Controlled Direct Liquid Cooling of Data Servers

Riccardo Lucchese, Damiano Varagnolo, Andreas Johansson

***Abstract*—We formulate a modelling and control framework aimed at direct liquid cooling of data servers. In our application scenario, the server's heat-load is rejected into a liquid cooling circuit that extends to the individual chips. We start with a comprehensive discussion of our modelling derivations and then we show how to manipulate the coolant's flow while 1) regulating the temperatures of any self-heating components within a safe operational envelope, 2) minimizing the coolant supply cost and 3) increasing the server outflow temperature (a key performance objective towards heat reuse systems). We confirm experimentally the benefits of the proposed controlled cooling strategy over several realistic scenarios corresponding to different inlet coolant temperatures and computational loads.**

***Index Terms*—Controlled Liquid Cooling, Thermal Networks, Thermal Management, Heat Reuse, Direct Liquid Cooling, Data Servers.**

I. Introduction

Modern data centers are energy intensive processes that can accommodate millions of computing cores and hundreds of thousands of data servers. While the energy efficiency of computing units has doubled every 18 months (a fact known as Koomey's law [1]), deployments have witnessed an increase in both total power loads and power densities (up to 120MW and 30kW/m^2 respectively). This state of practice presents a number of technological challenges concerning the design of the electronic equipment, its packaging and thermal management [2], [3]. In particular, the heat loads have approached the limits of traditional air-cooling solutions, exacerbating the power consumption and reliability issues. Indeed, air-cooling operates with high temperature gradients between air and the active components. This produces two main effects: first, the necessity to pre-cool the air, decreasing the overall energy efficiency; second, it results into low exergetic gains at the server outlet which hinder the repurposing of waste heat [4].

With acquisition costs being overwhelmed by the running costs, data center operators seek for more cost-effective cooling solutions [5]. A continuous development is thus focused on both making a more efficient use of air [6] and heat reuse systems [7], [8], [9], [10]. In this latter respect, a compelling alternative is to deploy liquid cooling loops that extend to the individual electronic chips ("direct liquid cooling"). Liquid coolants exhibit both higher thermal capacitance and lower thermal resistance then air. This allows compact designs that match higher power densities and result in generally smaller rates of exergy destruction. Furthermore, the cooling loops can run hotter, opening to both free-cooling and heat reuse implementations. Examples of the latter scenarios are supplying the basic heat load needs to indoor complexes, greenhouses [11], district heating [7], desalination and refrigeration processes [4], preheating of boiler feed water in power plants [12]. Remarkably, using hot water coolant enables heat recovery systems with efficiencies (up to 85 percent) which are not possible in air-cooled settings [4].

Air-cooled server enclosures have been subject to a substantial modelling effort with state of the art control strategies being based on economic polynomial optimization problems, see [13], [14], [15], [16], and references therein. However, to our current best knowledge, direct "on-chip" liquid cooling solutions have not been thoroughly investigated from a control perspective. In particular, there is a lack of studies evaluating the heat quality benefits of *dynamical* provisioning in this setting. A large part of the existing body of works focuses on energy efficiency at the data center level. For instance, [7], [17], [18], [19] pursue to quantify the prospective efficiency gains of direct liquid cooling over other liquid-cooling and air-cooling technologies. In hybrid air-and-liquid cooling problems, a portion of the compute capacity is upgraded with direct liquid cooling to address computer room hot-spots. Reducing the overall cooling cost corresponds then to 1) an off-line selection of the machines to be retrofitted and 2) the on-line optimal job allocation over the mix of air and liquid cooled platforms [20], [21]. We stress that, in the above liquid cooling studies, the coolant supply is matched to the peak heat load and thus not adaptive. This work, instead, sets out to assess specifically the performance of dynamical coolant provisioning whilst keeping a focus on heat recovery. To this aim, we first devise a *thermal networks* framework to describe the temperature dynamics of direct liquid cooled electronics. Then, on top of this modelling, we design a feedback provisioning law in the form of an optimizing supervisor.

We notice that temperature models that are similar in spirit to ours have been proposed at the chip level. [22] uses resistor-capacitor networks to capture heat conduction, generation and storage within a single chip. In addition, convective liquid cooling has been investigated for 3D stacked architectures where using air as the advection

Submitted on the 17th July 2018. This work is partly funded by the Celtic Plus project SENDATE-Extend (C2015/3-3). We moreover acknowledge support from the Swedish research council Norrbottens Forskinsråd project DISTRACT to acquire the liquid cooling parts. All the authors are with the Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, Luleå, Sweden. E-mails: `riccardo.lucchese@ltu.se, damiano.varagnolo@ltu.se, andreas.johansson@ltu.se`.

medium becomes inadequate due to the manifold increase in the power consumption [23], [24], [25]. Notice, however, that these approaches focus on a single package and disregard any dynamical interactions with other passive and active components. Network-oriented heat transfer models have moreover found application in other settings. For instance, describing how heating or cooling a room influences the temperatures of neighboring zones of a building is instrumental to derive holistic Heating, Ventilation, and Air Conditioning (HVAC) control strategies. In the this line of works, however, there is a lesser focus on the advection phenomena and practical implementations are limited to linear and bilinear models [26], [27]. Heat Exchange Networks (HENs) have been studied within the context of heat recovery systems in chemical processes [28]. This modelling approach is tailored to investigating the structural properties of the recovery systems (such as the energetic and exergetic yields) given their topology. The main focus is then on the *synthesis* problem of maximizing the HEN's performance subject to constraints on the heat sources and sinks and while minimizing the number of heat exchanger units [29]. At the same time, these settings marginalize the importance of deriving accurate temperature dynamics of other nodes in the network.

A. Statement of contributions

We propose to model the temperature dynamics of a liquid cooled data server using *thermal networks*: a graph theoretic formalism that concisely encodes which devices interact by exchanging heat and by which mechanism. The network's nodes correspond to thermal entities that can locally store, inject or transfer heat to and from any neighboring nodes. Our treatment is tailored to data servers since this is where the bulk of a data center heat-load is produced. At this level, we contribute a strategy to model the transient temperatures of characteristic parts such as the liquid cooled heat exchangers.

We then consider that implementing cost-effective heat reuse systems depends on the data center's ability to systematically act as a stable source of high quality heat, that is, to sustain outlet flows with high temperatures [18], [12]. Building on the thermal network model, we thus propose a dynamic provisioning strategy that achieves lower coolant supply costs and higher quality heat harvests. Finally we assess the performance benefits of dynamical versus static provisioning laws using a liquid cooled Open Compute Windmill V2 server as an experimental test bed.

Remark 1: We stress our present focus on the server level. In this work we thus neglect to treat explicitly other technological components that are necessary to operate the cooling circuits such as purifiers, pre-heaters and the Coolant Distribution Units (CDUs).

B. Organization of this manuscript

Section II introduces our thermal modelling framework in terms of networks of components with local thermal dynamics and heat exchange interactions described by graph overlays. Section III specializes our thermal model into a library of reusable nodes aimed at the server level. Section IV introduces a model-based strategy for regulating the volumetric flow rate of the liquid coolant dynamically. Section V accomplishes a validation of our modelling choices using a water cooled test server; then it characterizes the heat reuse performance of our dynamical control strategy on the same platform. Finally, Section VI collects concluding remarks and future directions.

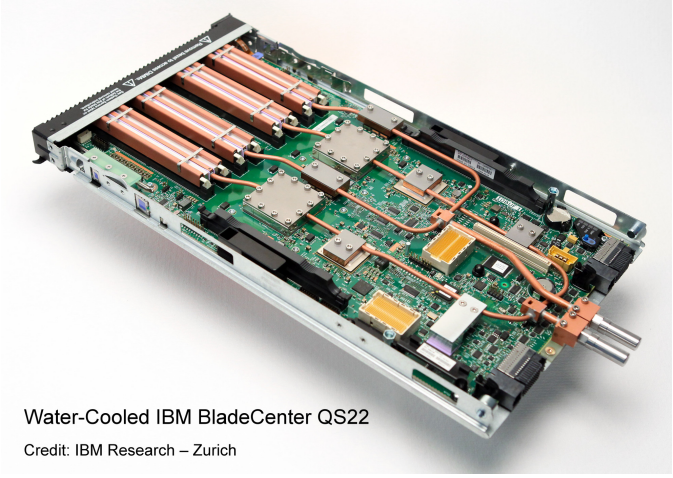

Figure 1: The IBM BladeCenter® QS22, a water cooled platform investigated in [4]. A symmetric cooling circuit extends to all the electronic parts with a power consumption above 3 Watts. Reproduced with permission.

II. A thermal modelling framework for controlled liquid cooling

We model a data server as a network of interacting components endowed with lumped thermal properties and local temperature dynamics. To this aim we introduce a graph-theoretic formalism where $\mathcal{N} = \{1, 2, \dots, n\}$ denotes the set of participating nodes/components ($n \doteq |\mathcal{N}|$ being the network size) and where the topology of the heat transfers is captured by two directed graph overlays $\mathcal{E}_{cd}, \mathcal{E}_{cv} \subset \mathcal{N} \times \mathcal{N}$: $\mathcal{E}_{cd}$ ($\mathcal{E}_{cv}$ respectively) encodes which components interact by exchanging heat through the mode of conduction (convection). Notice that the semantic meaning of a link in the two overlays differs (see the detailed discussion later in Sections II-A and II-B). A link $(j, h) \in \mathcal{E}_{cd}$ indicates a directed *heat flow* from node j to node h; a link $(j, h) \in \mathcal{E}_{cv}$ establishes instead a liquid cooling interconnection between node j and node h, indicating the flow of *both heat and mass* between the two nodes. For the generic graph overlay $\mathcal{E}$ on $\mathcal{N} \times \mathcal{N}$, we define the in-neighborhood and out-neighborhood sets of node $j \in \mathcal{N}$ respectively as

$$\delta^{\mathcal{E}}_{-}(j) \doteq \{h : (h, j) \in \mathcal{E}\}, \quad \delta^{\mathcal{E}}_{+}(j) \doteq \{h : (j, h) \in \mathcal{E}\}.$$

Example 1: Figure 1 shows a direct liquid cooled server in which a cooling network harvests the heat-load from

several on-board components using top-mounted heat exchangers. The generation of thermal energy is predominantly confined to small portions of space and the heat exchange phenomena are localized in the same volumes.

