# E3C3: Enhancing Energy Efficiency among Communication, Computation and Caching with QoI-Guarantee

Faheem Zafari[1,*], Jian Li[2,*], Kin K. Leung[3], Don Towsley[4] and Ananthram Swami[5]
[1,3]Imperial College London, [2,4]University of Massachusetts Amherst, [5]U.S. Army Research Laboratory
[1,3]{faheem16, kin.leung}@imperial.ac.uk, [2,4]{jianli, towsley}@cs.umass.edu, [5]ananthram.swami.civ@mail.mil
*Co-primary authors

***Abstract*—Energy efficiency is a fundamental requirement of modern data communication systems, and its importance is reflected in much recent work on performance analysis of system energy consumption. However, most works have only focused on communication and computation costs, but do not account for caching costs. Given the increasing interest in cache networks, this is a serious deficiency. In this paper, we consider the energy consumption trade-off between communication, computation, and caching (C3) under a Quality of Information (QoI) guarantee in a communication network. To attain this goal, we formulate an optimization problem to capture the C3 costs, which turns out to be a non-convex Mixed Integer Non-Linear Programming (MINLP) Problem. We then propose a variant of spatial branch and bound algorithm (V-SBB), that can achieve $\epsilon$-global [1] optimal solution to the original MINLP. We show numerically that V-SBB is more stable and robust than other candidate MINLP solvers under different network scenarios. More importantly, we observe that the energy efficiency under our C3 optimization framework improves by as much as 88% compared to any C2 optimization between communication and computation or caching.**

## I. Introduction

The rapid growth of smart environments, and advent of Internet of Things (IoT) have led to the generation of large amounts of data. However, it is a daunting task to transmit enormous data through traditional networks due to limited bandwidth and energy limitations [1]. These data need to be efficiently compressed, transmitted, and cached to satisfy the Quality of Information (QoI) required by end users. In fact, many wireless components operate on limited battery power supply and are usually deployed in remote or inaccessible areas, which necessitates the need for designs that can enhance the energy efficiency of the system with a QoI guarantee.

A particular example of modern systems that require high energy efficiency is the wireless sensor network (WSN). Consider a WSN with various types of sensors, which can generate enormous amount of data to serve end users. On one hand,

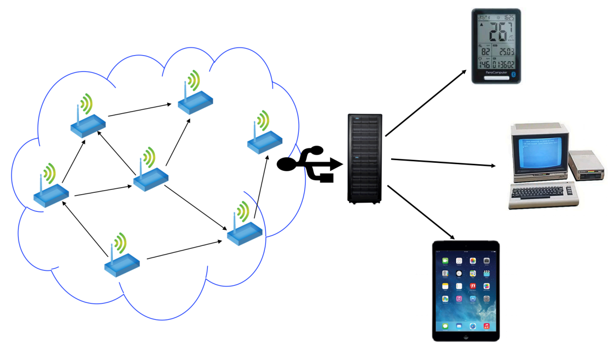

Fig. 1. A general wireless sensor network.

data compression has been adopted to reduce transmission (communication) cost at the expense of computation cost. On the other hand, caches can be used as a mean of reducing transmission costs and access latency, thus enhancing QoI but with the expense of the added caching cost. Hence, there exists a tradeoff in energy consumption due to data communication, computation and caching. This raises the question: what is the right balance between compression and caching so as to minimize the total energy consumption of the network?

In this paper, we formulate an optimization problem that characterizes the tradeoff among communication, computation, and caching energy cost with QoI guarantee, and then develop an efficient algorithm to solve the optimization problem. Each node has the ability to compress and cache the data with some finite storage capacity. We focus on wireless sensor networks as our motivating example. In particular, as shown in Figure 1, we assume that only edge sensors generate data, and there exists a single sink node that collects and serves the requests for the data generated in this network.

*Computation:* Data aggregation [2], [3] is the process of gathering data from multiple generators (e.g., sensors), compressing them to eliminate redundant information and then providing the summarized information to end users. Since only part of the original data is transmitted, data aggregation can conserve a large amount of energy. A common assumption in previous works is that energy required to compress data is smaller than that needed to transmit data. Therefore, data compression was considered a viable technique for reducing

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

[1] $\epsilon$-global optimality means that the obtained solution is within $\epsilon$ tolerance of the global optimal solution.

energy consumption. However, it has been shown [4] that computational energy cost can be significant and may cause a net-energy increase if data are compressed beyond a certain threshold. Hence, it is necessary to consider both transmission and computation costs, and it is important to characterize the trade-off between them [1].

*Caching:* Caches have been widely used in networks and distributed systems to improve performance by storing information locally, which jointly reduces access latency and bandwidth requirements, and hence improves user experience. Content Distribution Networks (CDNs), Software Defined Networks (SDNs), Named Data Networks (NDNs) and Content Centric Networks (CCNs) are important examples of such systems. The fundamental idea behind caching is to make information available at a location closer to the end-user. Again, most previous work focused on designing caching algorithms to enhance system performance without considering the energy cost of caching. However, caching itself can incur significant energy costs [5]. Therefore, capturing caching cost and characterizing the tradeoff between communication and caching energy cost are also critical for system design.

*Quality of Information (QoI):* The notion of QoI required by end users is affected by many factors. In particular, the degree of the data aggregation in a system is crucial for QoI. It has been shown that data aggregation can deteriorate QoI in some situations [6]. Thus an energy efficient design for appropriate data aggregation with a guaranteed QoI is desirable.

We focus on a tree-structured sensor network where each leaf node generates data, and compresses and transmits the data to the sink node in the network, which serves the requests for these data from devices outside this network. Examples of such a setting are military sites, wireless sensors or societal networks, where a large number of devices gather data, and desire to transmit the local information to any device outside this network that requires this information. The objective of our work is to develop an efficient algorithm to minimize the total energy cost by incorporating data communication, computation and caching energy costs with a desired QoI constraint into our model, so that an optimal data compression rate at each node, and an optimal caching location in the network can be determined. Such an algorithm should be lightweight and achieve a (sub-)optimal solution efficiently.

### A. Related Work

While optimizing energy costs in wireless sensor networks has been extensively studied [7], [8], existing work primarily is concerned with routing [9], MAC protocols [7], and clustering [10]. With the growing deployment of smart sensors in modern systems [1], in-network data processing, such as data aggregation, has been widely used as a mean of reducing system energy cost by lowering the data volume for transmission.

*Communication and Computation Energy Costs:* Energy efficient inference in a random fusion network without QoI guarantee was considered in [11]. Network Utility Maximization (NUM) framework was applied in [12] to obtain optimal compression rate for data aggregation as well as optimal locations for performing data compression. The optimal energy allocation between communication and sensing to maximize the total information received at the sink node was studied in [13], but they did not consider data computation. An efficient algorithm for data compression in a data gathering tree was proposed in [14]. [1] presented a distributed algorithm to minimize overall energy costs in a tree structured network by optimizing the compression factor at each node.

None of these works considered caching costs. Since caches are already integral components of many modern systems, including wireless sensor networks, they can be used to improve performance by making data available at locations closer to end users to reduce the communication cost [5].

To the best of our knowledge, there is no prior work that jointly considers communication, computation and caching costs in data communication networks. One of the important contributions of this paper is to develop an optimization algorithm that minimizes the total system energy costs by characterizing the tradeoff between communication, computation and caching costs with a QoI guarantee for end users.

