Performance-Aware Energy Saving for Data Center Networks

Motassem Al-Tarazi, *Student Member, IEEE* and J. Morris Chang, *Senior Member, IEEE*

***Abstract*—Today's data center networks (DCNs) tend to have tens to hundreds of thousands of servers to provide massive and sophisticated services. The architectural design of DCNs usually over-provisioned for peaks workloads and fault-tolerance. Statistically, DCNs remain highly under-utilized with typical utilization of around 30%. Network over-provisioning and under-utilization can be exploited for energy-saving. Most research efforts on data center network energy saving focus on how to save maximum energy with little or no consideration to the performance of the residual network. Thus, the DCN performance degraded and the network left vulnerable to sudden traffic surges. In this paper, we have studied energy-saving problem in DCNs while preserving network performance. The problem was formulated as MILP that is solvable by CPLEX to minimize the energy consumed by DCN, meanwhile, safety threshold constraints for links utilization are met. To overcome CPLEX high computational time, a heuristic algorithm to provide practical and efficient solution for the MILP is introduced. The heuristic algorithm uses switches grouping and links consolidation to switch the traffic to a small number of network devices and turn-off unused switches and links. Valiant load-balancing is used to distribute the loads over active links. Simulation experiments using synthetic and real packet traces were conducted to validate the heuristic in terms of energy consumption and network performance. The results show that the heuristic can save up to 45% of the network energy and improves the average imbalance-scores for links and switches by more than 50% with minimal effect on network performance.**

***Index Terms*—Data Center Networks, Energy Saving, Load Balancing.**

I. Introduction

CURRENTLY, data center networks (DCNs) tend to have tens to hundreds of thousands of servers to provide massive and sophisticated services, such as web searching, cloud storage, online social services, and scientific computing. As data centers become more popular, the importance of power consumption issues is increased due to the high number of powered devices [1]. EPA reported that the total electricity used by data centers in 2010 was about 1.3% of all electricity used in the world [2] and it is expected to reach 8% by 2020 [3].

Extensive research has been done on the energy saving techniques for the server side of the data centers, while the problem for the network side is still a substantial issue. Today's DCNs designed to accommodate peak loads in most reliable way without taking energy saving into consideration. Data center networks are built with many redundant links and heavily over-provisioned link bandwidth to handle link failures and traffic bursts. Although current data centers design increases reliability, it also decreases energy efficiency since all network devices are powered-on all the time with minimal link utilization. Statistics showed that most of the network devices are under-utilized, where the typical utilization of a DCN is only 30% [4]. DCNs' over-provisioning and under-utilization can be exploited for energy saving research.

Existing research proposed many techniques to overcome the energy saving problem. A stream of research [5], [6], [7] proposed energy efficient network topologies. Although these topologies reduce energy efficiently, for example, the optical-based topologies Proteus [8] and Petbit [9] were reported to save up to 75% of the data center power consumption, applying them to existing DCNs is expensive and require hardware modification. Another stream of research focused on traffic engineering and route consolidation as in [10], [11], [12], [13]. The main idea of this stream is to turn the network load to a minimal subset of network devices. Then it puts unused devices to sleep mode or shut them down to minimize the overall network power consumption. Using traffic engineering and route consolidation, there will always be a trade-off between energy saving and performance.

Few studies discussed this trade-off. Zhang et al. [14] proposed a traffic engineering technique to maximize the number of links that can be shut down under some network performance constraints, such as link utilization and packet delay. In their study no techniques were specified to handle traffic bursts. Shang et al. [15] proposed an energy aware routing, the idea is to use a few devices to satisfy the network demand with little or no degradation in the overall performance represented by the throughput of the original network. Initially, they compute the network throughput by routing all network devices; start to remove switches until the throughput decreases to a predefined threshold. Finally, switches not involved in the final routing are either powered off or put into sleep mode. This technique suffers from inefficient computational running time as it takes long time to calculate a near optimal solution.

In this paper, we studied the problem of saving data center network energy while maintaining network performance against traffic surges. The problem is formulated as a mix integer linear problem (MILP) to minimize the total network energy as a main objective. Moreover, the problem was constrained by network performance requirements, such as maximum link utilization with safety margin threshold. In general, MILPs are NP-hard problems, thus, the computational time to solve MILP increases exponen-

M. Al-Tarazi is with the Department of Computer Science, Iowa State University, Ames, IA 50011 USA. E-mail:altarazi@iastate.edu.

J. M. Chang is with the Department of Electrical Engineering, University of South Florida, Tampa, FL 33647 USA. E-mail:chang5@usf.edu.

Manuscript received May 18, 2017.

tially with the size of the problem. For example, the time to find the optimal solution for the MILP using a data center network with 54000 servers is more than 4 hours. Therefore, solving the energy saving problem for large data centers is impractical.

For practical implementation to large data center networks, we argue that setting margin threshold alone, as in existing methods such as [21], is not enough for saving energy and maintaining network performance from traffic surges. So, a light-weight heuristic algorithm that combines setting-up safety margin threshold and load balancing technique together is presented to save energy and maintain network performance to handle traffic surges.

The heuristic algorithm starts by setting up predefined safety thresholds on each link capacity. Then, it continuously monitors the utilization of network links and balances the loads on active links using Valiant Load Balancing (VLB) mechanism [16]. A decision to turn on new switches or links can be taken if these thresholds are exceeded. Using this algorithm, the safety margins and the load balancing mechanism allow the network to handle traffic surges, while maintaining its performance. On the other hand, switches grouping and links consolidation will also take place if the loads on the networks switches and links are under-utilized. This will allow turning off some active ports and switches to lower network power consumption.

To validate the effectiveness of the algorithm, extensive simulations conducted on data centers with classical three-tier and fat tree [17] topologies. The proposed algorithm was evaluated against data centers without any energy saving mechanism, data centers with greedy bin-packing energy saving mechanism, and Global First Fit energy saving mechanism in terms of energy saving, average end to end delay, throughput and drop packets at various data center loads. The results showed that the proposed algorithm can save up to 35% with three-tier topology and up to 45% with the folded clos fat tree topology with minor effect on network performance. To evaluate the load balancing mechanism used in the proposed algorithm, the imbalance scores for both links and switches are compared against the same proposed algorithm without any load balancing mechanisms. The imbalance score of switches/links is the standard deviation of the average switch/link utilization for all switches/links in a switching level. The proposed algorithm improves the imbalance scores for both links and switches by more than 50% and 60%, respectively. In addition, the proposed algorithm solution was evaluated against the optimal solution obtained by CPLEX [18] in terms of power consumption and computational running time. The results show that the power consumption gap between the solution provided by the proposed algorithm and the optimal solution provided by CPLEX is less than 4%. In comparison, the computational time for CPLEX is very high compared to the proposed algorithm.

The list of contributions in this paper is as follows:

- We propose a technique to save data center energy while preserving network performance from traffic surges. The problem was formulated as a mixed integer linear program. We identify that setting up link utilization threshold alone is not enough to preserve network performance as shown in our evaluation. We proposed that in addition of setting link utilization threshold a load balancing technique should be applied to preserve network performance and handle sudden traffic surges.
- For large scale data centers, we design a heuristic algorithm that sets safety thresholds on link capacities and uses valiant load balancing technique on active links. The proposed heuristic is abstract and can be applied to any switch-centric topology in similar fashion.
- We implement the proposed heuristic algorithm using GreenCloud simulator and compared to the base case, Greedy bin-packing, Global first fit, and the proposed heuristic without load balancing. Both synthetic and real traces demonstrate that the heuristic algorithm saves considerable amount of energy with minimum effect on the DCN.
- We propose the Average Imbalance Score metric for both switches and links to evaluate the performance of the load balancing mechanism. Using this metric, we show that the heuristic algorithm improves the imbalance score for links and switches by more than 50%.

The rest of the paper is organized as follows. Section II reviews previous related works. Section III formulates the power saving problem. Section IV presents the system model used. Section V proposes the heuristic algorithm. Section VI discusses how the heuristic algorithms achieves a desirable amount of load balancing. Section VII presents the simulation experiments and discusses the results and finally Section VIII concludes the paper.

II. Related works

Many approaches have been proposed to deal with the data center network energy saving problem. A number of researchers proposed designs of new topological structures that provide energy conservation while preserving performance. Examples may include flatted butterfly [19], Pcube [5], Small-World [6], NovaCube [7], 3D Torus based CamCube [20], and Proteus [8]. The primary drawback of these new topologies is that they cannot be applied to existing data centers as they require specific hardware and software capabilities.

On the other hand, some researchers found optimization problems for current DCNs and propose different techniques and heuristics to solve them. ElasticTree [21] proposed a power manager that adjusts the active switches and links to satisfy dynamic traffic loads. The authors in [21] proposed setting safety margins to provide performance insurance by delaying the point at which packets starts to drop and latency starts to degrade. Carpo [22] introduced a correlation-aware power optimization algorithm, it dynamically consolidates traffic loads into a minimal set of switches and links and shut down unused devices. REsPoNse [23] discussed the trade-off between optimal energy saving and scalability. It identifies a few critical routes offline, installs them to routing tables, then runs an online simple scalable traffic engineering to activate and deactivate network devices. GreenTE [14] proposed a power-aware traffic engineering model. They try to maximize the number of links that will be shut down

under certain constraints such as maximum link utilization and packet delay. Wang et al. proposed a rate adaptation approach for future data center networks to solve the oscillation brought by traffic engineering approaches [24].