A. The heat conduction overlay

The heat transfer rate among a pair of neighbors (j,h) in $\mathcal{E}_{cd}$ is approximated given the thermal state of the pair and the material/geometrical properties of the physical interaction medium.

Example 2: Consider the simple case of a thermal network with two electrical components connected through a solid thermal bridge. Let j,h be two generic and fixed indices and define $\mathcal{N} = \{j,h\}$. The thermal bridge induces two links in the heat conduction overlay, that is, $\mathcal{E}_{cd} = \{(j,h),(h,j)\}$. Let $t \mapsto x_j^c(t)$ and $t \mapsto x_h^c(t)$ be the continuous time trajectories of the proxy temperatures for the first and the second component respectively. We model the conduction heat flow from h to j as a one-dimensional phenomenon and describe the corresponding heat rate using Fourier's law:

$$q_j^{cd}(t) = -k_{hj}\big(x_j^c(t) - x_h^c(t)\big). \tag{1}$$

The constant $k_{hj} \in \mathbb{R}_{>0}$ above is then the lumped, overall, thermal conductivity of the bridge[1].

For a generic thermal network with size $n = |\mathcal{N}|$ we collect the thermal conductivity parameters in the matrix $K \in \mathbb{R}_{\geq 0}^{n\times n}$ and set $k_{jh} = 0$ whenever $(j,h) \notin \mathcal{E}_{cd}$. The rate of heat transfer due to conduction at the generic node $j \in \mathcal{N}$ can then be written as

$$q_j^{cd}(t) = -\sum_{h\in\delta_-^{\mathcal{E}_{cd}}(j)} k_{hj}\big(x_j^c(t) - x_h^c(t)\big). \tag{2}$$

We stress that the conduction overlay $\mathcal{E}_{cd}$ is directed. This asymmetry reflects situations in which a component rejects heat to an *environmental node*, that is, a specific node type describing heat reservoirs and thus characterized by a constant temperature. In particular, for an environmental node $j \in \mathcal{N}$ we have $\delta_-^{\mathcal{E}_{cd}}(j) = \emptyset$ and thus $q_j^{cd}(t) = 0$ at all times t. Finally, not all node types need to participate in the conduction overlay: for instance, *supply* and *collector* nodes (discussed in the next section) model the entry and exit points for the coolant in the liquid cooling circuit and have empty in- and out-neighborhoods in $\mathcal{E}_{cd}$.

B. The heat convection overlay

The heat convection overlay serves as a means to describe the flow of the coolant within the network: a directed link $(j,h) \in \mathcal{E}_{cv}$ models an actual interconnection (or pipe) where the liquid coolant can flow. We let $\varphi : t\times\mathcal{N}\times\mathcal{N} \to \mathbb{R}$ be the continuous time trajectory of the volumetric coolant flow rates over $\mathcal{E}_{cv}$: $\varphi(t,j,h)$ is then the (volumetric) flow rate of liquid coolant through link (j,h) at time t if $(j,h) \in \mathcal{E}_{cv}$, and is equal to zero otherwise. In the following, for the sake of a compact notation, we identify $\varphi_{jh}(t) \doteq \varphi(t,j,h)$.

The coolant enters the network at the supply nodes $\mathcal{S} \subset \mathcal{N}$ and exits it at the collector nodes $\mathcal{C} \subset \mathcal{N}$. We consider networks with a generic number $m \geq 1$ of supply nodes

$$\mathcal{S} = \{s_1,\dots,s_m\}, \quad |\mathcal{S}| = m \tag{3}$$

and a generic number $p \geq 1$ of collector nodes

$$\mathcal{C} = \{c_1,\dots,c_p\}, \quad |\mathcal{C}| = p. \tag{4}$$

Notice that supply nodes cannot act as collector nodes and vice versa: $\mathcal{S}\cap\mathcal{C} = \emptyset$. Moreover, supply nodes do not have inflows and, specularly, collector nodes have no outflows:

$$\delta_-^{\mathcal{E}_{cv}}(j) = \emptyset \quad \forall j \in \mathcal{S}, \quad \delta_+^{\mathcal{E}_{cv}}(j) = \emptyset \quad \forall j \in \mathcal{C}. \tag{5}$$

Instead, each supply node $j \in \mathcal{S}$ is univocally associated to a control variable $u_{I_s(j)}$ through the bijection $I_s : \mathcal{S} \mapsto \{1,\dots,m\}$. The order on $\mathcal{N}$ induces an order on the control variables such that they can be orderly stacked to form the control vector

$$\boldsymbol{u} \doteq \begin{bmatrix} u_{I_s(s_1)} & \dots & u_{I_s(s_m)} \end{bmatrix}^T \in \mathbb{R}_{\geq 0}^m. \tag{6}$$

The control input $t \mapsto \boldsymbol{u}(t)$ sets the supply flow rates to the liquid cooling network through

$$\sum_{h\in\delta_+^{\mathcal{E}_{cv}}(j)} \varphi_{jh}(t) = u_{I_s(j)}(t), \quad \forall j \in \mathcal{S}. \tag{7}$$

We assume that the liquid cooling circuit has no leaks and that the flow rates are conserved at all nodes except for the supply and collector nodes:

$$\sum_{h\in\delta_-^{\mathcal{E}_{cv}}(j)} \varphi_{hj}(t) = \sum_{h\in\delta_+^{\mathcal{E}_{cv}}(j)} \varphi_{jh}(t), \quad \forall j \in \mathcal{N}\setminus(\mathcal{S}\cup\mathcal{C}). \tag{8}$$

In light of (5) and (8) we define the total flow crossing the j-th node at time t as

$$f_j(t) \doteq \begin{cases} \displaystyle\sum_{h\in\delta_+^{\mathcal{E}_{cv}}(j)} \varphi_{jh}(t) & \text{if } j \in \mathcal{S} \\ \displaystyle\sum_{h\in\delta_-^{\mathcal{E}_{cv}}(j)} \varphi_{hj}(t) & \text{otherwise} \end{cases} \tag{9}$$

and define the network's flow vector by stacking the individual flows:

$$\boldsymbol{f}(t) \doteq \Big[f_1(t),\dots,f_n(t)\Big]^T. \tag{10}$$

We assume that the flow splitting ratios at nodes with multiple outflows are time-constant parameters of the physical system and independent from the flow rate. This independence from the working condition is motivated on a practical basis: a typical server platform accommodates the components (Central Processing Units (CPUs), Dual In-line Memory Modules (DIMMs), companion chips) in pairs leading to liquid cooling circuits with purposefully

[1] Notice that we let the term k_{hj} subsume also all the geometrical properties of the system and thus measure it in J/[s · K].

symmetric designs (see, for example, the platform in Figure 1). The ratios are formally encoded in the matrix $\Lambda \doteq (\lambda_{jh}) \in \mathbb{R}^{n\times n}_{\geq 0}$ defined through

$$\sum_{h\in\delta^{\mathcal{E}_{cv}}_{+}(j)} \lambda_{jh} = 1, \quad \varphi_{jh}(t) = \lambda_{jh} f_j(t), \quad \forall\, j \in \mathcal{N}. \tag{11}$$

The following result relates the control input in (6) and the flow vector in (10). For its proof (here omitted in the interest of space) we require two additional assumptions of both a practical and technical nature:

- $\mathcal{E}_{cv}$ induces a Directed Acyclic Graph (DAG). In particular, the DAG has no self-loops and admits a topological ordering;
- For each node $j \in \mathcal{N}$ such that $\delta^{\mathcal{E}_{cv}}_{-}(j) \neq \emptyset$ or $\delta^{\mathcal{E}_{cv}}_{+}(j) \neq \emptyset$ there exists a directed path over $\mathcal{E}_{cv}$ starting from j and reaching a collector node in $\mathcal{C}$. That is, the flow through each liquid cooled node must be able to reach a collector and exit the circuit.

Proposition 1: The instantaneous volumetric flow rate vector $\boldsymbol{f}(t)$ in (10) can be written explicitly as a linear function of the control vector $\boldsymbol{u}(t)$. In particular there exists a constant matrix $\Phi \in \mathbb{R}^{n\times m}$, function of the liquid cooling network topology and Λ, such that

$$\boldsymbol{f}(t) = \Phi \boldsymbol{u}(t). \tag{12}$$

Finally, we stress that not all nodes connected in the heat convection overlay will exchange heat with the liquid coolant (and thus influence its temperature); as discussed later in Section III-A, pure manifold nodes, devoid of local temperature dynamics, are used to model piping *joints* and *splitters* in the cooling circuit.

C. The nodes' thermal model

The temperature dynamics of the network are captured in a low order state space representation. We do not explicitly account for complex three dimensional geometries and non-heterogeneous thermal properties; instead, we consider lumped models that explain the average effect of the distinct heat contributions. To write the dynamics of the generic j-th node we first introduce the following temperature variables (see Figure 2):

- $x^i_j(t)$: the temperature of the coolant entering the node at time t (or *inflow temperature*);
- $x^c_j(t)$: the *local temperature* state of node j at time t (for instance, the average temperature of a CPU package);
- $x^o_j(t)$: the temperature of the coolant leaving the node at time t (or *outflow temperature*).

The flow of the coolant through the interconnections is assumed adiabatic and its temperature is thus determined only at the entry and exit points of each node, disregarding the explicit description of the in-transit thermal dynamics. The coolant flows into j, collects the heat produced within the node due to electrical dissipation, and exits at a higher temperature. During normal operation we have then $x^i_j(t) \leq x^o_j(t) \leq x^c_j(t)$ at all times.

Remark 2: We stress that thermal phenomena affecting the in-transit coolant can still be described in our framework by introducing opportune sub-networks that account for the local thermal inertiae and any parasitic resistances to the environment.