### B. Organization and Main Results

In Section II, we describe our system model in which nodes are logically arranged as a tree. Each node receives and compresses data from its children node(s). The compressed data are transmitted and further compressed towards the sink node. Each node can also cache the compressed data locally. In Section III, we formulate the problem of energy-efficient data compression, communication and caching with QoI constraint as a non-convex mixed integer non-linear programming (MINLP) problem, which is hard to solve in general. We then show that there exists an equivalent problem obtained through symbolic reformation [15] in Section IV, and propose a variant of the Spatial Branch-and-Bound (V-SBB) algorithm to solve it. We show that our proposed algorithm can achieve $\epsilon$-global optimality of the original MINLP efficiently. Since we have a discrete space, and a non-convex problem, showing that there exists an $\epsilon$-optimal solution and developing an efficient algorithm to achieving it are quite intricate. This is another contribution in this paper.

In Section V, we evaluate the performance of our optimization framework and the proposed V-SBB algorithm through extensive numerical studies. In particular, we make a thorough comparison with other MINLP solvers Bonmin [16], NOMAD [17], and Matlab's genetic algorithm (GA) under different network scenarios. The results show that our algorithm can achieve $\epsilon$-global optimality, and is either comparable to or outperforms Bonmin. Furthermore, our algorithm is more robust and stable in the context of varying network situations. In other words, Bonmin in certain cases is not able to provide a solution, even though the original problem is feasible. Furthermore, our algorithm easily outperforms NOMAD and Matlab's GA [18] in most of the scenarios that we have tested. More importantly, we observe that with the joint optimization of data communication, computation and caching (C3), energy efficiency can be improved by as much as $88\%$

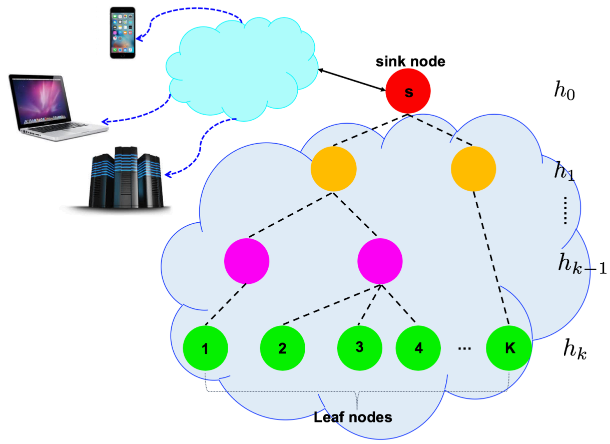


Fig. 2. Tree-Structured Network Model.

compared to only optimizing communication and computation, or communication and caching (C2). This further strengthens the contributions of our optimization framework. We provide concluding remarks in Section VI.

## II. Analytical Model

We represent the network as a directed graph $G = (V, E)$. For simplicity, we consider a tree, with $N = |V|$ nodes, as shown in Figure 2. Node $v \in V$ is capable of storing $S_v$ amount of data. Let $\mathcal{K} \subseteq V$ with $K = |\mathcal{K}|$ be the set of leaf nodes, i.e., $\mathcal{K} = \{1, 2, \cdots, K\}$. Time is partitioned in periods of equal length $T > 0$ and data generated in each period are independent. Without loss of generality (W.l.o.g.), we consider one particular period in the remainder of the paper. We assume that only leaf nodes $k \in \mathcal{K}$ can generate data, and all other nodes in the tree receive and compress data from their children nodes, and either cache or transmit the compressed data to their parent nodes during time T.

Let $y_k$ be the amount of data generated by leaf node $k \in \mathcal{K}$. The data generated at the leaf nodes are transmitted up the tree to the sink node $s$, which serves the requests for the data generated in the network. Let $h(k)$ be the depth of node $k$ in the tree. W.l.o.g., we assume that the sink node is located at level $h(s) = 0$. We represent a path from node $k$ to the sink node as the unique path $\mathcal{H}^k$ of length $h(k)$ as a sequence $\{h_0^k, h_1^k, \cdots, h_{h(k)}^k\}$ of nodes $h_j^k \in V$ such that $(h_j^k, h_{j+1}^k) \in E$, where $h_0^k \triangleq s$ (i.e., the sink node) and $h_{h(k)}^k \triangleq k$ (i.e., the node itself).

We denote the per-bit reception, transmission and compression cost of node $v \in V$ as $\varepsilon_{vR}, \varepsilon_{vT}$, and $\varepsilon_{vC}$, respectively. Each node $h_i^k$ along the path $\mathcal{H}^k$ can compress the data generated by leaf node $k$ with a *data reduction rate* $\delta_{k,i}$, where $0 < \delta_{k,i} \leq 1, \forall i, k$. The reduction rate characterizes the degree to which a node can compress the received data, which plays an important role for determining the QoI.

The higher the value of $\delta_{k,i}$, the lower the compression will be, and vice versa. The higher the degree of data compression, the larger will be the amount of energy consumed by compression. Similarly, caching the data closer to the sink node may reduce the transmission cost for serving the request, however,

TABLE I
Summary of Notations

| Notation | Description |
|---|---|
| $y_k$ | number of data (bits) generated at node $k$ |
| $\delta_{k,v}$ | reduction rate at node $v$, is the ratio of amount of output data to input data |
| $\gamma$ | the QoI threshold |
| $\varepsilon_{vR}$ | per-bit reception cost of node $v$ |
| $\varepsilon_{vT}$ | per-bit transmission cost of node $v$ |
| $\varepsilon_{vC}$ | per-bit compression cost of node $v$ |
| $b_{k,v}$ | 1 if node $v$ caches the data from leaf node $k$; otherwise 0 |
| $S_v$ | storage capacity of node $v$ |
| $w_{ca}$ | caching power efficiency |
| $R_k$ | request rate for data from node $k$ |
| $N$ | total number of nodes in the network |
| $C_v$ | set of leaf nodes that are descendants of node $v$ |
| $T$ | time length that data are cached |
| $\phi^u$ | upper bound of the objective function |
| $\mathcal{L}$ | list of regions |
| $\mathcal{R}$ | any sub-region in $\mathcal{L}$ |
| $\phi^{\mathcal{R},u}$ | upper bound on the objective function in subregion $\mathcal{R}$ |
| $\phi^{\mathcal{R},l}$ | lower bound on the objective function in subregion $\mathcal{R}$ |
| $\epsilon$ | difference between the upper and lower bound |

each node only has finite storage capacity. We study the trade-off among the energy consumed at each node for transmitting, compression and caching the data.

Denote the total energy consumption at node $v$ as $E_v$, which consists of the reception cost $E_{vR}$, transmission cost $E_{vT}$, computation cost $E_{vC}$ and storage (caching) cost $E_{vS}$; it takes the form

$$E_v = E_{vR} + E_{vT} + E_{vC} + E_{vS}, \tag{1}$$

where

$$E_{vR} = y_v \varepsilon_{vR}, \qquad E_{vT} = y_v \varepsilon_{vT} \delta_v,$$
$$E_{vC} = y_v \varepsilon_{vC} l_v(\delta_v), \qquad E_{vS} = w_{ca} y_v T. \tag{2}$$

Here, $l_v(\delta_v)$ captures the computation energy. As computation energy increases with the degree of compression, we assume that $l_v(\delta_v)$ is a continuous, decreasing and differentiable function of the reduction rate. One candidate function is $l_v(\delta_v) = 1/\delta_v - 1$ [1], [12]. Moreover, we consider an energy-proportional model [5] for caching, i.e., $E_{vS} = w_{ca} y_v T$ if the received data $y_v$ is cached for a duration of $T$ where $w_{ca}$ represents the power efficiency of caching, which strongly depends on the storage hardware technology. W.l.o.g., $w_{ca}$ is assumed to be identical for all the nodes. For simplicity, denote $f(\delta_v)$= $\varepsilon_{vR}$+$\varepsilon_{vT}$+$\varepsilon_{vC}$ as the sum of per-bit reception, transmission and compression cost at node $v$ per unit time.

