 5 minutes intervals and uses the 95th percentile for pricing. Because bandwidth prices are not geographically differentiated for Akamai [35], re-routing traffic to another datacenter does not increase the overall bandwidth cost. However, affecting the 95th percentiles does. In our simulations, we found that the sum of the 95/5 values for each datacenter using the request-routing assignments found by FORTE was 97% of the sum of 95/5 values found by Akamai's scheme. This indicates that FORTE does not significantly affect the 95th percentiles. Furthermore, bandwidth is relatively inexpensive today, with network bandwidth expenses only contributing a couple of percent to the total cost of operating a datacenter [16]. Finally, if bandwidth costs are a concern, then they can easily be incorporated into FORTE's optimization constraints.

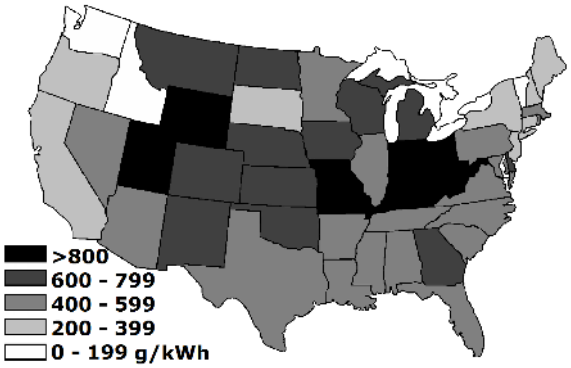

Figure 7: Carbon emission rate of each state in U.S. The average carbon emission rate of U.S. is 562g/kWh. Most of Akamai datacenters are in states with carbon emission rate between 200-599g/kWh.

	Nuclear	Coal	Gas	Oil	Hydro	Wind
$CO_2\ g/kWh$	15	968	440	890	13.5	22.5

Table 9: Carbon dioxide emission per kilowatt-hour for the most common fuel types [19].

4.1.3 Carbon footprint data

To estimate the carbon emission rate of each datacenter, we use electricity generation data from the U.S. Energy Information Administration's website [41]. They report the average electricity fuel mix of all the states in U.S. for the seven major types of fuel. Then, the average carbon emission rate of a state is found by summing the weighted contribution from each fuel type. This is defined as:

$$\text{state's avg. carbon emission rate} = \frac{\sum e_i \times r_i}{\sum e_i} \qquad (19)$$

where e_i is the electricity generated from fuel type i and r_i is the carbon emission rate of fuel type i. The carbon emission rate of the most common fuel types is shown in Table 9. Of course, data at a finer grain than state-level would improve the fidelity of our results. However, as carbon emission rates are often governed by state-regulated legislation and power-generation, we believe that state-level carbon emission data is sufficiently fine grain for our model.

Figure 7 shows our estimate of the carbon emissions caused by electricity generation in each state. The carbon emissions of locations with an Akamai datacenters is summarized in Figure 8. We see that, despite not directly optimizing for carbon emissions, Akamai's datacenters are typically in low-polluting locations, and most of its datacenters are located in states with carbon emission rate less than U.S. average (562 g/kWh).

As described in Section 2.2, the carbon emission rate of electricity generation varies with both location and time, because the fuel mix changes with the turning on/off of peak plants. We do not have hourly fuel mix data of states in the U.S.. However, we know that most of peak plants are powered by oil and gas [8], so peak plants emit more carbon than base load plants. We estimated the daily

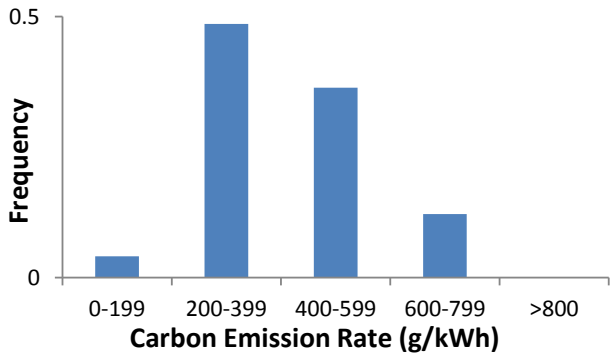

Figure 8: Distribution of the carbon emissions of Akamai datacenters.