1) Modelling the inflow temperature $x^i_j(t)$: Recall that the flow rates at the supply nodes are manipulable variables set through (7). The liquid coolant enters then the thermal network at the generic supply node $s_i \in \mathcal{S}$ at rate $u_{I_s(s_i)}(t)$ and given temperature $x^i_{s_i}(t)$. The latter temperature should be understood as an exogenous input since in practice the Coolant Distribution Unit (CDU) acts as a heat reservoir with a large thermal capacitance and a slow varying temperature. Using again the bijection $I_s(\cdot)$ in (6) we can introduce the vector of input temperatures by orderly stacking the supply coolant temperature of each supply node

$$\boldsymbol{x}^i(t) \doteq \begin{bmatrix} x^i_{I_s(s_1)}(t) & \dots & x^i_{I_s(s_m)}(t) \end{bmatrix}^T \in \mathbb{R}^m_{\geq 0}. \tag{13}$$

Those nodes that have more than one inflow act as mixing manifolds and correspond to points in the cooling circuit where multiple interconnections are channeled into a single pipe. The temperature of the coolant flow crossing the j-th node is captured through the average

$$x^i_j(t) \doteq \frac{1}{f_j(t)} \sum_{h\in\delta^{\mathcal{E}_{cv}}_{-}(j)} \varphi_{hj}(t) x^o_h(t), \quad j \in \mathcal{N}\setminus\mathcal{S}. \tag{14}$$

The above relation (14) weights the instantaneous temperature contribution of each incoming flow by its flow rate. This corresponds to consider the conservation law (8) together with two additional assumptions:

i) the coolant is perfectly mixed at the manifolds;
ii) the heat energy of the coolant is conserved during the mixing.

We notice that *ii*) above is motivated by the low flow rates while *i*) is supported by practical considerations: mixing and heat-exchange sites do not coincide in the hardware, and this allows flows to be mixed before they enter the active to-be-cooled parts.

Remark 3: For the sake of simplicity and in the light of the experimental results of Section V-C, we disregard to account for transport delays in this work. While these phenomena are inherent in a liquid cooling setting, their effect is negligible at the typical lengths of the server level.

2) Dynamics of the local temperature $x^c_j(t)$: Tracking the temperatures of any self-heating components is central to our modelling effort. Indeed, the safe operation of the network depends on being able to regulate these temperatures below specified thresholds (this aspect is further discussed and formalized in Section IV). To this aim, the generic j-th node is also seen as a dynamical subsystem with temperature dynamics $\dot{x}^c_j(t)$ and a lumped heat capacity d_j. The dynamics takes the form

$$d_j \dot{x}^c_j(t) = q^{cd}_j(t) + q^{cv}_j(t) + p_j(t), \tag{15}$$

and builds on top of the following three contributions:

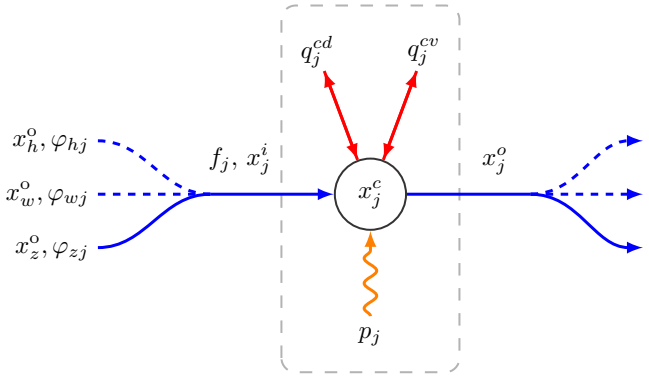

Figure 2: The temperature dynamics of the generic j-th node involves three heat transfer phenomena: *conduction* to and from neighboring nodes (q_j^{cd}), *convection* to and from the liquid coolant (q_j^{cv}) and *generation* due to the local conversion of electrical power (p_j). An energetic balance is used to estimate the inlet to outlet coolant temperature jump ($x_j^o - x_j^i$) in function of the volumetric flow rate (f_j).

- $q_j^{cd}(t)$ is the rate at which heat energy is transferred to/from the node purely through the mode of conduction;
- $q_j^{cv}(t)$ is the rate of heat exchange with the coolant in the liquid cooling circuit through the mode of convection;
- $p_j(t)$ is the rate at which electrical energy is converted into thermal energy locally at the node.

The functional structure of q_j^{cd} follows from (1) in Section II-A. Under constant flow conditions, the term q_j^{cv} in (15) can be approximated using Newton's law of cooling as

$$q_j^{cv}(t) \propto -\left(x_j^c(t) - x_j^l(t)\right), \tag{16}$$

where the effective temperature of the coolant $x_j^l(t)$ is, in first approximation, a function of $x_j^i(t)$, $x_j^o(t)$ and the physical parameters of the specific heat exchanger. More in general, q_j^{cv} exhibits a nonlinear dependence on the flow rate f_j. Later in Section III, we describe these nonlinear features by adopting a popular semi-empirical model suitable for a relevant class of heat exchangers when flows are time-varying. Finally, we notice that the power dissipation at the components is modelled as a uniform volumetric phenomenon with a first order effect on the proxy temperature x_j^c.

Remark 4: The nature of the thermal interfaces and the typical material properties found in data servers induce small Biot numbers. In turn, this motivates the adoption of the scalar temperature dynamics (15). We validate this choice experimentally in Section V-C.

Remark 5: The inclusion of computational aspects such as the CPU, memory, and I/O loads is thus accomplished in an aggregate manner by mapping these quantities into the corresponding electrical power consumption at the chip level.

3) Modelling the outflow temperature $x_j^o(t)$: Applying a thermal energy balance to a control volume containing only the generic node j gives the equation

$$q_j^o(t) - q_j^i(t) = -q_j^{cv}(t), \tag{17}$$

where $q_j^{cv}(t)$ is the heat rate in (15) and $q_j^i(t)$, $q_j^o(t)$ are the rates at which heat energy, transported by the coolant, enters and exits node j. Expanding these terms in function of the volumetric flow and the physical properties of the coolant, assuming that the pressures in the cooling circuit are in first approximation constant in time, yields

$$c_p \rho f_j(t) x_j^o(t) = c_p \rho f_j(t) x_j^i(t) - q_j^{cv}(t), \tag{18}$$

where ρ is the density of the coolant and c_p its specific heat capacity at constant pressure. By rearranging terms we obtain the outflow temperatures as

$$x_j^o(t) = x_j^i(t) - \frac{q_j^{cv}(t)}{c_p \rho f_j(t)}. \tag{19}$$

(12), (14), (15) and (19) are the salient ingredients of our control-oriented framework for liquid cooling applications. In the following section we show how they can be specialized to model the characteristic thermal networks of data centers at the server level.

III. A library of standard models at the server level

Here we develop the generic node model of Section II into a library of reusable node types. We start by categorizing nodes into two classes (see also Table I). *Thermal nodes* participate in the thermal dynamics by exchanging heat with their neighbors and by acting as local heat sources. Heat reservoirs and electrical components such as CPUs are examples of thermal nodes. *Transport nodes*, instead, have an infrastructure character: they do not participate directly in the thermal dynamics but rather support the liquid cooling operations as we discuss next.

A. The transport nodes

In practice, the liquid cooling overlay $\mathcal{E}_{cv}$ is implemented using pipes, *joints* and *flow splitters*. In our framework, these elements are modelled as transport nodes, that is, mathematical constraints describing how the coolant can flow in and out of each manifold and how the flow temperature propagates along the cooling circuits. Transport nodes are thus means to describe the topology of the cooling circuit while no heat is absorbed or rejected within these nodes. Therefore, we neglect the dynamics (15) and require $\left|\delta_-^{\mathcal{E}_{cd}}(j)\right| = \left|\delta_+^{\mathcal{E}_{cd}}(j)\right| = 0$ for all transport nodes $j \in \mathcal{N}$.

1) Joint nodes: This node type models joint couplings in the liquid cooling circuit. The generic joint node j has multiple inflows and one outflow. The inflow temperature is given by (14) and we assume that the flow crossing j exits the node instantaneously without exchanging heat:

$$x_j^o(t) = x_j^i(t). \tag{20}$$

Node type	$\left\|\delta_-^{\mathcal{E}_{cd}}(j)\right\|$	$\left\|\delta_+^{\mathcal{E}_{cd}}(j)\right\|$	$\left\|\delta_-^{\mathcal{E}_{cv}}(j)\right\|$	$\left\|\delta_+^{\mathcal{E}_{cv}}(j)\right\|$	Inflow temp.	Dynamics	Outflow temp.
Supply	0	0	0	1	-	-	(6)
Collector	0	0	1	0	(14)	-	-
Joint	0	0	≥ 2	1	(14)	-	(20)
Splitter	0	0	1	≥ 2	(14)	-	(20)
Environmental	0	≥ 1	0	0	-	(21)	-
MMC	≥ 0	≥ 0	1	1	(14)	(25)	(24)
Active	≥ 1	≥ 1	0	0	-	(26)	-

Table I: Each node type introduced in this section is specialized to address a specific modeling need in capturing the temperature dynamics of liquid cooling data servers.

2) Splitter nodes: Splitter nodes describe constant-ratio flow splitters. The generic splitter node j has one inflow and at least two outflows. The inflow temperature is again (14), with the summation reduced to the single inflow link, and the outflow temperature is evaluated as for the joint nodes through (20). Finally, splitter nodes are characterized by the splitting ratios Λ defined in (11).

3) Supply nodes: The generic supply node $j \in \mathcal{S}$ is characterized by the manipulable flow $u_{I(j)}(t)$ and the coolant temperature $x_j^o(t) = x_j^i(t)$ (considered here a measurable exogenous input).

4) Collector nodes: A collector node $j \in \mathcal{C}$ has one inflow and no outflows. The inflow temperature is evaluated using (14).

B. The thermal nodes

Nodes endowed with a local temperature state are called thermal nodes. They are further specialized in: *environmental nodes*, modelling heat reservoirs, *heat exchanger nodes*, introduced to describe heat transfers to and from the liquid cooling loop, and *active nodes*, modelling the self-heating components.

1) Environmental nodes: In our set-up a generic environmental node j is not connected to the liquid cooling circuit; in particular, the inflow and output temperatures are disregarded. Rather, the node acts as a heat reservoir being connected in the heat conduction overlay $\mathcal{E}_{cd}$, that is, $\left|\delta_-^{\mathcal{E}_{cd}}(j)\right| = 0, \quad \left|\delta_+^{\mathcal{E}_{cd}}(j)\right| \geq 1$. Environmental nodes are thus assumed to have constant temperature in time,

$$\dot{x}_j^c(t) = 0, \quad x_j^c(0) = \overline{x}_j^c, \tag{21}$$

for some given temperature $\overline{x}_j^c \in \mathbb{R}_{\geq 0}$ of the reservoir.

