Paper

Energy Aware Data Centers and Networks: a Survey

Piotr Arabas

Institute of Control and Computation Engineering, Warsaw University of Technology, Warsaw, Poland

https://doi.org/10.26636/jtit.2018.129818

Abstract—The past years have brought about a great variety of clusters and clouds. This, combined with their increasing size and complexity, has resulted in an obvious need for power-saving control mechanisms. Upon presenting a basis on which such solutions - namely low-level power control interfaces, CPU governors and network topologies – are constructed, the paper summarizes network and cluster resources control algorithms. Finally, the need for integrated, hierarchical control is expressed, and specific examples are provided.

Keywords*—*energy efficiency, green networks, resource allocation, HPC.

1. Introduction

Distributed computer systems – clusters and clouds – have gained in popularity over the past years due to their versatility and easily scalable processing power. The important drivers of this are of economic nature: leased infrastructure is usually cheaper than owned hardware, thanks to better utilization and greater efficiency of high-end equipment. On the other hand, the costs of running large installations are tremendous. Therefore any improvement, negligible in a small-scale scenario, is worth considering.

One of the major cost factors is energy consumption [1]–[3]. It is important to note that energy consumed by IT equipment is nearly fully transformed into heat. Therefore, efficient cooling systems must be built – a task which becomes more and more difficult and costly, as the packing density increases.

The electronics industry has made a great effort to lower energy consumption of hardware: processors, memory, etc., but the demand for computing power grows so quickly that the activities undertaken are insufficient to solve the problem [4]. The most viable strategy consists in the application of power-aware control algorithms enabling to reduce power consumption during periods of limited load. This is possible, as lower level mechanisms, namely measurement and power control interfaces of specific IT system components, are readily available. Furthermore, the most popular operating systems offer some power control functionalities as well. The main challenge now is to orchestrate all these mechanisms to build efficient and flexible power control systems encompassing all elements of cloud infrastructure. The remaining part of this paper is organized as follows. Section 2 briefly summarizes hardware-related, low level technologies, namely power-scaling, operating system level controllers and power-aware functionalities and the design of networks. Section 3 reviews a solution devoted to maximizing power efficiency of networks used to connect computing nodes of clusters and clouds. Section 4 describes selected resource allocation algorithms, and Section 5 proposes some integrated systems controlling all aspects of cloud operation. Section 6 concludes the survey.

2. Available Technologies

This section briefly presents technologies developed in recent years, enabling to control power consumption of computers and network equipment, and thus serving as a basis for designing more complex solutions.

2.1. Power-scaling Techniques

With the growing computing power of processors[1], the problem of energy dissipation and efficient cooling becomes important. So, the first attempts to reduce power consumption were more concerned with preventing the generation of excessive heat, rather than with energy savings. However, both targets were addressed by ACPI specification propositions. Now the ACPI standard [5] defines a relatively large number of energy-aware states for the operation of processors. *Performance states P*, associated with higher consumption levels, make it possible to choose the computing power required for processing tasks by means of decreasing processor clock frequency and supply voltage. This technique, known as dynamic voltage and frequency scaling (DVFS), is used by CMOS circuits which are controlled by the potential present at transistor gates.

As energy is required only when this potential is to be changed, lowering the clock frequency and voltage enables to reduce the power consumed. Additional reduction in power consumption is possible in the so-called *C states*, where the processor may enter a temporary sleep mode – i.e. may stop its operation and switch off some of its circuits. Figure 1 shows variations in computing power and energy consumption for specific energy-aware states. The

[1] Introduction of Intel 486 may be considered a time when the problem became broadly recognized, as it was the first consumer-use processor which needed fan cooling.

higher the number of processor subsystems switched off, the less energy is consumed, at the expense of a longer transition time, however. Common adoption of the ACPI standard has made it possible to implement processor power control modules in the majority of operating systems (e.g. Linux power governors) – see Subsection 2.2.

The adoption of similar techniques in network hardware is a little bit slower. However, IEEE802.3az [6] may be considered to be the most important standard, formerly known as Energy Efficient Ethernet employing Low Power Idle (LPI). It operates in a manner similar to C processor states. Implementation of LPI for highly redundant links present in typical cluster topologies may be the most natural way of reducing power consumed by the network. It must be also noted that some network devices are composed of hierarchically connected subsystems (e.g. Mellanox InfiniBand switches [9]) allowing to switch off some components, reducing capacity but preserving connectivity). Furthermore, modern routers and switches tend to use operating systems stemming from the main line of universal OSes (FreeBSD or Linux clones) running on PC-like equipment providing their control plane functions. This allows to accompany data plane-based techniques (e.g. 802.3az) with these inherited from baseline OS and data plane processors (e.g. ACPI and power governors).

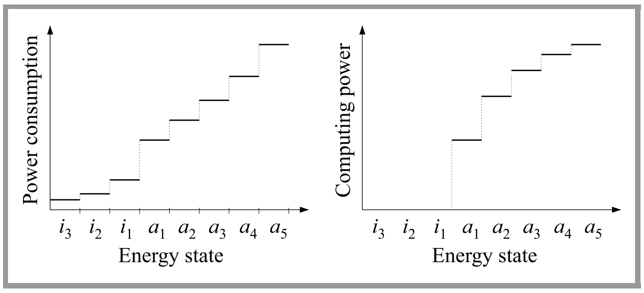

Fig. 1. Power consumption (a) and computation power (b) in subsequent energy-aware states. States i_3 to i_1 are idle (C states). States a_1 to a_5 are active (P states). For more detailed characteristics, see e.g. [4], [7], [8].

Successful and broad exploitation of power saving mechanisms requires standardized interfaces. The ACPI specification referred to above unifies control of the processor and some other devices. A number of other standards, e.g. RAPL for Intel processors [10] or PAPI specification [11], and, to some extent, the IPMI interface [12], allow to access power measurements related to some components of a computer system[2]. Unified interfaces, such as the green abstraction layer (GAL) [13], [14], are the next generation solution having the form of a generalized interface providing the functionality of setting energy states for all computer system components and also, which is another innovation, of querying energy-aware capabilities of the said components.

2.2. Local CPU Power Control

The availability of power consumption statistics and frequency scaling interfaces along with system load measurements makes the operating system kernel an ideal location to implement power saving processor control. In Linux, the most popular controller – on-demand [15] power governor – applies a simple yet robust mechanism responding to load fluctuations by selecting CPU frequency.

The operation relies on two thresholds. When the processor load grows above the higher threshold, the controller sets the maximum frequency value. If the load decreases below the lower threshold, the frequency is lowered by one step. As processor load is relative to frequency, this algorithm leads to selecting the clock rate which should be sufficient to handle the current tasks without causing excessive processor loads and delaying task execution. The resulting power characteristic, i.e. the ratio between power consumption and processor load, usually has the form of a nonlinear function – see Fig. 2. It must be noted, however, that even under the no-load condition, the processor still consumes power. Intel *p-state* [16] may be an example of a more complex governor, where the PI controller is relied upon to keep processor frequency close to the value ensuring optimized efficiency. Although these controllers offer significant energy savings, there is still some area for improvement, especially when the load characteristics are known. In such a case it is possible to construct specialized power governors suitable for specialized usage scenarios, e.g. a web server, large scale computations or network traffic filtering [7], [8], [17]. The savings achieved by the algorithm presented in [7] are attained mostly by exploiting identified dynamics of applications running, and thus designing a control law that makes it possible to adequately react to load changes. A discussion concerned with the possibility of designing PI and PID processor frequency

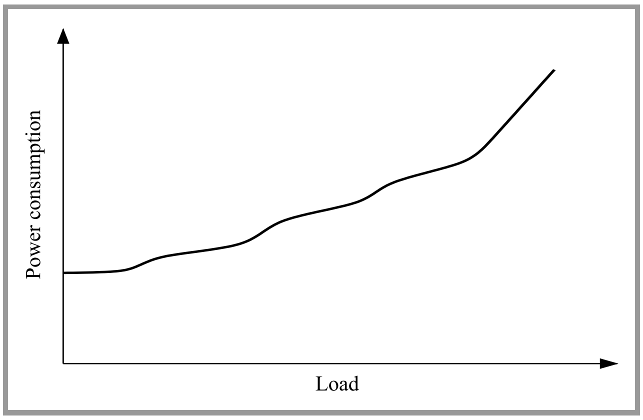

Fig. 2. Example of a power consumption curve for a processor with an on-demand governor. For more precise measurement data, see e.g. [4], [17].