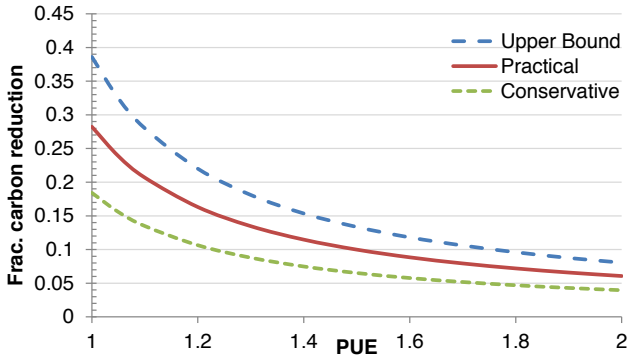

Figure 10: Percentage of carbon emission reduction with various PUEs and different schemes. The practical scheme increases the average user distance and the conservative scheme fixes both distance and electricity cost to their respective values under Akamai's routing scheme.

change on the carbon emission rate of each state by the change of total load and composition of fuel source. On average, the maximum carbon emission hour for a state emits 63% more carbon than its daily average.

4.1.4 *Approximating latency with distance*

The Akamai dataset does not include latency measurements, and latency estimation is a challenging problem. Therefore, we approximate access latency with geographical distance in our simulations. Although distance is not a perfect estimator of network latency, we believe that it is sufficiently accurate to determine the relative rank in latency from datacenters to each end-user. Therefore, for the remainder of this section, we use geographical distance as an approximation for latency. In practice, CDN providers have accurate measurements on network latency, so they can use their datasets to improve latency estimation.

4.2 Carbon Emission Reduction

We begin by determining the maximum amount of carbon reduction that is possible for Akamai. The reductions possible depends on the PUE of the datacenters in the distribution network, because the carbon output per unit of work is lower in datacenters with a lower PUE. We found the upper bound on carbon reduction possible as the PUE of the datacenters varies, assuming all datacenters have the same PUE. The results are shown in Figure 10. Here, carbon emission reduction is the main objective of FORTE's optimization—electricity costs and latency have a relatively smaller weight. Therefore, we can estimate the upper bound of carbon reduction for different PUEs.

As expected, as datacenter efficiency increases, FORTE achieves greater carbon emission reductions. For inefficient datacenters with a PUE of 2, we found that FORTE can reduce carbon emissions by 8%, while carbon emissions can be reduced over 38% if the datacenters have a PUE of 1. This graph demonstrates the incentive to further reducing PUE for a system whose current PUE is small. We do not have the PUE of the datacenters in the Akamai dataset; however, the industry average PUE was 2.0 in 2007 [10] and was 1.91 in 2009 [40]. The best datacenters have a PUE as low as 1.07 [13, 34].

Figure 10 also shows the sensitivity of FORTE to datacenter PUE. We vary the PUE of the CDN datacenters and found the carbon reduction achieved by two schemes, a practical scheme and a conservative scheme, as well as the upper bound on carbon reduction. For the practical scheme, we fixed the electricity cost at the current Akamai level. We found that this scheme increased the average distance from 745 km to 1000 km; however, FORTE does not minimize distance for throughput-sensitive traffic, so this traffic disproportionally affects the average distance. The conservative scheme fixes both distance and electricity cost to the current Akamai levels. From the figure, we find that PUEs of less than 1.5 is the sensitive region, which motivates datacenter owners to further reduce their PUE. For the remainder of our simulations, we assume that all datacenters have a PUE of 1.2. In the 2007 E.P.A. report [10], they indicate that best-practices yield datacenters with PUEs of 1.3. Therefore, we believe that a 1.2 PUE is a reasonable estimate for a modern best-practice datacenter.

Next, we evaluated the effects of changing the weight of carbon emissions in FORTE's objective function. That is, we evaluate the effect of changing λ_1 on carbon emissions. We used four values for λ_1 and fixed all other parameters. The results are shown in Figure 11. The chart shows carbon emission levels, normalized to Akamai's carbon emissions, for the duration of the Akamai trace. We see that even if λ_1 is set to a very small value, we have around 4% carbon emission reduction. This is because users can be assigned to cleaner datacenters within a similar distance, so carbon emission reductions are found without increasing latency. When λ_1 is set to a large value, carbon emissions can be reduced up to 20% on average.

We also observed that:

- The major difference in carbon reduction between $\lambda_1 =$ small and $\lambda_1 =$ medium is at non-peak hours.
- The major difference in carbon reduction between $\lambda_1 =$ medium and $\lambda_1 =$ large is at peak hours.

This shows that when the load is low, FORTE has more freedom to assign users to cleaner datacenters. Therefore, as presented in Section 3.3, FORTE indicates where are the desirable places for datacenter upgrading. The result of datacenter upgrading is demonstrated in Section 4.4.

