Intent-based Decentralized Orchestration for Green Energy-aware Provisioning of Fog-native Workflows

Mays AL-Naday
School of Computer Science and Electronic Engineering
University of Essex
Colchester, UK
mfhaln@essex.ac.uk

Tom Goethals, Bruno Volckaert
IDLab, Department of Information Technology (intec)
Ghent University - imec
Gent, Belgium
tom.goethals@UGent.be, bruno.volckaert@UGent.be

***Abstract*—The cloud native paradigm is emerging as a pathway to developing applications for intrinsic operation on the cloud. This prompted application modularity, leveraging the adoption of the microservices architecture. Meanwhile, fog computing is emerging as a geo-dispersed cloud, bringing services closer to the end-user for localization and improved responsiveness. Transitioning to fog-native applications, i.e. managing microservice workflows over the fog, is a non-trivial challenge. On one hand, engineering workflows require awareness of the dependencies across microservices, as they impact the perceived quality of service. On the other hand, the heterogeneity of capacities, energy prices and supply, introduce challenges that can negate the sought advantages of the fog. This work proposes a novel algorithm based on Alternating Direction Method of Multipliers for intent-based workflow mapping and admission, *iADMM*. The performance of the algorithm is evaluated analytically and experimentally and compared to a baseline compute-network cost minimization alternative. Evaluation results show that iADMM achieves near optimal decisions in minimizing operational costs without violating workflow intents.**

***Index Terms*—cloud-native applications, fog networks, microservices, resource optimization, intent-based allocation, green fog ecosystem**

I. Introduction

The proliferation of cloud networks and services has prompted the cloud-native paradigm, referring to the development of applications for intrinsic operation on distributed and elastic resources. This has demanded greater modularity of applications, which incentivised larger adoption of the microservices architecture. So far, cloud-native applications run within central clouds, having largely homogeneous infrastructure of virtually unlimited capacity. However, the tight control and homogeneity have resulted in incorporating oversimplified assumptions about applications and resources. This ultimately restricted operations over multiple, heterogeneous, clouds [1].

Meanwhile, fog computing networks are rapidly growing as a geo-distributed and heterogeneous infrastructure [2], [3]. Their adoption is widening to meet stringent latency and privacy requirements, among others. The variant constraints on fog resources increases the likelihood of application *geo-dispersion*. That is to distribute application's components over different fog clusters and interconnect them to compose the application. Facilitating dispersed, yet intent-compliant, deployment of applications under current orchestration mechanisms is not straightforward. This is due to overlooking the heterogeneity and network impact outside cluster boundaries [4].

Specifically, the variation of compute-network capacities, energy prices and quality of energy supply across clusters introduce non-trivial trade-offs. On one hand, capacity constraints at the edge of the fog increase the likelihood of congestion, which ultimately impact processing and/or network latencies. On the other hand, variant energy prices and types of energy supply at different clusters vary the operational costs. Existing evidence shows that smaller data centers have higher energy prices [5], [6]. This translates into higher operational costs per application. Meanwhile, there is a globally growing need to transition cloud operations to using clean energy [7]. However, the inconsistent supply and high price of such energy are causing challenges for workload management that slowed the speed of transitioning [8]. Emerging indicators such as [9] show that purchase of green energy correlates strongly with the size of cloud providers. This suggest that large clouds are having the larger share of green energy.

The above creates multiple workload attraction forces, which impact compliance with application intents and provision costs. These forces correlate with each other across the hosting clusters, and create a ripple effect that impact overall application performance. Furthermore, there is need for efficient utilization of the sparse edge. This includes conserving small, "near-user", clusters for time-critical processes while pushing counterparts to cheaper clusters, within latency intents. To facilitate such intent-based orchestration, there is need for solutions that: 1) have knowledge of application intents; and 2) can act on the fog underlying heterogeneity.

This work proposes a novel optimization algorithm for solving the problem of intent-based mapping and admission of fog-native microservice applications (*workflows*). The algorithm incorporates awareness of latency and resource intents while optimizing operational costs, given: compute-network capacity, energy price and ratio of green energy supply. The solving approach is fully decentralized; allowing for significant flexibility and scalability. Hence, the contributions of this work are three-fold: 1) **model** the fog ecosystem and formulate the problem of intent-based workflow mapping and admission; 2) **develop** a decentralized algorithm for solving the problem, based on Alternating Direction Method of Multipliers

(ADMM); and, 3) **evaluate** the performance of the algorithm both analytically and experimentally, compared to a baseline for joint minimization of compute-network cost (MinCB).

The remainder of this paper is structured as follows: Section II reviews state of the art related work, Section III models the fog ecosystem, including interactions among clusters' orchestrators, and formulates the problem of intent-based fog-native workflow mapping and admission. Section IV describes the proposed intent-based ADMM (iADMM) algorithm, while Section V evaluates the performance of the algorithm; and finally, Section VI draws the conclusions.

II. Related work

Fog ecosystems are proliferating and have been investigated extensively [1], [3]. Increasing research effort in this realm are tackling problems of workflow and resource management. For example, the work of [10] tackles the problem of workflow offloading in mobile edge computing, given energy and cost awareness. They developed a multi-objective optimization that follow a computationally-expensive global solving method. Stavrinides et al. [11] consider the orchestration of real-time QoS-limited workflows. Instead of optimizing service selection, they allow trading accuracy for QoS.

Distributed algorithms are showing higher potential to support intent-based workflow orchestration. For example, the work of [12] proposes an ADMM algorithm that solves the service mapping and routing problems at the node-level. Although the solution is focused on unitary services, it provides an attractive baseline to a newer solution for fog-native microservice workflows. The work of [13] in Service Function Chaining (SFC) proposes a solution for SFC graph embedding based on a combination of Bender decomposition and ADMM. Their solution lacks consideration of latency and requires a centrally-calculated global solution, from local ones.