[2] Running average power limit (RAPL) provides access to various processor registers containing, inter alia, measurements of power consumed by processor components, cache memory, etc. While the original aim was primarily to provide power capping, these measurements are precise, short term averages relevant for designing dynamic processor frequency controllers [7].

controllers may be also found in [18]. Similar mechanisms may be applied to control network devices, either software (Linux)-based routers implemented on general grade PC-class machines [19], [20] or specialized network devices – see, for instance, [9].

2.3. Data Center Interconnect Network

Networks connecting machines that operate in clusters are usually designed to maximize throughput between any two components while providing extremely low latency and high availability. As a result, typical topologies are highly regular, usually hierarchical. Dominant technologies for machine-to-machine networks are Ethernet 1 Gbps, 10 Gbps and higher speeds or InfiniBand [21]. The use of a consistent technology across the data center enables to build a switched network limiting delays and complexity. Traditional topologies include two or three layers of switches with the lowest level switches installed on top of the rack (ToR) in the cabinets housing the servers (Fig. 3). To attain high reliability and to multiply bandwidth, the computers may use more than one network interface, connected to different switches. Similarly, ToR switches are interconnected with upper level devices [23]. It must be noted that providing full bandwidth between any pair of hosts requires that links connecting the individual levels offer capacity being at least equal to the sum of lower level links connected to the node. Such a topology requires using costly high-end switches to provide appropriate switching capacity and the number of ports needed, e.g. a 24-port 10 Gbps switch requires six 40 Gbps uplinks terminated at the upper level switch, to provide full bisection bandwidth. The example demonstrates the high cost involved, which grows exponentially as the network expands. In practice, based on the available size (i.e. number of ports) and the switching capacity of devices, it is possible to build clusters of up to several thousands of nodes using two levels of switches: ToR and core [22]. For larger installations, it is necessary to use at least three levels: ToR, aggregation, and core. The cost may be reduced by oversubscription, i.e. by connecting more nodes to ToR switches than their aggregated uplink bandwidth allows in the base, fully provisioned scenario. However, it also reduces the bandwidth available to hosts[3].

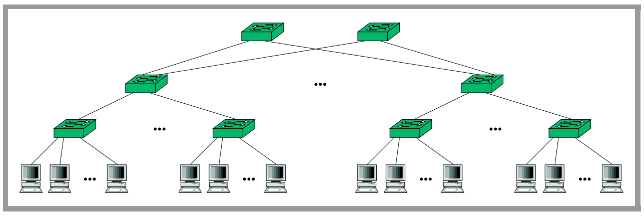

Fig. 3. Traditional topology of a cluster interconnect network [22], [23].

While oversubscription makes it possible to shift the limit, it does not allow to scale the network economically. To lower the cost and provide scalability, a number of topologies using baseline switches have been proposed (see e.g. [22]). They all exploit, to some point, the concept of a Clos network [24] to build a mesh network of a large number of simple devices having the same number of inputs, i.e. links connecting to the lower level of the tree, and outputs (uplinks). As the cost of higher class devices rises rapidly, it is possible to build a cheaper network, at the expense of more complex wiring[4]. A number of similar topologies attempting to solve some of the complexity-related issues have been proposed, with fat tree [25], [26] being the most common of them (along with some variations – see, e.g. [27] – flattened fat trees or [28], [29] – numerous variants of butterfly networks).

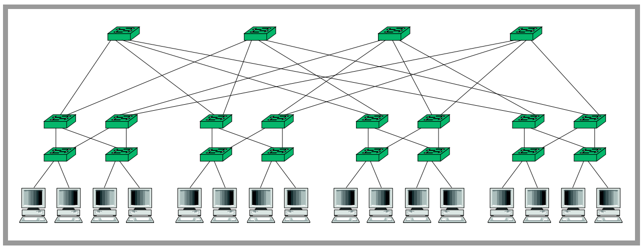

Fig. 4. Example of fat tree topology of a cluster interconnect network [22].

Regardless of which class of equipment is used, the resulting multiple tree topologies are substantially redundant, as they tend to provide full mesh connectivity and an equal level of service to all nodes. However, as the traffic pattern depends on the type of tasks performed by the cluster, it is often possible that the load is not spread equally across the network, so a substantial amount of energy may be saved if some of the devices are switched off or can operate with reduced performance. Provided they are equipped with some of the mechanisms mentioned in Subsection 2.1, it is possible to incorporate power saving functions into the network management system. Furthermore, as the entire network is managed by the same institution, application of the centralized controller is possible and in many aspects favorable. On the other hand, the large number of nodes and links makes the task of mathematical programming (e.g. known from [30]) formidable. Section 3 contains some proposals on how to solve this problem.

3. Energy Efficient Network Control

The power saving mechanism described in Subsection 2.1 may be helpful for constructing a local control mechanism that is relevant for a single device. However, especially when a whole network is considered, it may be more beneficial to rely on global, network-wide control. Such a mechanism, apart from acting reactively by setting the speed of

[3] At least in the worst-case scenario, when all nodes communicate simultaneously. In scenarios in which statistical multiplexing is possible, oversubscription is viable, provided its level is properly calculated.

[4] Which, in turn, may also be relatively cheap, as simpler copper cables may be used.

a single device in response to load, may reconfigure traffic paths to optimize energy levels of all devices within the network. Results of recent studies considering general purpose networks suggest that the overall energy consumption may be considerably reduced by the application of appropriate control mechanisms [31], [32]. In short, the proposed solutions may be divided into two major groups:

- distributed algorithms,
- centralized controllers.

The first group of algorithms is typically built on the base of existing routing protocols. Their main advantage is good scalability and relative robustness achieved by decomposition. Therefore, they are supposed to be correct (if not the sole feasible) approach in the case of large, general purpose networks. However, networks connecting machines within a cluster are very specific. Not only is their topology specialized (see Subsection 2.3), but they often use homogenic equipment and, more importantly, they are usually controlled by a single organization with the help of dedicated software. All these characteristics make it more acceptable to build a centralized controller in a scenario involving a free topology network.

Techniques used in this case are mostly related to traffic engineering and provisioning and require solving complex optimization tasks. However, as most large installations use some kind of a network management system, implementation of control may be relatively easy.

Both approaches are based on the obvious observation that redundant links, so abundant in cluster topologies, could be switched off or could operate at a limited rate during periods of lower loads. An additional source of redundancy present in cluster networks has the form of bundled links used to multiply their bandwidth. Typically, distributed power saving algorithms rely also on distributed sensing of the network load and react to it by increasing or decreasing the set of active paths. Centralized algorithms may rely on the monitoring system (if present) and request load statistics or even try to forecast traffic patterns based on historic data. The problem that most centralized algorithms suffer from is the availability of a traffic matrix necessary to define target demands between network nodes and to assess QoS. A situation in which the traffic matrix is provided by upper control level (e.g. tasks scheduling, see Section 5) and reflects the real needs of end users is beneficial, but not common. Deriving the traffic matrix from on-line measurements – an activity performed by many distributed controllers – may lead to an oscillatory behavior of flows due to interference with low level flow control mechanisms, e.g. TCP. Decoupling control by using different time scales (much longer for traffic engineering then for flow control) may be a solution here. A more detailed discussion is presented, for instance, in [33]–[36].