In [10], the authors introduced a combination between energy-aware routing and preemptive flow scheduling to maximize energy efficiency. [15] Introduced a model that uses few network devices as possible to provide routing services with little or no sacrifice of the network throughput. They compute the network throughput according to routing overall network devices; start to remove switches until the throughput decreases to a predefined threshold, and finally switches not involved in the final routing are either powered off or put into sleep mode. [11] proposed PowerNets, a power optimization framework that minimizes DCN, server, and cooling power together. For more energy saving, a workload correlation analysis is done during the server and traffic consolidation processes. The authors in [25] present PowerFCT, an energy saving scheme that combines flow consolidation and DCNs' switch components power throttling. They also consider flow completion time of delay sensitive flows to preserve network performance. [12] designed a power efficient network system that is based on an artificial intelligence abstraction model called blocking island. The idea is to produce a set of different B-blocking islands and blocking island hierarchy tree (BIH) based on the available bandwidth. For each traffic demand, bandwidth allocation mechanism and power-aware routing algorithm are applied on BIH to compute and allocate the best routing path. After having a set of routes that satisfy all the demands, backup routes will be added for fault tolerance. Finally, the system turns off or puts to sleep the switches, line cards, or links not in the solution set.

A distributed flow consolidation framework with correlation analysis and delay constraints presented by [26]. They present two distributed heuristics that provide different trade-offs between scalability, power saving, and network performance. A flow consolidation that considers the flow completion time (FCT) introduced by [13]. It is designed based on control theory to dynamically control the FCT of delay of delay-sensitive traffic flows. Merge network [27] considers minimizing the power consumed by a switch. It tries to consolidate links loads to a smaller subset of links within the same switch, then turning the unused links to low power mode. This approach focuses on reducing energy within switches, which tend to have less energy-saving compared to traffic engineering approaches which tries to merge traffic at a subset of the network. Most of these techniques focus on the optimization without setting load balancing and safeguards policies to maintain the performance of the network.

Another stream of research tried to combine the network energy-saving problem with the server energy saving problem to maximize the overall energy saving as in [28], [29] or to combine it with VM placement problem as in [30], [31], [32], [33]. This combination will add extra load to the network due to VM migration overhead. Although this combined mechanism will provide extra energy saving, the network performance will suffer due to route consolidation and VM migration.

Table I
Definition of important symbols

Symbol	Definition
S	Set of all switches
S_C, S_{Agg}, S_{Acc}	Sets of core, aggregation, access switches respectively
N	Set of all ports
E	Set of all links
D	Set of all traffic Demands
N_i	Ports in switch i
i, j	A link connects two nodes i and j
ϵ^{Port}	Energy consumed by a port
ϵ^{Fixed}	Fixed energy consumed by a switch
$f^t_{i,j}$	Traffic flow t through link i, j
C	Capacity Matrix for all links
$On(i,j)$	A 0,1 Decision variable indicates if the link is on or off
$On(s)$	A 0,1 Decision variable indicates if the switch is on or off
$P(s)$	The total power consumed by switch s
U^{upper}	Upper link utilization threshold
$u_{i,j}$	Utilization of link i, j

III. Problem Formulation

Consider a data center network $G = (V, E)$ where V is the set of nodes and E is the set of links. A port link can be turned-off if there is no traffic on the link and a switch can be turned-off if all its ports are turned-off.

Let S be the set of all switches in the network where $S \subseteq V$. The power consumed by a single switch $s \in S$ consists of fixed power ϵ^{Fixed}, which consumed by components like (chassis, fans, etc), and ports power ϵ^{Port}. The power saving gained from turning off a single port is ϵ^{Port}, and from turning off an entire switch is $\epsilon^{Fixed} + \sum_{l \in N_i} \epsilon^{Port}$. We use $On(i)$ and $On(i,j)$ as decision variables to denote that switch i and link (i,j) are active or not.

Assume that the traffic demand matrix D consists of a number of flows $\{f^0, f^1, \ldots, f^t\}$. Each flow f^t will be passing through a number of links from source to destination that satisfies the flow load. $f^t_{i,j}$ represents the flow load of t that is passing through link (i,j). The traffic matrix τ is the summation of all flow loads passing through each link in the data center network G. Note that each link $(i,j) \in E$ has a bidirectional bandwidth capacity $C_{i,j} \in C$, where C is the capacity matrix for all links in E. With the notations summarized in Table I, we can formulate our problem as a mixed integer linear program that is solvable by CPLEX as the following: The MILP takes the data center network $G(V,E)$, the demand matrix D, the capacity matrix C, and the upper utilization threshold U^{upper} as input. Equation 1 is the objective function. It minimizes the network power consumption function $P(x)$ for every switch. Thus, minimizing the total power consumed by the data center network.

$$Minimize \sum_{x \in S} P(x) \tag{1}$$

The constraints are divided into three categories: links constraints, switches constraints, and utilization constraints. Equations 2-5 present network links constraints.

Equation 2 introduces the active link constraint [29]. It states that an active link connects two active switches or a switch and a server.

$$On(i,j) \leq On(i), On(i,j) \leq On(j)\ , \forall i,j \in E, \forall i \forall j \in S \tag{2}$$

Equation 3 states the bidirectional link power constraint which means both directions of a link (i,j) should have the same on/off power status. Likewise, equation 4 ensures that for every active link $On(i,j) = 1$, both directions have the same capacity limits $C_{i,j}$.

$$On(i,j) = On(j,i)\ , \forall i,j \in E \tag{3}$$

$$On(i,j) \cdot C_{i,j} = On(i,j) \cdot C_{j,i}\ , \forall i,j \in E \tag{4}$$

Equation 5 introduces the satisfiability constraint. It shows that the summation of all traffic flow loads $\sum_{t=0}^{n} f^t$ passing through link (i,j) is always less than or equal to the capacity limit of that link $C_{i,j}$. Where n is the number of all traffic flows.

$$\sum_{t=0}^{n} f_{i,j}^t \leq On(i,j) \cdot C_{i,j}\ , \forall i,j \in E \tag{5}$$

Equations 6-7 present network switch constraints. Equation 6 shows the active switch constraint. Let $N_i \in N$ be the set of ports in a switch and $|N_i|$ is the cardinality of N_i, then equation 6 ensures that a switch will be turned off only if all its ports are turned off.

$$|N_i| \cdot (1 - On(i)) \leq \sum_{j \in N_i} (1 - On(i,j))\ , \forall i,j \in E, \forall i \in S \tag{6}$$

Equation 7 calculates the power consumed by a switch. Which is the power consumed by its fixed components ϵ^{Fixed}, such as chassis, fans, line cards, ... etc., in addition to the power consumed by each active port ϵ^{Port}.

$$P(x) = \epsilon^{Fixed} \cdot On(x) + \sum_{n \in N_i} \epsilon^{Port} \cdot On(x,n) \tag{7}$$

Equations 8-9 present utilization constraints. Equation 8 calculates the link utilization u for each link. Where link utilization is the summation of every traffic flow load passing link (i,j) to the capacity of that link. Equation 9 ensures that the utilization of every link is always less than or equal to a predefined upper link utilization threshold U^{upper}(in this paper $U^{upper} = 0.80$).

$$u_{i,j} = \frac{\sum_{t=0}^{n} f_{i,j}^t}{C_{i,j}}\ , \forall i,j \in E \tag{8}$$

$$u_{i,j} \leq U^{upper}\ , \forall i,j \in E \tag{9}$$

Equations 10-11 show the problem decision variables. $On(i,j)$ and $On(i)$ are binary decision variables indicate the power status for network links and switches, respectively. Since $On(i,j)$ and $On(i)$ are binary integers, the formulated problem is $MILP$.

$$On(i) \in 0,1 \tag{10}$$

$$On(i,j) \in 0,1 \tag{11}$$

Since mixed integer linear programming is NP-hard, the proposed formulation is not practical for large data center networks. Thus, it can be used as a benchmark tool to evaluate practical heuristic approaches.

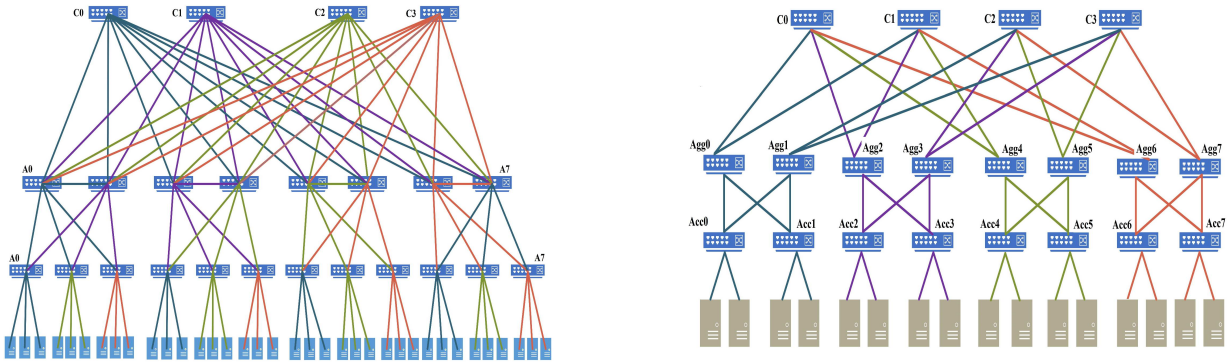
IV. System Model

This section provides a brief background about network topologies and traffic model used.