2) Heat exchanger nodes: This type of nodes is used to model the heat transfer interfaces between the on-board electrical components and the cooling loop. Thus, the j-th heat exchanger node can transfer heat over both the conduction and convection overlays. The inflow and outflow temperatures are given by (14) and (19), respectively. The local temperature dynamics is modelled by specializing (15). The rate of local heat generation is zero and the dynamical contribution due to thermal conduction is given by (2). The heat rate $q_j^{cv}(t)$ due to convection is instead approximated by the following nonlinear resistive thermal model

$$q_j^{cv}(t) = -\frac{x_j^c(t) - x_j^l(t)}{R_j\big(f_j(t)\big)}, \tag{22}$$

where

- $x_j^l(t) - x_j^c(t)$ is the effective temperature difference between the liquid coolant and the component;
- $f_j \mapsto R_j(f_j)$ is the lumped thermal resistance of the heat exchanger in function of the (volumetric) flow rate f_j.

Here we specifically consider Manifold Micro-Channel (MMC) heat exchangers, a specific liquid cooling technology that has been investigated extensively, both outside [30], [31], [32], [33], [34] and inside data centers [35], [36]. We thus model the flow dependence of the thermal resistance R_j in (22) through the following rational form

$$f_j \mapsto R_j(f_j) \doteq R_j^p + R_j^s + \frac{R_j^b}{f_j}, \tag{23}$$

where R_j^p, R_j^s, R_j^b are positive parameters defining the heat transfer performance of the physical device. The thermal resistance R_j corresponds then to the series connection of three thermal resistances:

- R_j^p: the thermal resistance given by the component's package and the metal plate at the base of the heat sink;
- R_j^s: the resistance of the heat transfer structure, that is, the channel's fins in a MMC design;
- R_j^b: the bulk resistance between the heat transfer structure and the liquid coolant.

We stress that the nonlinear model (23) has been shown to capture accurately the heat exchange profiles of MMC devices. The three resistance parameters depend on the physical microchannel design (layout and topology) and can be either estimated from first principles or evaluated numerically [33], [34], [36], [37]. As for the effective coolant temperature $x_j^l(t)$ in (22), following [34], [36], we propose to set it equal to the inflow coolant temperature, in symbols $x_j^l(t) = x_j^i(t)$.

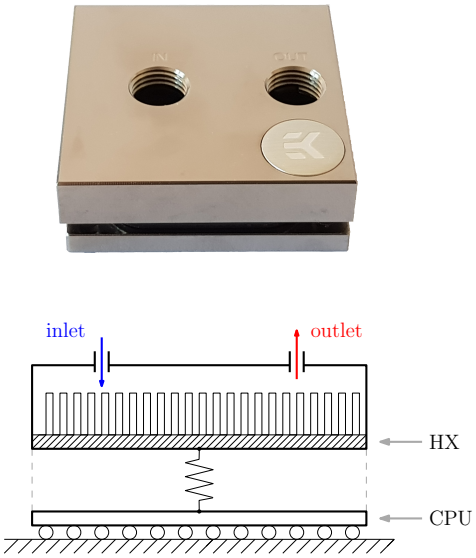

Figure 3: Above, photo of an off the shelf EK-Supremacy EVO water cooling block (with the mounting plate removed). The 60 × 60 × 20mm block is manufactured in nickel-plated brass and weights 380 grams with about 25 percent of the mass corresponding to its plate. Below, sketch of a CPU chip dissipating heat into an MMC heat exchanger.

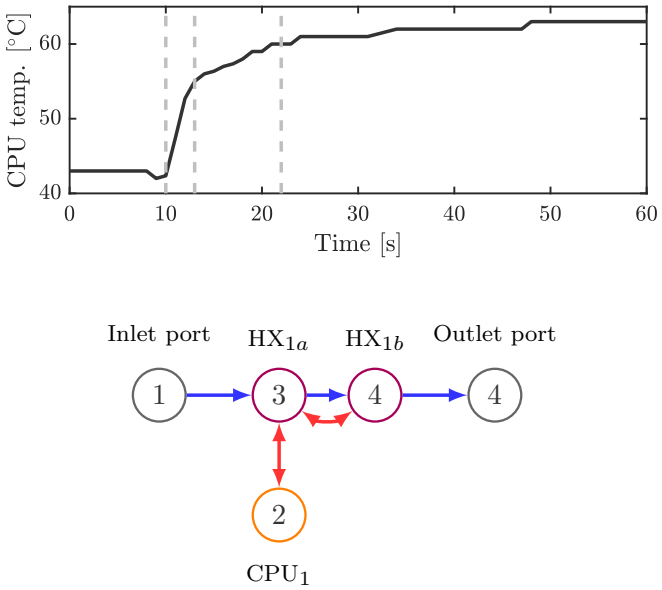

Figure 4: Above, temperature recording of a liquid cooled CPU subject to a step in the computational workload at time $t = 10$s. The coolant is supplied at a constant rate (see Example 3 for the details). Below, graph representation of the corresponding thermal network model: the blue arrows indicate links in the convection overlay while red arrows depict links in the conduction overlay. Two heat exchangers nodes connected in series are used to describe the temperature dynamics of a single MMC device.

By using (22) and (23) in (19) we obtain the heat exchanger's outflow temperature

$$x_j^o(t) = x_j^i(t) + \frac{\big(x_j^c(t) - x_j^i(t)\big)}{c_p\rho\left(R_j^b + (R_j^p + R_j^s)f_j(t)\right)}. \tag{24}$$

Finally, substituting (2) and (22) in (15) yields the full continuous time temperature dynamics of the node:

$$\begin{aligned}\dot{x}_j^c(t) = &-\frac{f_j(t)}{d_j\left(R_j^b + (R_j^p + R_j^s)f_j(t)\right)}\big(x_j^c(t) - x_j^i(t)\big) + \dots \\ &- \sum_{h\in\delta_-^{\mathcal{E}_{cd}}(j)} \frac{k_{hj}}{d_j}\big(x_j^c(t) - x_h^c(t)\big).\end{aligned} \tag{25}$$

3) Active nodes: In a data server the active nodes coincide with the electrical components that necessitate cooling such as CPUs, DIMMs and any companion chips. Heat is either injected into the network through local generation or exchanged over the conduction overlay. The generic active node j has no inflows or outflows, while the corresponding temperature dynamics takes the form:

$$\dot{x}_j^c(t) = -\sum_{h\in\delta_-^{\mathcal{E}_{cd}}(j)} \frac{k_{hj}}{d_j}\big(x_j^c(t) - x_h^c(t)\big) + \frac{p_j(t)}{d_j}. \tag{26}$$

Example 3: Consider the uni-CPU liquid cooled set-up in Figure 3. The cool inlet liquid enters the MMC and is channeled into the micro-channels by a jet-plane. The warmer coolant then recirculates within the MMC exchanging heat with the exchanger's casing until it reaches the outlet port. In Figure 4, we recorded a CPU temperature trajectory where we transition the workload from idle to peak utilization. By inspection of the trace we notice two fast modes and one slow mode indicating the interplay of three main thermal inertiae: 1) the CPU package, 2) the MMC plate and heat-transfer structure and 3) the MMC casing. We propose to use the thermal network in Figure 4 to model this setting. More in detail we consider that the CPU core is interposed between the server motherboard and the heat exchanger, and we assume any heat losses from this interstice to be negligible. We then model the chip as having homogeneous thermal properties and approximate the heat transfer from the CPU as a one-dimensional thermal flux directed towards the heat exchanger as in (26). We approximate the micro-channel structure of the heat exchanger as an isothermal surface and thus let a single scalar temperature be a proxy for the temperature of the whole plate. To account for flow dependency in the heat transfer rates we capture the corresponding dynamics using a heat exchanger node (HX$_{1a}$). Finally, the thermal dependency between the bottom and upper part of the water block is taken into account by introducing a thermal bridge in the conduction overlay to a second heat exchanger node (HX$_{1b}$) with higher thermal capacitance and lower convective heat transfer performance.

IV. Controlled liquid cooling

We propose a controlled liquid cooling strategy that aims at simultaneously minimizing the supply flow rates (to decrease the actuation costs) while increasing the temperature of the coolant at the outlet (to improve its quality in heat reuse applications). To this aim, we formulate a polynomial optimal control problem in a Receding Horizon Control (RHC) fashion. The minimization objective is then to minimize the coolant pumping cost subject to the temperature constraints induced by the underlying thermal network model.

Remark 6: Our focus on an optimizing strategy is motivated by the impact that exogenous inputs have on the temperature dynamics. For example, the heat-loads at a single server can vary instantaneously with the workload from tens to hundreds of Watts. Moreover, the inlet coolant temperatures are dependent on the past workload at the computer room level and the amount of thermal energy that the data center injects into the end-user heat reuse application. The latter being a time-varying quantity depending on the thermal state of the subordinate system. We thus suggest that dynamical controlled liquid cooling strategies should be *designed* with the knowledge of the attainable performance as measured by the system's thermal model and an optimizing controller. An on-line *implementation* of the feedback could, instead, simply mimic the optimal control law by approximating it using one of many nonlinear regression tools.

A. Discretization

We assume a uniform sampling schedule with period Δ. The dynamics of the thermal network is discretized using Euler's forward rule and propagated over a horizon of length H sampling periods. With a slight abuse of notation, we let $x_j^i(k)$ denote the inflow temperature of node j at time $k\Delta$ and adopt the same convention for all the time-varying quantities. All manipulable and exogenous inputs are assumed zero-order held.