4.3 Costs of Carbon Reduction

We have shown that FORTE can reduce an Internet application's carbon emissions; however, these reductions may come at the price of increased access latency and electricity cost. To explore the tradeoffs an operator can make, we show the effects of varying distance and electricity costs on carbon emissions in Figure 12.

We find that we can reduce carbon emissions caused by Akamai's datacenters by 10% without increasing latency nor electricity cost from their existing levels. When carbon emissions are at 81% of the current Akamai level, the electricity cost reduces with the reduction of carbon emissions. This is because for some regions, carbon emission is positively correlated with electricity prices. In this situation, assigning users to datacenters powered by cleaner electricity will reduce both carbon emissions and electricity cost.

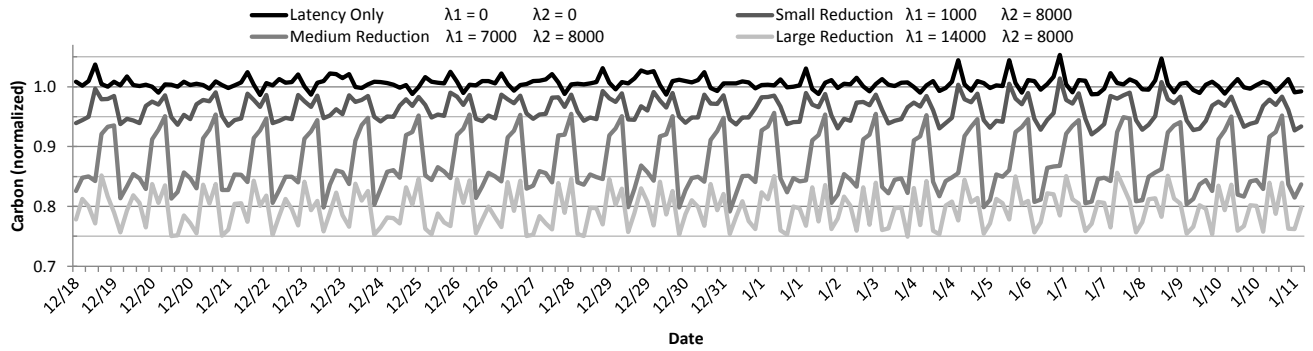

Figure 11: Carbon emissions with various weights of carbon in the FORTE objective function. The emissions shown are normalized against the carbon emissions by carbon-oblivious request-routing. The "latency only" one is our simulation on Akamai request-routing, which is very close to Akamai's emission level.

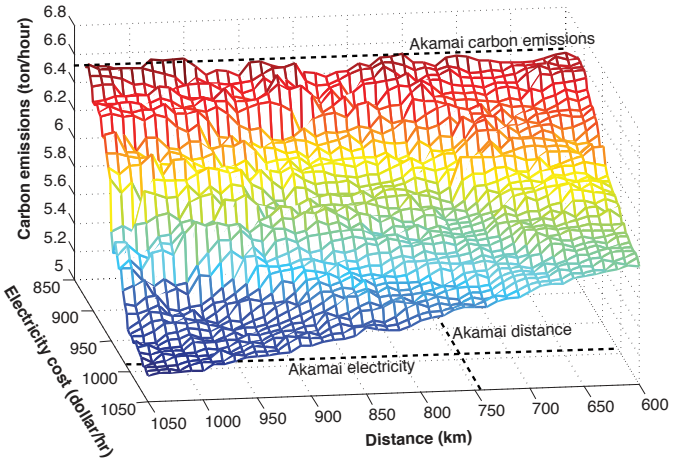

Figure 12: Tradeoff between carbon emissions, average distance, and electricity costs.

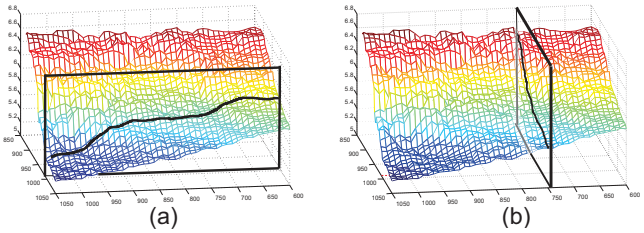

Figure 13: Locations of the slices plotted in Figures 14 and 15.