3.1. Centralized Power Save Network Control

Before offering specific, cluster-related formulations, some general proposals will be presented. Typically, they tend to formulate an optimization problem similar to the network design problem [30] or to the QoS provisioning task [34], [37], but with a cost function defined as total energy consumed by all components of the network. Such mathematical programming tasks are presented in [2], [33], [38], [39] and [40]. In [38], Chiaraviglio *et al.* uses integer linear programming to identify network nodes and links that can be switched off. Chabarek *et al.* proposes, in [2], reducing power consumption by finding links and line cards that can be switched off when a large mixed-integer linear problem is solved for a given traffic matrix. Similarly, [41] tries to scale link rates by selectively switching the fibers they are made of on and off. Optimization of a two-level structure with an IP network built with the use of optical equipment is covered in [42].

Optimization of an energy-aware network involves more difficulties than in the case of typical shortest path calculations, as paths are not independent but should be typically aggregated on a subset of links to allow switching off the remaining ones. Furthermore, energy consumption of devices often depends, in a non-convex way, on their load due to intrinsic nonlinearities and must be modeled using binary variables. All this makes the fully formulated problem NP-complete and large. Relaxing some constraints to simplify the solution may result in suboptimality or instability [43] of the system. In [44], the following variants of the mathematical programming task were presented, from an exact mixed integer programming (MIP) formulation, including complete routing and energy-state decisions, to simplified, continuous formulations:

1. complete network management problem with full routing and energy state control – MIP task,
2. energy state control with predefined paths – simplification of above MIP task,
3. continuous relaxation of formulation 1,
4. continuous relaxation of formulation 2.

Unfortunately, formulation 1 is a complex, NP-complete problem, so finding the solution for larger networks is usually impossible. Avoiding routing, like in the case of formulation 2, makes the computation easier, at the cost of the earlier path generation procedure referred to earlier[5]. Application of continuous relaxation variants yields multipath solutions, which are typically avoided due to the unacceptable jitter level and packet reordering.

Application of heuristics to eliminate some paths and to consolidate flows may be the solution here, at the expense of power efficiency or QoS. However, in some data center network topologies (e.g. fat tree), multipath routing is a must to balance the load of multiplied links, so the continuous solution may be easier to apply (although

[5] The resulting mechanism is usually suboptimal and, depending on network complexity, path generation may be a difficult and a time consuming task. On the other hand, for highly regular topologies, preparation of some predefined sets of paths seems viable.

some rounding off to meet equipment capacities may be necessary).

To demonstrate its complexity, the definition of the full MIP task [45]–[47] is provided below:

$$\min_{x_c, y_{ek}, z_r, u_{ed}} \left[\sum_{r=1}^{R} T_r z_r + \sum_{c=1}^{C} W_c x_c + \sum_{e=1}^{E} \sum_{k=1}^{K} \xi_{ek} y_{ek} \right] , \qquad (1)$$

subject to the following constraints:

$$\forall_{e=1,\dots,E} \quad \sum_{k=1}^{K} y_{ek} \leq 1 , \qquad (2)$$

$$\forall_{\substack{d=1,\dots,D,\\ c=1,\dots,C}} \quad \sum_{p=1}^{P} l_{cp} \sum_{e=1}^{E} a_{ep} u_{ed} \leq x_c , \qquad (3)$$

$$\forall_{\substack{d=1,\dots,D,\\ c=1,\dots,C}} \quad \sum_{p=1}^{P} l_{cp} \sum_{e=1}^{E} b_{ep} u_{ed} \leq x_c , \qquad (4)$$

$$\forall_{\substack{r=1,\dots,R,\\ c=1,\dots,C}} \quad g_{rc} x_c \leq z_r , \qquad (5)$$

$$\forall_{\substack{d=1,\dots,D,\\ r=1,\dots,R,\\ p=s_d}} \sum_{c=1}^{C} g_{rc} l_{cp} \sum_{e=1}^{E} a_{ep} u_{ed} - \sum_{c=1}^{C} g_{rc} l_{cp} \sum_{e=1}^{E} b_{ep} u_{ed} = 1 , \qquad (6)$$

$$\forall_{\substack{d=1,\dots,D,\\ r=1,\dots,R\\ p\neq t_d, p\neq s_d}} \sum_{c=1}^{C} g_{rc} \sum_{p=1}^{P} l_{cp} \sum_{e=1}^{E} a_{ep} u_{ed} - \sum_{c=1}^{C} g_{rc} \sum_{p=1}^{P} l_{cp} \sum_{e=1}^{E} b_{ep} u_{ed} = 0 , \qquad (7)$$

$$\forall_{\substack{d=1,\dots,D,\\ r=1,\dots,R,\ p=t_d}} \sum_{c=1}^{C} g_{rc} l_{cp} \sum_{e=1}^{E} a_{ep} u_{ed} - \sum_{c=1}^{C} g_{rc} l_{cp} \sum_{e=1}^{E} b_{ep} u_{ed} = -1 , \qquad (8)$$

$$\forall_{e=1,\dots,E} \quad \sum_{d=1}^{D} V_d u_{ed} \leq \sum_{k=1}^{K} M_{ek} y_{ek} , \qquad (9)$$

where: W_c and T_r are power consumption values of the card c and the router r, respectively, M_{ek} is throughput and ξ_{ek} power consumption of link e in the state k, $y_{ek} = 1$ if the energy state of link e is set to k (0 otherwise), $z_r = 1$ if router r transmits any flow (0 otherwise), $x_c = 1$ if card c transmits any flow (0 otherwise), $l_{cp} = 1$ if port (one of link endpoints) p is on the card c (0 otherwise), $u_{ed} = 1$ if path d leads through the link e (0 otherwise), binary constants a_{ep} and b_{ep} are used to define ingress and egress links (e) of port p, g_{rc} is set to 1 if card c belongs to the router r.

The complexity of the problem results from the fact that authors have combined the routing task - see flow continuity constraints (6)–(8) and link capacity constraint (9) – with hierarchic layout of the network node (router) – constraints (3)–(5) and the multiple energy state model of a link – constraint (2).

An efficient solution of such a complex task is possible by relying on heuristics, usually built as repetitive solving of simpler (usually relaxed) mathematic programming tasks. Relevant examples may be found in [41], [48], where the algorithm is run for a predefined set of links or in [49], [50], where the solution of a full routing task is replaced by the selection of paths from a set provided. Similarly, Garroppo *et al.* [51] solves a relaxed task to determine the use of links, and then runs heuristics to find out which links within the bundle may be switched off. Aggregation of nodes and demands may provide a precise solution by mathematical programming. However, one must remember that additional operations are needed to de-aggregate the results [38].

As it was explained in Subsection 2.3, the topologies used in cluster networks are highly regular and redundant. Although they provide full mesh connectivity, active links form a relatively sparse tree during periods of limited load. Such behavior allows to aggregate link loads, easily leading to reduced power consumption. The design of heuristics may be greatly simplified as well. The most popular strategies are greedy algorithms, attempting to pack as many flows as possible within a limited number of switches, moving from the lower layer up, to reduce power consumption. In typical two-layer fat tree topologies, the upper layer (i.e. aggregation) switches form a full mesh. Therefore, after consolidating flows to reduce the number of active lower layer switches, upper layer switches may be chosen easily, as the selection task boils down to powering up an appropriate number of such switches. In topologies with more than two layers, the procedure described should be repeated recursively, just like in HERO [52] (although authors discussed only two-layer case).