A. Network Topologies

In this paper, we considered applying our technique to two of the most popular topologies in data center networks: Three-tier and Fat-tree topologies.

The three-tier is the mostly used topology in data center networks [34]. Three-tier topology consists of three switching layers; core or border routers, aggregation switches, and top-of-rack (ToR) access switches. Each ToR connects up to 48 servers placed in a rack with 1 Gbps links, while for redundancy issues; a ToR is connected to two aggregation switches. Furthermore, each aggregation switch is connected to core switches with multiple high speed 10 Gbps links [35]. Unfortunately, three-tier topology suffers



(a) (b)

Figure 1. Data center network topologies: (a) Three-Tier. (b) Fat Tree with K = 4.

from various issues such as: scalability, cost, energy consumption, cross-section bandwidth, and agility [36]. Figure 1(a) shows the three-tier topology.

The fat-tree topology in data center networks was proposed by [17] to deal with the issues of traditional data centers. Fat tree is a multi-rooted tree, where its links - unlike the traditional tree topologies - became larger in term of capacity as they move toward the roots. Fat-tree is one of the Clos technologies that has been adopted for Google data centers [37]. Figure 1(b) illustrates the fat-tree topology with $k = 4$.

In general, if k-port switches are used to construct a fat-tree, then $(k/2)^2$ core switches are needed to connect k pods, each pod consists of $k/2$ access switches and $k/2$ aggregation switches. Within a pod, aggregation switches and access switches are connected with each other to form a complete bipartite graph. Since each access switch connected to $k/2$ aggregate switches, each access switch is also connected to $k/2$ servers. So the number servers supported by a k-port fat tree are $k^3/4$.

B. Traffic Model

The traffic model used follows the ingress/egress traffic model. The ingress traffic represents accepted task requests by the data center, which travels from core switches down to its designated server. On the other hand, the egress traffic are the outputs of the tasks which originated at servers and traverse upward to core switches.

Let n_s denote the number of servers/switches connected by one switch at any switching level on a switch-centric topology. Thus, the ingress/egress capacity limit of each switch is bounded by $n_s \cdot C_s$, where C_s is the capacity of link s. Taking the upper link utilization threshold into account, the capacity limit will be bounded by $n_s \cdot C_s \cdot U^{upper}$. So, any valid traffic matrix $\tau \in D$ should satisfy the following constraint:

$$\sum_{s=0}^{n_s-1} \sum_{t=0}^{n-1} f_s^t \leq n_s \cdot C_s \cdot U^{upper}, \quad where\ s \in E \qquad (12)$$

V. Heuristic Approach

To overcome the exponential increase in CPLEX computation time, a heuristic algorithm solving the data center energy-saving problem was developed. In data center environment, traffic demands fluctuate frequently. For that reason, heuristic algorithm is preferred to solve our optimization model in real time.

Algorithm 1 illustrates the heuristic pseudocode, it takes similar inputs as in CPLEX. The output includes a set of active switches and ports that satisfies the traffic demands as well as the load balancing requirements. The heuristic algorithm devised to solve the problem under any $switch-centric$ topology [38] in similar way.

The algorithm starts by taking the data center topology; the set of current active switches and ports, the flow to be assigned, and the upper utilization threshold as inputs. After initialization, the network will power on a minimum spanning tree of switches (MST) if the current flow is the first flow to assign, otherwise, the set of all switches that currently powered-on (Set A') will be used. Set A''' is the set of all powered-off switches.

Algorithm 1 Heuristic Algorithm

```
Input: G(V,E), A, C, flow, U^upper
Output: Set of active switches and ports A
S_C ⊆ V; i ← 0; l ∈ N_s; N_s ∈ E
if A = φ then
    A = MST( )
A' = A
A'' = φ; A''' = S_C − A'
while A' ≠ φ do
    Randomly select i ∈ A'
    if ∃ l ∈ i is active then
        if LinkChecker(flow, l, i, C_{i,l}) then
            Update(A, A', flow, i); break;
    else
        if ∃ l ∈ i is inactive then
            A'' = A'' + i
    End if
    A' = A' − i
End While
if A' = φ then
    if A'' ≠ φ then
        Randomly select i ∈ A''
        SET l to active
    else
        Randomly select i ∈ A'''
        SET i, l to active
    End if
    A' = A' + i
    Update(A, A', flow, i)
End if
SwitchGrouping(S_core, A, U^upper)
ValidateAndConsalidate(S_core, A, U^upper)
Return A
```

For an incoming flow at core switching level, the algorithm randomly selects a switch i from the current set of active switches A'. Then, it searches $i's$ routing table to check if there exists an active port l that can lead to the target destination. If so, **LinkChecker** function will be called to compute the current link load and to verify that the new load will not exceed the capacity of the link (Equation 5) and the predefined upper utilization threshold (Equations 8 and 9). As the flow assigned to a specific link, the current link load will be adjusted accordingly. Meanwhile, any active switch that holds an inactive target link l will be added to set A''.

In case no active switch has an active target port that can handle the flow, the algorithm checks if there is an active switch with an inactive target port in set A''. If found, it will randomly select a switch from set A'' and powered-on the target port l on that switch. If the algorithm failed to find any active switch with a target link that can handle the incoming flow, a new switch i will be randomly selected from set A''', powered-on, and a target port l will be activated.

As the incoming flow being assigned, several already assigned flows might be expired. Thus, some switches might be ended up with a light load on its ports and/or replicated switches which might be using different ports. The **SwitchGrouping** function tries to find any matches

for switches grouping and elimination. SwitchGrouping goal is to increase the number of switches to be turned off using two techniques. The first technique involves searching for two replicated candidate switches which are powered on, connected to the same switches, and they use different ports, to group them into one switch by re-allocating all the flows on the lighter switch to the other one and shut it down. For example, in the fat-tree topology with k = 4 (Figure 1.b), suppose that the first and second switches at core level (C0 and C1) are powered-on, switch C0 is using ports (0,3) while switch C1 is using port (1). The flows in switch C1 port (1) can be re-allocated to switch C0 port (1) and switch C1 can be turned-off. The second technique calculates the current traffic load for each switch and starts with the lightest traffic load switch trying to move its traffic load to other switches and turn it off.

For further increase in energy saving, **ValidateAndConsolidate** function attempts to consolidate links and turns off unused ports in a greedy fashion. It computes active links utilization and chooses candidate links with the lowest utilization for consolidation. The SwitchGrouping and ValidateAndConsolidate functions will assure that the MST connectivity property, the link capacity, and the utilization threshold requirements are satisfied. Finally, the final solution of set A after the grouping and consolidation processes will be returned.

The heuristic algorithm assures that minimum number of switches and link will be active. Thus, maximizing the energy saving. From network performance point of view, the switches random selection in the heuristic algorithm will distribute the flow load among active links. The link capacity safety threshold maintains extra space within a link to be use in case of sudden traffic surge.

The same algorithm can be used to handle the communications in aggregation level. The only change needed is to use S_{Agg} rather than S_C if it runs over three-tier topology. For aggregation switching level in fat-tree, the search space for the designated switch will be within a pod not the whole aggregate switches.

To maintain the minimum spanning tree property of reaching all servers all the time, access level switches and ports will not be turned off. It should be noted that the heuristic algorithm is flexible and can modified to handle situations that affect the network traffic such as virtual machine migrations. The worst case computational complexity of the algorithm is $O(S_C \cdot N_s)$, where S_C is the number of switches in the core switching level and N_s is the number of ports per switch.

A. Performance Bound Analysis

In this subsection we analyze the consolidation performance bound of our proposed heuristic algorithm. Let OPT be the smallest number of switches to be used in a consolidation problem (i.e. the optimal solution).

Proposition 1. *The number of switched to be used by the heuristic algorithm at each switching level is upper bounded by* $\lfloor \frac{17}{10} OPT \rfloor$.

Proof. For a set of flows to be assigned (F) to a set of switches at core level (S_{core}), the proposed heuristic assigns each incoming flow to the lowest indexed active switch that can handle the flow load based on random proposition. This consolidation similar to applying First Fit (FF) method to the bin packing problem. The only difference is that First Fit method searches for a bin (server) sequentially starting from the lowest indexed non-empty bin and our heuristic uses random proposition. Both will pack an item to the first bin that fits in. Since it has been shown that FF has a worst-case result bounded by $\lfloor \frac{17}{10} OPT \rfloor$ [39], then the performance of the consolidation process of the heuristic algorithm is no worse than $\lfloor \frac{17}{10} OPT \rfloor$. □

It should be noted that if the incoming flows are presorted in a decreasing order, the consolidation process would be similar to the First Fit Decreasing (FFD) method which has a tight upper bound of $\frac{11}{9} OPT + \frac{6}{9}$ [40].