We assume a time-scale separation between the temperature dynamics of the server and that of the storage Coolant Distribution Unit (CDU). In particular, the temperature of the supply inflows $x_j^i(k), j \in \mathcal{S}$, are measured at the beginning of the receding horizon and assumed to remain constant over it. We assume moreover that the power consumption of the server's active nodes is unknown, that the computational loads are also unknown, and that they are difficult to forecast. To cope with this minimal-information setting we consider the following worst-case scenario where each component dissipates the highest plausible power:

$$p_j(k) = p_{j,\max}, \quad \forall\, k \geq 0 \tag{27}$$

for all active nodes $j \in \mathcal{N}$. Considering (27) leads then to a feedback law that satisfies the operation constraints over the specified horizon irrespective of the unknown future computational loads.

B. Static and dynamical constraints

The safe operation of the server requires to keep the temperature of the main components below established thresholds[2]:

$$x_j^c(k) \leq x_{j,\max}^c, \quad \forall\, k \geq 0 \tag{28}$$

for all active nodes $j \in \mathcal{N}$. Moreover, the supply flow rates must satisfy box constraints of the form

$$\boldsymbol{u}_{\min} \preceq \boldsymbol{u}(k) \preceq \boldsymbol{u}_{\max} \quad \forall\, k \geq 0, \tag{29}$$

with $\boldsymbol{u}_{\min}, \boldsymbol{u}_{\max} \in \mathbb{R}^m_{\geq 0}$.

The continuous time thermal dynamics of the network are approximated under the assumption of piece-wise constant flows, discretized using Euler's forward rule and then rewritten as polynomial constraints. For instance, discretization of (25) yields

$$\begin{aligned} \frac{x_j^c(k+1) - x_j^c(k)}{\Delta} &= \dots \\ &- \frac{f_j(k)\big(x_j^c(k) - x_j^i(k)\big)}{d_j\left(R_j^b + (R_j^p + R_j^s) f_j(k)\right)} \cdots \\ &- \sum_{h \in \delta^{\mathcal{E}_{cd}}_{-}(j)} \frac{k_{hj}}{d_j}\big(x_j^c(k) - x_h^c(k)\big), \end{aligned} \tag{30}$$

which can be rewritten into an equivalent polynomial constraint by multiplying both its left and right hands by the positive affine term $R_j^b + (R_j^p + R_j^s) f_j(k)$.

C. The cost function

Denote a generic candidate control sequence through

$$\boldsymbol{u}_{k:k+H-1} \doteq \big(\boldsymbol{u}(k), \boldsymbol{u}(k+1), \dots, \boldsymbol{u}(k+H-1)\big). \tag{31}$$

and define the corresponding control cost $g(\cdot)$ by

$$\boldsymbol{u}_{k:k+H-1} \mapsto g(\boldsymbol{u}_{k:k+H-1}) \doteq \sum_{z=k}^{k+H-1} P_{\text{flow}}(\boldsymbol{u}(z)), \tag{32}$$

where P_{flow} quantifies the pumping costs incurred to sustain the given volumetric flow rates. The main feature of $g(\cdot)$ is to penalize the overprovision of the liquid coolant while attaining the auxiliary objective of increasing the coolant temperature throughout the circuit. Indeed, by inspection of (24) we expect lower flow rates to induce higher outflow temperatures with a first order effect. Moreover, since the thermal convection resistance (23) is monotonic decreasing with respect to f_j, the required temperature gap to transfer a given heat-load increases as the flow rate decreases. This leads to a higher amount of thermal energy that is stored within the network and thus higher x_j^c and higher outflow temperatures. The cost (32) thus pairs a natural economic control formulation with the benefit of improving the eventual quality and value of the outlet heat.

[2]We notice that dew point safety considerations potentially introduce a new set of lower temperature constraints. In practice, however, dew point controllers are naturally located at a higher level of the cooling infrastructure (at and above the CDU level). Indeed, while condensation phenomena entail serious hazards, it would be impractical and cost-inefficient to regulate the inlet coolant temperature at the server level.

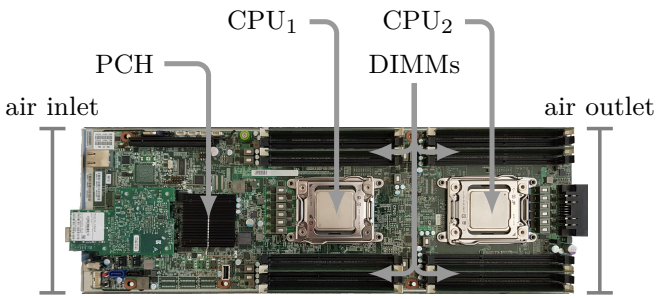

Figure 5: Top view of the test bed server (slid outside of its enclosure) before the liquid cooling upgrade. This Windmill V2 platform has dimensions 16.4×50.7cm

D. The RHC problem formulation

Let g be the cost function in (32) and ψ an opportune vector-field of polynomial constraints obtained by stacking all the static and dynamical polynomial constraints of Section IV-B. For instance, design ψ by first stacking the $2m$ scalar inequalities corresponding to (29), then, for all active components j, consider the static and dynamical constraints corresponding to (28) and (30). Finally, append the inflow and outflow constraints derived by reformulating the rational forms (14) and (19). Let moreover $\boldsymbol{X}_0$ be a compatible vector corresponding to the measured state of the thermal network at time k_0. Our control policy aimed at heat recovery corresponds then to solve the following optimal control problem:

$$\begin{aligned} &\min_{\boldsymbol{u}_{k_0:k_0+H-1}} g(\boldsymbol{u}_{k_0:k_0+H-1}) \\ &\text{subject to:} \\ &\quad \psi\big(\boldsymbol{x}^i(k_0), \boldsymbol{X}_0, \boldsymbol{u}_{k_0:k_0+H-1}\big) \preceq 0 \end{aligned} \quad . \tag{33}$$

V. Experimental results

In this section we derive the thermal network model of a direct liquid cooled server and assess the heat-reuse performance of (33) in a realistic scenario. All the field data supporting the remainder of this work has been acquired from an Open Compute *Windmill V2* server, a Facebook design aimed at hyperscale deployments [38]. This platform deploys two CPU sockets, 8 low-power DIMMs, a Platform Controller Hub (PCH) chip, support electronics for I/O and power distribution and two axial fans for cooling. The board's layout is highlighted in Figure 5. The maximum heat-loads are categorized by component in Table II. Notice that all tests have been carried out with the server deployed in its original 1.5U tray enclosure.

A. Experimental setup

The experimental set-up is schematized in Figure 6. A 0.8 liter tank is used as the only water reservoir. The main cooling circuit (depicted below the reservoir) circulates the coolant through the liquid cooled server and back to the storage. A three-way valve is used to limit the maximum volumetric flow through the test bed. Four temperature sensing points are used to monitor the air (T2, T3) and the water coolant (T1, T4) at the inlet and outlet faces of the server. An auxiliary cooling circuit (depicted above the reservoir), circulates the coolant through a water-to-air heat exchanger; an independent pump and a fans' array are actuated to maintain a desired set-point for the inflow temperature T1. We henceforth omit the operative details of the auxiliary cooling circuit which is employed with the sole purpose of simulating a constant temperature water reservoir.

Part	Count	Model	TDP
CPU	2	Intel Xeon E5-2670	115W
PCH	1	Intel C600/X79	7.8W
DIMM	8	2GiB DDR3-1600	2.1W

Table II: The two CPUs are responsible for more than 90 percent of the total heat load at peak usage on the adopted Windmill V2 test bed.

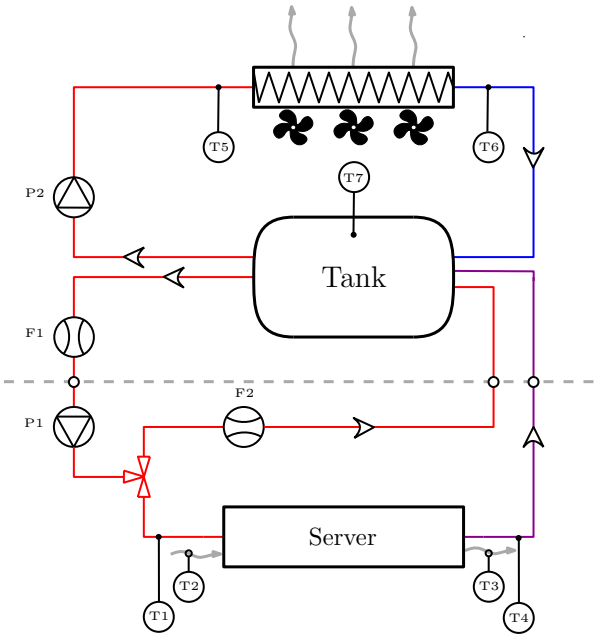

Figure 6: Schematic representation of our experimental liquid cooling setup. The pump P1 provisions the coolant to the server. The inlet and outlet temperatures of both air and water are monitored through sensors T1 to T4. An auxiliary cooling circuit is used to regulate the water temperature within the tank. Notice that at each temperature sensing point multiple sensors are used to average the measurements.

The two CPUs on the Windmill server have been retrofitted using off the shelf components acquired from EK Water Blocks. Differently from the engineered platform in Figure 1, we did not connect other self-heating

parts in the liquid cooling loop[3]. We stress, however, that the two CPUs are the largest contributors to the server heat-load and thus a natural focus point. During all the experiments, a constant volumetric air flow rate equal to $0.00816\text{m}^3/\text{s}$ was provisioned through the server enclosure to cool down these remaining components.

To confirm the capability of thermal networks to capture thermal dependencies among nodes we have chosen to implement a series cooling circuit. The coolant thus flows from the main inlet to the main outlet while crossing the two water cooled CPUs in series. Depending on the current heat-load, this configuration leads the two packages to operate at markedly different temperature conditions (as demonstrated later in Section V-C).

Monitoring and control of the test bed has been performed through a dedicated software toolkit. We collected information about the logical and thermal state of the server (power load, CPU temperatures, inlet and outlet air temperatures) by querying the server's Intelligent Platform Management Interface (IPMI) through an out-of-band Ethernet connection. Ad-hoc electronics have been designed to acquire the additional temperature and tachometric measurements from the other external sensors and actuators. At run-time, this state information is fed back to our supervision algorithm, residing on a remote machine, and used to compute the optimal pump control of Section IV in real-time. Time-varying computational workloads have been injected in the instrumented server according to either deterministic or random utilization patterns[4].