ElasticTree [53] may serve as another example of such an approach, where flows are assigned starting from leftmost switch, based on their declared peak rate. While the algorithm is time efficient, its main drawback consists in the fact that it needs a correct traffic matrix and allocates flows based on peak rate, which leads to a relatively low network utilization level. To achieve higher loads on selected links and to enable better consolidation, the algorithm must be aware, in some way, of the traffic level and must act adaptively. In [54], Wang *et al.* proposed CARPO – Correlation-Aware Power Optimization, based on observations showing that in the usual cluster operation not only is the network underutilized, but also many if not most of flows are negatively correlated. This observation implies that a much higher number of flows may be served by a single link than when it is computed based solely on the declared peak rate. Instead, the authors propose to use 90 percentile rates and to construct a greedy flow consolidation algorithm using flow correlation to calculate traffic mixing coefficients. The algorithm is repeated regularly and starts with calculating the flow correlation matrix. Then, it packs flows on switches, left to right, and taking into account the computed coefficients.

The packing flows from left to right may result in (possibly short-term) overload of the leftmost switches and in lowering QoS. To prevent this, authors of AgreeFlow [55] propose to balance the load of active switches, and propose another interesting concept known as Flow-set routing and Lazy rerouting. The former is based on the presence of many flows which can be routed along the same path. To simplify the solution and speed-up flow consolidation, they are assigned to the common flow set based on a hash function. The algorithm's calculations are meant to be regularly repeated to react to flow fluctuations that may lead to a change in a large number of routing paths. To limit the stress of the underlying control plane protocols (OpenFlow), they apply Lazy rerouting, i.e. delay the setting paths until transmitting the first packet of a new flow.

PowerFCT [56] attempts to deal with QoS by very precise modeling of transmit queues in the switches. Thanks to the application of an ECN derived protocol (namely DCTCP), the queue length may be calculated in a simple manner. To facilitate the algorithm's operation, different classes of service flows are divided into two groups: long lasting non-critical flows and the ones with a defined completion time. The flows are routed by a heuristic algorithm which sets the energy states of network devices to meet QoS requirements.

3.2. Distributed Power Save Network Control

In general networks, the distributed control scheme is preferred over the centralized one due to its reliability and scalability. Some solutions of this class may be suitable for cluster networks as well, especially for larger ones that rely on IP routing protocols to simplify management, because distributed energy-aware mechanisms are typically built as extensions of existing routing protocols – e.g., OSPF [57], [58], [59] or MPLS. Extension of the signaling infrastructure with green functions allows to partially overcome the absence of the traffic matrix [48] – the past state of the network can be used to compute flows [58]. Apart from traditional traffic engineering, the Software Defined Network (SDN) concept may be exploited here as well. Although SDN usually uses a central controller, decisions may be implemented locally with the help of existing routing protocols.

In *GRiDA* [58], Bianzino *et al.* relied on information inferred from augmented OSPF LSA messages to build the topology and find out the congested links. With this data, nodes may decide to reduce their power consumption by switching links off. The decision is taken based not only on the current network state, but also on past observations. The node tries to set the configuration of links minimizing its cost function being a sum of power consumption and penalty. Penalty is used to accumulate the knowledge about the node's role in the network. If the decision leads to congestion, penalty is increased additively, while for beneficial decisions, it is decreased multiplicatively. The most important feature of GRiDA is that it does not need the traffic matrix. Instead, it learns the network's topology and state from LSA. As sending an augmented LSA may be spawned by topology change or congestion, it allows to construct a reactive algorithm suitable for networks of medium dynamics (i.e. measured in seconds). A similar algorithm was proposed in [60]. The main difference is the use of a rule-based mechanism for switching the links leading to the router on and off. Relatively good results (at least in low-load periods) were obtained for the roll-back last (RL) activation strategy and the least loaded link (LLL) deactivation strategy. The former switches on the link whose deactivation has caused the congestion, the latter selects the link carrying the least (possibly no) load to be switched off.

SENATOR [61] may serve as an example of the algorithm exploiting OSPF and SDN infrastructure, in which the SDN controller is used to switch off the unnecessary links, while OSPF is used to propagate topology changes. To ensure a smooth topology change, temporary tunnels are used to redirect traffic along the envisaged paths. All this makes the algorithm is a hybrid solution – the decision to switch off links is computed globally and is executed using an SDN controller. The new routing paths are computed in a distributed manner, however.

4. Power-saving Resource Allocation

As most of energy consumed by any cluster is used by computing servers, it is crucial to manage resource usage in a manner allowing to limit their power needs without deteriorating QoS significantly. This implies cooperation between the resource allocation system and low level power monitoring and power-saving mechanisms described in Subsection 2.1. To achieve this, one can formulate a power-aware resource allocation task similar to the network control task described in Subsection 3.1. Defining this task requires the knowledge of power characteristics of the servers. As the servers are not fully power-proportional, its solution typically results in some kind of task and server consolidation. The solution of this task may be either precise, determined with the help of mathematic programming, or approximated, relying on various heuristics, and serves as a basis for designing many algorithms.

Another group of purely heuristic algorithms is based solely on the server consolidation postulate – i.e. they are policy-based mechanisms assuming that assigning tasks to a minimum number of servers allowing to maintain the required QoS level is the optimal or close-to-optimal approach. It may be easily shown by a computational experiment that when long term average power consumption and QoS metrics are considered and the equipment is homogenic, such a solution is optimal [62]. In a more dynamic case, however, especially when task characteristics are uncertain or varying, a more complicated mechanism should be applied. Following the classification proposed in [63], it is possible to distinguish three types of resource allocation mechanisms:

- predictive allocation algorithms,
- reactive allocation strategies,
- a hybrid of the above.

The predictive approach takes into account historical observations of the load imposed on servers by tasks to forecast its future evolution and to assign resources to meet power and QoS criteria. On the contrary, the reactive mechanism relies on system observations and tries to execute some previously prepared actions to bring the system close to the defined working point. Typically, predictive algorithms operate based on longer time frames of hours, while reactive algorithms need to be much faster, with repetition occurring every few minutes. The shortcoming of the reactive approach is that large workload changes are managed with difficulty, while they can be easily handled by predictive mechanisms, provided that forecasts are available. On the other hand, predictive mechanisms cannot react to small workload fluctuations between repetitions, which may lead to unsatisfactory performance.

The proposal of hybrid mechanisms is a simple consequence of these observations. The predictive mechanism may be applied in the long term to facilitate long calculations needed to find the solution, while the reactive one acts between repetitions to accommodate workload fluctuations.

4.1. Predictive Allocation Algorithms

The graph coloring algorithm [64] may serve as an example of the predictive approach. It uses graph coloring to assign, to the servers, resource units demanded by tasks represented by graph nodes. Links in the graph describe time dependencies among tasks. The resulting problem may be solved by mathematic programming. However, a more efficient heuristic is proposed. The pMapper mechanism described in [62] is a type of a predictive algorithm aiming to migrate virtual machines between servers and, hence, packing tasks in a power efficient manner. Importantly, the mechanism takes into account not only current power consumption and QoS requirements of the individual applications, but also the migration history to prevent excessive overheads.

The allocation algorithm proposed in [65] is much more detailed, as it computes task allocations and fine-grained energy configuration of servers (namely P-states), simultaneously modeling heat generation and air conditioning costs. Two versions of mathematical programming tasks are proposed. One tends to minimize the aggregated tasks' completion time based on the total power consumption constraint, while the other minimizes power consumption with limited completion time. The allocations are found by solving relaxed continuous problems and then applying a heuristic algorithm to get a feasible integer solution.