VI. Load Balancing

In this section, we show how the heuristic algorithm balances traffic loads while turning off a number of network devices. In general, load balancing mechanisms tend to disperse the network traffic among network devices to minimize packet delay, packet loss, and congestion problems [41].

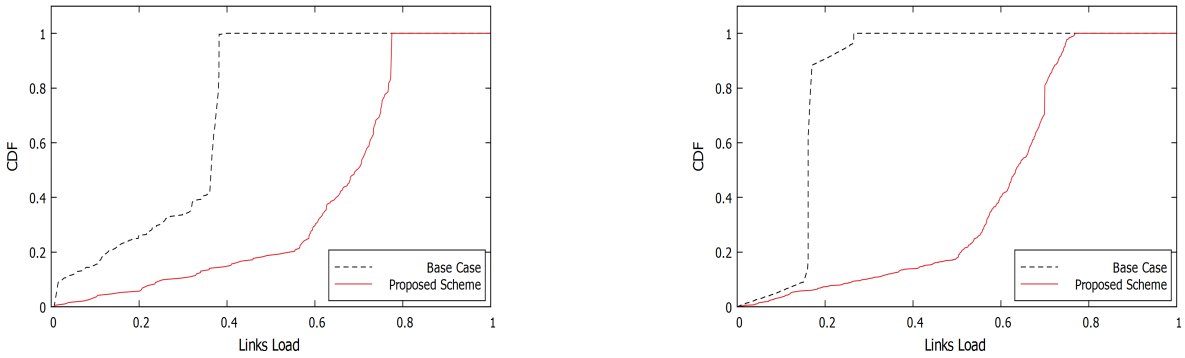
In general, when formulating the energy-saving problem, having a joint objective function to maximize energy saving and load balancing will introduce a contradiction. While energy saving tends to concentrate the load to a small subset of devices, load balancing tries to evenly distribute the load to all links.

In our problem formulation, the main objective is to save data center network energy while setting constraints to preserve network performance. The load balancing requirement can be satisfied (although not perfectly balanced) through the maximum link utilization constraints (Equations 8 and 9). The maximum link utilization constraints will assure that most of the active links are utilized up to the upper utilization threshold, thus fulfilling the load balancing requirement.

The heuristic algorithm deals with the contradiction by balancing the loads only over active links. It uses a similar idea used by Valiant Load Balancing mechanism (VLB) [16], it randomly selects an active switch and checks if the target link load can handle the demand without violating the upper link utilization constraint. Although the heuristic algorithm - as in VLB- do not guarantee perfect balancing between active links, it satisfies the objective of load balancing by ensuring that any active link load will not be congested.

In fat tree and three-tier topologies, the load balancing mechanism is needed at the core and aggregation levels only. Since the access level should be always active to maintain the server reachability property as a requirement for minimum spanning tree and the fact that each server is connected to one access switch.

Figure 2 shows the cumulative distribution function (CDF) for active links load at core and aggregation levels of a fat tree topology for both the proposed scheme (heuristic algorithm) and base case while tasks are distributed equally to all servers. The base case uses round



(a) (b)

Figure 2. Comparison of links loads: (a) CDF at Core Level. (b) CDF at Aggregation Level.

robin load balancing without energy saving mechanism (i.e. all switches and links powered-on all the time). For the base case, both CDFs illustrate that the active links are not utilized efficiently, most of the active links are only between 25% - 38% for the core level and 15% - 27% for the aggregation level. On the other hand, the heuristic algorithm uses group switching and link consolidation mechanisms to maximize link utilization with respect to the upper utilization threshold and balances the load among active links. In our proposed scheme, most of the links loads are ranging between 55% - 77% of links capacities at core and aggregation levels.

VII. Performance Evaluation

This section presents the evaluation of our proposed heuristic algorithm. The evaluation is conducted to show that the algorithm achieves a considerable amount of energy saving with minor effect on the network performance. The evaluation divided into two parts: first, we implemented our algorithm using GreenCloud simulator[42], which is based on ns2 [43], to show how the network load will be distributed among core and aggregation switches and how it will affect network energy consumption, network performance metrics, and average imbalance score [44]. The simulation results were compared with the ones obtained by data centers that do not use any energy saving mechanism (Base Case), data centers that use greedy bin-packing energy saving mechanism [21], data centers that use Global First Fit energy saving mechanism [45], and our proposed scheme without load balancing mechanism using synthetic and real packet traces. The base case uses round robin load balancing technique and all network devices are on. Thus, it would produce the best network performance that can be achieved and almost the worst energy consumption. Greedy bin-packing dynamically changes the power state of network devices (links and switches) based on traffic load fluctuation and turns off idle devices. Greedy bin-packing uses full link capacity with no load balancing mechanism. The Global First Fit assigns the current flow in a greedy fashion. The flow will be allocated to the first path that can handle it. To show that the proposed scheme can be applied to various switching-centric topologies, we compared the scheme using both fat tree and traditional tree-tier topologies. Second, we investigate the computation time and power consumption obtained by CPLEX and compare them to the ones achieved by the heuristic algorithm. This shows the applicability and solution optimality/near optimality of our proposed heuristic algorithm.

A. Simulation Setup

We have designed two sets of experiments using GreenCloud simulator. The first set is based on fat-tree topology with $k = 30$ port switches. The topology includes 6,750 high computing servers; 225 core switches with 100 Gbps links, 450 aggregation switches with 10 Gbps links, and 450 access switches with 1 Gbps links. The second set is based on three-tier topology, where the topology includes 6,144 high computing servers; 16 core switches with 10 Gbps links, 32 aggregation switches with 1 Gbps links, and 128 access switches with 1 Gbps links. The servers have homogeneous set of resources includes computation, memory, storage resources. Specifically, each server can provide 238,310 MIPS [48], 32 Gigabytes of memory, and 250 Gigabytes of storage. The main topologies parameters considered in the simulations are tabulated in Table II. Also, Table III shows the power rates of various commodity switches used in the simulation.

Table II
System parameters

Parameter	Three-Tier	Fat Tree
Core switches	16	225
Aggregation switches	32	450
Access switches	128	450
Servers	6144	6750
Access Links	1 Gbps/3.3μs	1 Gbps/3.3μs
Aggregation Links	1 Gbps/3.3μs	10 Gbps/3.3μs
Core Links	10 Gbps/3.3μs	100 Gbps/3.3μs
Resource_Computational	238,310 MIPS	238,310 MIPS
Resource_Memory	32 Gigabyte	32 Gigabyte
Resource_Storage	250 Gigabyte	250 Gigabyte

Clients send high performance computing (HPC) task requests to be executed in the data center. Each task

Table III
POWER RATES OF VARIOUS COMMODITY SWITCHES IN WATTS

Topology	Switch Type	Fixed Power	Port Power
Three-Tier [46]	Core	2770	27
	Aggregate	2770	27
	Access	146	0.42
Fat Tree [47]	Core	3350	60
	Aggregate	3184	25
	Access	1250	13.3

Table IV
NETWORK AND TASKS PARAMETERS

Parameter	Value
Queue limit	100 Packets (150 Kbytes)
Traffic generator	Exponential
Packet size	1500 byte
Task_MIPS	100,000
Task_Memory	1 Gigabyte
Task_Storage	10 Gigabyte
Task_Duration	5 Seconds
Task_Size	8500 byte
Task_Output	2.5 Megabyte

request has a size of 8,500 bytes (6 packets needed). For each request a green scheduler searches for a target server, which has enough resources to handle the HPC task, from left to right. The green scheduler assigns the requested task to the first server that satisfies its requirements so it aims to consolidate all the tasks to a small subset of servers [46].

Each HPC task consumes 100,000 MIPS, 1 Gigabyte of memory, 10 Gigabytes of storage, and duration of 5 seconds. The output of the HPC task has a size of 2.5 megabytes which is sent from the server back to the client. To simulate traffic surges, each client sending agent has an exponential traffic generator with 1500 bytes packet size. Flows are initiated from clients and their targeted server, and no data traffic generated at any switching level. Each port on a switch has an independent FIFO queue with a limit of 150 Kilobytes (100 packets). To evaluate the performance of the DCN at different data center loads, the task requests rate is increased accordingly. Table IV summarizes the network and tasks parameters.

B. Network Energy Consumption

Figures 3(a) and 3(b) show the network energy consumed by base case (conventional data center), greedy bin-packing, Global First Fit and proposed schemes with/without load balancing with both fat tree and three-tier topologies, respectively. Greedy bin-packing tends to be the most energy saving mechanism. It consolidates routes and uses links at full capacity without setting safety thresholds. Using full link capacity, minimum number of switches will be used, thus, saving a lot of energy. The Global First Fit is able to save energy for low data center loads since it uses the links full capacity and the process of finding a path that can handle flows is easy. On the other hand, as the load increases, data center links became more saturated. This will make finding a path to handle the flows more difficult and time consuming, thus increase power consumption. Greedy bin-packing and Global First Fit are using links at full capacity which in turn will leave the network vulnerable to sudden traffic surges. The proposed scheme tended to save energy while setting up safety thresholds to deal with traffic surges. The results also show that using a load balancing mechanism will cause better energy conservation for the proposed scheme, where the energy consumption is almost similar to the greedy bin-packing even with reserving part of the links capacity as a safety threshold.