During time period $T$, we assume that there are $R_k$ requests at the sink node $s$ for data $y_k$ generated by leaf node $k$. For simplicity, we assume that the number of requests for the data of a node $k$ is constant. The boolean variable $b_{k,i}$ equals 1 if the data from node $k$ is stored along the path $\mathcal{H}^k$ at node $h_i^k$, otherwise it equals 0. For ease of notation, we define $b_{k,h(k)}$ by $b_k$. Let $C_v$ denote the set of leaf nodes $k \in \mathcal{K}$ that are descendants of node $v$.

We also assume that the energy cost for searching for data at different nodes in the network is negligible [1], [19]. For convenience, let $f_{k,h(k)} \triangleq f_k$ and $\delta_{k,h(k)} \triangleq \delta_k$. For ease of exposition, the parameters used throughout this paper are summarized in Table I.

## III. Energy Efficiency Optimization

In this section, we first define the cost function in our model and then formulate the optimization problem. Data produced by every leaf node is received, transmitted, and possibly compressed by all nodes in the path from the leaf node to the root node, consuming energy

$$E_k^{\mathrm{C}} = \sum_{i=0}^{h(k)} y_k f(\delta_{k,i}) \prod_{m=i+1}^{h(k)} \delta_{k,m}, \tag{3}$$

where $\prod_{m=i}^{j} \delta_{k,m} := 1$ if $i \geq j$. Let $E^{\mathrm{R}}$ be the total energy consumed in responding to the subsequent $(R_k - 1)$ requests. We have

$$E_k^{\mathrm{R}} = \sum_{i=0}^{h(k)} y_k (R_k - 1) \Bigg\{ f(\delta_{k,i}) \prod_{m=i+1}^{h(k)} \delta_{k,m} \Bigg( 1 - \sum_{j=0}^{i-1} b_{k,j} \Bigg) + \Bigg( \prod_{m=i}^{h(k)} \delta_{k,m} \Bigg) b_{k,i} (w_{ca} T + \varepsilon_{kT}) \Bigg\}. \tag{4}$$

The first term captures the energy cost for reception, transmission and compression up the tree from node $v_{k,i-1}$ to $v_{k,0}$ and the second term captures the energy cost for storage and transmission by node $v_{k,i}$ The total energy consumed in the network is $E^{\mathrm{total}}$,

$$E^{\mathrm{total}}(\boldsymbol{\delta}, \boldsymbol{b}) \triangleq \sum_{k \in \mathcal{K}} \Bigg( E_k^{\mathrm{C}} + E_k^{\mathrm{R}} \Bigg), \tag{5}$$

where $\boldsymbol{\delta} = \{\delta_{k,i}, \forall k \in \mathcal{K}, i = 0, \cdots, h(k)\}$ and $\boldsymbol{b} = \{b_{k,i}, \forall k \in \mathcal{K}, i = 0, \cdots, h(k)\}$. Our objective is to minimize the total energy consumption of the network with a QoI constraint for end users by choosing the compression ratio vector $\boldsymbol{\delta}$ and caching decision vector $\boldsymbol{b}$ in the network $G$. Therefore, the optimization problem is,

$$\begin{aligned} \min_{\boldsymbol{\delta}, \boldsymbol{b}} \quad & E^{\mathrm{total}}(\boldsymbol{\delta}, \boldsymbol{b}) \\ \text{s.t.} \quad & \sum_{k \in \mathcal{K}} y_k \prod_{i=0}^{h(k)} \delta_{k,i} \geq \gamma, \\ & b_{k,i} \in \{0,1\}, \forall k \in \mathcal{K}, i = 0, \cdots, h(k), \\ & \sum_{k \in C_v} b_{k,h(v)} y_k \prod_{j=h(k)}^{h(v)} \delta_{k,j} \leq S_v, \forall\, v \in V, \\ & \sum_{i=0}^{h(k)} b_{k,i} \leq 1, \forall k \in \mathcal{K}, \end{aligned} \tag{6}$$

where $h(v)$ is the depth of node $v$ in the tree. The first constraint is the QoI constraint i.e. the total data available at the sink node [1]. The second constraint indicates that our decision/caching variable $b_{k,i}$ is binary. The third constraint is on what can be cached at each node. The fourth constraint is that at most one copy of the generated data should be cached on the path between the leaf node and the sink node. The optimization problem in (6) is a non-convex MINLP problem with $M$ continuous variables, the $\delta_{k,i}$'s and $M$ binary variables, the $b_{k,i}$'s where, $M = \sum_{k \in \mathcal{K}} h(k)$.

### A. Properties

**Theorem 1.** *The optimization problem defined in* (6) *is NP-hard.*

*Proof.* The optimization problem (6) can be reduced to a general non-convex MINLP problem. Due to space limitations, the general form of a non-convex MINLP and the reduction steps are presented in [20]. Since non-convex MINLP is NP-hard [21], the optimization problem described in (6) is NP-hard. □

**Remark 1.** *The objective function $E^{total}$ defined in* (6) *is monotonically increasing in the number of requests $R_k$ for all $k \in \mathcal{K}$ provided that $\boldsymbol{\delta}$ and $\boldsymbol{b}$ are fixed.*

Notice that (3) is independent of $R_k$ and (4) is linear in $R_k$, and its multipliers are positive. Hence, for any fixed $\boldsymbol{b}$ and $\boldsymbol{\delta}$, (5) increases monotonically with $R_k$.

**Remark 2.** *Given a fixed network scenario, if we increase the number of requests $R_k$ for the data generated by leaf node $k$, then these data will be cached closer to the sink node or at the sink node, if there exists enough cache capacity, to reduce the overall energy consumption.*

For fixed $\boldsymbol{\delta}$, observe from (4) that energy consumption decreases if the cache is moved closer to the root as the nodes deep in the tree do not need to retransmit.

### B. Relaxation of Assumptions

In our model, we make several assumptions for the sake of simplicity. In the following, we discuss the relaxation of these assumptions.

While we assume that the network is structured as a tree, this assumption can be easily relaxed as long as there exists a simple fixed path from each leaf node to the sink node. The tree structure represents a simple topology that captures the key parameters in the optimization formulation without the complexity introduced by a general network topology. Furthermore, for simplicity, we assume that all parameters across the nodes are identical, which is not necessary as seen from the cost function. We also assume that only leaf nodes generate data. However, our model can be extended to allow intermediate nodes to generate data at the cost of added complexity. Finally, rather than having a constant $R_k$, we can generalize our approach to the case where $R_k$ are drawn from a distribution such as the Zipf distribution [5].

## IV. Variant of Spatial Branch and Bound Algorithm

In this section, we present a variant of the Spatial Brand-and-Bound algorithm (V-SBB). Instead of solving the MINLP problem (6) directly, we use V-SBB to solve a *standard form* of the original MINLP. We first introduce the *Symbolic*

*Reformulation* [15] method that reformulates the MINLP (6) into a standard form needed by V-SBB.