B. The thermal network model

The liquid cooling topology discussed above and schematized in Figure 6 is captured by the thermal network model of Figure 7. A single source node, corresponding to pump P1, provisions the coolant to the server subnetwork and back to the reservoir through two modelled collector nodes. Along the same reasoning of Example 3, each MMC heat exchanger has been modelled by the series of two heat exchanger nodes. Finally, an environmental node is introduced to capture heat losses from the MMC casings due to the constant-rate provisioning of air.

Instrumental to estimating the thermal network parameters, we have measured the water pump power consumption at different rotational speeds and the corresponding volumetric flow rate through the server enclosure (see Figure 8). With a slight abuse of notation, in what follows, we write $u(k)$ to denote the volumetric flow rate of water through the server at time k. Through a thermal energy balance applied at the server's inlet and outlet ports we have moreover estimated the amount of harvested heat and the heat losses under a medium CPU loading condition. Figure 9 shows how modulating the water supply rate u corresponds to modulating the temperature at which the water is harvested, x^i_{10}. Remarkably, at the lower flow rates, the outlet water temperature approaches 60°C when the server heat-load is approximately half of the peak value. At the same time, low control values imply higher CPU temperatures which in turn induce higher losses to the environment as shown in Figure 10. The static performance of the system is summarized graphically in Figure 11 by plotting the flow-dependent equivalent thermal resistance $\overline{R}$ that relates the temperature gap between the inlet and outlet water and the power rejected into the cooling loop (cf. Figure 7)

$$\overline{R} \doteq \frac{x^i_{10} - x^i_1}{q^{cv}_4 + q^{cv}_7}. \tag{34}$$

In other words, $\overline{R}$ quantifies the attainable outlet water temperature in function of the heat-load and the coolant supply rate; thus, it highlights the overall relevance of provisioning the coolant dynamically. However, to be able to predict also the temperatures of the individual network nodes' (a requirement induced by (28)) it is necessary to identify the full model of the network.

C. Identification of the full dynamics

In order to train the complete dynamical model we have let the server run computational workloads designed by sampling a Pseudo-Random Binary Sequence (PRBS) with random switchings every 60 seconds. We mapped the binary low and high values of the PRBS into normalized CPU usage values of 0 (the CPU is idle) and 1 (the CPU is fully utilized), respectively. By continuously monitoring the test bed we have then acquired (recall Figure 7) the inlet temperature $x^i_1(k)$, the CPU temperatures $x^c_4(k)$, $x^c_7(k)$, the water flow rate through the server $u(k)$ and the power consumptions of the CPUs, $p_4(k)$ and $p_7(k)$.

The thermal network model has been fitted to the data by minimizing a cost quantifying the H-steps ahead ($H = 60$) prediction capabilities of the model:

$$J(\boldsymbol{\theta}) \doteq \sum_{j=4,7} \sum_{k=0}^{K-1-H} \left(\widehat{x}^c_j(k, k+H; \boldsymbol{\theta}) - x^c_j(k+H)\right), \tag{35}$$

where K is the number of samples in the training data set, $\boldsymbol{\theta}$ is the current candidate estimand (whose entries correspond to all the unknown parameters of the thermal network) and $\widehat{x}^c_j(k, k+H; \boldsymbol{\theta})$ is the H-steps ahead prediction of the proxy temperature of node j given knowledge of the state of the system at time k and the parameters $\boldsymbol{\theta}$.

We considered a sampling interval $\Delta = 1\,\text{s}$, and used four hours of measurements to minimize the cost (35). The performance of the estimated thermal network model is quantified in terms of the model-fit errors

$$\varepsilon^{\text{p}}_j(k) \doteq \widehat{x}^{\text{p}}_j(k-1, k; \boldsymbol{\theta}) - x^{\text{p}}_j(k), \quad j = 4, 7 \tag{36}$$

[3]This choice reflects both the lack of opportune fixations (as in the case of the PCH) and the lack of on-chip temperature information that would have been necessary to validate the augmented model (as in the case of the DIMMs).

[4]The CPU stressor that we used is based on the open-source software *stress-ng* [39]

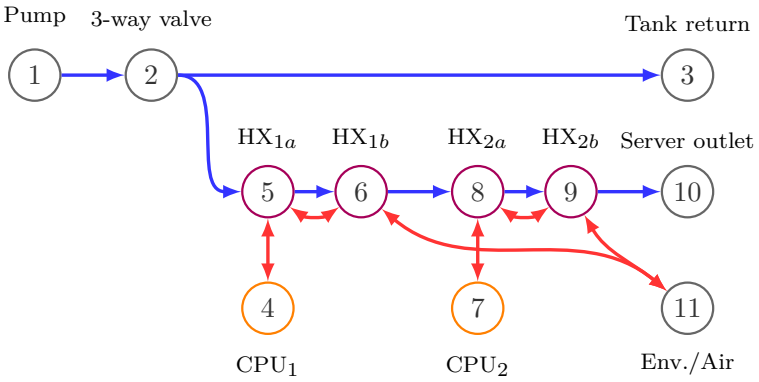

Figure 7: Graph representation of the thermal network used to model the Windmill V2 test bed of Figure 5. The blue arrows indicate the liquid cooling advection paths. The red arrows indicate the modelled heat conduction paths between the CPUs (drawn in orange) and the corresponding, top-mounted, heat exchangers (drawn in violet). An environmental node is introduced to capture heat losses due to the constant rate provisioning of air.

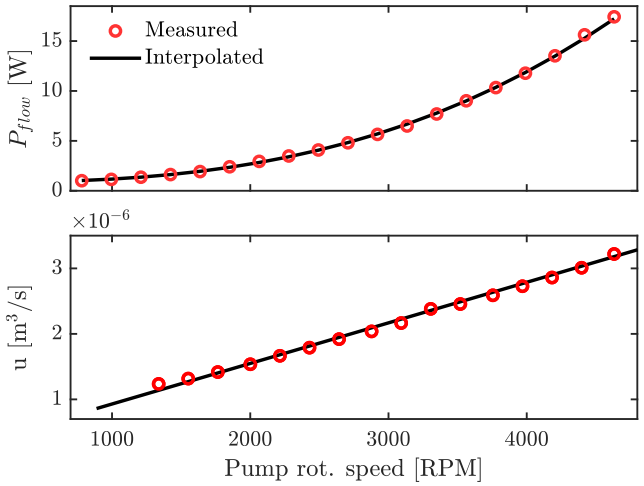

Figure 8: Pumping power and volumetric water flow rate through the server in function of the pump's rotational speed.

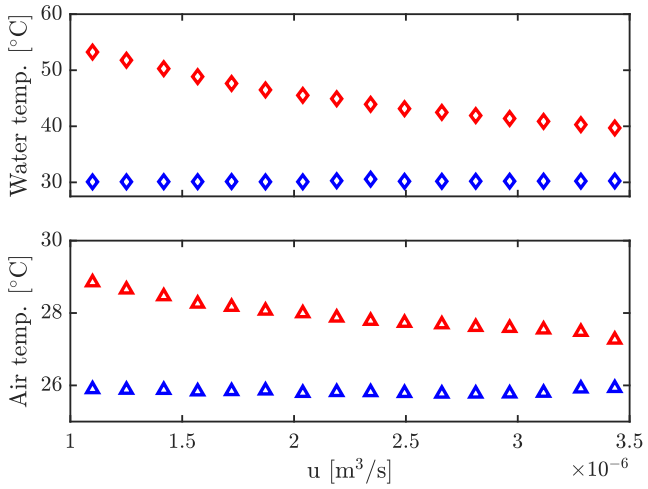

Figure 9: The average inlet (drawn in blue) and outlet (red) temperatures of the air and water entering and leaving the server in function of the volumetric water flow rate through the server. To be able to safely measure these quantities at the lower flow rates we limited the heat-load dissipated during the test to about 135 Watts.

namely, the difference between the measured and predicted temperatures of the two CPUs, computed on a validation data set. The quality of the J-optimal model used in the following is assessed graphically in Figure 12 and in Table III. The model demonstrates uniformly good performance within the standard operational ranges of exogenous and control inputs. For both CPU temperatures, the mean model-fit errors are smaller then 0.158 °C in absolute value and their standard deviations are smaller then 1.52 °C. Moreover, the two errors are smaller than one standard deviation in absolute value with a normalized frequency of 0.77, while the absolute fit errors are smaller than 2°C more than 83% of the time.

	Mean	St. Dev.
$\varepsilon_4^{\mathrm{P}}$	-0.1526	1.3476
$\varepsilon_7^{\mathrm{P}}$	0.1584	1.5194

Table III: Mean and standard deviation statistics of the validation model-fit errors in (36) evaluated over a four hours long trace.

D. Assessment of the control performance

We compared a static provisioning policy (henceforth shortened as "STA") where the coolant is supplied to the server at a constant rate (chosen as the smallest volumetric flow rate that maintains the servers's on-board temperatures below the safety thresholds (28)) against the dynamic policy (33) (in short "DYN"). To this aim we monitored the test bed while reproducing six sets of pre-determined CPU workload traces. The latter traces have been generated by sampling a PRBS as in Section V-C while varying the PRBS's minimum switching

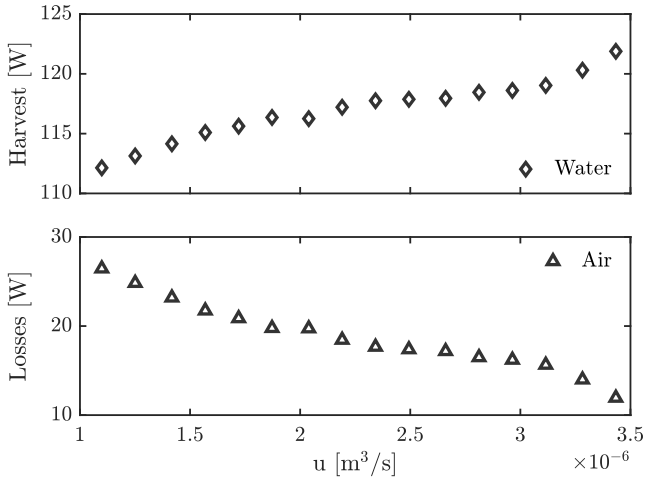

Figure 10: The estimated amount of thermal energy rejected into the air (losses) and into the water cooling loop corresponding to the experiment in Figure 9.