4.2. Reactive Allocation Strategies

Most virtual machine migration algorithms may be considered reactive, they are typically spawned by the loss of QoS or power efficiency and do not analyze any historical data (pMapper [62] seems to be a special, very complex example). One of the typical strategies used to allocate virtual machines to servers is best fit decreasing (BFD) [66], which starts with VMs sorted in the descending order of the demanded resources and assigns them to the server having the minimum computing capacity. Power and computation capacity BFD (PCA-BFD) is a mutation of this method, and it computes the servers' power consumption to computing capacity ratio to take into account power efficiency as well. Similarly, low perturbation bin packing (LPBP) attempts to migrate VMs from the most power consuming server to the one with the lowest power demand. The variation of the method proposed in [66] differs in terms of the sorting order: it starts from the least demanding VM and allocates it to the server with the lowest power consumption. Similar strategies are described in [67], the most trivial being first fit (FF), in which the task is assigned to the first machine with sufficient capacity. Next fit (NF) tries to assign the task to the last allocated machine, as the easiest way of consolidation. Max rest on requirements (MRR) is also called the worst fit method, as it tends to allocate tasks to servers with the maximum remaining capacity. The idea of delayed allocation is the most important contribution presented in [67]. Task are allocated in groups – such an approach utilizes server capacity much better than in the case of allocating single tasks. The authors of [68] not only allocate tasks to servers based on the best fit strategy, but also control processor frequency to attain best efficiency.

4.3. Hybrid Allocation Algorithms

The hybrid mechanism proposed in [69] comprises two complementary tasks. The first one is a predictive algorithm based on Fourier analysis of historic workload traces. Such an approach makes it possible to find out regular changes of the workload profile and to prepare base resource allocation. On the other hand, it cannot account for sudden workload surges, which are controlled by the reactive mechanism adding resources to the tasks violating QoS thresholds.

The algorithm presented in [70] may be considered of the hybrid variety, as it uses predictions for the allocation of virtual machines, while implementing a reactive migration algorithm as well. Apart from a rather sophisticated framework incorporating some network monitoring functions, a proposition of novel statistics to estimate thresholds used for host overload and VM selection is another important contribution of this particular solution. The allocation mechanism presented in [71] may be considered a trivial case of a hybrid algorithm. It uses an auto regressive model to predict the future load for the task relied upon by the machine allocation mechanism, and enters unused machines into idle state. The reactive mechanism switches off those machines that have been idle past the declared time-out. By applying such a delayed power-off strategy, the authors attempt to refrain from switching off all unused servers, at least during periods of high load variations.

5. Hierarchical Control

While network control algorithms may help reduce the power consumed by network devices, and while task allocation or scheduling may economize power necessary for computers, the overall result may be greatly improved by coordinating operation of both components. It is obvious that consolidation of tasks on the part of a cluster may lead to reducing power consumption of both servers and the network. To build a manageable control structure, it is usually necessary to divide responsibility between a number of controllers with task allocation performed on the upper levels. Then it is possible to predict the traffic matrix and to identify transmission paths reducing power consumption.

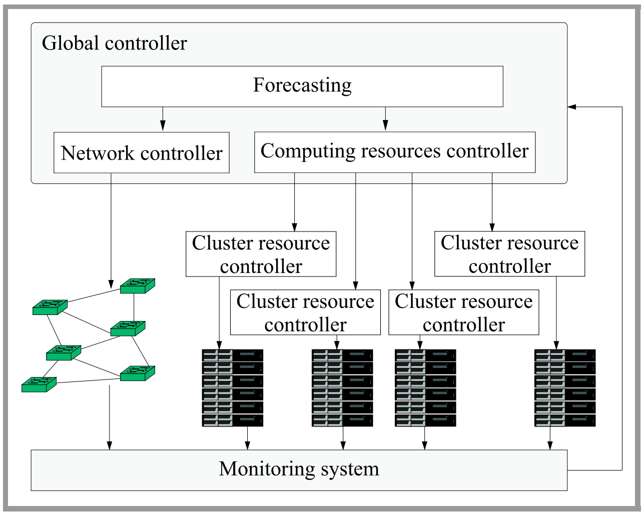
Global controller
Forecasting
Network controller
Computing resources controller
Cluster resource controller
Cluster resource controller
Cluster resource controller
Cluster resource controller
Monitoring system

Fig. 5. Hierarchical control system [72].

The control framework proposed in [72] comprises two levels: the global level with separate network and computing resources controllers, and the local level with resource managers assigned to subsequent clusters (see Fig. 5). The global controller is responsible for network configuration (including selecting power states of links) and for assignment of tasks to the clusters. The allocation of tasks to processors is carried out by cluster resource controllers. The proposed mathematical programming task resembles that of the network control case [46], Eqs. (1)–(9) with their performance index augmented by the addition of the total power consumed by computing servers:

$$E_F(\mathbf{x_f}) = \sum_{f=1}^{F}\sum_{k=1}^{K_f} \bar{P}_f^k x_f^k \ , \tag{10}$$

where $x_f^k = 1$ if the cluster f is in the state k (0 otherwise), and $\bar{P}_f^k$ is power consumed by the cluster f in state k. Constraints related to network function are defined in a manner similar to (2)–(9), and are accompanied by a set of inequalities describing the limitations of clusters:

$$\forall_{f=1,\ldots,F} \quad \sum_{k=1}^{K_f} x_f^k \leq 1 \ , \tag{11}$$

$$\forall_{f=1,\ldots,F} \quad \sum_{j=1}^{J} W_j \vartheta_{fj} \leq \sum_{k=1}^{K_f} \bar{\Theta}_f^k x_f^k \ , \tag{12}$$

$$\forall_{f=1,\ldots,F} \quad \sum_{j=1}^{J} M_j \vartheta_{fj} \leq \Psi_f \ , \tag{13}$$

$$\forall_{j=1,\ldots,J} \quad \frac{W_j}{\sum_{f=1}^{F} \vartheta_{fj}\left(\sum_{k=1}^{K_f} \bar{\Theta}_f^k x_f^k - \sum_{i=1,i\neq j}^{J} W_i \vartheta_{fi}\right)} \leq T_j \ , \tag{14}$$

where: $\vartheta_{fj} = 1$ is assignment of task j to cluster f, $\bar{\Theta}_k^f$ is the computing capacity of cluster f in energy state k, W_j, M_j and T_j are workload (in MIPS), data size (in MB) and completion time T_j of task j.

The constraint (11) forces the cluster to work in a single energy-aware state, (12) and (13) are processing and memory capacity constraints. The most complicated, non-linear constraint (14) is the makespan limitation. Although the constraint (14) may be easily linearized, the resulting task complexity resembles that of the network control case. It must be also noted that hierarchical decomposition allows to use a simplified model of the cluster – it is described by a single processing element of the aggregated capacity and multilevel energy state. The authors suggest to solve the problem using a heuristic method based on successive continuous relaxation.

An example of a relatively complex, hierarchical structure is presented in [73]. It involves three levels: local controllers find the power-saving paths in their domain (part of cluster network), resource allocation controllers allocate tasks to processors, while the global controller coordinates actions concerning the network and the servers by controlling paths leading to top level switches. In general, both the task allocation algorithm and the path selection algorithm tend to consolidate task on servers and paths on links, respectively. The algorithms used are relatively simple heuristics (reuse of link/server till capacity limit). The authors demonstrated some power saving capabilities of their approach which may be fine-tuned by selecting a proper version of the algorithm. However, some trade-offs between efficiency and quality of service (e.g. longer delay) may be observed. The path selection algorithms draw heavily on the fat-tree structure of the network. In fact, it is the adoption of a regular topology that allows to construct a simple but effective heuristic.