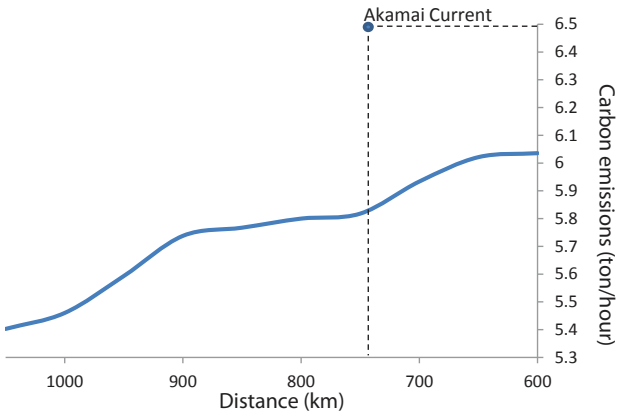

Figure 14: Carbon emissions vs. average user distance when electricity costs are fixed to Akamai's current costs.

To better understand the 3D surface plotting carbon emissions, electricity cost, and distance, shown in Figure 12, Figures 14 and 15 show the results of fixing electricity cost to the current Akamai cost and of fixing distance to the current Akamai distance respectively. Figure 13 shows the locations of these fixed points.

Figure 14 shows carbon emission vs. distance when we fix the electricity cost to Akamai's current electricity cost. We can reduce carbon emission by 10% without affecting distance. Electricity cost reductions beyond 20% increase distance linearly. After this point, further carbon emission reductions are cost prohibitive.

Figure 15 shows carbon emission vs. electricity cost when the average distance is fixed to Akamai's current distance. We observe that FORTE reduces carbon emissions by nearly 10% without increasing electricity cost. However, after this point, further carbon reduction is very expensive, as the reduction curve flattens out. To reduce the carbon emission by 0.1 ton per hour, the extra electricity cost is $23. The cost of unit carbon reduction is $230/ton, which is significantly higher than the current carbon tax ($5/ton to $39/ton [21]). Hence, paying the carbon tax is less expensive than reducing carbon emission.

4.4 Upgrading Datacenters

To evaluate FORTE's datacenter upgrade algorithm, we again use Akamai as a case study; however, we now assume that their traffic growth rate is 40% per year. (This growth rate is consistent with recent measurements [25].) We assume that each year, the datacenter operator has a budget large enough to increase the total number of servers by 15%. To simplify our analysis, we assume that upgrades are performed in June each year.

Each month's traffic estimation is used by FORTE to find the upgrading plan for each month. Let $up_{y,m}(n_j)$ denote the percentage of servers to be upgraded in datacenter n_j at month m of year y. We use the 12 month running average of $up_{y,m}(n_j)$ to calculate the upgrading plan for year y. Mathematically, we upgrade datacenter n_j in year y by:

$$up_y(n_j) = \alpha \sum_{m=1}^{12} (1-\alpha)^{12-m} up_{y,m}(n_j) \qquad (20)$$

where α is a parameter controlling the weight of recent months.

We compared FORTE's upgrading algorithm with a carbon oblivious uniform upgrading scheme, which adds the same percentage of server each month. The total number of servers added for FORTE and the carbon oblivious scheme are the same. Figure 16 shows the carbon emissions of both schemes over a 36 month period. Up-

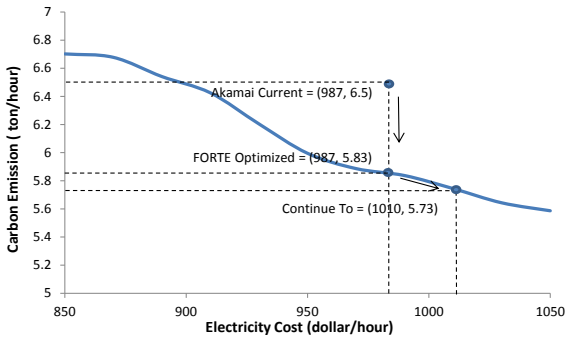

Figure 15: Carbon emissions vs. electricity costs when average distance is fixed at current Akamai levels.