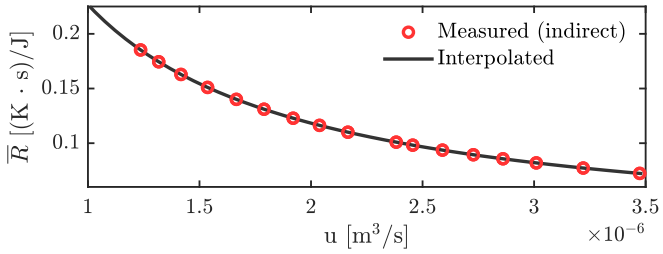

Figure 11: The equivalent thermal resistance defined in (34). The interpolating curve has the functional form $\overline{R}(u) = \overline{R}' + \overline{R}''/u$.

time, T_{PRBS}, and the average workload, w_{AVG} (as measured by the ratio of active versus idle periods in the trace) to represent different stress patterns. We designed thus 12 scenarios by crossing two different values for the switching time $T_{\text{PRBS}} \in \{10, 60\}$, three values for the average CPU utilization $w_{\text{AVG}} \in \{0.25, 0.5, 0.75\}$ and two different values for the coolant inlet temperature $x_1^i \in \{30, 35\}$ degrees Celsius[5]. Each scenario has been accurately replayed twice to compare the cooling performance of the static (STA) and dynamic (DYN) controllers.

The length of each experiment (involving the triple of a given control strategy, a workload scenario and an inlet water temperature) has been set to 240 minutes. In the case of DYN, we used $\Delta = 1$ seconds, considered a prediction horizon of $H = 60$ steps, and set the temperature thresholds in (28) as $x^c_{j,\max} = 80\,^\circ$C for both CPUs.

For each scenario, we quantified the control performance in terms of the following two indexes: *1)* the average amount of water coolant supplied to the system, and *2)* the average temperature of the coolant at the servers's outlet.

[5]The higher coolant temperature was limited by the heat losses in the supporting cooling circuit.

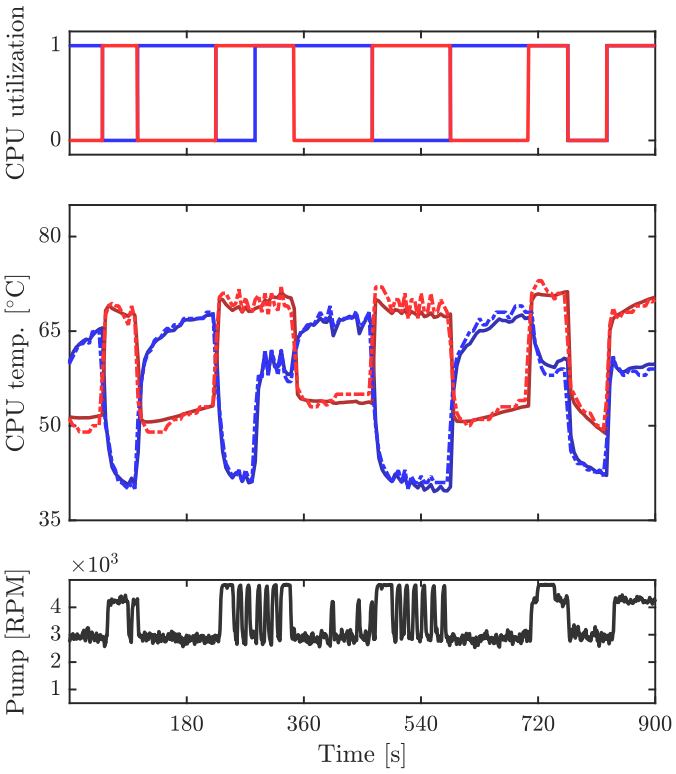

Figure 12: Validation CPU temperatures (dashed lines in light blue and light red) acquired on the liquid cooled Windmill server while running a computational stress loader. During the experiment, the water flow rate was regulated by letting the server air-cooling algorithm control the rotational speed of the pump. The identified thermal network model accurately predicts the temperatures of both CPU1 (dark blue) and CPU2 (dark red). The sampling period Δ is one second. The measured temperatures are quantized at the hardware level with a interval of 1°C.

The above indexes are formally defined as

$$\widehat{u} \doteq \frac{1}{T}\sum_{k=0}^{T-1} u(k), \qquad \widehat{x} \doteq \frac{1}{T}\sum_{k=0}^{T-1} x^o_{10}(k) \tag{37}$$

where $u(\cdot)$ and $x^o_{10}(\cdot)$ corresponds now to the sampled sequences of a given experimental scenario. Since the pumping cost P_{flow} is a monotonic increasing function of the water flow rate u, lower values of $\widehat{u}$ indicates lower pumping costs. As for the temperature index $\widehat{x}$, higher values indicate a higher quality heat harvest at the server's outlet which in turn relates directly to the economic value of the outflowing coolant in heat-reuse applications [4]. To remove any dependence on the initial conditions when evaluating (37) we discarded the first 30 minutes of each test.

A qualitative comparison of the STA and DYN control strategies is shown in Figure 13. The different trajectories of the state variables highlight how DYN achieves higher outlet water temperatures by minimizing the supply cost while managing the thermal state of the CPUs and the large thermal inertiae of the heat exchangers. Within the

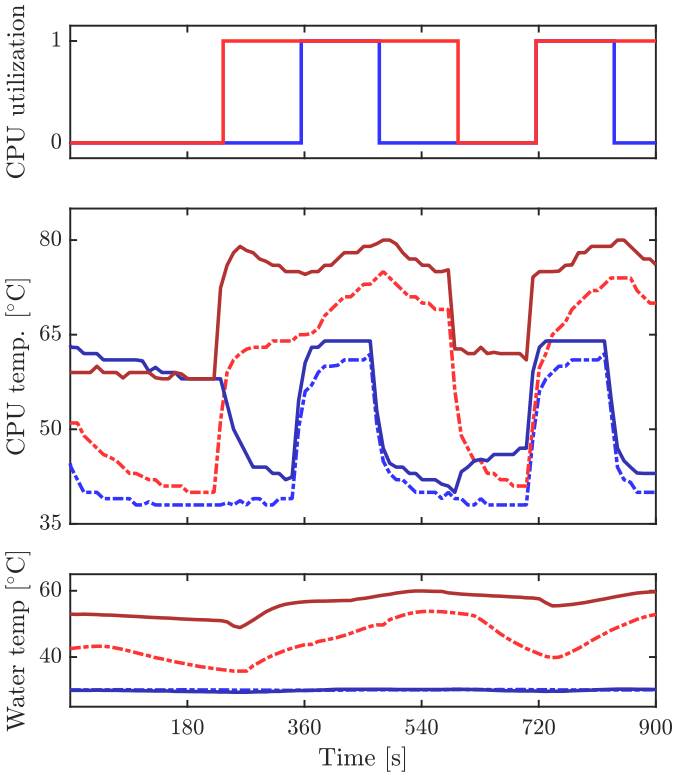

Figure 13: Example temperature trajectories induced by the static (dashed lines) and dynamic (solid lines) provisioning strategies. Remarkably, as the amount of heat energy stored in MMC devices increases, the optimizing supervisors steers the CPU states away from the upper temperature envelope to comply with constraint (28). The mean water temperatures achieved by STA and DYN over the full length of the test are 44.52°C and 55.04°C respectively.

testing conditions the dynamic provisioning laws achieved approximately 10 degrees higher outlet temperatures while reducing the water supply by 29 percent on average.

The experimental results relative to $x_1^i = 30°\text{C}$ and $x_1^i = 35°\text{C}$ are summarized in Figure 14. The panels in the first row, dedicated to $\widehat{u}$, highlight the savings that can be realized by considering the proposed dynamic provisioning policy (33) relative to the coolant supply performance of the static policy. The panels in the second row, dedicated to $\widehat{x}$, quantify the benefits of controlled liquid cooling in terms of the temperature of the coolant at the server outlet. When the server operates at a low computational load ($w_{\text{AVG}} = 0.25$) the static supply policy overprovisions the cooling resources leading to the unnecessary usage of water, increased actuation cost and the overall decrease of the heat quality index $\widehat{x}$. In this specific utilization scenario, the dynamic policy can reduce the average coolant supply rate up to a factor of two. Gains, from 19% to 45% can be achieved at a medium-high computational load depending on the coolant inlet temperature. Throughout our tests, the heat-quality index $\widehat{x}$ increases from 6 to 11.5 degrees Celsius when the dynamical policy is used instead of the static one.

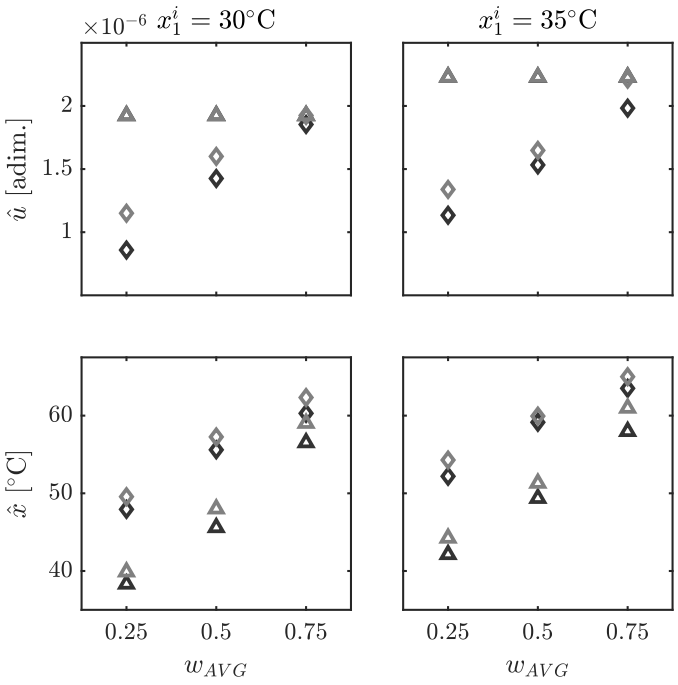

Figure 14: The average coolant supply rate $\widehat{u}$ (panels in the first row) and the average outlet coolant temperature $\widehat{x}$ (panels in the second row) evaluated for the static (triangles) and dynamic (diamonds) provisioning strategies when $T_{\text{PBRS}} = 10$ (gray marks) and $T_{\text{PBRS}} = 60$ (black marks), as w_{AVG} varies. The dynamic strategy produces higher quality heat harvests with a lower actuation cost throughout all the considered scenarios.

