UC Davis

UC Davis Previously Published Works

Title

Parallelizing Under-Determined Inverse Problems for Network Applications

Permalink

https://escholarship.org/uc/item/6s886344

Authors

Malboubi, Mehdi
Garrison, Joshua
Chuah, Chen-Nee
et al.

Publication Date

2016

Peer reviewed

eScholarship.org Powered by the California Digital Library
University of California

Parallelizing Under-Determined Inverse Problems for Network Applications

Mehdi Malboubi, *Student Member, IEEE*, Joshua Garrison, Chen-Nee Chuah, *Fellow, IEEE* and Puneet Sharma, *Fellow, IEEE*

***Abstract*—In this paper, we introduce a new technique for partitioning a large-scale under-determined linear inverse problem into multiple smaller sub-problems that can be efficiently solved independently, and in parallel. When it is impossible or inefficient to solve a large-scale under-determined linear inverse problem, this technique can be used to significantly speed up the computation process without compromising the accuracy of the solution. We present numerical results that show the effectiveness of this approach when applied to network inference problems including traffic matrix estimation and network anomaly detection, both are important for managing large, complex networks and cyber-security. Our proposed framework is applicable to other emerging applications in computational intelligence that can be formulated as UDLI problems.**

Keywords: **Network Measurement and Inference, Traffic Matrix Estimation, Anomaly Detection.**

I. Introduction

Many problems in networking, signal processing, communications, machine learning, and Computational Intelligence (CI) are formulated as Under-Determined Linear Inverse (UDLI) problems [1] [2] [3] [4] [5]. In UDLI problems, the number of measurements/observations are smaller than the number of unknown attributes of interests. This is mainly due to the practical limitations and hard constraint of measurement resources where it is expensive and impossible or infeasible to collect as many measurements as needed. Accordingly, UDLI problems are naturally ill-posed as there are not sufficient observations to uniquely and accurately estimate the solution [5]. Therefore, side information from different perspectives and multiple sources must be incorporated into the problem formulation and/or underline intelligent computation engine to improve the estimation precision, and cope with the complexity of inference process [1] [5] [6] [7] [8] [9] [10] [11] [12] [13].

A class of important UDLI problems is Network Inference (NI) or Network Tomography (NT) problems. Network Inference (NI) or network tomography is the study of the internal characteristics of a network using a set of limited end-to-end and/or aggregated observations that can be measured directly. An important NI problem is Traffic Matrix Estimation (TME) which provides essential information for many networking applications including network design, operation/management, and security. A Traffic Matrix (TM) is the vector representation of the size of all Origin-Destination Flows (ODF) where a flow is the sequence of packets, sharing common network identification attributes that can be extracted from packet header fields. In TME, the main goal is to provide the fine-grained estimate of the size of all network ODFs [8] [14]. Due to limited measurement resources (e.g. flow-table/TCAM entries, storage capacity, and processing power) in network monitoring infrastructures, it is often infeasible and/or inefficient to directly measure the size of every flow (using NetFlow or sFlow) in large-scale networks [15]. In this case, a TME problem is formulated as an UDLI problem where the number of *known* SNMP link-load and/or flow aggregated measurements are less than the number of *unknown* ODFs [8].

Existing studies generally attempt to infer TMs in a centralized manner where all measurements are collected and processed at a central node by applying domain-specific inference techniques and/or machine learning/intelligence algorithms [1] [7] [8] [9] [10] [11] [12] [13] [16]. However, the high computational complexity of these centralized inference techniques/algorithms hinders their deployment in large-scale and dynamic production networks where network inference processes/algorithms must be performed at much faster time scales for practical deployment [17]. Therefore, it is important to provide computationally-efficient solutions for solving large-scale NI problems.

Recently in [18] and [19], we proposed an efficient and robust framework for solving large-scale UDLI problems in a decentralized manner. Our goal was to significantly reduce the computational burden of large-scale network inference problems without compromising the accuracy of the solution. In this framework, called Multiple Description Fusion Estimation (MDFE), a large-scale network inference problem is decentralized by intelligently dividing it into smaller sub-problems (using greedy algorithms) and solving them independently and in parallel. The results, solved in respective sub-spaces and referred to as multiple local descriptions, are then fused together to reconstruct the ultimate global estimate. When (1) the observation matrix of a large-scale UDLI problem is sparse, and (2) the *partitioning* of a large-scale UDLI problem into multiple sub-problems with independent set of unknowns is impossible, the MDFE framework is able to compute an alternative solution by finding a convex combination of redundant (overlapping) local estimates.