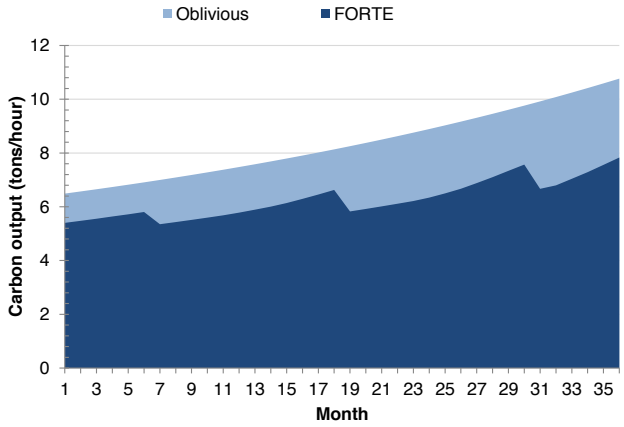

Figure 16: Carbon emissions over time in the datacenter upgrade scenario. FORTE performs upgrades every June, hence the drop in carbon emissions every 12 months.

grades take place at months 7, 19, and 31, which can be seen as dips in the carbon emission. The actual percentage of servers purchased each year is shown in Table 17. The carbon emission growth rate is also lower using FORTE's algorithm. At month 1, FORTE's solution has 83.22% of the carbon emission that the oblivious uniform upgrading scheme's solution has. At the end of month 36, FORTE's carbon emission is 72.80% of the carbon oblivious one. Therefore, FORTE lowers the carbon emission growth rate.

The datacenters upgraded are primarily near the most densely populated regions in United States such as the west coast, the northeast and the south. We selected some typical upgraded datacenters and plot the percentage of servers upgraded each year in Figure 18. When we compare Figure 18 with Figure 7, we see that datacenters selected for upgrades are primarily from the states with cleaner electricity generation. The state of Washington, California, New York, and New Jersey all have carbon emission rates less than 400 g/kWh. Although Texas has a emission rate of 562 g/kWh, it is a relatively clean state in the southern U.S., so upgrading is beneficial because it reduces latency for users in the southern part of the country.

5. DISCUSSION

Impact on application service providers: Application service providers fall into three broad classes. The first class, like Amazon, Google, Facebook, and Microsoft, are vertically integrated and own both the application as well as the underlying request-routing framework and datacenters. Such providers can use FORTE as-is to optimize their request-routing algorithms and datacenter upgrades. The second class of service providers, like Amazon Cloudfront,

Month	Upgrading Percentage
6	5.365%
18	9.872%
30	10.107%

Table 17: Percentage of servers upgraded among all servers

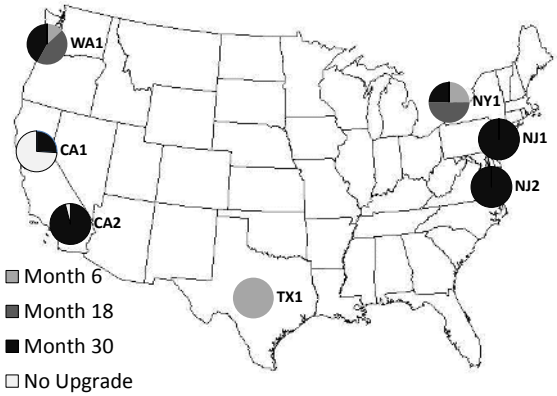

Figure 18: Three year upgrade plan for selected datacenters

Microsoft Azure, AT&T, and Telefonica, own and operate datacenters that host applications and also provide a request-routing framework, but do not own the applications that they host. Such providers can also use FORTE as-is by monitoring the usage of the applications that they host. The third class of service providers, like Akamai, Limelight, Cotendo, and CDNetworks, own neither datacenters nor applications, and primarily provide a request-routing service and access to servers collocated at datacenters around the world. Although such providers can use FORTE to modify their request-routing algorithms, our approach is less applicable to them because they do not actually pay for energy but the capacity of power provisioned to the servers (a fact also noted by [35]). However, they can still use FORTE to decide which collocation sites to upgrade in an effort to minimize their overall carbon footprint.

Will carbon taxes or credits work? The price of electricity today does not reflect externalities such as its carbon footprint. It has been suggested[2] that this can be corrected by a carbon tax (or its converse, a carbon credit for green generators). Can these affect the behavior of Internet-scale application providers?

We believe that, at least for the near future, the answer is negative, because the carbon cost of a server is under 2% of the electricity cost. To see this, consider a typical server that uses approximately 250 W = 0.25 kW [23] in a region where the electricity cost is 11.2 c/kWh, the US average [41]. The hourly electricity cost of operating the server would be 0.25 * 0.112 = $0.28/hour. Recall that the mean carbon footprint of electricity in the US is 562g/kWh. The server's carbon emissions are therefore 0.25*562g = 140.5g/hour. The US currently does not impose a carbon tax. However, the European Union suggests a carbon tax of $5 to $39 per 1000 kg = $0.000039/g [21]. Even at the high end of this scale, the carbon cost of the server would be only $0.00548/hour, which is less than 2% of the electricity cost. Given this disparity, unless carbon taxes are at least an order of magnitude larger, which is unlikely for the next few years, they will have little effect on application service providers.