Orthogonally, several efforts started investigating the role of green energy in facilitating cloud services. For instance, a review by Malik et al. [14] includes recent studies of optimal use of green energy in fog computing for 6G-enabled IoT. The work of [15] proposes a Mixed Integer Linear Programming (MILP) approach to reduce CO2 emissions in fog-cloud architectures, showing a 71% CO2 reduction compared to resource placement in distributed clouds. The work of [16] focuses on request dispatching based on real-time availability of green energy, showing a significant improvement in service time, QoS violations, and green energy utilization simultaneously. Similar to earlier reviewed efforts, these solutions address the problem from different perspectives and while their solutions provide a strong baseline for the work proposed here, they fall short in incorporating critical parts of workflow intents.

III. The fog ecosystem model

Recently, the fog-native architecture has been proposed in [4] to enable intrinsic development and operations of workflows within a fog ecosystem. A workflow is defined as a set of microservices, selected and organized such that to satisfy the intents of application developers and users. This requires

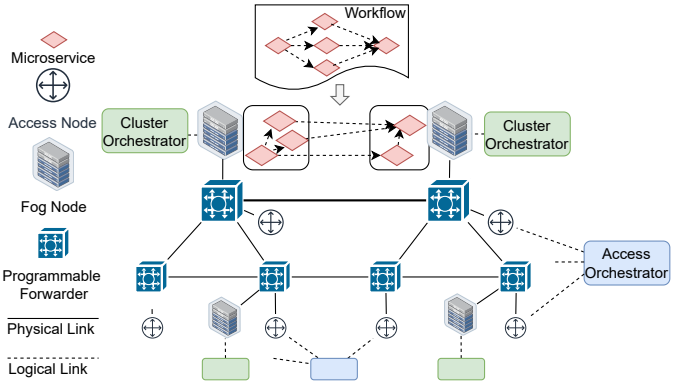


Fig. 1. Network view of the fog ecosystem showing fog and access orchestrators and the underlying programmable network

the two actors to define the desired behavior, i.e. *intents*, rather than deployment plans. The architecture includes a flexible management plane for intent-based composition of workflows. This work propose to extend the architecture of [4] with intent-based workflow orchestration. This section models the fog ecosystem and formulate the problem of intent-based workflow mapping and admission.

*a) **The network**:* A fog network (shown in Figure 1) is a modeled as a set of access nodes $\mathcal{U}$ and a set of fog nodes $\mathcal{N}$. Each $u \in \mathcal{U}, n \in \mathcal{N}$ are connected by a set of paths $\mathcal{P}_u$, i.e. u is multi-homed by $\mathcal{N}$. Each $n \in \mathcal{N}$ is typically characterized by a wide set of metrics, this work focuses on four: the computing capacity, c_n in number of CPU cores; the average computation speed of a node, μ_n in GHz (i.e. equivalent to clock cycles/s); the energy price at the node site, e_n in Penny per milliCPU (mCPU); and the ratio of green energy supply, η_n, relative to the total energy consumption of the node. Hence, the tuple $\langle c_n, \mu_n, e_n, \eta_n \rangle$ characterize each $n \in \mathcal{N}$. Similarly, each path $p \in \mathcal{P}_u$ can be described by a variety of metrics. Here, the focus is on: bandwidth capacity, b_p in Gb/s and the metric distance, l_p in km. Hence, the tuple $\langle b_p, l_p \rangle$ describes each $p \in \mathcal{P}_u$. Notably, energy cost in data transmission is neglected in this work due to space limitation and since existing evidence [17], [18] reveals that network energy consumption is a marginal $3 - 8\%$ of total energy consumption in cloud and Internet systems.

*b) **Microservices & workflows**:* Application developers advertise a set of microservices, $\mathcal{S}$ in the fog ecosystem. Each $s \in \mathcal{S}$ can be characterized by functional and non-functional intents. Functional intents are outside the scope of this work. They are primarily considered by the workflow manager of [4] when constructing an intent matching workflow, $w(\mathcal{S}^w, \mathcal{R}^w)$. The latter is a subset of microservices, $\mathcal{S}^w \subseteq \mathcal{S}, |\mathcal{S}^w| = m$, connected by $\mathcal{R}^w$ logical relationships in a dependency graph. A user sends their request to the first $s_0 \in \mathcal{S}^w$ while expecting a response back from the last microservice $s_m \in \mathcal{S}^w$, after execution of the other microservices of $\mathcal{S}^w$. $\mathcal{R}^w$ can be organized as a set of chains $\mathcal{I}^w$ that connect s_0 to s_m. Once constructed, the non-functional intents of w become the primary focus and

they can be specified on a microservice and workflow level. A non-functional intent describes a requirement rather than function of a microservice.

*c) **Intents**:* Here, the focus is on non-functional intents. Application developers specify the resource and QoS requirements, along with the input/output model of each microservice. Resource requirements include: the average task size of a microservice, c^s in mCPU; and the average input and output data size, q^s and r^s respectively. The input/out model assumes the completion of $\mathcal{S}^w_{k-1}$ microservices to trigger the start of s_k and independent retrieval of input data from a data store. The latter can be embedded as microservices in the workflow such that at least one $s'_k \in \mathcal{S}^w_{k-1}$ is the data store for s_k, providing $q^{s_k} = r^{s'_k}$. QoS requirements can be wide. Here they are narrowed down to the total latency of a workflow, τ^w in milliseconds (msec), being the elapsed time between a user sending their request to s_0 and receiving a response back from s_m. Thus, workflow intents can be formulated as a set of tuples $\{\langle c^s, q^s, r^s\rangle \mid \forall s \in \mathcal{S}^w\}$ and $\langle \tau^w \rangle$.