**Definition 1.** *A MINLP problem is said to be in a* standard form *if it can be written as*

$$\begin{aligned}\min_{\boldsymbol{w}} \quad & \boldsymbol{w}_{obj} \\ \textit{s.t.} \quad & \mathbf{A}\mathbf{w} = \mathbf{b}, \\ & \mathbf{w}^l \le \mathbf{w} \le \mathbf{w}^U, \\ & \mathbf{w}_k \equiv \mathbf{w}_i \mathbf{w}_j, \quad \forall (i,j,k) \in \mathcal{T}_{bt}, \\ & \mathbf{w}_k \equiv \mathbf{w}_i / \mathbf{w}_j, \quad \forall (i,j,k) \in \mathcal{T}_{lft}, \end{aligned} \tag{7}$$

*where the vector of variables* $\mathbf{w}$ *consists of continuous and discrete variables in the original MINLP. The sets* $\tau_{bt}$ *and* $\tau_{lft}$ *contain all relationships that arise in the reformulation.* $\boldsymbol{A}$ *and* $\boldsymbol{b}$ *are a matrix and a vector of real coefficients, respectively. The index obj denotes the position of a single variable corresponding to the objective function value within the vector* $\boldsymbol{w}$*.*

**Theorem 2.** *The non-convex MINLP problem (6) can be transformed into a standard form.*

Due to space constraints, we relegate detailed reformulations and standard form of (6) to [20].

Here, we give an example to illustrate the above reformulation process.

**Example 1.** We consider $k = 1$ and $h(k) = 1$ in (6), i.e., one leaf node and one sink node. Then (3) and (4) reduce to

$$\begin{aligned} & E_1^C = y_1 f(\delta_{1,0})\delta_{1,1} + y_1 f(\delta_{1,1}), \\ & E_1^R = y_1(R_1 - 1)\left[f(\delta_{1,0}\delta_{1,1} + \delta_{1,0}\delta_{1,1} b_{1,0}(w_{ca}T + \delta_{1T})\right] \\ & \quad + y_1(R_1 - 1)\left[f(\delta_{1,1}(1 - b_{1,0}) + \delta_{1,0}\delta_{1,1} b_{1,1}(w_{ca}T + \delta_{1T})\right], \end{aligned} \tag{8}$$

and non-convex MINLP problem is

$$\begin{aligned}\min_{\boldsymbol{\delta},\boldsymbol{b}} \quad & E^{\text{total}}(\boldsymbol{\delta}, \boldsymbol{b}) = E_1^C + E_1^R \\ \text{s.t.} \quad & y_1 \delta_{1,0}\delta_{1,1} \ge \gamma, \\ & b_{1,0}, b_{1,1} \in \{0,1\}, \\ & b_{1,0} y_1 \delta_{1,0}\delta_{1,1} \le S_0, \\ & b_{1,1} y_1 \delta_{1,1} \le S_1, \\ & b_{1,0} + b_{1,1} \le 1. \end{aligned} \tag{9}$$

$\delta_{1,0}\delta_{1,1}$ is a bilinear term. Based on symbolic reformulation rules (see [20] for details), a new bilinear auxiliary variable $w_{1,0}^{\text{bt}}$ needs to be added. The first constraint in (9) is then transformed into $y_1 w_{1,0}^{\text{bt}} \ge \gamma$, which is linear in auxiliary variable $w_{1,0}^{\text{bt}}$. Similarly, we add $w_{1,0}^{\text{lft}}$ for linear-fractional term $\delta_{1,1}/\delta_{1,0}$ that appears in $f(\cdot)$. $b_{1,0}\delta_{1,0}\delta_{1,1}$ in the third constraint of (9) is a tri-linear term. Since $\delta_{1,0}\delta_{1,1}$ is replaced by $w_{1,0}^{\text{bt}}$, we obtain a bilinear term $b_{1,0}w_{1,0}^{\text{bt}}$. Again, based on symbolic reformulation rules, $b_{1,0}w_{1,0}^{\text{bt}}$ is replaced by a new auxiliary variable $\overline{w}_{1,0}^{\text{bt}}$. Similarly we add new auxiliary variables $\tilde{w}_{1,1}^{\text{b}}, \tilde{w}_{1,0}^{\text{bt}}$, and $\tilde{w}_{1,0}^{\text{lft}}$. The objective function in (9) can be then expressed as a function of these new auxiliary variables. Therefore, the standard form of (9) is

$$\begin{aligned}\min_{\boldsymbol{\delta},\boldsymbol{b}} \quad & w_{\text{obj}} \\ \text{s.t.} \quad & y_1 w_{1,0}^{\text{bt}} \ge \gamma, \\ & b_{1,0}, b_{1,1} \in \{0,1\}, \\ & y_1 \overline{w}_{1,0}^{\text{bt}} \le S_0, \\ & y_1 \tilde{w}_{1,1}^{\text{bt}} \le S_1, \\ & b_{1,0} + b_{1,1} \le 1, \\ & w_{1,0}^{\text{bt}} = \delta_{1,1} \times \delta_{1,0}, \\ & w_{1,0}^{\text{lft}} = \delta_{1,1}/\delta_{1,0}, \\ & \overline{w}_{1,0}^{\text{bt}} = b_{1,0} \times w_{1,0}^{b}, \\ & \tilde{w}_{1,1}^{\text{bt}} = b_{1,1} \times \delta_{1,1}, \\ & \tilde{w}_{1,0}^{\text{bt}} = \delta_{1,1} \times b_{1,0}, \\ & \tilde{w}_{1,0}^{\text{lft}} = b_{1,0}/\delta_{1,1}, \\ & w_{\text{obj}} = y_1\varepsilon_{1R}\delta_{1,1} + \varepsilon_{1T} y_1 w_{1,0}^{\text{bt}} + y_1\varepsilon_{1C} w_{1,0}^{\text{lft}} - y_1\varepsilon_{1C}\delta_{1,1} \\ & + y_1\varepsilon_{1R} + \varepsilon_{1T} y_1 \delta_{1,1} + y_1\varepsilon_{1C}/\delta_{1,1} - y_1\varepsilon_{1C} + y_1 R_1 \varepsilon_{1R}\delta_{1,0} \\ & + y_1 R_1 \varepsilon_{1T} w_{1,0}^{\text{bt}} + y_1 R_1 \varepsilon_{1C} w_{1,0}^{\text{lft}} - y_1 R_1 \varepsilon_{1C}\delta_{1,1} - y_1 R_1 \delta_{1,1} \\ & - y_1\varepsilon_{1T} w_{1,0}^{\text{bt}} - y_1\varepsilon_{1C} w_{1,0}^{\text{lft}} + y_1\varepsilon_{1C}\delta_{1,1} + y_1 R_1 w_{ca} T \overline{w}_{1,0}^{\text{bt}} \\ & + y_1 R_1 \varepsilon_{1T} \overline{w}_{1,0}^{\text{bt}} - y_1 w_{ca} T \overline{w}_{1,0}^{\text{bt}} - y_1 \varepsilon_{1T} \overline{w}_{1,0}^{\text{bt}} + y_1 R_1 \varepsilon_{1R} \\ & + y_1 R_1 \varepsilon_{1T}\delta_{1,1} + y_1 R_1 \varepsilon_{1C}/\delta_{1,1} - y_1 R_1 \varepsilon_{1C} - y_1 \varepsilon_{1C} \\ & - y_1 \varepsilon_{1T}\delta_1 - y_1\varepsilon_{1C}/\delta_{1,1} + y_1\varepsilon_{1C} - y_1 R_1 \varepsilon_{1R} b_{1,0} \\ & - y_1 R_1 \varepsilon_{1T} \tilde{w}_{1,0}^{\text{bt}} - y_1 R_1 \varepsilon_{1C} \tilde{w}_{1,0}^{\text{lft}} + y_1 R_1 \varepsilon_{1C} b_{1,0} + y_1 \varepsilon_{1R} b_{1,0} \\ & + y_1 \varepsilon_{1T} \tilde{w}_{1,0}^{\text{bt}} + y_1 \varepsilon_{1C} \tilde{w}_{1,0}^{\text{lft}} - y_1 \varepsilon_{1C} b_{1,0} + y_1 R_1 w_{ca} T \tilde{w}_{1,1}^{\text{bt}} \\ & + y_1 R_1 \varepsilon_{1T} \tilde{w}_{1,1}^{\text{bt}} - y_1 w_{ca} T \tilde{w}_{1,1}^{\text{bt}} - y_1 \varepsilon_{1T} b_{1,1}. \end{aligned} \tag{10}$$