We believe, instead, that carbon footprint reductions can be achieved using a combination of intelligent request-routing and non-cash incentives. Section 4 demonstrates that, using FORTE, a service

[2]http://www.carbontax.org/

provider can reduce its carbon footprint by up to 10% with no impact on its electricity cost or mean access latency. The increased electricity cost of additional reductions could be balanced by the goodwill this can generate with the general public: a cost, like advertising, that corporations are often willing to bear.

A principled approach to datacenter location: FORTE provides a principled approach to decide where to build new datacenters and how large these should be. A service provider indicates candidate locations to FORTE by placing zero-size data centers in these locations and then running the upgrade algorithm presented in Section 3.3. An upgrade to a datacenter with zero servers to a datacenter with n servers is readily interpreted as a selection of this location for a new datacenter with n servers. In reality, location decisions are often heavily influenced by various incentives offered by authorities trying to attract datacenters to their location. It is straightforward to reflect these incentives in the objective function by adjusting the cost to build a datacenter in a given location.

Datacenter emergy: Electricity consumption is only part of an Internet service's carbon footprint. A more complete life-cycle assessment should incorporate additional environmental impacts, such as the energy and materials used to produce the datacenters it is deployed on. *Emergy* is the amount of energy used to manufacture a product, that is, it is the embodied energy of the product. The emergy of a product can be significant; for example, a study from 2004 estimated production to account for 81% of a computer and display's total life-cycle energy use [43]. Additionally, Raghavan and Ma estimate that embodied power is 53% of the Internet's total power use [36]. The relative contribution of emergy to a server's life-cycle energy use is less: it accounts for at most 21% to 10% of its total energy use, assuming an average power of 250 W to 500 W and a three-year lifespan and that a typical server has 5 GJ of emergy [6]. Because a datacenter is not just servers, the emergy of a datacenter may be considerably higher. Therefore, estimating the relative importance of datacenter emergy is an interesting direction for future work.

6. RELATED WORK

Request-routing and electricity: There has been considerable interest in reducing the electricity consumption of datacenters and networks in recent years. Qureshi et al. proposed a request-routing scheme to minimize the electricity bill of multi-datacenter systems [35]. They showed that this sort of optimization can reduce the electricity bill by 13% when assuming power-proportional datacenters. Later work, such as [30, 37], has improved the algorithms of Qureshi et al.. Other proposals consider how request-routing can be used to help with load balancing in the electric grid [32].

Recently, there has been work investigating modifying request-routing to reduce carbon emissions. Le et al. [26] considered the joint optimization problem of minimizing carbon emission and electricity cost. Their user assignment algorithm does not attempt to minimize the distance between users and the datacenters that serve them, so they are not able to capture the full space of tradeoffs. Liu et al. [29] also proposed an algorithm to geographically load balance users while taking carbon into account. Their algorithm assigns users to datacenters based on access latency and costs only; however, they put a dollar cost on carbon emissions for their analysis. As previously discussed, the per dollar cost of carbon is low compared to the cost of electricity, so carbon emission needs to be more explicitly considered in the optimization procedure. Liu et al. use their algorithm to investigate whether user load balancing can facilitate the introduction of stochastic energy sources (like wind and solar) to the electric grid. In [9], Doyle et al. proposed adjusting the number of servers running in datacenters to tradeoff between latency and carbon emissions. In this preliminary work, the authors use a simple model of the problem, which does not take into account load distribution, network latency, and the temporal dynamics of carbon emission rates.

None of the previously proposed request-routing schemes allow operators to tune all three parameters: electricity costs, access latency, and carbon emissions. FORTE solves this problem, and specifically we make the following contributions beyond previous works: (1) we model data at datacenters, whereas previous work ignore this and assume full replication of the data; (2) we propose a new algorithm for user assignment that accounts for the location of data, whereas previous works ignore data locality; (3) we propose an algorithm to plan datacenter upgrades such that carbon emissions are minimized; and, (4) we use this algorithm to thoroughly explore the tradeoffs between latency, electricity costs, and carbon emissions.