*d) **Demand and decision variables**:* Each access node $u \in \mathcal{U}$ generate requests for workflow w at a rate of α^w_u. Notably, $\alpha^w_u \equiv \alpha^s_u,\ \forall s \in \mathcal{S}^w$. accordingly the workload and traffic demands offered to the fog network can be defined as:

$$\delta^s_u = \alpha^s_u c^s,\ \omega^s_u = \alpha^s_u r^s,\ \forall u \in \mathcal{U} \tag{1}$$

Where δ^s_u and ω^s_u are the workload and traffic demand per-microservice in a workflow, respectively. Notice that the traffic equation in (1) can be modified on a per-workflow basis to account for the largest of $(q^s, r^s),\ \forall s \in \mathcal{S}^w$. Here, r^s is assumed the largest. An access node u decides $\beta^s_{un} \in [0,1]$ fraction of requests for $s \in \mathcal{S}^w$ to be *mapped* to fog node n. Equivalently, n fog node decides $\gamma^s_{un} \in [0,1]$ the fraction of requests received by u to be *admitted* for processing.

*e) **Operational cost**:* Two forms of costs are incurred in such a fog ecosystem: first, the cost of processing admitted requests for s at fog node n, θ^s_n. This is driven by the structure of energy supply in terms of price and greenness, along with the task size c^s. Incorporating c^s in the admission cost allows a fog node to tune the attraction of microservices based the effect of their size on the node's welfare. The second cost is that of data transmission from either n to u, θ^s_{un}; or from n to another fog node $m \in \mathcal{N}$, θ^s_{nm}. The transmission cost is driven by the input/output data size of s, bandwidth capacity on the path and the length of the path. Given that inter-fog node paths generally have higher capacity than access-fog paths $\mathcal{P}_u$, we assume $\theta^s_{nm} < \theta^s_{un}$. For simplicity, we optimize for θ^s_{un} given $\mathcal{P}_u$ then utilize p_{nm} direct paths instead.

*f) **Workflow latency**:* The observable latency of a workflow w can be defined as:

$$\tau^{\varnothing(w)} \quad = \quad \sum_{s \in \varnothing(w)} \tau^s \tag{2}$$

where $\varnothing(w)$ is the diameter of w, specifying the longest chain in $\mathcal{I}^w$. Length is defined by the combination of: the number of microservices in the chain, $|\varnothing(w)|$; the task size c^s of each $s \in \varnothing(w)$; and, the output data size r^s. τ^s is the latency incurred by a single microservice $s \in \varnothing(w)$, including the processing latency along with the output data transmission and propagation latency. The processing latency τ^s_n is dependent on the service rate of s at n, defined as μ_n/c^s requests/s. Now, n can be modeled as an M/M/1 queue, given Little's law [19] and assuming balanced distribution of load across all CPUs:

$$\tau^s_n = \frac{c^s c_n}{c_n \mu_n - c^s \sum_{u \in \mathcal{U}} \beta^s_{un} \delta^s_u} \tag{3}$$

The transmission latency τ^s_{un} is derived by the forwarding rate of the underlying network on path $p_{un} \in \mathcal{P}_u$. Notably, p_{un} is expected to traverse a series of forwarders, each can be modeled as a M/M/1 queue and collectively they form a tandem queue. Assuming the number of forwarders on p_{un} is known, h_p, and all forwarders on p_{un} has the same average service rate equal to b_p/r^s the transmission delay can be formulated as:

$$\tau^s_{un} = \frac{h_p r^s}{b_p - \gamma^s_{un} \omega^s_u} \tag{4}$$

The propagation latency is dependent on the link medium and metric length, here we assume an underlying network is optical and hence the proposition latency is $\tau_p = l_p/(2 \times 10^8)$.

A. Problem formulation

Given the above, the problem of intent-based workflow mapping and admission can be formulated as a constrained minimization of two costs: provision of compute resources of microservices running on fog nodes, and provision of bandwidth resources between access and fog nodes. Recall that for simplicity, we pose the problem and solve it for access-fog paths. Once an optimal solution is found, we substitute access-fog paths with the direct and cheaper inter-fog node alternatives, by applying principles of trigonometry. Mathematically the problem can be defined as:

$$\min_{\beta,\gamma} \sum_{s \in \mathcal{S}^w} \sum_{u \in \mathcal{U}} \sum_{n \in \mathcal{N}} \delta^s_u \theta^s_n \beta^s_{un} + \omega^s_u \theta^s_{un} \gamma^s_{un} \tag{5}$$