Through this reformulation, the non-convex and non-linear terms in the original problem are transformed into bilinear and linear fractional terms, which can be easily used to compute the lower bound of each region in V-SBB, which are discussed in details later. This is the reason V-SBB requires reformulating the original problem into a standard form.

**Theorem 3.** *Reformulated problem and the original MINLP are equivalent.*

Proof is available in [20].

Due to the reformulation, the number of variables in the reformulated problem is larger than in the original MINLP. In the following, we show that the number of auxiliary variables that arise from symbolic reformulation is bounded.

**Remark 3.** *The number of auxiliary variables in the symbolic reformulation is* $O(n^2)$*, where* $n = 2M$ *is the number of variables in the original formulation.*

From [22], a way to transform a general form optimization problem into a standard form (7) is through basic arithmetic operations on original variables. To be more specific, any algebraic expression results from the basic operators including the five basic binary operators, i.e., addition, subtraction, multiplication, division and exponentiation, and the unary operators, i.e., logarithms etc. Therefore, in order to construct a standard problem consisting of simple terms corresponding to these binary or unary operations, new variables need to be added corresponding to these operations. From the symbolic reformulation process [22]–[24], any added variable results from the basic operations between two (including possibly the same) original variables or added variables. Hence, based on the basic operations, there are at most $n^2$ combinations of these variables, given that there are $n$ variables in the original problem (6). Therefore, the number of added variables in the symbolic reformulation is bounded as $O(n^2)$. In the remainder of this section, we present the V-SBB to solve the equivalent problem.

### A. Variant of Spatial Branch-and-Bound Algorithm

In contrast with conventional SBB, our newly proposed V-SBB eliminates the bound-tightening steps. This has two advantages: (i) bound tightening step does not always guarantee faster convergence; (ii) removal significantly reduces the computational complexity of the algorithm. Algorithm 1 gives an overview of V-SBB. We briefly describe the key steps due to space limitations. A detailed explanation of each step is given in [20].

*Step* 2*:* We use the *least lower bound rule*[2] to choose a subregion $\mathcal{R}$ from $\mathcal{L}$ among all feasible subregions. This lower bound is obtained by solving a convex relaxation of the reformulated problem such as (10). McCormick linear overestimators and underestimators [25] are used to obtain the convex relaxation for bilinear terms (bt) and linear fractional terms (lft). This leads to a Mixed Integer Linear Program (MILP), that can be solved by the *SCIP* solver [26]. Denote the optimal solution of this subregion as $\phi^{\mathcal{R},l}$. Note that if the convex relaxation is infeasible or the obtained lower bound is greater than the current upper bound $\phi^u$, we move to *Step* 5, otherwise we move to *Step* 3.

*Step* 3*:* We compute the upper bound $\phi^{\mathcal{R},u}$ for the subregion $\mathcal{R}$ through local MINLP solver such as *Bonmin* [16]. If this upper bound cannot be obtained or is greater than $\phi^u$, we move to *Step* 4. Otherwise, we set it as the current best solution $\phi^u$, and delete all other subregions that have higher lower bounds than this region's upper bound. If the difference between the upper and lower bounds for this subregion is within $\epsilon$-tolerance, we delete this subregion by moving to *Step* 5, otherwise move to *Step* 4.

*Step* 4*:* known as the *branching* step, is used to select a variable and its corresponding value at which the region is further divided. Here, we use the variable and value selection rule specified in [23], under which the variable that causes the maximal reduction in the feasibility gap between the solution of *Step* 2 and the exact problem, is branched on. Then we partition $\mathcal{R}$ into $\mathcal{R}_{\text{right}}$ and $\mathcal{R}_{\text{left}}$, and add them into $\mathcal{L}$ as well as delete $\mathcal{R}$.

### B. Convergence of V-SBB

**Definition 2.** *A solution to an optimization problem is said to be $\epsilon$-global optimal if the difference between this solution and the optimal one is within $\epsilon$.*

[2]Select a subregion $\mathcal{R} \in \mathcal{L}$ , whose convex relaxation provides the lowest objective function value.

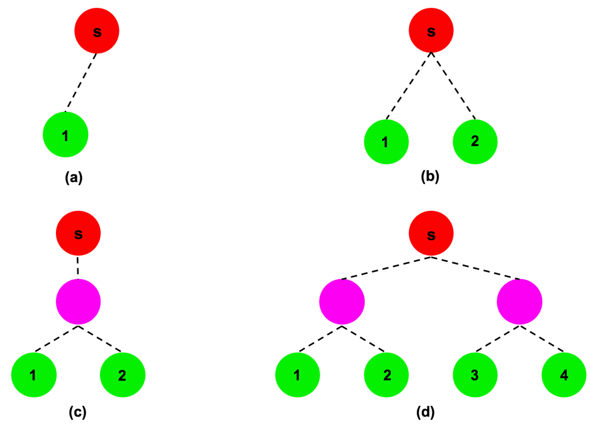


Fig. 3. Candidate network topologies used in the experiments: (a) one sink node and one leaf node; (b) one sink node and two leaf nodes; (c) one sink node, one intermediate node and two leaf nodes; and (d) one sink node, two intermediate nodes and four leaf nodes.

**Theorem 4.** *Our V-SBB described in Algorithm 1 converges to an $\epsilon$-global optimal solution of its standard problem given in [20].*

Though we made critical modifications to obtain our V-SBB algorithm, the proof of convergence follows an argument similar to that of *Branch-and-select* given in [27]. We present the poof in [20] for completeness.