The distributed nature of the MDFE framework is com-

At the time of this work, Mehdi Malboubi and Joshua Garrison were with the Dept. of ECE at UC Davis. E-mail: {mmalboubi, jcgarrison@ucdavis.edu}

Chen-Nee Chuah is Professor at the Dept. of ECE at UC Davis. E-mail: chuah@ucdavis.edu

Puneet Sharma is principal research scientist at HP-Labs, Palo Alto. E-mail: puneet.sharma@hpe.com

patible with today's multi-core/virtualized computing architectures and cloud-computing infrastructures where a large-scale problem can be divided into smaller sub-problems and distributed among multi-parallel processing units. In addition, by exploiting the redundancy between sub-spaces, MDFE can enhance the robustness against noise and failures in network measurement and monitoring infrastructures. The MDFE framework can also be applied to solve large-scale NI problems in Software Defined Networks (SDNs) where the data plane and control planes are separate [9]. A SDN controller typically has a global view of the network and can poll various switches and routers for measurement statistics. Under MDFE, the SDN controller or central network controller collects flow-aggregated measurements from network devices, partitions the UDLI problem into sub-problems, and solves them independently. The intermediate results (or local estimates) are then fused together to derive the global estimate, by considering the redundancy among local descriptions. In [18] and [19], we demonstrated that the MDFE is a flexible framework that can be applied to different UDLI problems, and it is also complementary to the inference techniques proposed previously for solving specific network inference problems. In our previous studies in [18], [19] and [20], we have shown the compatibility of the MDFE framework with different NI techniques in different applications including traffic matrix estimation, traffic matrix completion, and loss inference.

Building upon our MDFE framework in [18], [19] and [20], this paper introduces a new partitioning technique for the MDFE framework which remarkably speeds up the computation process of solving large-scale UDLI problems without compromising the accuracy of estimation. This partitioning technique is based on our upper bound for the least square estimation error. We evaluate the performance of our proposed MDFE framework by applying it to TM estimation and a new application, namely, network anomaly detection. Due to space limitations, we occasionally refer to [21] for further results and mathematical proofs.

It should be noted that although in this paper we consider TME in the context of large-scale IP networks, the MDFE is also applicable to a variety of emerging CI problems which can be formulated as UDLI problems. For instance, estimating computation loads in cloud networks or multi-tenant data centers, inferring origin-destination vehicular traffic streams based on traffic intensity observed at smart intersections or road side units along highways, or detecting anomalous behaviors (e.g., attacks, fraud) of new threats in large-scale and complex computer networks with increasing attack surface that arise with the proliferation of Internet-of-Things (IoT) devices [22] [23] [24] [25] [26] [27]. In such cases, the MDFE framework acts as a complementary solution to overcome the high computational cost of implementing large-scale CI algorithms.

II. Case Study: TM Estimation with MDEF

Traffic matrix estimation is the main network inference problem that is used to demonstrate the efficacy of our MDFE framework in this paper. TME is formulated as an UDLI problem. Eq.(1) represents the general form of TME problem in a network with N nodes, $n := N(N-1)$ ODFs, and m links where $m < n$. In this equation, first, Y is an $(m \times 1)$ measurement vector of known link-loads and/or flow-aggregated measurements (provided via network management or OpenFlow protocols [28] [29]); second, H is typically a sparse and binary routing and/or aggregation matrix where the ij^{th} entry h_{ij} shows the contribution of the j^{th} flow on the i^{th} link and/or flow-table entry, and third, X is an $(n \times 1)$ vector where the j^{th} entry x_j represents the size of j^{th} ODF over a specific time interval. Given Y and H, the main goal in solving the TME problem is estimating the unknown vector X. The general solution to this problem is of the form $\hat{X} = X + X_H^{\mathscr{N}}$ with $X_H^{\mathscr{N}}$ denoting a solution from the span of the null space of H, denoted by $\mathscr{N}(H)$.

$$Y = HX \tag{1}$$

Figure 1 shows the general block-diagram of the MDFE framework where link-load and flow-aggregated measurements from all network devices, including SDN enabled devices, are collected and processed at the Central Network Controller (CNC). In the MDFE framework, the original (global) UDLI problem represented in Eq.(1) is partitioned into L local sub-problems represented in Eq.(2). These sub-problems are independently solved and local/sub-space estimates $\{\hat{X}_i\}_{i=1}^{L}$ (i.e. local descriptions) are then fused together to provide an accurate solution in a computationally-efficient way. The fusion process is accomplished by applying appropriate weights to each local estimate. This process is done at CNC where all local descriptions are available. Eq.(3) describes the fusion process (denoted by operator $\oplus$) where a convex combination of local descriptions is computed as the ultimate global estimate $\hat{X}_P^F$. Note that, assuming H is a sparse matrix, then not all entries of X can be observed in all sub-spaces. The accuracy of the ultimate estimate $\hat{X}_P^F$ is a *joint* function of sub-space estimation technique, partition P, and fusion process F. Since there is an NP-hard set partitioning problem at the core of this joint optimization problem, we *decouple* and independently address these three steps [18].