Subject to:

$$C_\tau: \quad \sum_{s \in i} \tau^s_n + \tau^s_{un} \leqslant \tau^{\varnothing(w)}, \tau^{\varnothing(w)} \leqslant \tau^w\ \forall i \in \mathcal{I}^w \tag{6}$$

$$C_c: \quad \sum_{s \in \mathcal{S}^w} \sum_{u \in \mathcal{U}} \delta^s_u \beta^s_{un} \leqslant c_n,\ \forall n \in \mathcal{N} \tag{7}$$

$$C_b: \quad \sum_{s \in \mathcal{S}^w} \omega^s_u \gamma^s_{un} \leqslant b_p,\ \forall u \in \mathcal{U}, n \in \mathcal{N}, p \in \mathcal{P}_u \tag{8}$$

$$C_a: \quad \sum_{n \in \mathcal{N}} \gamma^s_{un} = 1,\ \forall s \in \mathcal{S}^w, u \in \mathcal{U} \tag{9}$$

$$C_e: \quad \beta^s_{un} = \gamma^s_{un},\ \forall s \in \mathcal{S}^w, u \in \mathcal{U}, n \in \mathcal{N} \tag{10}$$

$$C_v: \quad \beta^s_{un}, \gamma^s_{un} \geqslant 0 \tag{11}$$

C_τ constraint of (6) ensures the latency observed on any chain $i \in \mathcal{I}^w$ does not exceed the latency on $\varnothing(w)$. The latter, in turn, does not exceed the response time intent τ^w. C_c constraint of (7) ensures that the CPU resources reserved at any fog node for the workflow demand from all users do not exceed the CPU capacity of the node. C_b constraint of (8) ensures that the traffic incurred by mapping u's demand to n

does not exceed the bandwidth capacity of $p_{un} \in \mathcal{P}_u$ path. C_a constraint of (9) ensures that all demand of an access node is allocated, C_e and C_v constraints ensure that: 1) consensus is reached on the fraction of demand mapped from u to n, and admitted by n; and 2) all decision variables are positive.

The problem space for multiple workflows $\mathcal{W}$ is the size of: $|\mathcal{W}| \times |\mathcal{S}^w| \times |\mathcal{U}| \times |\mathcal{N}|$ with a number of variables easily in excess of 10^6 for a small size network of no more than 10 nodes. This renders global solving methods highly expensive, if at all feasible. Hence, in the next section we propose a scalable approach to solving the problem by decomposing it into per-node sub-problems; each of which is solved locally.

IV. Decentralized solution

This section describes the proposed intent-based ADMM algorithm, iADMM, for solving the problem of (5) in a scalable manner. To achieve the latter, we decompose the problem into two-parts, solve them individually and then combine the solutions iteratively until convergence is reached.

A. Decomposing the latency constraint

C_τ of (6) couples all variables of the problem, which would have otherwise been easily decomposed. To address this, we propose to decompose C_τ into the elementary components: τ^s_n and τ^s_{un}, and associate them with the respective problem part. Each element is then calculated by the respective orchestrator at u or n, and shared with others to calculate the workflow latency. Notice that the left hand side (LHS) of C_τ has two-levels coupling: a) on microservice-level, between processing and transmission latency of a microservice, τ^s_n and τ^s_{un} respectively; and b) on a chain-level across, multiple microservices of a chain $i \in \mathcal{I}^w$. To decouple the chain-level latency, we calculate a weight factor, χ^s, of each microservice relative to the workflow diameter $\varnothing(w)$:

$$\chi^s = \frac{c^s + r^s}{\varnothing(w)}, \ \forall s \in \mathcal{S}^w \tag{12}$$

This allows for setting up an upper bound on the latency exhibited by any microservice, irrespective of others. Specifically, if $\tau^{\varnothing(w)} \leqslant \tau^w$ holds then $\tau^s_n + \tau^s_{un} \leqslant \chi^s \tau^w, \forall s \in \mathcal{S}^w$ holds too. This allows for substituting C_τ with the simpler C^s_τ: a microservice-level constraint that can be formulated as:

$$C^s_\tau : \tau^s_n + \tau^s_{un} \leqslant \chi^s \tau^w \tag{13}$$

Now, C^s_τ still couples the problems at microservice level, which poses dependency on sharing state information across fog and access orchestrators. To address this, we propose to decompose the LHS of C^s_τ into the two terms and let each orchestrator calculate its' respective term for the variable of its problem. Then each share the result with the other so that both calculate τ^s. This allows for transforming C^s_τ to $C^{s,\beta}_\tau$ and $C^{s,\gamma}_\tau$, which can be defined as:

$$C^{s,\beta}_\tau : \quad \frac{c^s c_n}{c_n \mu_n - c^s \sum_{u \in \mathcal{U}} \beta^s_{un} \delta^s_u} + \tau^s_{un} \leqslant \chi^s \tau^w \tag{14}$$

$$C^{s,\gamma}_\tau : \quad \tau^s_n + \frac{h_p r^s}{b_p - \gamma^s_{un} \omega^s_u} \leqslant \chi^s \tau^w \tag{15}$$

Now, the problem can be decomposed into two parts as described next.

B. Problem decomposition and solution

First, the augmented Lagrangian of the problem is formulated as:

$$\begin{aligned} \mathcal{L}(\beta, \gamma, \lambda) = \sum_{s \in \mathcal{S}^w} \sum_{n \in \mathcal{N}} \sum_{u \in \mathcal{U}} \delta^s_u \theta^s_n \beta^s_{un} + \omega^s_u \theta^s_{un} \gamma^s_{un} + \\ \lambda^s_{un}(\beta^s_{un} - \gamma^s_{un}) + \frac{\rho}{2}(\beta^s_{un} - \gamma^s_{un})^2 \end{aligned} \tag{16}$$

Where $\rho > 0$ is the penalty parameter of $\mathcal{L}$. Notably, the augmented Lagrangian is strictly convex, irrespective of the original form of the problem. The mapping part of (16) is posed as a per-access-node set of sub-problems in the form:

$$\min_{\gamma^w_u} \sum_{s \in \mathcal{S}^w} \sum_{n \in \mathcal{N}} \gamma^s_{un} \Big(\omega^s_u \theta^s_{un} - \lambda^s_{un} + \frac{\rho}{2}(\gamma^s_{un} - 2\beta^s_{un}) \Big) \tag{17}$$

Subject to: $C^{s,\gamma}_\tau$ of (15), C_b of (8), C_e of (10) and C_v of (11). Equivalently, the admission problem term of (16) can be posed as a per-fog-node set of sub-problems in the form:

$$\min_{\beta^w_n} \sum_{s \in \mathcal{S}^w} \sum_{u \in \mathcal{U}} \beta^s_{un} \Big(\delta^s_u \theta^s_n + \lambda^s_{un} + \frac{\rho}{2}(\beta^s_{un} - 2\gamma^s_{un}) \Big) \tag{18}$$

Subject to: $C^{s,\beta}_\tau$ of (14), C_c of (7), C_e of (10) and C_v of (11). Recall that C_e links the two problems so that consensus is reached on the volume of served demand; consequently giving one solution of the problem. Now, we apply an ADMM method [20] to solve the problems of (18) and (17) iteratively.