Overall, controlled liquid cooling is effective at reducing the actuation cost across all inlet temperature regimens and workload scenarios. As expected, however, the benefit of using DYN decreases as the server approaches the peak heat-load.

VI. Conclusions

This manuscript unfolds a control theoretic approach to enabling heat reuse in data centers that deploy direct liquid cooling systems. We have proposed a thermal modelling framework, built on first principles, that approximates the thermal dynamics of liquid cooled data servers. We have then shown that the resulting dynamics can be discretized and exploited in optimal control problems suitable to the design of dynamic flow-provisioning policies. We have considered in particular a receding horizon strategy where the cost to be minimized is taken to be power dissipated to supply the liquid coolant.

An implementation in a realistic scenario has highlighted the effectiveness of our strategy in both minimizing the control effort and increasing the temperature of the coolant harvested at the server's outlet, enabling thus higher quality heat at lower pumping costs, and eventually improving the prospective potential of controlled liquid cooling for heat reuse purposes.

Future directions include the scaling up of our strategy to a full rack of data servers at the research data center

SICS-ICE in Luleå. Of particular interest is the study of the coupling between this higher power heat source and the end-user of the heat in thermal storage and heat-injection scenarios.

Acknowledgments: We thank the team at the research data center SICS-ICE in Luleå for lending us the Open Compute Project test bed used in Section V.

References

[1] J. G. Koomey, S. Berard, M. Sanchez, and H. Wong, "Implications of Historical Trends in the Electrical Efficiency of Computing," *IEEE Annals of the History of Computing*, vol. 33, 2011.

[2] B. Agostini, M. Fabbri, J. E. Park, L. Wojtan, J. R. Thome, and B. Michel, "State of the Art of High Heat Flux Cooling Technologies," *Heat Transfer Engineering*, vol. 28, no. 4, 2007.

[3] P. Kogge, K. Bergman, S. Borkar, D. Campbell, W. Carson, W. Dally, M. Denneau, P. Franzon, W. Harrod, K. Hill, J. Hiller, M. Richards, A. Scarpelli, S. Scott, A. Snavely, T. Sterling, S. R. Williams, and K. Yelick, "ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems," Tech. Rep., 2008.

[4] S. Zimmermann, I. Meijer, M. K. Tiwari, S. Paredes, B. Michel, and D. Poulikakos, "Aquasar: A hot water cooled data center with direct energy reuse," *Energy*, vol. 43, 2012.

[5] C. D. Patel, "A vision of energy aware computing from chips to data centers," in *International Symposium on Micro-Mechanical Engineering*, 2003.

[6] Y. Fulpagare and A. Bhargav, "Advances in data center thermal management," *Renewable and Sustainable Energy Reviews*, 2015.

[7] T. Brunschwiler, B. Smith, E. Ruetsche, and B. Michel, "Toward zero-emission data centers through direct reuse of thermal energy," *IBM Journal of Research and Development*, vol. 53, 2009.

[8] M. Iyengar, M. David, P. Parida, V. Kamath, B. Kochuparambil, D. Graybill, M. Schultz, M. Gaynes, R. Simons, R. Schmidt, and T. Chainer, "Extreme energy efficiency using water cooled servers inside a chiller-less data center," in *InterSociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems*, 2012.

[9] M. K. Patterson and J. M. Walters, "On Energy Efficiency of Liquid Cooled HPC Datacenters," in *15th IEEE ITHERM Conference*, 2016.

[10] T. green grid, "White paper #70: Liquid cooling technology update," 2016.

[11] T. Brunschwiler, I. G. Meijer, S. Paredes, W. Escher, and B. Michel, "Direct Waste Heat Utilization from Liquid-cooled Supercomputers," in *Proceedings of the 14th International Heat Transfer Conference*, 2010.

[12] J. B. Marcinichen, J. A. Olivier, N. Lamaison, and R. John, "Advances in Electronics Cooling," vol. 7632, 2016.

[13] J. Choi, Y. Kim, A. Sivasubramaniam, J. Srebric, Q. Wang, and J. Lee, "A CFD-based tool for studying temperature in rack-mounted servers," *IEEE Transactions on Computers*, vol. 57, 2008.

[14] Z. Wang, C. Bash, N. Tolia, M. Marwah, X. Zhu, and P. Ranganathan, "Optimal fan speed control for thermal management of servers," in *ASME InterPACK*, 2009.

[15] L. Parolini, B. Sinopoli, B. H. Krogh, and Z. K. Wang, "A cyber-physical systems approach to data center modeling and control for energy efficiency," *Proceedings of the IEEE*, vol. 100, 2012.

[16] R. Lucchese, J. Olsson, A.-L. Ljung, W. Garcia-Gabin, and D. Varagnolo, "Energy savings in data centers: A framework for modelling and control of servers' cooling," in *IFAC World Congress*, 2017.

[17] B. A. Rubenstein, H. Packard, F. Collins, R. Zeighami, R. Lankston, and E. Peterson, "Hybrid cooled data center using above ambient liquid cooling," in *IEEE ITherm*, 2010.

[18] J. B. Marcinichen, J. A. Olivier, and J. R. Thome, "On-chip two-phase cooling of datacenters: Cooling system and energy recovery evaluation," *Applied Thermal Engineering*, vol. 41, 2012.

[19] S. J. Ovaska, R. E. Dragseth, and S. A. Hanssen, "Direct-to-chip liquid cooling for reducing power consumption in a subarctic supercomputer centre," *International Journal of High Performance Computing and Networking*, 2016.

[20] L. Li, W. Zheng, X. Wang, and X. Wang, "Coordinating Liquid and Free Air Cooling with Workload Allocation for Data Center Power Minimization," *11th International Conference on Autonomic Computing*, 2014.

[21] ——, "Placement optimization of liquid-cooled servers for power minimization in data centers," *2014 International Green Computing Conference*, 2015.

[22] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature-Aware Microarchitecture: Modeling and Implementation," *ACM Transactions on Architecture and Code Optimization*, vol. 1, 2004.

[23] T. Brunschwiler, B. Michel, H. Rothuizen, U. Kloter, B. Wunderle, H. Oppermann, and H. Reichl, "Interlayer cooling potential in vertically integrated packages," *Microsystem Technologies*, vol. 15, no. 1, 2009.

[24] A. K. Coskun, J. L. Ayala, D. Atienza, and T. S. Rosing, "Modeling and dynamic management of 3D multicore systems with liquid cooling," in *17th IFIP International Conference on Very Large Scale Integration*, 2011.

[25] A. Coşkun, J. Ayala, D. Atienza, and T. Rosing, "Thermal Modeling and Management of Liquid-Cooled 3D Stacked Architectures," in *VLSI-SoC: Technologies for Systems Integration*, 2011, vol. 360.

[26] F. Oldewurtel, A. Parisio, C. N. Jones, D. Gyalistras, M. Gwerder, V. Stauch, B. Lehmann, and M. Morari, "Use of model predictive control and weather forecasts for energy efficient building climate control," *Energy and Buildings*, vol. 45, 2012.

[27] D. Sturzenegger, D. Gyalistras, M. Morari, and R. S. Smith, "Model predictive climate control of a swiss office building: Implementation, results, and cost–benefit analysis," *IEEE Transactions on Control Systems Technology*, vol. 24, 2016.

[28] K. C. Furman and N. V. Sahinidis, "A Critical Review and Annotated Bibliography for Heat Exchanger Network Synthesis in the 20th Century," *Industrial & Engineering Chemistry Research*, vol. 41, 2002.

[29] C. A. Floudas, A. R. Ciric, and I. E. Grossmann, "Automatic synthesis of optimum heat exchanger network configurations," *AIChE Journal*, vol. 32, 1986.

[30] D. Copeland, M. Behnia, and W. Nakayama, "Manifold Microchannel Heat Sinks: Isothermal Analysis," *IEEE Transactions on components, packaging and manufacturing technology*, vol. 20, 1997.

[31] J. H. Ryu, D. H. Choi, and S. J. Kim, "Three-dimensional numerical optimization of a manifold microchannel heat sink," *International Journal of Heat and Mass Transfer*, 2003.

[32] T. Brunschwiler, H. Rothuizen, M. Fabbri, U. Kloter, B. Michel, R. J. Bezama, and G. Natarajan, "Direct liquid jet-impingement cooling with micronsized nozzle array and distributed return architecture," in *Thermomechanical Phenomena in Electronic Systems*, 2006.

[33] W. Escher, B. Michel, and D. Poulikakos, "A novel high performance, ultra thin heat sink for electronics," *International Journal of Heat and Fluid Flow*, vol. 31, 2010.

[34] W. Escher, T. Brunschwiler, B. Michel, and D. Poulikakos, "Experimental Investigation of an Ultrathin Manifold Microchannel Heat Sink for Liquid-Cooled Chips," *Journal of Heat Transfer*, vol. 132, 2010.

[35] R. Wälchli, T. Brunschwiler, B. Michel, and D. Poulikakos, "Combined local microchannel-scale CFD modeling and global chip scale network modeling for electronics cooling design," *International Journal of Heat and Mass Transfer*, vol. 53, 2010.

[36] P. Kasten, S. Zimmermann, M. K. Tiwari, B. Michel, and D. Poulikakos, "Hot water cooled heat sinks for efficient data center cooling: Towards electronic cooling with high exergetic utility," *Frontiers in Heat and Mass Transfer*, vol. 1, 2010.

[37] A. F. Al-Neama, N. Kapur, J. Summers, and H. M. Thompson, "An experimental and numerical investigation of the use of liquid flow in serpentine microchannels for microelectronics cooling," *Applied Thermal Engineering*, 2017.

[38] "Open Compute Project, http://opencompute.org/," 2017.

[39] "stress-ng - CPU and memory stressor."