The energy consumption of the greedy bin-packing and the proposed schemes with/without load balancing is in direct proportion to the data center load; when the load is low, the number of network devices to turn off is increased so the energy consumption is low. When the load is high, most of the devices need to be active to deal with this load. The base case doesn't use any energy saving mechanism, thus all network devices are powered-on all the time.

All schemes have the same pattern in both fat tree and three-tier topologies but since fat tree has more network switches and links at core and aggregation levels, it consumes more energy. Since fat tree has more switches and links with more capacity than three-tier, the number of switches and link to be turned off are much larger. Thus, fat tree can save more energy than three-tier.

C. Network Throughput

The network throughput evaluates the network transmission capability based on the used approach. Figures 4(a) and 4(b) present the network throughput of the base case, Global First Fit, greedy bin-packing, and proposed schemes with/without load balancing with fat tree and three-tier topologies at various data center loads.

The base case has the highest practical network throughput since it sends packets over the whole set of links and uses round robin-load balancing mechanism which will minimize transmission time and increase throughput. The proposed scheme with a load balancing mechanism has the closest throughput to the base case, it outperforms greedy bin-packing, Global First Fit, and the proposed scheme without load balancing. This is mainly because the aim of load balancing is satisfied and setting threshold, then less congestion will occur in DCN. For greedy bin-packing and Global First Fit, when the data center load is low, more network devices can be turned off forcing packets to be sent over a small number of links. This will increase the transmission time since more DCN links would be congested, thus, decreasing throughput. Whereas when the data center load is high, all schemes will transmit packets over almost the same number of links thus the network throughput will be almost the same.

The effect of using the load balancing mechanism in our proposed scheme is significant, it introduces an up to 7% improvement in throughput compared to the same scheme without load balancing. The network throughput with fat tree topology is much higher than three-tier topology; this is because fat tree has more available links with larger capacities at the core and aggregation switches than three-tier. So, three-tier will send packets over less number of links thus decrease throughput.

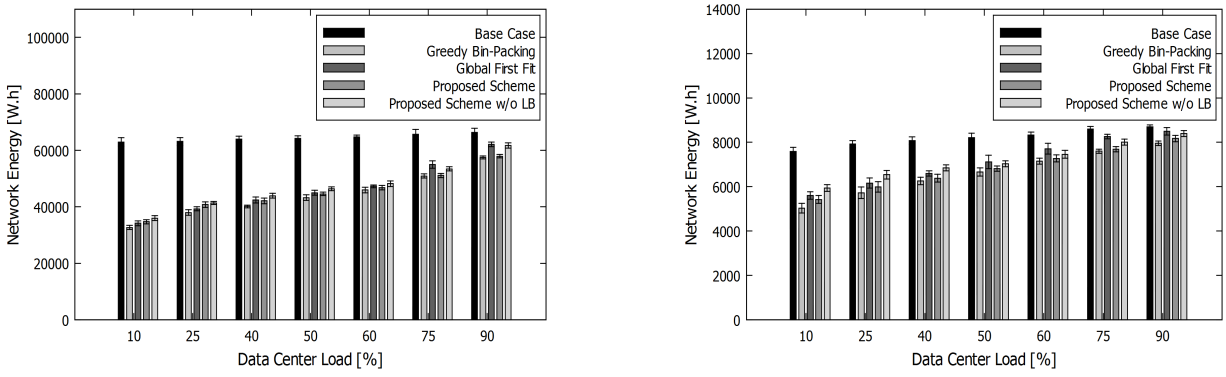


Figure 3. Network energy consumption at different data center loads. (a) Fat tree. (b) Three-Tier.

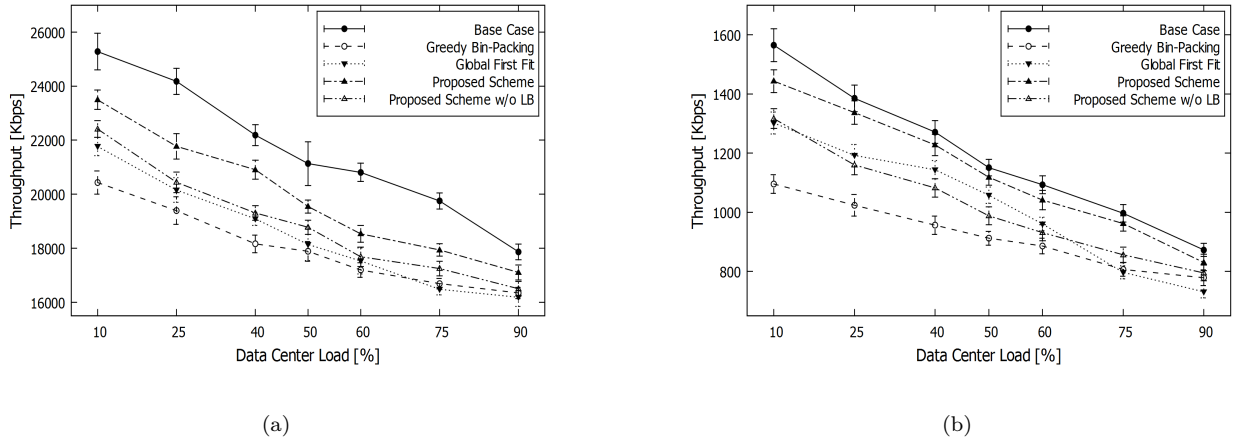


Figure 4. Network throughput at different data center loads. (a) Fat tree. (b) Three-Tier.

D. Average End-to-End Delay

End-to-end delay is the time for a packet to be transmitted across the network from source to destination. The end-to-end delay includes transmission delay, propagation delay, and queuing delay. It is an indication of the overall network performance. Figures 5(a) and 5(b) show the network average end-to-end delay for all schemes.
Similar to network throughput, the base case tends to have the lowest end to end delay. The proposed scheme with load balancing has the nearest average end-to-end delay to the base case. For the greedy bin-packing, Global First Fit, and the proposed scheme with/without load balancing, when the data center load is low, the packets transmitted over less number of links thus the end to end delay will increase, as transmission and queuing delays will increase, compared to the base case. When the data center load is high, the proposed scheme will send packets over almost the same number of links as the base case, thus, the end to end delay is almost the same. On the other hand, the greedy bin-packing, Global First Fit, and the proposed scheme without load balancing lack of load balancing mechanism increases the average end-to-end delay significantly. The results show that the Global First Fit has lower end to end delay compared to the greedy pin-packing at low loads, while it became worst as the load increases. This is because the process of finding and assigning paths in Global First Fit is easy and fast when the load is low, but when the load is high, finding paths is more difficult and time consuming which will increase the end to end delay.
The results clearly show the effect of load balancing on end-to-end delay. The end-to-end delay of the proposed scheme with load balancing decreased by up to 14% compared to the same scheme without load balancing. Again, since fat tree contains more links and larger capacities in core and aggregation levels, end to end delay with fat tree is much lower than end to end delay in three-tier.

E. Ratio of Dropped Packets

As data packets transmitted across the data center network, some of them may be lost or dropped and fail to reach their destination due to many reasons. In data center networks with energy-aware mechanisms, data packets may drop due to link errors, reaching a queue that is already full, or reaching an intermediate link or switch that is turned off. These drops may cause significant network

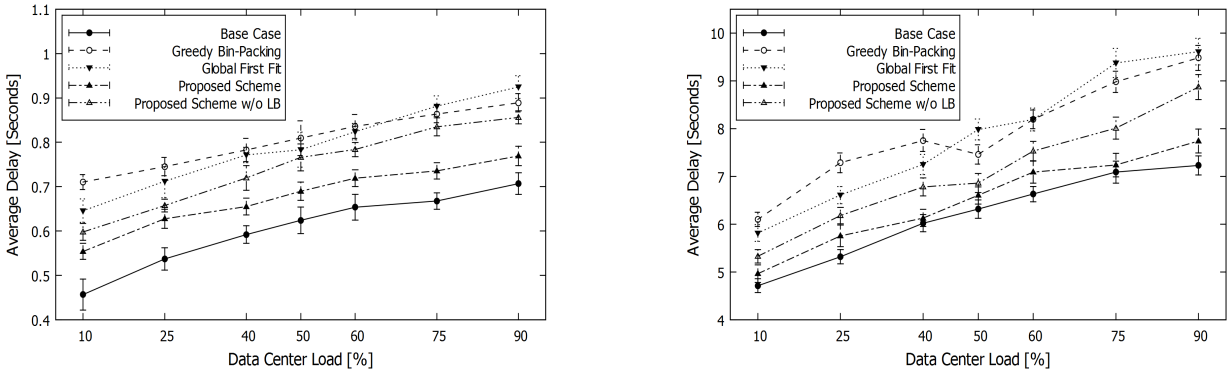


Figure 5. Network end-to-end delay at different data center loads. (a) Fat tree. (b) Three-Tier.