6. Conclusions

This survey shows that many energy aware algorithms have been worked out for particular purposes: controlling utilization of processors by choosing the operating frequency, managing networks by routing and traffic engineering methods, allocating resources of clusters and grids. The need for this mechanism is obvious: energy should be used carefully, as it is one of the major components of costs involved, and, even more importantly, it is converted into harmful heat. The environmental factors should also be considered, as we all should take care to limit emissions. The last issue may be to some extent mitigated by the application of heat recuperation systems (relied upon, for instance, to heat offices) or by incorporating green energy sources. In order to take advantage of all these particular solutions, an integrated hierarchical control system is needed. As presented in Section 5, designing such a system is possible, as most necessary interfaces (to control the network and the computing nodes, to provide power and utilization measurements, etc.) have been already standardized and implemented.

Acknowledgements

This research was supported by the National Science Centre (NCN) under the grant no. 2015/17/B/ST6/01885.

References

[1] S. Nedevschi, L. Popa, G. Iannacone, D. Wetherall, and S. Ratnasamy, "Reducing network energy consumption via sleeping and rate adaptation", in *Proc. 5th USENIX Symp. on Netw. Syst. Design and Implement. NSDI 2008*, San Francisco, CA, USA, 2008, pp. 323–336 [Online]. Available: https://www.usenix.org/legacy/events/nsdi08/tech/full_papers/nedevschi/nedevschi.pdf

[2] J. Chabarek *et al.*, "Power awareness in network design and routing", in *Proc. 27th Conf. on Comp. Commun. INFOCOM 2008*, Phoenix, AZ, USA, 2008, pp. 457–465 (doi: 10.1109/INFOCOM.2008.93).

[3] A. Shehabi *et al.*, "United States data center energy usage report", 06/2016, 2016 [Online]. Available: https://eta.lbl.gov/publications/united-states-data-center-energy

[4] R. Bolla, R. Bruschi, A. Carrega, and F. Davoli, "Theoretical and technological limitations of power scaling in network devices", in *Proc. Australasian Telecommun. Netw. and Appl. Conf. ATNAC 2010*, Auckland, New Zealand, 2010, pp. 37–42 (doi: 10.1109/ATNAC.2010.5680253).

[5] "Advanced Configuration and Power Interface Specification, Revision 5.0", Hewlett-Packard, Intel, Microsoft, Phoenix Technologies, and Toshiba, 2011 [Online]. Available: http://www.acpi.info/DOWNLOADS/ACPIspec50.pdf

[6] "IEEE 802.3az Energy Efficient Ethernet Task Force", IEEE, 2012 [Online]. Available: http://grouper.ieee.org/groups/802/3/az/public/index.html

[7] M. Karpowicz, "Energy-efficient CPU frequency control for the Linux system", *Concur. and Comput.: Pract. and Exper.*, vol. 28, no. 2, pp. 420–437, 2015 (doi: 10.1002/cpe.3476).

[8] M. Karpowicz, P. Arabas, and E. Niewiadomska-Szynkiewicz, "Design and implementation of energy-aware application-specific CPU frequency governors for the heterogeneous distributed computing systems", *Future Gener. Comp. Syst.*, vol. 78, pp. 302–315, 2018 (doi: 10.1016/j.future.2016.05.011).

[9] "Power saving features in Mellanox products", Melanox Technologies, 2013 [Online]. Available: http://www.mellanox.com/related-docs/whitepapers/WP_ECONET.pdf

[10] "Intel 64 and IA-32 Architectures Software Developer's Manual", 2015 [Online]. Available: http://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-manual-325462.pdf

[11] S. Terpstra, H. Jagode, H. You, and J. Dongarra, "Collecting performance data with PAPI-C", in *Tools for High Performance Computing 2009. Proceedings of the 3rd International Workshop on Parallel Tools for High Performance Computing, September 2009, ZIH, Dresden*, M. S. Müller, M. M. Resch, A. Schulz, and W. E. Nagel, Eds. Berlin, Heidelberg: Springer, 2010, pp. 157–173.

[12] "IPMI – Intelligent Platform Management Interface Specification, Second Generation", Intel, Hewlett-Packard, NEC, and Dell, 2015 [Online]. Available: https://www.intel.com/content/www/us/en/servers/ipmi/ipmi-intelligent-platform-mgt-interface-spec-2nd-gen-v2-0-spec-update.html

[13] R. Bolla *et al.*, "The green abstraction layer: A standard power-management interface for next-generation network devices", *IEEE Internet Comput.*, vol. 17, no. 2, pp. 82–86, 2013 (doi: 10.1109/MIC.2013.39).

[14] R. Bolla *et al.*, "Large-scale validation and benchmarking of a network of power-conservative systems using ETSI's green abstraction layer", *Trans. on Emerg. Telecommun. Technol.*, vol. 27, no. 3, pp. 451–468, 2016 (doi: 10.1002/ett.3006).

[15] V. Pallipadi and A. Starikovskiy, "The ondemand governor: past, present and future", in *Proc. Linux Symp.*, Ottawa, Ontario, Canada, 2006, vol. 2, pp. 223–238 [Online]. Available: https://www.kernel.org/doc/ols/2006/ols2006v2-pages-223-238.pdf

[16] K. Accardi, "Balancing power and performance in the Linux kernel", 2015 [Online]. Available: https://events.static.linuxfound.org/sites/events/files/slides/LinuxConEurope_2015.pdf

[17] P. Arabas and M. Karpowicz, "Server power consumption: measurements and modeling with measurements", in *Challenges in Automation, Robotics and Measurement Techniques. Proceedings of AUTOMATION-2016, March 2-4, 2016, Warsaw, Poland*, R. Szewczyk, C. Zieliński, and M. Kaliczyńska, Eds. Springer, 2016, pp. 233–244 (ISBN 9783319293578).

[18] J. Gong and Ch. Xu, "A gray-box feedback control approach for system-level peak power management", in *Proc. 39th Int. Conf. on Parallel Process. ICPP-2010*, San Diego, CA, USA, 2010, pp. 555–564 (doi: 10.1109/ICPP.2010.63).

[19] R. Bolla, R. Bruschi, and A. Ranieri, "Green support for PC-based software router: Performance evaluation and modeling", in *IEEE Int. Conf. on Commun.*, Dresden, Germany, 2009, pp. 1–6 (doi: 10.1109/ICC.2009.5199050).

[20] R. Bolla and R. Bruschi, "Energy-aware load balancing for parallel packet processing engines", in *Online Conf. on Green Commun. GreenCom 2011*, New York, NY, USA, 2011, pp. 105–112 (doi: 10.1109/GreenCom.2011.6082516).

[21] M. Benito, E. Vallejo, and R. Beivide, "On the use of commodity Ethernet technology in exascale HPC systems", in *Proc. IEEE 22nd Int. Conf. on High Perform. Comput. HiPC 2015*, Bangalore, India, 2015, pp. 254–263 (doi: 10.1109/HiPC.2015.32).

[22] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture", in *Proc. ACM SIGCOMM 2008 Conf. on Data Commun.*, Seattle, WA, USA, 2008, pp. 63–74 (doi: 10.1145/1402958.1402967).

[23] "Cisco Data Center Infrastructure 2.5 Design Guide", Cisco Systems, Inc., 2011 [Online]. Available: https://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Data_Center/DC_Infra2_5/DCI_SRND_2_5a_book.html

[24] C. Clos, "A study of non-blocking switching networks", *The Bell Syst. Tech. J.*, vol. 32, no. 2, pp. 406–424, 1953 (doi: 10.1002/j.1538-7305.1953.tb01433.x).

[25] Ch. E. Leiserson, "Fat-trees: Universal networks for hardware-efficient supercomputing", *IEEE Trans. Comput.*, vol. 34, no. 10, pp. 892–901, 1985 (doi: 10.1109/TC.1985.6312192).

[26] "Introduction to cloud design four design principals for IaaS", Melanox Technologies, 2012 [Online]. Available: http://www.mellanox.com/pdf/whitepapers/WP_Cloud_Computing.pdf

[27] Y. Xia and T. S. E. Ng, "Flat-tree: A convertible data center network architecture from CLOS to random graph", in *Proc. 15th ACM Worksh. on Hot Topics in Netw. HotNets'16*, Atlanta, GA, USA, 2016, pp. 71–77 (doi: 10.1145/3005745.3005763).