**Algorithm 1** Variant of Spatial Branch-and-Bound (V-SBB)

**Step 1**: Initialize $\phi^u := \infty$ and $\mathcal{L}$ to a single domain
**Step 2**: Choose a subregion $\mathcal{R} \in \mathcal{L}$ using *least lower bound rule*
**if** $\mathcal{L} = \emptyset$ **then** Go to Step 6
**if** for chosen region $\mathcal{R}$, $\phi^{\mathcal{R},l}$ is infeasible or $\phi^{\mathcal{R},l} \geq \phi^u - \epsilon$ **then** Go to Step 5
**Step 3**: Obtain the upper bound $\phi^{\mathcal{R},u}$
**if** upper bound cannot be obtained or if $\phi^{\mathcal{R},u} > \phi^u$ **then** Go to Step 4
**else** $\phi^u := \phi^{\mathcal{R},u}$ and, from the list $\mathcal{L}$, delete all subregions $\mathcal{S} \in \mathcal{L}$ such that $\phi^{\mathcal{S},l} \geq \phi^u - \epsilon$
**if** $\phi^{\mathcal{R},u} - \phi^{\mathcal{R},l} \leq \epsilon$ **then** Go to Step 5
**Step 4**: Partition $\mathcal{R}$ into new subregions $\mathcal{R}_{\text{right}}$ and $\mathcal{R}_{\text{left}}$
**Step 5**: Delete $\mathcal{R}$ from $\mathcal{L}$ and go to Step 2
**Step 6**: Terminate Search
**if** $\phi^u = \infty$ **then** Problem is infeasible
**else** $\phi^u$ is $\epsilon$-global optimal

## V. Evaluation

We evaluate the performance of our V-SBB algorithm as well as the energy efficiency of our communication, compression and caching (C3) joint optimization framework through a series of experiments on several network topologies as shown in Figure 3. Our key objective is to gain preliminary insights into our algorithm when compared with a few other well-known techniques. The highlights of the evaluation results are:

- Our V-SBB algorithm can obtain an $\epsilon$-global optimal solution in most situations within a reasonable time. Also it is robust and stable to various parameters in different network scenarios.
- When Bonmin [16] can achieve a solution, it is faster. However, the solution obtained through Bonmin is not always comparable to that of V-SBB. We observe that when higher compression is done (i.e., smaller value of $\gamma$), V-SBB always outperforms Bonmin. More importantly, we find that Bonmin has poor performance in stability and robustness, i.e., it cannot even produce feasible solutions in some cases although they exist. NOMAD [17] and GA [18] often produce objective-function values much larger than V-SBB.
- Our C3 joint optimization framework improves energy efficiency by as much as $88\%$ compared to the C2 optimization over communication and computation, or communication and caching.

### A. Methodology

***Performance metrics:*** Our primary metrics for comparisons are:
*(1) The best solution to the objective function:* Since obtaining the global optimum for the NP-hard problem is daunting, we are primarily interested in $\epsilon$-global optimum;
*(2) Convergence Time,* which is the time an algorithm needs to obtain the best solution;
*(3) Stability and Robustness,* which is characterized by the frequency or ability of the algorithm to provide feasible solutions, provided that they are known to exist;
*(4) Energy efficiency* in joint optimization. We compare the energy cost of our joint optimization framework for communication, computation and caching (C3) with that of the optimization of any of the two types of resources (denoted by C2) under the same situation. The energy efficiency $\mathcal{E}$ defined as:

$$\mathcal{E} = \frac{E^{\text{total}*}(\text{C2}) - E^{\text{total}*}(\text{C3})}{E^{\text{total}*}(\text{C2})} \times 100\%, \quad (11)$$

where $E^{\text{total}*}(\text{C3})$ and $E^{\text{total}*}(\text{C2})$ are the optimal energy costs under the C3 optimization framework in (6) and the C2 optimization, respectively. $\mathcal{E}$ reflects the reduction of energy efficiency for the C3 over the C2 optimization.

TABLE II
CHARACTERISTICS OF THE SOLVERS USED IN THIS PAPER

| Solver | Characteristics |
|---|---|
| **Bonmin** [16] | A deterministic approach based on Branch-and-Cut method that solves relaxation problem with Interior Point Optimization tool (IPOPT), as well as mixed integer problem with Coin or Branch and Cut (CBC). |
| **NOMAD** [17] | A stochastic approach based on Mesh Adaptive Direct Search Algorithm (MADS) that guarantees local optimality. It can be used to solve non-convex MINLP and has a relatively good performance. |
| **GA** [18] | A meta-heuristic stochastic approach that can be tuned to solve global optimization problems. We use Matlab *Optimization Toolbox*'s implementation. |

***Setup:*** We implement V-SBB in Matlab on a Core i7 3.40 GHz CPU with 16 GB RAM. The candidate MINLP solvers in this work include Bonmin, NOMAD and GA, which are implemented with Opti-Toolbox [28]. We summarize the characteristics of these solvers in Table II. Note that these solvers can be applied directly to solve the original optimization problem in (6), while our V-SBB solves the equivalent problem. The reformulations needed are executed by a Java based module and we derive the bounds on the auxiliary variables. We also relax the integer constraint in (6) to obtain a non-linear programming problem, which is solved by IPOPT [29] and use it as a benchmark for comparison. V-SBB terminates when $\epsilon$-optimality is obtained or a computation timer of 200 seconds expires. We take $\epsilon = 0.001$ in our study. If the timer expires, the last feasible solution is taken as the best solution. Our simulation parameters are provided in Table III, which are the typical values used in the literature [1], [7], [8].

TABLE III
PARAMETERS USED IN SIMULATIONS

| Parameter | Value | Parameter | Value (Joules) |
|---|---|---|---|
| $y_k$ | 1000 | $\varepsilon_{vR}$ | $50 \times 10^{-9}$ |
| $R_k$ | 100 | $\varepsilon_{vT}$ | $200 \times 10^{-9}$ |
| $w_{ca}$ | $1.88 \times 10^{-6}$ | $\varepsilon_{cR}$ | $80 \times 10^{-9}$ |
| $T$ | 10s | $\gamma$ | $[1, \sum_{k\in\mathcal{K}} y_k]$ |

### B. The Best Solution to the Objective Function

We compare the performance of V-SBB with three other candidate solvers for the networks in Figure 3. The results for two nodes and seven nodes are presented in Tables IV and V. We observe that V-SBB achieves the lowest value comparable to Bonmin for larger values of $\gamma$, and significantly outperforms Bonmin for smaller values of $\gamma$, which we discuss in detail later. However, Bonmin cannot generate a feasible solution even if it exists for some cases. This is because Bonmin is built on the Branch-and-Cut method, which sometimes cuts regions where a lower value exists. NOMAD and GA in general yield a higher objective-function value than V-SBB does. This is because both NOMAD and GA are based on a stochastic approach which cannot guarantee convergence to the $\epsilon$-global optimum. Similar trends are observed for three and four node networks, details can be found in [20] due to space limitations.

Figure 4 verifies that the optimal energy cost is monotonically increasing with the number of requests, as stated in Remark 1 for a two node and seven node network. The results are obtained using our C3 framework for $\gamma = 0.25\sum_{k\in\mathcal{K}} y_k$ and $\gamma = 0.75\sum_{k\in\mathcal{K}} y_k$, respectively. For the network parameters under consideration, we note that there is a turning point on the curves, and the total energy cost increases much faster with the number of requests before the turning point than that after it. This is because the data has already been cached at the root node at this point and there is no need to retrieve data from other nodes in the network, which reduces transmission costs. This is the benefit that caching brings, and we will further discuss the advantage of C3 optimization over the C2 later in Section V-E.