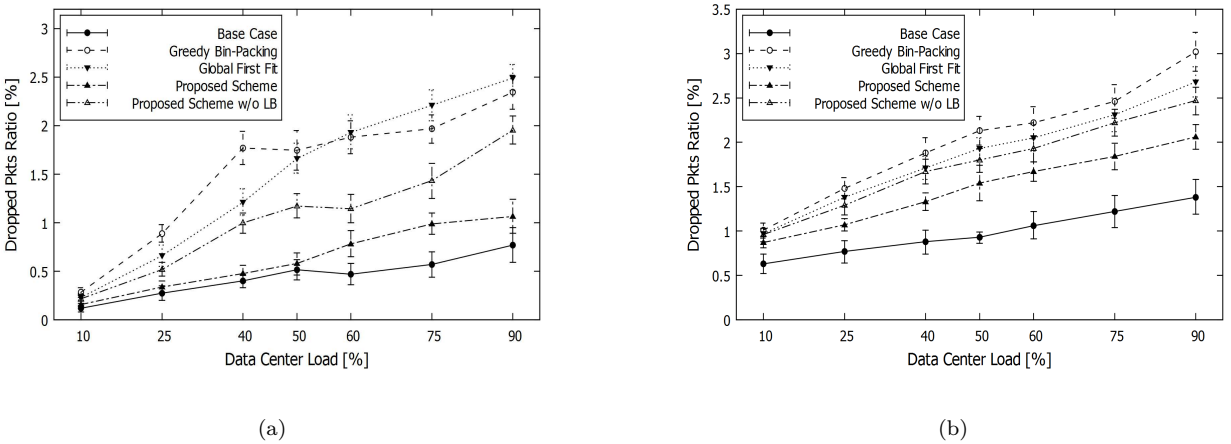


Figure 6. Ratio of dropped packets at different data center loads. (a) Fat tree. (b) Three-Tier.

performance degradation as the delay will increase. This is because the data packets drop and their re-transmission happens at the transport layer in the TCP protocol [49]. Figures 6(a) and 6(b) illustrate the ratio of dropped data packets for all schemes with fat tree and three tier topologies. The results show that the proposed scheme with load balancing provides the nearest drop ratio to the base case. It outperforms the greedy bin-packing, Global First Fit, and the proposed scheme without load balancing for all data center loads with both fat tree and three tier topologies. The proposed scheme maintains the network performance using two techniques; setting up safety threshold to always have extra spaces for incoming packets and adopting load balancing technique for fair load distribution over active links. Using a fat tree, when the load is high, the ratio of drop packets with greedy bin-packing, Global First Fit, and the proposed scheme without load balancing has raise to 2.49%, 2.34%, and 1.95%, respectively, as compared to the proposed schemes' 1.06%.

The results clearly state that the drop packet ratios with three tier topology are much higher than those with fat tree topology. This is because fat tree has more switches and fatter links capacities compared to three tier, thus, lower data packet drop ratio.

F. Average Imbalance Score

In section V, we showed how the proposed scheme with load balancing consolidates and balances links to achieve maximum link utilization with respect to the upper utilization threshold. In this part, we evaluated the efficiency of the load balancing mechanism adopted by our proposed scheme using the notion of imbalance score I [44]. Generally, the imbalance score is calculated using the standard deviation (Equation 13), where $x_1, ..., x_N$ are the values of a finite data set, N is the cardinality of that set, and μ is the standard mean.

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}, \quad where\ \mu = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad (13)$$

In data center networks, the average imbalance score for switches (I_s) or links (I_l) is the standard deviation of the average switch/link utilization across all switches/links in the switching level. Since our proposed scheme minimizes active links and switches, it is essential to compute the

average imbalance scores of them for both core and aggregation levels. Suppose that $P_i(t)$ provides the instantaneous link throughput at time t, equation 14 calculates link utilization over time period when the link is active [50]. C_i is the capacity of link i, T is the time interval between measurements, T_i^* is the time when link i is active.

$$u_i = \frac{1}{T_i^*} \int_t^{t+T} \frac{P_i(t)}{C_i} \cdot dt \tag{14}$$

From equations 13 and 14, the average imbalance score for links at core switching level can be calculated in equation 15, where $|P_C|$ is the cardinality of the set of links at core switching level, μ_1 is the standard mean of all average links load which is calculated in equation 16.

$$I_l = \sqrt{\frac{1}{|P_C|} \sum_{i=1}^{|P_C|} \big(u_i - \mu_1\big)^2} \tag{15}$$

Where:

$$\mu_1 = \frac{1}{|P_C|} \sum_{i=1}^{|P_C|} u_i \tag{16}$$

Similarly, equation 17 shows the imbalance score for switches at core level, where $|S_C|$ is the number of switches at core level, $|N_i|$ is the number of ports within a switch. It calculates each switch utilization based on its links utilization. Equation 18 calculates μ_2, which is the standard mean of all average switches utilization at core level.

$$I_s = \sqrt{\frac{1}{|S_C|} \sum_{i=1}^{|S_C|} \big(\frac{1}{|N_s|} \sum_{j=1}^{|N_s|} u_j - \mu_2\big)^2} \tag{17}$$

Where:

$$\mu_2 = \frac{1}{|S_C|} \sum_{i=1}^{|S_C|} \frac{1}{|N_s|} \sum_{j=1}^{|N_s|} u_j \tag{18}$$

Table V shows the average imbalance scores for both active links and switches at the core and aggregation levels using a fat tree with $k = 30$ and 30% data center load. The results illustrate that the average imbalance scores for the proposed scheme with load balancing are less than the scores for the proposed scheme without load balancing for the links and switches at both core and aggregate levels. The proposed scheme with load balancing uses a VLB like load balancing mechanism which improves the average imbalance scores for links by more than 65% and 50% for core and aggregate switching levels, respectively. Moreover, it also improves the average imbalance scores for switches by more than 68% and 63% for core and aggregate switching levels, respectively. Thus, our proposed scheme balances the load over links and switches efficiently.

Table V
AVERAGE IMBALANCE SCORES

Switch Level	Type	Proposed w/o LB	Proposed with LB
Core	Link	0.1723	0.0602
	Switch	0.2404	0.0752
Aggregate	Link	0.1833	0.0915
	Switch	0.2566	0.0932

G. Real Packet Traces

The proposed scheme was also evaluated against the base case, Greedy-bin packing, Global First Fit, and the proposed heuristic without load balancing technique using packet traces collected in real data center UNIV1 [51]. The traces include the user application data alongside ARP, ICMP, OSPF, and RIP flows. The flows in the traces have small size ranges (less than 10KB). They were applied to traditional three tier and fat tree topologies ($k = 6$) with 54 servers.

Tables VII and VI show the network energy consumption and network performance metrics (throughput, average delay, and ratio of dropped packets) for all schemes using three tier and fat tree topologies, respectively. The results show that greedy bin-packing achieves the highest energy saving in both topologies since it uses full links capacities. However, greedy bin-packing has the worst network performance as it has the lowest throughput, highest average delay, and highest packet drop ratio for both topologies. This shows that the greedy bin-packing, although achieves the highest energy saving, is not suitable for data centers with short network flows.

The Global first fit can save around 16.5% and 22.4% of the network energy with fat tree and three tier topologies, respectively, but the network performance is not as good as the proposed scheme nor the proposed scheme without load balancing.

The proposed scheme without load balancing saves around 19.7% and 25.6% of the network energy with fat tree and three tier topologies, respectively. The lack of load balancing mechanism affects the network performance as throughput, average delay, and the ratio of dropped packets are worst compared to the proposed scheme.

The proposed scheme sacrifices part of the network energy that can be saved by setting up links utilization threshold. This threshold alongside the load balancing mechanism preserve the data center network performance. The results clearly show that the proposed scheme has the nearest performance to the base case for all network performance metrics under consideration for both topologies. Furthermore, the proposed scheme is still able to save around 38% and 35% of the network energy with fat tree and three tier topologies, respectively.

H. More Comparisons

In this subsection we compare our proposed scheme to Virtual machine Placement and Traffic Configuration Algorithm (VPTCA) [52] and Deadline-Constrained Flow Scheduling and Routing (DCFSR) [53] based on the results reported in [52]. VPTCA uses genetic algorithm based VM placement and multiple QoS constrained routing algorithm to save network energy and avoid congestion. In DCFSR, the authors proved that the joint deadline flow scheduling and routing problem is an NP-hard. After that, they proposed an approximation algorithm based on a relaxation and randomized rounding technique. For a fair comparison, we experiment our proposed scheme using the same simulator (i.e. NS2) and same simulation parameters [52]. The algorithms were evaluated using a fat-tree topology ($k = 6$) with 54 servers in terms of DCN energy consumption, average End-to-End delay, and ratio

Table VI
TRACES RESULTS WITH FAT TREE ($k = 6$)

Scheme	Network Energy [W.h]	Throughput [Kbps]	Average Delay [Seconds]	Dropped Pkts Ratio [%]
Base Case	29399.79	5759.53	1.056	3.11%
Greedy Bin-Packing	17894.11	3336.56	5.934	6.91%
Global First Fit	24543.28	3525.19	4.951	5.22%
Proposed Scheme	18187.45	4934.29	1.472	4.16%
Proposed Scheme w/o LB	23598.06	4261.63	2.388	4.86%

Table VII
TRACES RESULTS WITH THREE TIER

Scheme	Network Energy [W.h]	Throughput [Kbps]	Average Delay [Seconds]	Dropped Pkts Ratio [%]
Base Case	3381.67	1499.05	4.839	5.90%
Greedy Bin-Packing	2058.24	839.715	11.367	11.24%
Global First Fit	2623.06	866.179	9.024	10.32%
Proposed Scheme	2191.99	1233.38	5.847	6.97%
Proposed Scheme w/o LB	2514.33	1074.95	7.317	8.63%

of dropped data packets with constant bit rates (CBR) 200 and 800 Kbps.