New architectures for service energy reduction: Others have considered new architectures to reduce the electricity costs of content distribution. Vytautas et al. proposed the use of nano datacenters [42], which act similar to peer-to-peer distribution networks, to replace the current centralized model. They argue that fixed costs such as cooling consume a large proportion of energy in datacenters, which is avoided by nano datacenters, because a nano datacenters are freely cooled by ambient air. Jourjon et al. [20] also proposed a peer-to-peer-based architecture; however, they have not yet evaluated their proposal. Uichin et al. [27] and Guan et al. [17] proposed architectures to reduce content distribution energy costs using content-centric networking (CCN). A CCN incorporates storage components into switches and routers in the network, enabling them to cache content. This may reduce energy use for serving static content distribution, but does not work for dynamic content.

These alternative approaches are promising, but it is unclear whether they will save energy. Others have argued that peer-to-peer content delivery architectures use more electricity overall than centralized models like CDNs [11]. However, these architecture do reduce the electricity use of datacenters, but because they increase overall electricity use, they will increase carbon emissions overall.

Datacenter upgrade and expansion: Site selection for new datacenters has traditionally been performed based on factors such as susceptibility to natural disasters, electricity prices, land prices, tax breaks, workforce, and availability of Internet connection points [38]. Recently, Goiri et al. [12] proposed a rigorous framework for datacenter site selection. Their solution automates site selection using optimization, and their algorithms simultaneously optimize many objectives, including minimizing costs and carbon emissions. Expanding the capacity of existing datacenters has not been as widely studied in the literature. However, there has been work on datacenter network upgrades and expansions [7].

7. CONCLUSIONS

The carbon footprint of Internet-scale applications is a relatively small but still rapidly growing fraction of total emissions. To address this issue, we propose FORTE, a principled approach based on flow optimization to route users to datacenters. FORTE allows application service providers, such as content distribution networks, to navigate the three-way tradeoff between carbon footprint, access latency and electricity cost. To deal with the scale of the problem, we also describe Fast-FORTE, a greedy algorithm that obtains nearly the same results as FORTE but runs faster by at least an order of magnitude. Due to their underlying principled approach, both algorithms can be easily enhanced to take other factors, such as tax incentives and upgrade budgets, into account.

Using FORTE and Fast-FORTE to analyze traces from the Aka-

mai CDN, we find that our approach to request-routing can reduce Akamai's carbon footprint by 10% without increasing electricity cost, while simultaneously bounding the access latency. We also find that further reductions in carbon footprint come at the expense of either latency or electricity cost, and that these costs are unlikely to be offset by carbon credits or taxes. We modify FORTE to determine how best to upgrade datacenters in response to increases in request traffic. We find that, using our approach, under some simplifying assumptions, Akamai can reduce its carbon footprint by about 25% over three years, compared to a carbon-oblivious upgrade algorithm.

Acknowledgement

We thank Bruce Maggs for giving us access to the Akamai trace. We also thank our shepherd Fabián Bustamante and the anonymous reviewers for their insightful comments. This work is supported by HP Inc., National Science and Engineering Research Council, Canada, and the Canada Research Chairs Program.

8. REFERENCES

[1] Light-duty automotive technology, carbon dioxide emissions, and fuel economy trends: 1975 through 2011. *U.S. Environmental Protection Agency*.

[2] Akamai. Akamai reports fourth quarter 2010 and full-year 2010 financial results, 2011. http://tiny.cc/afyu9, Last visited Jan. 2012.

[3] H. Amur, J. Cipar, V. Gupta, G. R. Ganger, M. A. Kozuch, and K. Schwan. Robust and flexible power-proportional storage. In *SoCC*, 2010.

[4] L. Barroso and U. Holzle. The case for energy-proportional computing. *Computer*, 40(12):33 –37, dec. 2007.

[5] California ISO. Renewables watch, 2011. http://www.caiso.com/green/renewableswatch.html.

[6] J. Chang, J. Meza, P. Ranganathan, C. Bash, and A. Shah. Green server design: beyond operational energy to sustainability. In *HotPower*, 2010.

[7] A. R. Curtis, S. Keshav, and A. López-Ortiz. LEGUP: Using heterogeneity to reduce the cost of data center network upgrades. In *CoNEXT*, 2010.

[8] B. Dennis. Estimating a data center's electrical carbon footprint. *Schneider Electric White Paper 66*, 2011.

[9] J. Doyle, D. O'Mahony, and R. Shorten. Server selection for carbon emission control. In *GreenNet*, 2011.

[10] EPA. EPA report to congress on server and data center energy efficiency. Technical report, U.S. Environmental Protection Agency, 2007.

[11] A. Feldmann, A. Gladisch, M. Kind, C. Lange, G. Smaragdakis, and F.-J. Westphal. Energy trade-offs among content delivery architectures. In *CTTE*, 2010.