At any iteration t, each cluster orchestrator uses the previous solution of the mapping problem ($\gamma^{s,t-1}_{un}$) together with $\lambda^{s,t-1}_{un}$ to solve its instance of (18) and share the solution, $\beta^{s,t}_{un}$, with access orchestrators. Each of the latter uses $\beta^{s,t}_{un}$ to solve its instance of (17) and obtain $\gamma^{s,t}_{un}$. The two solutions are then used to calculate the dual variable $\lambda^{s,t}_{un}$. The iterations continue until the residual parameters of the problem are below the primary and dual error gaps, i.e. $s_{pri} \leqslant \epsilon_{pri}, s_{dual} \leqslant \epsilon_{dual}$.

Notice that τ^s_{un} is calculated using $\beta^{s,t}_{un}$ rather than $\gamma^{s,t}_{un}$ to avoid churning the latency values by the error gap between the two solutions, which would hinder the convergence of the algorithm. The iADMM algorithm is described in Algorithm (1). To expedite convergence without compromising performance, we apply the penalty variation technique proposed by [20] using: $\zeta > 1$, $\kappa_{inc} > 1$ and $\kappa_{dec} > 1$ as the multiplicative parameters.

C. Complexity Analysis

The computation complexity is comprised of the calculations of: 1) the latency terms for each microservice of the workflow, $\mathcal{O}(\tau^w) = |\mathcal{S}^w| \mathcal{O}(\tau^s)$; 2) the mapping decision, $\mathcal{O}(\gamma^s_{un})$; and 3) the admission decision, $\mathcal{O}(\beta^s_{un})$. The complexity in calculating the latency terms is: $\mathcal{O}(\tau^w) = |\mathcal{S}^w| \mathcal{O}(N+1)$. The mapping decision complexity, without considerations of parallel computations, is $\mathcal{O}(\gamma^w_u) = |\mathcal{S}^w| t\big(\mathcal{O}(N^3) + 3\mathcal{O}(N^2)\big)$. The first term is the complexity of decomposing the matrix, while the second term corresponds to solving operations of the system of equations. The admission decision complexity has

Algorithm 1 iADMM

Given: $\epsilon_{pri}, \epsilon_{dual}, \rho$
Initialize: $t \leftarrow 0$, $\beta_{un}^{s,t}, \gamma_{un}^{s,t}, \lambda_{un}^{s,t} \leftarrow \{0 | \forall s \in \mathcal{S}^w, n \in \mathcal{N}, u \in \mathcal{U}\}$
Each $n \in \mathcal{N}$ calculates τ_n^s and advertise θ_n^s, τ_n^s to the access orchestrators.
Each $u \in \mathcal{U}$ calculates τ_{un}^s and advertise $\theta_{un}^s, \tau_{un}^s$ to the fog orchestrators.
while $s_{pri} > \epsilon_{pri}$ OR $s_{dual} > \epsilon_{dual}$ **do**
    Each $n \in \mathcal{N}$: 1) solves the admission problem of (18) to obtain $\beta_{un}^{s,t+1}$; 2) calculates new τ_n^s based on $\beta_{un}^{s,t+1}$; and 3) advertise $\beta_{un}^{s,t+1}$ and τ_n^s to access orchestrators
    Each $u \in \mathcal{U}$: 1) calculates τ_{un}^s using $\beta_{un}^{s,t+1}$; 2) solves the mapping problem of (17) using $\beta_{un}^{s,t+1}$; 3) calculates the new value of the dual variable, $\lambda_{un}^{s,t+1}$; and 4) publish $\tau_{un}^s, \gamma_{un}^{s,t+1}$ and $\lambda_{un}^{s,t+1}$ to fog orchestrators.
    Both access and fog orchestrators use τ_n^s and τ_{un}^s to calculate new τ^s value.
    Fog orchestrators calculate s_{pri} and s_{dual}
    Update ρ^{t+1}
    $t \leftarrow t + 1$
end while
Each $n \in \mathcal{N}$ publishes the converged solution $\hat{\beta_{un}^w}$ to each access orchestrator $u \in \mathcal{U}$.

a similar complexity of $\mathcal{O}(\beta_n^w) = |\mathcal{S}^w| t\big(\mathcal{O}(U^3) + 3\mathcal{O}(U^2)\big)$. On a 64-bit server with CPU of 1.70GHz and L2 cache of 1024K; the algorithm's run takes $\approx 10 - 20 msec$. This is for: a number of iterations $t = 20$; $|\mathcal{N}|, |\mathcal{U}| = 11$ and $|\mathcal{S}^w| = 5$.