### C. Convergence Time

The time taken to obtain the best solution is important in practice. The amount of time that an algorithm requires

TABLE IV
THE BEST SOLUTION TO THE OBJECTIVE FUNCTION (OBJ.) AND CONVERGENCE TIME FOR TWO NODES NETWORK

| **Solver** | $\gamma = 1$ | | $\gamma = 250$ | | $\gamma = 500$ | | $\gamma = 750$ | | $\gamma = 1000$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Obj.** | **Time** (s) | **Obj.** | **Time** (s) | **Obj.** | **Time** (s) | **Obj.** | **Time** (s) | **Obj.** | **Time** (s) |
| **Bonmin** | 0.010 | 0.076 | 0.018 | 0.07 | 0.026 | 0.071 | 0.032 | 0.077 | 0.039 | 0.102 |
| **NOMAD** | 0.012 | 1.036 | 0.038 | 0.739 | 0.033 | 0.640 | 0.038 | 0.203 | 0.039 | 0.263 |
| **GA** | 0.010 | 0.286 | 0.018 | 2.817 | 0.026 | 7.670 | 0.042 | 11.020 | 0.064 | 3.330 |
| **V-SBB** | 0.010 | 18.231 | 0.018 | 17.389 | 0.026 | 12.278 | 0.032 | 7.327 | 0.039 | 19.437 |
| **Relaxed** | 0.010 | 0.075 | 0.018 | 0.048 | 0.026 | 0.046 | 0.032 | 0.050 | 0.039 | 0.059 |

TABLE V
THE BEST SOLUTION TO THE OBJECTIVE FUNCTION (OBJ.) AND CONVERGENCE TIME FOR SEVEN NODES NETWORK

| **Solver** | $\gamma = 1$ | | $\gamma = 1000$ | | $\gamma = 2000$ | | $\gamma = 3000$ | | $\gamma = 4000$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Obj.** | **Time** (s) | **Obj.** | **Time** (s) | **Obj.** | **Time** (s) | **Obj.** | **Time** (s) | **Obj.** | **Time** (s) |
| **Bonmin** | 0.0002 | 0.214 | 0.039 | 0.164 | 0.078 | 0.593 | 0.117 | 0.167 | 0.156 | 0.212 |
| **NOMAD** | 0.004 | 433.988 | 0.121 | 381.293 | 0.108 | 203.696 | 0.158 | 61.093 | 0.181 | 26.031 |
| **GA** | 0.043 | 44.538 | 0.096 | 30.605 | 0.164 | 44.970 | 0.226 | 17.307 | 0.303 | 28.820 |
| **V-SBB** | 0.0001 | 1871.403 | 0.039 | 25.101 | 0.078 | 30.425 | 0.117 | 23.706 | 0.156 | 19.125 |
| **Relaxed** | 0.0002 | 0.201 | 0.039 | 0.111 | 0.078 | 0.095 | 0.117 | 0.102 | 0.156 | 0.105 |

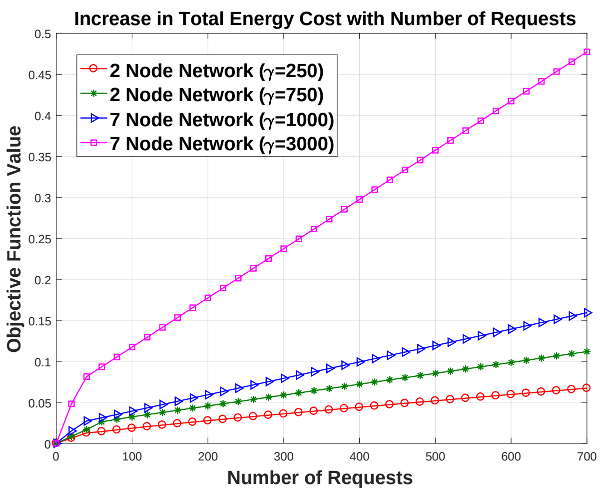


Fig. 4. Total Energy Costs vs. Number of Requests.

TABLE VI
INFEASIBILITY OF BONMIN FOR NETWORKS IN FIGURE 3

| **Networks** | (a) | (b) | (c) | (d) |
|---|---|---|---|---|
| **# of test values** | 1000 | 2000 | 2000 | 4000 |
| **# of infeasible solutions** | 0 | 0 | 1 | 216 |
| **Infeasibility (%)** | 0 | 0 | 0.05 | 5.4 |

to obtain its best solution as discussed in Section V-B are shown in Tables IV and V for the two node and seven node networks, respectively. It can be see that Bonmin is the fastest method since it uses the branch-and-cut approach which cuts certain domains to accelerate the branching process. As discussed earlier, the Bonmin algorithm is fast at the expense of algorithm stability, i.e., sometimes it cannot find a solution although it exists. This will be further discussed in the following section. V-SBB takes longer to obtain a better solution, because our reformulation introduces auxiliary variables and additional linear constraints. Different applications can tolerate various degrees of algorithm speed. For the sample networks and applications under consideration, the speed of V-SBB is considered to be acceptable [21].

### *D. Stability and Robustness*

From the analysis in Sections V-B and V-C, we know that Bonmin is faster but unstable in some situations. We further characterize the stability of Bonmin with respect to the threshold value of QoI $\gamma$ as follows. Specifically, we fix all other parameters in Table III, and vary only the maximal possible value of $\gamma$ in different networks. The results are shown in Table VI. For each maximal value, we test all the possible integer values of $\gamma$ between 1 and itself. Hence, the number of tests equals the maximal value. We see that the number of instances where the Bonmin method fails to produce a feasible solution increases as the network size increases. This is mainly due to the cutting phase in the Bonmin method, which cuts the feasible regions that need to be branched.

Although Bonmin can provide a feasible solution for smaller values of $\gamma$ at a faster time, we observe that the value of the solution is larger than that of V-SBB. We compare the performance of V-SBB and Bonmin for smaller values of $\gamma$ in Table VII. We see that V-SBB outperforms Bonmin by as much as $52.45\%$ when searching for an $\epsilon$-global optimum, though it requires more time. The timer is set to 7200s for results shown in Table VII.

### *E. Energy Efficiency*

We compare the total energy costs under joint C3 optimization with those under C2 optimization. We consider two cases for the C2 optimization: (i) C2o (Communication and Computation), where we set $S_v = 0$ for each node to avoid any data caching; (ii) C2a (Communication and Caching), where we set $\gamma = \sum_{k \in \mathcal{K}} y_k$, which is equivalent to $\delta_v = 1, \forall v \in V$, i.e., no computation. Comparison between C3, C2o and C2a is shown in Figure 5.

First, we observe that as the number of requests increases, the total energy cost increases, as reflected in Remark 1. Second, the energy cost for the C3 joint optimization is lower than that for C2o optimization for the same parameter setting. This captures the tradeoff between caching, communication and computation. In other words, although C3 incurs caching costs, it may significantly reduce the communication and

TABLE VII
COMPARISON BETWEEN V-SBB AND BONMIN FOR SMALLER VALUES OF $\gamma$ IN SEVEN NODE NETWORK

| **Solver** | $\gamma$ =1 | | $\gamma$=3 | | $\gamma$ =5 | | $\gamma$ =8 | | $\gamma$ =50 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Obj.** | **Time (s)** | **Obj.** | **Time** | **Obj.** | **Time** | **Obj.** | **Time** | **Obj.** | **Time** |
| **Bonmin** | 0.0002 | 0.214 | 0.0003 | 0.211 | 0.0003 | 0.224 | 0.0005 | 0.23 | 0.0021 | 0.364 |
| **V-SBB** | 0.00011 | 1871.403 | 0.00015 | 2330 | 0.00019 | 1243.77 | 0.00047 | 1350.016 | 0.0020 | 3325.302 |
| **Improvement (%)** | 52.45 | | 49.43 | | 50.30 | | 7.59 | | 4.62 | |

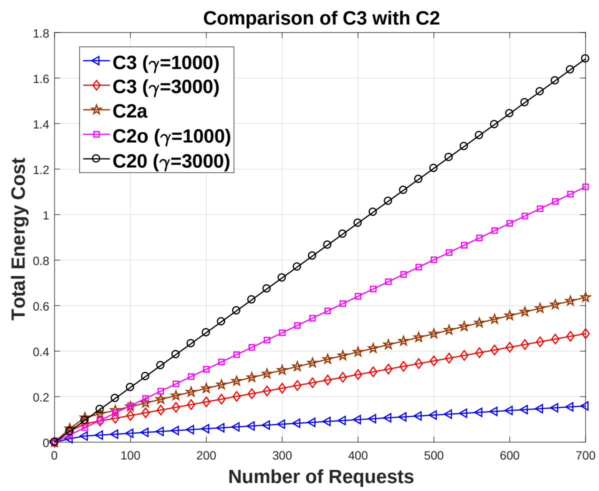


Fig. 5. Comparison of C3 and C2 optimization for the seven node network in Figure 3.

computation, which in turn brings down total energy cost. Similarly, C3 optimization outperforms C2a although C3 incurs caching cost. Using Equation (11), energy efficiency improves by as much as 88% for the C3 framework when compared with the C2 formulation. These trends are observed in other candidate network topologies and readers are referred to [20] for details due to space limitations.