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

[13] Google. Efficient computing—step 2: efficient datacenters. http://www.google.com/corporate/green/datacenters/step2.html.

[14] Google green. http://www.google.com/green/, Last visited Jan. 2012.

[15] Google's Data Center Efficiency. http://www.google.com/about/datacenters/inside/efficiency/power-usage.html, Last visited Jan. 2012.

[16] A. G. Greenberg, J. R. Hamilton, D. A. Maltz, and P. Patel. The cost of a cloud: research problems in data center networks. *SIGCOMM CCR*, 39(1):68–73, 2009.

[17] K. Guan, G. Atkinson, D. Kilper, and E. Gulsen. On the energy efficiency of content delivery architectures. In *ICC*, 2011.

[18] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown. Elastictree: saving energy in data center networks. In *NSDI*, 2010.

[19] B. H. Joseph V. Spadaro, Lucille Langlois. Greenhouse gas emissions of electricity generation chains: Assessing the difference. *IAEA Bulletin*, 2000.

[20] G. Jourjon, T. Rakotoarivelo, and M. Ott. Models for an energy-efficient p2p delivery service. In *PDP*, 2010.

[21] J. Kanter. Europe considers new taxes to promote 'clean' energy, 2010. http://nyti.ms/xSlobE, Last visited Jan. 2012.

[22] J. Koomey. Growth in data center electricity use 2005 to 2010. *Analytics Press*, Aug. 2011.

[23] J. G. Koomey. Worldwide electricity used in data centers. *Environmental Research Letters*, 3(3):034008, 2008.

[24] A. Krioukov, P. Mohan, S. Alspaugh, L. Keys, D. Culler, and R. H. Katz. Napsac: design and implementation of a power-proportional web cluster. In *GreenNet*, 2010.

[25] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian. Internet inter-domain traffic. In *SIGCOMM*, 2010.

[26] K. Le, R. Bianchini, T. Nguyen, O. Bilgir, and M. Martonosi. Capping the brown energy consumption of internet services at low cost. In *IGCC*, 2010.

[27] U. Lee, I. Rimac, and V. Hilt. Greening the internet with content-centric networking. In *e-Energy*, 2010.

[28] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska. Dynamic right-sizing for power-proportional data centers. In *INFOCOM*, 2011.

[29] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew. Greening geographical load balancing. In *SIGMETRICS*, 2011.

[30] V. Mathew, R. Sitaraman, and P. Shenoy. Energy-aware load balancing in content delivery networks. In *INFOCOM*, 2012.

[31] R. Miller. Report: Google Uses About 900,000 Servers. Data Center Knowledge, http://tiny.cc/gservers, 2011. Last visited Jan 2012.

[32] A. Mohsenian-Rad and A. Leon-Garcia. Coordination of cloud computing and smart power grids. In *SmartGridComm*, 2010.

[33] E. Nygren, R. K. Sitaraman, and J. Sun. The akamai network: a platform for high-performance internet applications. *SIGOPS OSR*, 44:2–19, August 2010.

[34] J. Park. Designing a Very Efficient Data Center. Facebook.com, http://tiny.cc/fbservers, 2011. Last visited Jan 2012.

[35] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs. Cutting the electric bill for Internet-scale systems. In *SIGCOMM*, 2009.

[36] B. Raghavan and J. Ma. The Energy and Emergy of the Internet. In *HotNets*, 2011.

[37] L. Rao, X. Liu, L. Xie, and W. Liu. Minimizing electricity cost: optimization of distributed internet data centers in a multi-electricity-market environment. In *INFOCOM*, 2010.

[38] J. Rath. Data center site selection. In *Rath Consulting White Paper*, 2007.

[39] E. Schurman and J. Brutlag. The user and business impact of server delays, additional bytes, and HTTP chunking in web search. Presentation at the O'Reilly Velocity Web Performance and Operations Conference, 2009.

[40] A. Sullivan. ENERGY STAR for Data Centers. Technical report, U.S. Environmental Protection Agency, 2009.

[41] U.S. Energy Information Administration. http://www.eia.gov, Last visited Jan. 2012.

[42] V. Valancius, N. Laoutaris, L. Massoulié, C. Diot, and P. Rodriguez. Greening the internet with nano data centers. In *CoNEXT*, 2009.

[43] E. Williams. Energy intensity of computer manufacturing: hybrid assessment combining process and economic input-output methods. *Environ. Sci. Technol.*, 38(22), Nov 2004.