V. Evaluation

This section evaluates the proposed iADMM algorithm, and compares it to a baseline greedy algorithm for joint CPU and bandwidth cost minimization (MinCB). The baseline does not consider the latency intent and allocates microservices independently from each other in a workflow. The evaluation is conducted both analytically and experimentally. The former is realized using simmer Discrete Event Simulator[1] for the Abilene topology(11 nodes, 28 links) [21], as an example network of the ecosystem. The experimental evaluation is realized using the IDlab Virtual Wall[2] for a tiered-mesh topology of (9 nodes, 11 links). A summary of the analytical and experiment settings along with the source code for the experimental setup are made available on GitHub[3]. Four key performance indicators are analyzed: the ***Latency Residual Budget (LRB)***, an indicator of compliance with the latency intent; the ***CPU Utilization*** per fog node [workload/capacity], an indicator of workload distribution when meeting resource intents; ***Bandwidth Utilization*** per network path [traffic/capacity], an indicator of traffic distribution; and ***Workload Greenness*** an indicator of the fraction of workload executed using green energy.

[1]https://r-simmer.org/

[2]https://www.ugent.be/ea/idlab/en/research/research-infrastructure/virtual-wall.htm

[3]https://github.com/togoetha/IntentFogWorkflows

A. Analytical Evaluation

Here, performance is evaluated for 25 workflows with frequency of requests that follows a Zipf distribution, with an exponent factor of 0.8. This is inline with existing models of applications' workload in realistic cloud networks [22]. Within the network, each access node presents requests from a group of users between 500 and 2000. A set of fog nodes of different tiers are colocated with a subset of forwards. The tier is indicative of the node structure and attributes: 1) *Tier-1* nodes are large cloud data-centers; 2) *Tier-2* nodes are cloudlet mini/micro-data-centers of larger number; and, 3) *Tier-3* nodes are nano-data-centers of the largest number. Detailed settings are provided in the GitHub repository.

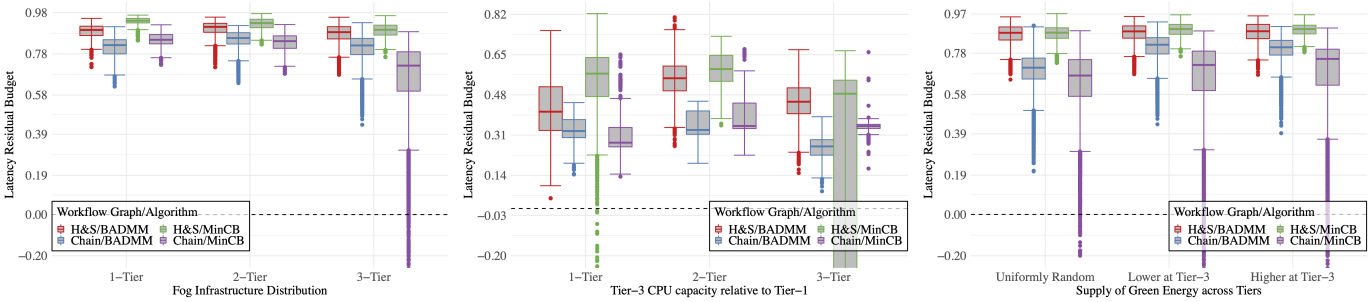
Two scenarios have been investigated: first, the fraction of green energy supply is assumed higher in tier-1 data centers compared to tier-3 edge. In the second scenario, the trade-off between green energy supply and energy prices is analyzed when: 1) supply of green energy is uniformly random and energy prices are the same across fog nodes; 2) supply of green energy and energy price are highest at tier-3 and lowest at tier-1; and, 3) supply of green energy is highest at tier-1 while energy prices are at their lowest. The first setting is a baseline of the influence of energy supply in the overall placement cost, while the second setting corresponds to use-cases where edge data-centers benefit from greater access to green energy yet energy prices remains higher. The third setting corresponds to existing ecosystems where energy prices are lower for tier-1 clouds and they have higher supply of green energy.

B. Experimental Evaluation

Experimental evaluations are run on fog nodes of specifications provided in GitHub. A unit of processing work (mCPU) is defined as the CPU time required to bubble sort 1000 integers, equal to $6/5$ physical mCPU on a fog node. A tiered architecture is simulated by limiting the total CPU budget of all microservices on each node. Each node runs Ubuntu 20.04, and microservices are deployed and managed by a customized version of FLEDGE [23]. The latter provides containerd-based container deployment and networking.

Three REST microservices are used in each topology: 1) A generator (i.e. user) creates requests for a workflow at a configurable rate and with variable payload size. The frequency of requests follows a normal distribution, with the mean and standard deviation determined by configuration; 2) five processors (*microservices*), each execute 1 mCPU per request and forward the result to the next microservice(s) in the workflow; 3) A sink is the terminator for a workflow, receiving requests and logging them with latency metadata.

To ensure statistically valid results, each workflow topology is run with a variable amount of total requests, so that all users are simultaneously generating load for at least 20 seconds. For practical reasons, the experimental evaluation is performed with only 3 users, each starting an average of 40 requests per second. Notably, the choice of Golang REST services imposes a significant communication overhead due to JSON (de)serialization, and in some cases tier-3 nodes require up to



(a) Infrastructure distribution - analytical (b) Infrastructure distribution - experimental (c) Supply of green energy - analytical

Fig. 2. The latency residual budget, shown for both analytical and experimental evaluations when varying the infrastructure distribution and the supply of green energy per node.

10% extra CPU (i.e. 0.13 physical CPU core) to avoid crashing overloaded services; both effects are considered in the results.