**Remark 4.** *Note that the above results are based on parameter values typically used in the literature, as shown in Table III. From our analysis, it is clear that the larger the ratio between* $\varepsilon_{vT}$ *and* $\varepsilon_{vR}$, $\varepsilon_{vC}$, *the larger will be the improvement provided by our C3 formulation.*

## VI. CONCLUSION

We have investigated energy efficiency tradeoffs among communication, computation and caching with QoI guarantee in communication networks. We first formulated an optimization problem that characterizes the energy costs for communication, computation and caching. This optimization problem belongs to the non-convex class of MINLP, which is hard to solve in general. We then proposed a variant of the spatial branch-and-bound (V-SBB) algorithm, which can solve the MINLP with $\epsilon$-optimality guarantee. Finally, we show numerically that the newly proposed V-SBB algorithm outperforms the existing MINLP solvers, Bonmin, NOMAD and GA. We also observed that C3 optimization framework, which to the best of our knowledge has not been investigated in the literature, leads to an energy saving of as much as 88% compared with either of the C2 optimizations which have been widely studied.

Going further, we aim to extend our results in two ways. The first is to refine and improve the symbolic reformulation to reduce the number of needed auxiliary variables in order to shorten the algorithm execution time. Second, since many networking problems involve the optimization of both continuous and discrete variables as in this work, we plan to apply and extend the newly proposed V-SBB to solve those problems.

## REFERENCES


[1] S. Nazemi, K. K. Leung, and A. Swami, “QoI-aware Tradeoff Between Communication and Computation in Wireless Ad-hoc Networks,” in *Proc. IEEE PIMRC*, 2016.

[2] R. Rajagopalan and P. K. Varshney, “Data Aggregation Techniques in Sensor Networks: A Survey,” *IEEE Commun. Surveys Tuts.*, vol. 8, no. 4, pp. 48–63, 2006.

[3] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, “In-network Aggregation Techniques for Wireless Sensor Networks: a Survey,” *IEEE Wireless Communications*, vol. 14, no. 2, 2007.

[4] K. C. Barr and K. Asanović, “Energy-aware Lossless Data Compression,” *ACM Transactions on Computer Systems*, 2006.

[5] N. Choi, K. Guan, D. C. Kilper, and G. Atkinson, “In-network Caching Effect on Optimal Energy Consumption in Content-Centric Networking,” in *Proc. IEEE ICC*, 2012.

[6] S. A. Ehikioya, “A Characterization of Information Quality Using Fuzzy Logic,” in *NAFIPS*, 1999.

[7] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-Efficient Communication Protocol for Wireless Microsensor Networks,” in *System sciences*, 2000.

[8] W. Ye, J. Heidemann, and D. Estrin, “An Energy-Efficient MAC Protocol for Wireless Sensor Networks,” in *IEEE INFOCOM*, 2002.

[9] A. Manjeshwar and D. P. Agrawal, “TEEN: a Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks,” in *IPDPS*, 2001.

[10] M. Ye, C. Li, G. Chen, and J. Wu, “EECS: an Energy Efficient Clustering Scheme in Wireless Sensor Networks,” in *IEEE IPCCC*, 2005.

[11] A. Anandkumar, J. E. Yukich, L. Tong, and A. Swami, “Energy Scaling Laws for Distributed Inference in Random Fusion Networks,” *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 7, 2009.

[12] S. Eswaran, J. Edwards, A. Misra, and T. F. L. Porta, “Adaptive In-Network Processing for Bandwidth and Energy Constrained Mission-Oriented Multihop Wireless Networks,” *IEEE Transactions on Mobile Computing*, vol. 11, no. 9, pp. 1484–1498, Sept 2012.

[13] C. Zhang, J. Kurose, Y. Liu, D. Towsley, and M. Zink, “A Distributed Algorithm for Joint Sensing and Routing in Wireless Networks with Non-steerable Directional Antennas,” in *IEEE ICNP*, 2006.

[14] Y. Yu, B. Krishnamachari, and V. K. Prasanna, “Data Gathering with Tunable Compression in Sensor Networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 2, pp. 276–287, 2008.

[15] E. M. Smith and C. C. Pantelides, “Global Optimisation of General Process Models,” in *Glo. Opt. Eng. Des.* Springer, 1996, pp. 355–386.

[16] P. Bonami *et al.*, “An Algorithmic Framework for Convex Mixed Integer Nonlinear Programs,” *Disc. Opt.*, vol. 5, no. 2, pp. 186–204, 2008.

[17] S. Le Digabel, “Algorithm 909: NOMAD: Nonlinear Optimization with the MADS Algorithm,” *ACM TOMS*, vol. 37, no. 4, p. 44, 2011.

[18] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II,” *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[19] S. Ioannidis and E. Yeh, “Adaptive Caching Networks with Optimality Guarantees,” in *ACM SIGMETRICS*, 2016.

[20] Anonymous, “E3C3: Enhancing Energy Efficiency among Communication, Computation and Caching with QoI-Guarantee,” Tech. Report available at goo.gl/ugzi91.

[21] C. A. Floudas, *Deterministic Global Optimization: Theory, Methods and Applications*. Springer Science & Business Media, 2013, vol. 37.

[22] E. M. Smith and C. C. Pantelides, "A Symbolic Reformulation/Spatial Branch-and-Bound Algorithm for the Global Optimisation of Nonconvex MINLPs," *Comp. & Chem. Eng.*, vol. 23, no. 4, pp. 457–478, 1999.

[23] E. M. Smith, "On the Optimal Design of Continuous Processes," Ph.D. dissertation, Imperial College London (University of London), 1996.

[24] L. Liberti, "Reformulation and Convex Relaxation Techniques for Global Optimization," *4OR: A Quarterly Journal of Operations Research*, vol. 2, no. 3, pp. 255–258, 2004.

[25] G. P. McCormick, "Computability of Global Solutions to Factorable Nonconvex Programs: Part IConvex Underestimating Problems," *Mathematical Programming*, vol. 10, no. 1, pp. 147–175, 1976.

[26] T. Achterberg, "SCIP: Solving Constraint Integer Programs," *Mathematical Programming Computation*, vol. 1, no. 1, pp. 1–41, 2009.

[27] L. Liberti, "Reformulation and Convex Relaxation Techniques for Global Optimization," Ph.D. dissertation, Imperial College London, 2004.

[28] OPTI Toolbox, "A Free Matlab Toolbox for Optimization," https://www.inverseproblem.co.nz/OPTI/index.php/Main/HomePage, [Online; accessed 28-Jun-2017].

[29] A. Wächter and L. T. Biegler, "On the Implementation of an Interior-point Filter Line-search Algorithm for Large-scale Nonlinear Programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.