[28] J. Kim, W. J. Dally, and D. Abts, "Flattened butterfly: A cost-efficient topology for high-radix networks", in *Proc. of the 34th Ann. Int. Symp. on Comp. Architec. ISCA'07*, San Diego, CA, USA, 2007, pp. 126–137 (doi: 10.1145/1273440.1250679).

[29] A. Shpiner, Z. Haramaty, S. Eliad, V. Zdornov, B. Gafni, and E. Zahavi, “Dragonfly+: Low cost topology for scaling datacenters”, in *Proc. IEEE 3rd Int. Worksh. on High-Perform. Interconnec. Netw. in the Exascale and Big-Data Era HiPINEB 2017*, 2017, Austin, TX, USA, pp. 1–8 (doi: 10.1109/HiPINEB.2017.11).

[30] M. Pióro, M. Mysłek, A. Juttner, J. Harmatos, and A. Szentesi, “Topological Design of MPLS Networks”, in *Proc. Global Telecommun. Conf. GLOBECOM'2001*, San Antonio, TX, USA, 2001 (doi: 10.1109/GLOCOM.2001.965071).

[31] F. Bianco, G. Cucchietti, and G. Griffa, “Energy consumption trends in the next generation access network – a telco perspective”, in *Proc. 29th Inter. Telecommun. Energy Conf. INTELEC 2007)*, Rome, Italy, 2007, pp. 737–742 (doi: 10.1109/INTLEC.2007.4448879).

[32] S. N. Roy, “Energy logic: a road map to reducing energy consumption in telecommunications networks”, in *Proc. 30th Int. Telecommun. Energy Conf. INTELEC 2008*, San Diego, CA, USA, 2008 (doi: 10.1109/INTLEC.2008.4664025).

[33] A. Karbowski and P. Jaskóła, “Two approaches to dynamic power management in energy-aware computer networks – methodological considerations”, in *Proc. Federated Conf. on Comp. Sci. and Inform. Syst. FedCSIS*, Łódź, Poland, 2015 (doi: 10.15439/2015F228).

[34] P. Jaskóła and K. Malinowski, “Two methods of optimal bandwidth allocation in TCP/IP networks with QoS differentiation”, in *Proc. Symp. on Perform. Eva. of Comp. and Telecommun. Systems SPECTS'04*, San Jose, CA, USA, 2004, pp. 373–378, 2004.

[35] A. Kozakiewicz and K. Malinowski, “Network traffic routing using effective bandwidth theory”, *Eur. Trans. on Telecommun.*, vol. 20, no. 7, pp. 660–667, 2009 (doi: 10.1002/ett.1383).

[36] M. Kamola and P. Arabas, “Dynamically established transmission paths in the future internet – proposal of a framework”, *Bull. of the Polish Acad. of Sciences: Tech. Sciences*, vol. 59, no. 3, pp. 357–366, 2011 (doi: 10.2478/v10175-011-0043-9).

[37] K. Malinowski, E. Niewiadomska-Szynkiewicz, and P. Jaskóła, “Price method and network congestion control”, *J. of Telecommun. and Inform. Technol.*, no. 2, pp. 73–77, 2010.

[38] L. Chiaraviglio, M. Mellia, and F. Neri, “Minimizing ISP network energy cost: formulation and solutions”, *IEEE/ACM Trans. on Netw.*, vol. 20, no. 2, pp. 463–476, 2011 (doi: 10.1109/TNET.2011.2161487).

[39] E. Niewiadomska-Szynkiewicz, A. Sikora, P. Arabas, and J. Kołodziej, “Control framework for high performance energy aware backbone network”, in *Proc. of 26th Eur. Conf. on Modell. and Simul. ECMS 2012*, Koblenz, Germany, 2012, pp. 490–496 (doi: 10.7148/2012-0490-0496).

[40] J. Restrepo, C. Gruber, and C. Machuca, “Energy profile aware routing”, in *Proc. IEEE Int. Conf. on Commun. Workshops ICC 2009*, Dresden, Germany, 2009, pp. 1–5, 2009 (doi: 10.1109/ICCW.2009.5208041).

[41] W. Fisher, M. Suchara, and J. Rexford, “Greening backbone networks: reducing energy consumption by shutting off cables in bundled links”, in *Proc. 1st ACM SIGCOMM Worksh. on Green Networking Green Networking'10*, New Delhi, India, 2010, pp. 29–34 (doi: 10.1145/1851290.1851297).

[42] F. Idzikowski, S. Orlowski, Ch. Raack, H. Rasner, and A. Wolisz, “Saving energy in IP-over-WDM networks by switching off line cards in low-demand scenarios”, in *Proc. 14th Conf. on Opt. Netw. Design and Model. ONDM'10*, Kyoto, Japan, 2010 (doi: 10.1109/ONDM.2010.5431569).

[43] N. Vasić and D. Kostić, “Energy-aware traffic engineering”, in *Proc. 1st Int. Conf. on Energy-Efficient Comput. and Netw. E-ENERGY 2010*, Passau, Germany, 2010 (doi: 10.1145/1791314.1791341).

[44] P. Arabas, K. Malinowski, and A. Sikora, “On formulation of a network energy saving optimization problem”, in *Proc. of 4th Int. Conf. on Commun. and Electron. ICCE 2012*, Hue, Vietnam, 2012, pp. 122–129 (doi: 10.1109/CCE.2012.6315903).

[45] E. Niewiadomska-Szynkiewicz *et al.*, “Network-wide power management in computer networks”, in *Proc. 22nd ITC Special. Seminar on Energy Effic. and Green Netw. SSEEGN 2013*, Riccarton, New Zealand, 2013, pp. 25–30 (doi: 10.1109/SSEEGN.2013.6705398).

[46] E. Niewiadomska-Szynkiewicz *et al.*, “Dynamic power management in energy-aware computer networks and data intensive systems”, *Future Gener. Comp. Syst.*, vol. 37, pp. 284–296, 2014 (doi: 10.1016/j.future.2013.10.002).

[47] M. Karpowicz, P. Arabas, and E. Niewiadomska-Szynkiewicz, “Energy-aware multilevel control system for a network of Linux software routers: design and implementation”, *IEEE Syst. J.*, vol. 12, no. 1, pp. 571–582, 2018 (doi: 10.1109/JSYST.2015.2489244).

[48] L. Chiaraviglio, M. Mellia, and F. Neri, “Energy-aware backbone networks: a case study”, in *Proc. IEEE Int. Conf. on Commun. Worksh. ICC2009*, Dresden, Germany, 2009, pp. 1–5 (doi: 10.1109/ICCW.2009.5208038).

[49] M. Zhang, Ch. Yi, B. Liu, and B. Zhang, “GreenTE: power-aware traffic engineering”, in *Proc. IEEE Inter. Conf. on Netw. Protoc. ICNP'2010*, Kyoto, Japan, 2010 (doi: 10.1109/ICNP.2010.5762751).

[50] G. Shen and R. S. Tucker, “Energy-minimized desig for IP over WDM networks”, *J. of Optical Commun. and Netw.*, vol. 1, no. 1, pp. 176–186, 2009 (doi: 10.1364/JOCN.1.000176).

[51] R. G. Garroppo, S. Giordano, G. Nencioni, and M. G. Scutella, “Power-aware routing and network design with bundled links: Solutions and analysis”, *J. of Comp. Netw. and Commun.*, vol. 2013, Article ID 154953, 2013 (doi: 10.1155/2013/154953).