C. Satisfying latency intents

This section evaluates the LRB per workflow, both analytically and experimentally. The results are shown in Figure 2, where a dashed line is plotted to illustrate the violation threshold. That is: a negative LRB implies the response time is higher than the latnecy intent τ^w. Figures 2a and 2b show the LRB in analytical and experimental setups, when extending the distribution of fog infrastructure from a 1-Tier cloud to a 3-Tier cloud-to-edge. Both results show the LRB increases (i.e. response time decreases) in a 2-Tier setting compared to 1-Tier and 3-Tier fog. In the latter, LRB drops significantly with MinCB violating the threshold for a large number of requests. iADMM, while generally having a lower LRB than MinCB, it does not violate the threshold at any of the settings. The awareness of response time intents and microservice dependency allows iADMM not only to deploy workflows on less constrained nodes, but also to balance the workload over multiple Points of Presence (PoPs) of a microservice. Furthermore, when τ^w permits, iADMM allocates demand further away from the edge. This introduces network latency that contributes to the lower LRB, but it spares CPU capacity on constrained fog nodes. This reduces the likelihood of creating black holes in the ecosystem where processing latency violates the threshold, as observed for MinCB.

Figure 2c illustrate the analytical evaluation of LRB when varying the supply of green energy per fog node. The results confirm those shown in the first two figures. The LRB is lowest for iADMM when the fraction of green energy supply is uniformly random and the energy price is fixed across tiers. This results in a dominating effect of the capacity weight in the mapping and admission costs. Under these conditions, iADMM spreads the workflow wider in the network, which results in lower LRB instigated by network latency.

D. Fog CPU and Bandwidth Capacities

Figures 3a and 3b show the CPU and bandwidth utilization, respectively as a fraction of capacity. Due to space limits, the former is shown in experimental settings while the latter is shown in the analytical alternative. Figure 3a show the results when moving from a homogeneous cloud to a heterogeneous fog. iADMM maintains low CPU utilization per node even in a 3-Tier setting, while utilization under MinCB increases exponentially. This is caused by the tendency of iADMM to distribute the workload among all available PoPs, resulting in a lower overall CPU use when additional tiers are available. MinCB on the other hand, tends to cluster all services of a workflow on as few nodes as possible. Practically, the violations of MinCB are caused by a marginal increase in cgroup limits to avoid crashing microservices on tier-3 nodes.

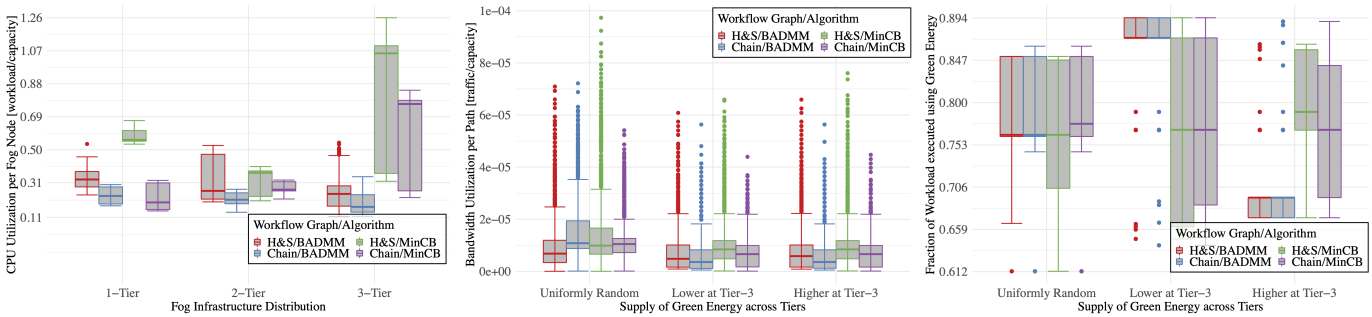
Figure 3b show the results when varying the supply of green energy. When energy prices are fixed and supply of green energy is random, bandwidth utilization is the highest overall. This is because the quality of energy supply is dominant in the cost of admission, causing the allocation to spread randomly that leads to higher bandwidth usage. When the supply of green energy is lower or higher at tier-3, iADMM bandwidth utilization does not change much. This is because of the added effect of energy prices in the cost of admission, being the lowest at tier-1. This raises the trade-off between energy supply, price and latency, ultimately resulting in no change to bandwidth utilization when only the energy supply is varied.

E. Workload Greenness

This section analytically evaluates the fraction of workload served using green energy, with the results shown in Figure 3c. When the supply of green energy is random across nodes, H&S workflows have a wider variation in workload greenness than their Chain counterparts. This shows a larger number of H&S workflows deployed over tier-2 and tier-3 nodes. However, when the green supply is better at tier-3, workload greenness under iADMM drops to $\approx 0.66-0.7$. This indicates a continued high execution at tier-1, with a marginal increase in using tier-2 nodes. This is because of the lower energy prices at tier-1, outweighing the attractiveness of green energy supply.

VI. Conclusions

This work proposed a novel intent-based Alternating Direction Method of Multipliers algorithm (iADMM), for solving the problem of workflow mapping and admission. The algorithm presents a decentralized approach, which allows for



(a) CPU: Infrastructure Distribution-Experimental (b) Bandwidth: green energy - analytical (c) Greenness: Supply of green energy - analytical

Fig. 3. CPU and bandwidth capacity utilization, and workload greenness. CPU utilization is shown for the experimental evaluation. The bandwidth utilization and workload greenness are shown for the analytical evaluation when varying the supply of green energy per fog node.

significant flexibility and scalability in workload management. The performance of the algorithm has been evaluated analytically and experimentally, and compared to a joint CPU and bandwidth cost minimization alternative. Evaluation results illustrated the superiority of iADMM in deploying workflows with minimum joint cost and without violating their intents. Furthermore, evaluation results revealed non-trivial trade-offs between workflow intents, the heterogeneity of energy price and supply of green energy. This illustrated the need to accompany greater supply of green energy with a drop in energy price, to increase workload greenness. Additionally, higher attention is needed to support the edge with cheaper green energy, to reduce the cost of time-critical workflows.