For energy consumption in DCN, the proposed scheme outperforms VPTCA and DCFSR for both light and heavy traffic loads. In particular, the proposed scheme can approximately save around 13% and 28% compared to VPTCA and DCFSR respectively. This is because the proposed scheme uses links consolidation and VLB on active links where as VPTCA relays on optimal initial VM placement of interrelated VMs within a server or a pod to reduce traffic. VPTCA don't provide any mechanism to deal with congested links nor VM migrations in case of resource usage changes. DCFSR uses full links capacities without using any load balancing mechanism. The flows are prioritized using Early Deadline First (EDF) policy.

For average end-to-end delay and ratio of dropped data packets, satisfying the EDF policy in DCFSR increases its average end-to-end delay and ratio of dropped data packets compared to the proposed scheme and VPTCA. The proposed scheme and VPTCA have similar average end-to-end delay and ratio of dropped data packets. Specifically, the average end-to-end delay using light loads were 0.17, 0.08, and 0.09 milliseconds (ms) and using heavy loads 0.28, 0.27, and 0.25 ms for DCFSR, VPTCA, and the proposed scheme, respectively. Moreover, using light loads, the proposed scheme and VPTCA have almost no dropped data packets while DCFSR has a dropped data packets ratio of around 0.4%. Using heavy loads, the dropped data packets ratios were 1.8%, 0.70%, and 0.78% for DCFSR, VPTCA, and the proposed scheme, respectively.

I. CPLEX versus Heuristic Algorithm

The MILP formulation is solved using CPLEX. CPLEX results provide the optimal solutions which are taken as a benchmark to evaluate the difference between them and the proposed heuristic algorithm. All experiments were conducted on an identical platform, a Linux machine with 24 Intel Xeon CPUs x5650@2.67 GHz and 47 GB of memory.

To show the validity of optimality for our proposed algorithm, we compared the results with the optimal ones obtained by CPLEX. We found that the final objective values of our proposed algorithm are fairly close to the optimum ones for all the cases under consideration.

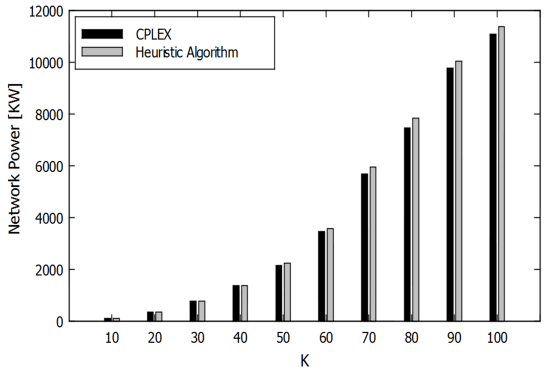


Figure 7. Comparison of power consumption between heuristic algorithm and CPLEX.

Figure 7 shows the differences between CPLEX and the heuristic algorithm in terms of power consumption with fat tree topology. The comparison was conducted for data center sizes ranging from 250 hosts ($k = 10$) to 250000 hosts ($k = 100$) with data center load of 30%. Note that K represents the number ports in a switch at a fat tree and the number of servers of a K-ary fat tree can be calculated as $K^3/4$. The results show that the gap between the optimal power consumption and the proposed heuristic algorithm power consumption is less than 4%. Although the proposed heuristic algorithm can provide solutions slightly less than the optimal, it is much more computationally efficient.

The proposed heuristic algorithm demonstrates high computational efficiency compared to CPLEX as shown in figure 8. The growth of computational time for the proposed algorithm increases linearly with the size of the data center, whereas the growth of computational time in CPLEX increases exponentially. There is a slight difference

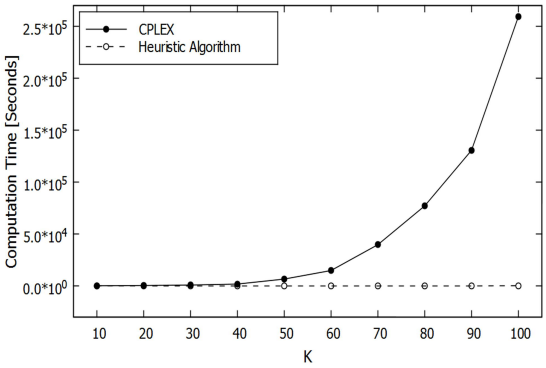


Figure 8. Comparison of computation time between heuristic algorithm and CPLEX.

between the solutions obtained by CPLEX and proposed algorithm; however, solving the problem in CPLEX will introduce high computational cost. As the size of the data center goes up, in contrast with the significant boost in computation time for CPLEX, the proposed algorithm solves the problem efficiently. The ratios of the solving time of CPLEX to that of the heuristic algorithm are considerable, which demonstrates the applicability and scalability for our proposed heuristic especially for large-scale (exascale) data centers.

VIII. Conclusion

The large number of redundant paths and low link utilization in data center networks can be exploited for energy saving. Most research on literature focuses on optimizing energy without any concern about the performance of the network or the ability to handle traffic bursts. In this paper, we conducted a study on saving energy in data center networks while guaranteeing same or similar performance to the original network. We formulate the problem as MILP, where the objective is to minimize energy consumption while introducing load balancing and link utilization thresholds as constraints to maintain network performance and to deal with traffic bursts. The problem solution succeeded to calculate the minimum energy; however, it showed high computational complexity. Thus, for implementation purposes to large data center networks, a suboptimal heuristic algorithm is proposed to solve the problem. The heuristic algorithm switches traffic to a subset of links, turns off unused switches and links, and uses valiant load balancing mechanism on active routes. Simulation experiments for the proposed model under different network topologies and data center loads show that the proposed model is able to save a considerable amount of energy, improve load balancing for both links and switches with minor effect on network performance.

References

[1] A. Hammadi and L. Mhamdi, "A survey on architectures and energy efficiency in data center networks," *Computer Communications*, vol. 40, pp. 1–21, 2014.

[2] J. Koomey, "Growth in data center electricity use 2005 to 2010," 2011.

[3] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, "It's not easy being green," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 211–222, 2012.

[4] J. Liu, F. Zhao, X. Liu, and W. He, "Challenges towards elastic power management in internet data centers," in *Distributed Computing Systems Workshops, 2009. ICDCS Workshops' 09. 29th IEEE International Conference on*. IEEE, Conference Proceedings, pp. 65–72.

[5] L. Huang, Q. Jia, X. Wang, S. Yang, and B. Li, "Pcube: Improving power efficiency in data center networks," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*. IEEE, Conference Proceedings, pp. 65–72.

[6] J.-Y. Shin, B. Wong, and E. G. Sirer, "Small-world datacenters," in *Proceedings of the 2nd ACM Symposium on Cloud Computing*. ACM, 2011, p. 2.

[7] T. Wang, Z. Su, Y. Xia, B. Qin, and M. Hamdi, "Novacube: A low latency torus-based network architecture for data centers," in *2014 IEEE Global Communications Conference*. IEEE, 2014, pp. 2252–2257.

[8] A. Singla, A. Singh, K. Ramachandran, L. Xu, and Y. Zhang, "Proteus: a topology malleable data center network," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*. ACM, 2010, p. 8.

[9] Y. Chen, R. Griffith, J. Liu, R. H. Katz, and A. D. Joseph, "Understanding tcp incast throughput collapse in datacenter networks," in *Proceedings of the 1st ACM workshop on Research on enterprise networking*. ACM, 2009, pp. 73–82.

[10] Y. Shang, D. Li, and M. Xu, "Greening data center networks with flow preemption and energy-aware routing," in *Local & Metropolitan Area Networks (LANMAN), 2013 19th IEEE Workshop on*. IEEE, 2013, pp. 1–6.

[11] K. Zheng, W. Zheng, L. Li, and X. Wang, "Powernets: Coordinating data center network with servers and cooling for power optimization," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 661–675, 2017.

[12] T. Wang, Y. Xia, J. Muppala, and M. Hamdi, "Achieving energy efficiency in data centers using an artificial intelligence abstraction model," *IEEE Transactions on Cloud Computing*, vol. PP, no. 99, pp. 1–1, 2015.

[13] K. Zheng and X. Wang, "Dynamic control of flow completion time for power efficiency of data center networks," in *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*. IEEE, 2017, pp. 340–350.

[14] M. Zhang, C. Yi, B. Liu, and B. Zhang, "Greente: Power-aware traffic engineering," in *Network Protocols (ICNP), 2010 18th IEEE International Conference on*. IEEE, Conference Proceedings, pp. 21–30.