[52] Y. Zhang and N. Ansari, “Hero: Hierarchical energy optimization for data center networks”, *IEEE Systems J.*, vol. 9, no. 2, pp. 406–415, 2015 (doi: 10.1109/JSYST.2013.2285606).

[53] B. Heller *et al.*, “ElasticTree: Saving energy in data center networks”, in *Proc. 7th USENIX Conf. on Network. Syst. Design and Implemen. NSDI'10*, San Jose, CA, USA, 2010, p. 17.

[54] X. Wang, X. Wang, K. Zheng, Y. Yao, and Q. Cao, “Correlation-aware traffic consolidation for power optimization of data center networks”, *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 4, pp. 992–1006, 2016 (doi: 10.1109/TPDS.2015.2421492).

[55] Z. Guo, Sh. Hui, Y. Xu, and H. J. Chao, “Dynamic flow scheduling for power-efficient data center networks”, in *Proc. IEEE/ACM 24th Int. Symp. on Quality of Service IWQoS 2016*, Beijing, China, 2016 (doi: 10.1109/IWQoS.2016.7590399).

[56] K. Zheng, X. Wang, and X. Wang, “PowerFCT: Power optimization of data center network with flow completion time constraints”, in *Proc. IEEE Int. Paral. and Distrib. Proces. Symp.*, Hyderabad, India, 2015 (doi: 10.1109/IPDPS.2015.22).

[57] A. Cianfrani, V. Eramo, M. Listani, M. Marazza, and E. Vittorini, “An energy saving routing algorithm for a green OSPF protocol”, in *Proc. IEEE Conf. on Comp. Commun. INFOCOM 2010*, San Diego, CA, USA, 2010, pp. 1 – 5 (doi: 10.1109/INFCOMW.2010.5466646).

[58] A. P. Bianzino, L. Chiaraviglio, and M. Mellia, GRiDA: a green distributed algorithm for backbone networks”, in *Online Conf. on Green Commun. GreenCom 2011*, New York, NY, USA, 2011, pp. 113–119 (doi: 10.1109/GreenCom.2011.6082517).

[59] F. Cuomo, A. Abbagnale, A. Cianfrani, and M. Polverini, “Keeping the connectivity and saving the energy in the Internet”, in *Proc. IEEE INFOCOM 2011 Workshop on Green Communications and Networking*, Shanghai, China, 2011, pp. 319–324 (doi: 10.1109/INFCOMW.2011.5928831).

[60] M. Kamola and P. Arabas, “Shortest path green routing and the importance of traffic matrix knowledge”, in *Proc. 24th Tyrrhenian Int. Worksh. on Digit. Commun. – Green ICT (TIWDC)*, Genoa, Italy, 2013 (doi: 10.1109/TIWDC.2013.6664215).

[61] H. Huin *et al.*, “Bringing energy aware routing closer to reality with SDN hybrid networks”, in *Proc. Global Commun. Conf. GLOBECOM'2017*, Singapore, 2017, pp. 1101–1107 (doi: 10.1109/GLOCOM.2017.8254456).

[62] A. Verma, P. Ahuja, and A. Neogi, “pMapper: Power and migration cost aware application placement in virtualized systems”, in *Middleware 2008. ACM/IFIP/USENIX 9th International Middleware Conference Leuven, Belgium, December 1-5, 2008 Proceedings*, V. Issarny and R. Schantz, Eds. LNCS, vol. 5346, pp. 243–264. Springer, 2008 (doi: 10.1007/978-3-540-89856-6_13).

[63] A. Hameed *et al.*, "A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems", *Computing*, vol. 98, no. 7, pp. 751–774, 2016 (doi: 10.1007/s00607-014-0407-8).

[64] Ch. Ghribi and D. Zeghlache, "Exact and heuristic graph-coloring for energy efficient advance cloud resource reservation", in *Proc. 7th IEEE Int. Conf. on Cloud Comput. CLOUD 2014*, Anchorage, AK, USA, 2014, pp. 112–119 (doi: 10.1109/CLOUD.2014.25).

[65] A. M. Al-Qawasmeh, S. Pasricha, A. A. Maciejewski, and H. J. Siegel, "Power and thermal-aware workload allocation in heterogeneous data centers", *IEEE Trans. on Comp.*, vol. 64, no. 2, pp. 477–491, 2015 (doi: 10.1109/TC.2013.116).

[66] K. Gupta and V. Katiyar, "Energy aware virtual machine migration techniques for cloud environment", *Int. J. of Comp. Appl.*, vol. 141, no. 2, pp. 11–16, 2016 (doi: 10.5120/ijca2016909551).

[67] V. Armant, M. De Cauwer, K. N. Brown, and B. O'Sullivan, "Semi-online task assignment policies for workload consolidation in cloud computing systems", *Future Gener. Comp. Syst.*, vol. 82, pp. 89–103, 2018 (doi: 10.1016/j.future.2017.12.035).

[68] M. Zhang, S. Wang, G. Yuan, Y. Li, and Z. Qian, "Energy-efficient real-time task allocation in a data center", in *Proc. IEEE Int. Conf. on Internet of Things and Green Comput. and Commun. and Cyber, Phys. and Soc. Comput. and Smart Data iThings-GreenCom-CPSCom-SmartData 2016*, Chengdu, China, 2016, pp. 680–687 (doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2016.147).

[69] A. Gandhi, Yuan Ch., D. Gmach, M. Arlitt, and M. Marwah, "Minimizing data center SLA violations and power consumption via hybrid resource provisioning", in *Proc. Int. Green Comput. Conf. and Worksh. IGCC'11*, Orlando, FL, USA, 2011, pp. 1–8 (doi: 10.1109/IGCC.2011.6008611).

[70] S. Hasan and E. Huh, "Heuristic based energy-aware resource allocation by dynamic consolidation of virtual machines in cloud data center", *TIIS*, vol. 7, no. 8, pp. 1825–1842, 2013 (doi: 10.3837/tiis.2013.08.005).

[71] Y. Shao, C. Li, W. Dong, and Y. Liu, "Energy-aware dynamic resource allocation on hadoop YARN cluster", in *Proc. 18th Int. Conf. on High Perform. Comput. and Commun., 14th Int. Conf. on Smart City, 2nd Int. Conf. on Data Sci. and Syst. HPCC/SmartCity/DSS 2016*, Sydney, NSW, Australia, 2016, pp. 364–371 (doi: 10.1109/HPCC-SmartCity-DSS.2016.0059).

[72] E. Niewiadomska-Szynkiewicz and P. Arabas, "Resource management system for HPC computing", in *Automation 2018. Advances in Automation, Robotics and Measurement Techniques, Warsaw, Poland*, R. Szewczyk, C. Zieliński, and M. Kaliczyńska, Eds. *Advances in Intelligent Systems and Computing*, vol. 743, pp. 52–61. Springer, 2018 (doi: 10.1007/978-3-319-77179-3_5).

[73] M. Ray, S. Sondur, J. Biswas, A. Pal, and K. Kant, "Opportunistic power savings with coordinated control in data center networks", in *Proc. 19th ICDCN Int. Conf. on Distrib. Comput. and Netw.*, Varanasi, India, 2018, Article no. 48, pp. 48:1–48:10 (doi: 10.1145/3154273.3154328).

Piotr Arabas received his Ph.D. in Computer Science from the Warsaw University of Technology, Poland, in 2004. Currently he is an Assistant Professor at the Institute of Control and Computation Engineering at the Warsaw University of Technology. He has been with the Research and Academic Computer Network (NASK) since 2002. His research interest focuses on energy-efficient control of computer systems and networks, analysis of social networks, predictive control and hierarchical systems.

https://orcid.org/0000-0002-4173-3249
E-mail: parabas@ia.pw.edu.pl
Institute of Control and Computation Engineering
Warsaw University of Technology
Nowowiejska 15/19
00-665 Warsaw, Poland