References

[1] O. Tomarchio, D. Calcaterra, and G. D. Modica, "Cloud resource orchestration in the multi-cloud landscape: a systematic review of existing frameworks," *Journal of Cloud Computing*, vol. 9, no. 1, p. 49, 2020. [Online]. Available: https://doi.org/10.1186/s13677-020-00194-7

[2] J. Singh, P. Singh, and S. S. Gill, "Fog computing: A taxonomy, systematic review, current trends and research challenges," *Journal of Parallel and Distributed Computing*, vol. 157, pp. 56–85, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0743731521001349

[3] R. K. Naha, S. Garg, D. Georgakopoulos, P. P. Jayaraman, L. Gao, Y. Xiang, and R. Ranjan, "Fog computing: Survey of trends, architectures, requirements, and research directions," *IEEE Access*, vol. 6, pp. 47 980–48 009, 2018.

[4] M. Sebrechts, B. Volckaert, F. De Turck, K. Yangy, and M. AL-Naday, "Fog native architecture: Intent-based workflows to take cloud native towards the edge," *IEEE Communications Magazine*, pp. 1–7, 2022.

[5] A. M, B. P, and C. L, "Trends in data centre energy consumption under the european code of conduct for data centre energy efficiency," *ENERGIES*, vol. 10, no. 10, p. 1470, 2017.

[6] Y. Sverdlik, "What is the Data Center Cost of 1kW of IT Capacity?" https://www.datacenterknowledge.com/archives/2016/08/23/what-is-the-data-center-cost-of-1kw-of-it-capacity, Aug 2016, [Online; accessed 12-March-2022].

[7] C. Koronen, M. Åhman, and L. J. Nilsson, "Data centres in future european energy systems—energy efficiency, integration and policy," *Energy Efficiency*, vol. 13, no. 1, pp. 129–144, 2020.

[8] X. Hu, P. Li, and Y. Sun, "Minimizing energy cost for green data center by exploring heterogeneous energy resource," *Journal of Modern Power Systems and Clean Energy*, vol. 9, no. 1, pp. 148–159, 2021.

[9] SKhynix, "Beyond Clean Energy: Technology Must Help Address Data Center Challenges," https://news.skhynix.com/hed-beyond-clean-energy-technology-must-help-address-data-center-challenges/, Dec 2021, [Online; accessed 08-July-2022].

[10] K. Peng, M. Zhu, Y. Zhang, L. Liu, J. Zhang, V. C. M. Leung, and L. Zheng, "An energy- and cost-aware computation offloading method for workflow applications in mobile edge computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, p. 207, 2019. [Online]. Available: https://doi.org/10.1186/s13638-019-1526-x

[11] G. L. Stavrinides and H. D. Karatza, "Orchestrating real-time IoT workflows in a fog computing environment utilizing partial computations with end-to-end error propagation," *Cluster Computing*, vol. 24, no. 4, pp. 3629–3650, jul 2021.

[12] H. Xu and B. Li, "Joint request mapping and response routing for geo-distributed cloud services," in *2013 Proceedings IEEE INFOCOM*, Apr 2013, pp. 854–862.

[13] Y. Yu, X. Bu, K. Yang, H. K. Nguyen, and Z. Han, "Network function virtualization resource allocation based on joint benders decomposition and admm," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 1706–1718, 2020.

[14] U. M. Malik, M. A. Javed, S. Zeadally, and S. u. Islam, "Energy efficient fog computing for 6g enabled massive iot: Recent trends and future opportunities," *IEEE Internet of Things Journal*, pp. 1–1, 2021.

[15] M. Aldossary and H. A. Alharbi, "Towards a green approach for minimizing carbon emissions in fog-cloud architecture," *IEEE Access*, vol. 9, pp. 131 720–131 732, 2021.

[16] A. Karimiafshar, M. R. Hashemi, M. R. Heidarpour, and A. N. Toosi, "A request dispatching method for efficient use of renewable energy in fog computing environments," *Future Generation Computer Systems*, vol. 114, pp. 631–646, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X19327979

[17] E. Innovation, "How Much Energy Do Data Centers Really Use?" https://energyinnovation.org/2020/03/17/how-much-energy-do-data-centers-really-use/, March 2020, [Online; accessed 21-June-2022].

[18] L. Altamira, J. Viegand, D. Polverini, B. Huang, and S. Flucker, "The role of data centres in reducing energy consumption through policy measures," vol. 2019-June, 2019, pp. 1581–1591. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85085203856partnerID=40md5=3718881d8893fa4c958f54fc025ea57e

[19] J. L. Gustafson, *Little's Law*. Boston, MA: Springer US, 2011, pp. 1038–1041.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, p. 1–122, jan 2011. [Online]. Available: https://doi.org/10.1561/2200000016

[21] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The Internet Topology Zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, Oct 2011.

[22] S. Di, D. Kondo, and F. Cappello, "Characterizing cloud applications on a google data center," in *2013 42nd International Conference on Parallel Processing*, 2013, pp. 468–473.

[23] T. Goethals, F. De Turck, and B. Volckaert, "Fledge: Kubernetes compatible container orchestration on low-resource edge devices," in *Internet of Vehicles. Technologies and Services Toward Smart Cities*, C.-H. Hsu, S. Kallel, K.-C. Lan, and Z. Zheng, Eds. Cham: Springer International Publishing, 2020, pp. 174–189.