[15] Y. Shang, D. Li, and M. Xu, "Energy-aware routing in data center network," in *Proceedings of the first ACM SIGCOMM workshop on Green networking*. ACM, Conference Proceedings, pp. 1–8.

[16] W. Ni, C. Huang, and J. Wub, "Provisioning high-availability datacenter networks for full bandwidth communication," *Computer Networks*, vol. 68, pp. 71–94, 2014.

[17] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4, pp. 63–74, 2008.

[18] International Business Machines (IBM) Corporation, "Cplex 12.5 user manual." [Online]. Available: http://pic.dhe.ibm.com/infocenter/cosinfoc/v12r5/index.jsp

[19] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu, "Energy proportional datacenter networks," in *ACM SIGARCH Computer Architecture News*, vol. 38. ACM, Conference Proceedings, pp. 338–347.

[20] H. Abu-Libdeh, P. Costa, A. Rowstron, G. O'Shea, and A. Donnelly, "Symbiotic routing in future data centers," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 4, pp. 51–62, 2010.

[21] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "Elastictree: Saving energy in data center networks," in *NSDI*, vol. 10, Conference Proceedings, pp. 249–264.

[22] X. Wang, Y. Yao, X. Wang, K. Lu, and Q. Cao, "Carpo: Correlation-aware power optimization in data center networks," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, Conference Proceedings, pp. 1125–1133.

[23] N. Vasi, P. Bhurat, D. Novakovi, M. Canini, S. Shekhar, and D. Kosti, "Identifying and using energy-critical paths," in *Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*. ACM, Conference Proceedings, p. 18.

[24] L. Wang, F. Zhang, C. Hou, J. A. Aroca, and Z. Liu, "Incorporating rate adaptation into green networking for future data centers," in *Network Computing and Applications (NCA), 2013 12th IEEE International Symposium on*. IEEE, 2013, pp. 106–109.

[25] K. Zheng, X. Wang, and X. Wang, "Powerfct: Power optimization of data center network with flow completion time constraints," in *Parallel and Distributed Processing Symposium (IPDPS), 2015 IEEE International*. IEEE, 2015, pp. 334–343.

[26] K. Zheng, X. Wang, and J. Liu, "Disco: Distributed traffic flow consolidation for power efficient data center network," in *IFIP Networking Conference (IFIP Networking) and Workshops, 2017*. IEEE, 2017, pp. 1–9.

[27] A. Carrega, S. Singh, R. Bolla, and R. Bruschi, "Applying traffic merging to datacenter networks," in *Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet*, 2012, p. 3.

[28] S. Saha, J. S. Deogun, and L. Xu, "Energy models driven green routing for data centers," in *Global Communications Conference (GLOBECOM), 2012 IEEE*. IEEE, Conference Proceedings, pp. 2529–2534.

[29] H. Jin, T. Cheocherngngarn, D. Levy, A. Smith, D. Pan, J. Liu, and N. Pissinou, "Joint host-network optimization for energy-efficient data center networking," in *Parallel & Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on*. IEEE, Conference Proceedings, pp. 623–634.

[30] L. Wang, F. Zhang, A. V. Vasilakos, C. Hou, and Z. Liu, "Joint virtual machine assignment and traffic engineering for green data center networks," *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, no. 3, pp. 107–112, 2014.

[31] L. Wang, F. Zhang, J. Arjona Aroca, A. V. Vasilakos, K. Zheng, C. Hou, D. Li, and Z. Liu, "Greendcn: a general framework for achieving energy efficiency in data center networks," *Selected Areas in Communications, IEEE Journal on*, vol. 32, no. 1, pp. 4–15, 2014.

[32] A. Dalvandi, M. Gurusamy, and K. C. Chua, "Time-aware vmflow placement, routing, and migration for power efficiency in data centers," *IEEE Transactions on Network and Service Management*, vol. 12, no. 3, pp. 349–362, 2015.

[33] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Energy-efficient resource allocation and provisioning framework for cloud data centers," *IEEE Transactions on Network and Service Management*, vol. 12, no. 3, pp. 377–391, 2015.

[34] Cisco press, "Cisco data center infrastructure 2.5 design guide."

[35] K. Bilal, S. U. R. Malik, O. Khalid, A. Hameed, E. Alvarez, V. Wijaysekara, R. Irfan, S. Shrestha, D. Dwivedy, and M. Ali, "A taxonomy and survey on green data center networks," *Future Generation Computer Systems*, vol. 36, pp. 189–208, 2014.

[36] K. Bilal, S. U. Khan, L. Zhang, H. Li, K. Hayat, S. A. Madani, N. Min-Allah, L. Wang, D. Chen, and M. Iqbal, "Quantitative comparisons of the state-of-the-art data center architectures," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 12, pp. 1771–1783, 2013.

[37] A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano *et al.*, "Jupiter rising: a decade of clos topologies and centralized control in google's datacenter network," *Communications of the ACM*, vol. 59, no. 9, pp. 88–97, 2016.

[38] D. Li, Y. Yu, W. He, K. Zheng, and B. He, "Willow: Saving data center network energy for network-limited flows," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 9, pp. 2610–2620, 2015.

[39] G. Dósa and J. Sgall, "First fit bin packing: A tight analysis," in *LIPIcs-Leibniz International Proceedings in Informatics*, vol. 20. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2013.

[40] G. Dósa, "The tight bound of first fit decreasing bin-packing algorithm is ffd (i)âĽď 11/9opt (i)+ 6/9," in *Combinatorics, Algorithms, Probabilistic and Experimental Methodologies*. Springer, 2007, pp. 1–11.

[41] S. Mahapatra and X. Yuan, "Load balancing mechanisms in data center networks," in *the 7th Int. Conf. & Expo on Emerging Technologies for a Smarter World (CEWIT)*, Conference Proceedings.

[42] D. Kliazovich, P. Bouvry, and S. U. Khan, "Greencloud: a packet-level simulator of energy-aware cloud computing data centers," *The Journal of Supercomputing*, vol. 62, no. 3, pp. 1263–1283, 2012.

[43] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu *et al.*, "Advances in network simulation," *IEEE computer*, vol. 33, no. 5, pp. 59–67, 2000.

[44] A. Gulati, A. Holler, M. Ji, G. Shanmuganathan, C. Waldspurger, and X. Zhu, "Vmware distributed resource management: Design, implementation, and lessons learned," *VMware Technical Journal*, vol. 1, no. 1, pp. 45–64, 2012.

[45] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat, "Hedera: dynamic flow scheduling for data center networks." in *Nsdi*, vol. 10, no. 8, 2010, pp. 89–92.

[46] D. Kliazovich, S. T. Arzo, F. Granelli, P. Bouvry, and S. U. Khan, "e-stab: Energy-efficient scheduling for cloud computing applications with traffic load balancing," in *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing*. IEEE, 2013, pp. 7–13.

[47] Cisco, "Cisco nexus 7000 series site preparation guide - technical specifications [cisco nexus 7000 series switches]," Jan 2018.

[48] "Core i7-5960x extreme edition review: Intel's overdue desktop 8-core is here," Techgage, accessed: 2014-08-29.

[49] C. Fiandrino, D. Kliazovich, P. Bouvry, and A. Zomaya, "Performance and energy efficiency metrics for communication systems of cloud computing data centers," *IEEE Transactions on Cloud Computing*, vol. PP, no. 99, pp. 1–1, 2015.

[50] D. Boru, D. Kliazovich, F. Granelli, P. Bouvry, and A. Y. Zomaya, "Energy-efficient data replication in cloud computing datacenters," *Cluster Computing*, vol. 18, no. 1, pp. 385–402, 2015.

[51] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM, 2010, pp. 267–280.

[52] T. Yang, Y. Lee, and A. Zomaya, "Collective energy-efficiency approach to data center networks planning," *IEEE Transactions on Cloud Computing*, 2015.

[53] L. Wang, F. Zhang, K. Zheng, A. V. Vasilakos, S. Ren, and Z. Liu, "Energy-efficient flow scheduling and routing with hard deadlines in data center networks," in *Distributed Computing Systems (ICDCS), 2014 IEEE 34th International Conference on*. IEEE, 2014, pp. 248–257.

Motassem Al-Tarazi (S'16) received his B.S. and M.S. degrees in computer information systems and computer science from Jordan University of Science and Technology, Jordan. He is currently working toward his Ph.D. degree in computer science at Iowa State University. His research interests include cloud computing, wireless network, communication network and Internet technology. He is a student member of IEEE.

Morris Chang (SM'08) a professor in the Department of Electrical Engineering at the University of South Florida. He received the Ph.D. degree from the North Carolina State University. His past industrial experiences include positions at Texas Instruments, Microelectronic Center of North Carolina and AT&T Bell Labs. He received the University Excellence in Teaching Award at Illinois Institute of Technology in 1999. His research interests include: cyber security, wireless networks, and energy efficient computer systems. In the last five years, his research projects on cyber security have been funded by DARPA. Currently, he is leading a DARPA project under Brandeis program focuses on privacy-preserving computation over Internet. He is a handling editor of Journal of Microprocessors and Microsystems and the Associate Editor-in-Chief of IEEE IT Professional. He is a senior member of IEEE.