$$Y = HX \Leftrightarrow \begin{bmatrix} Y_1 \\ \hline \vdots \\ \hline Y_L \end{bmatrix} = \begin{bmatrix} H_1X_1 \\ \hline \vdots \\ \hline H_LX_L \end{bmatrix} \tag{2}$$

$$\hat{X_P^F} = \oplus_{i=1}^{L} \omega_i^F \hat{X}_i \quad \text{where} \quad \hat{x_P^F}_j = \sum_{i=1}^{L} \omega_{ji} \hat{x}_{ji}$$

$$\sum_{i=1}^{L} \omega_{ji}^F = 1 \ \text{ and } \ 0 \le \omega_{ij}^F \le 1 \quad \text{for } i = 1, ..., L \text{ and } j = 1, ..., n \tag{3}$$

Although our illustrative example based on Figure 1 considers one central SDN controller with the global view of the network, the MDFE framework is also compatible to the distributed controller scenarios. After obtaining partition P, each partition can be assigned one SDN controller that will poll various switches and routers to collect measurement statistics and compute local estimates. Through coordination between the SDN controllers (and perhaps involving a parent

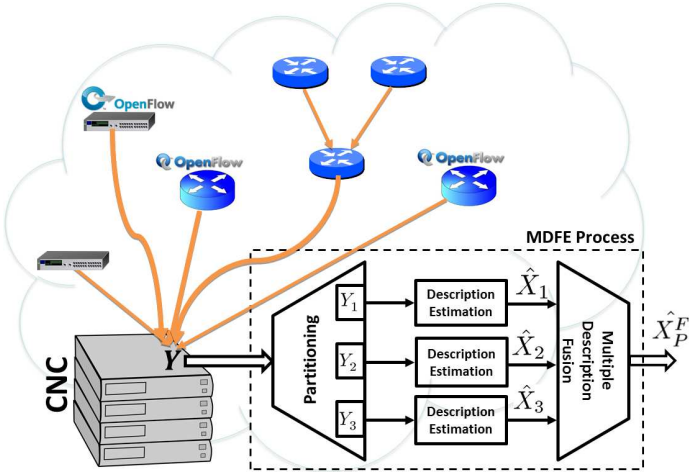

Fig. 1: The general block diagram of MDFE process where SDN enabled devices use OpenFlow protocol, as an example.

central network controller), these local estimates can then be fused to derive the global estimate. Due to space limitation, we focus on the case of one central SDN controller for the rest of our discussion.

III. The Implementation of MDFE

Many network inference problems in networking, communication, and signal processing are formulated as UDLI problems that are solved by computing the Least Norm Estimates (LNE). To develop the basic theory of MDFE, it is assumed that the underlying sub-space estimation method is the LNE. It should be noted that a well-designed MDFE framework is compatible with different underline network inference techniques [18], and its applications can be effectively extended into cases where computationally expensive machine learning/intelligence algorithms are the underline NI methods.

A. Construction of Multiple Descriptions

Let $I = \{1, 2, ..., m\}$ denotes the set of all indices of observations and I_i denotes the i^{th} set of disjoint indices of measurements where $I = \bigcup_{i=1}^{L} I_i$ and $I_i \bigcap I_j = \emptyset$ for $i \neq j$. Then, the partition P of set I is formed, and thus, $P = \bigcup_{i=1}^{L} I_i$. Let $J = \{1, 2, ..., n\}$ denotes the set of all indices of unknowns and J_i denotes the i^{th} set of indices of unknowns (observed at i^{th} sub-space) where $J = \bigcup_{i=1}^{L} J_i$. It should be noted that: 1) the intersection of J_i and J_j (for all i and j in $\{1, ..., L\}$ where $i \neq j$) is not necessarily empty, and 2) considering the sparsity of matrix H in practical network inference problems, then in general, $J \neq J_i$ (for $i \in \{1, ..., L\}$). Now, let $Y_i := \{y_k\}_{k \in I_i}$, $H_i := H(I_i, J_i)$ and $X_i := \{x_k\}_{k \in J_i}$. Thus, the original global problem $Y = HX$ in Eq.(1) (where H is of size $m \times n$) is divided into L sub-problems as $Y_i = H_i X_i$ (see Eq.(2)) where H_i is an $m_i \times n_i$ matrix. Considering the fact that the original routing (i.e. observation) matrix is sparse, the i^{th} local routing matrix H_i is also sparse where $m_i < m$, $n_i < n$ and with high expectations $m_i < n_i$. These local under-determined inverse problems can be achieved by removing unknown entries that are not observed in corresponding sub-problems.

Assuming that the input vector X does not include unusual inputs that differ in size by large order of magnitudes, the least norm estimates for both global UDLI problem $Y = HX$ and local UDLI problems $Y_i = H_i X_i$ (for $i = 1, ..., L$) are computed using the pseudoinverse of observation matrices H and H_is [30], denoted by operator †. Eq.(4) and Eq.(5) respectively denote the closed-form solutions of global and local LNEs where the observation matrices H and H_i (for all $i \in 1, ..., L$) are sparse and full low-rank. Here, the pseudoinverses of H and H_is are accurately computed using the Singular Value Decomposition (SVD) with computation complexity $O(mn^2)$ and $O(m_i n_i^2)$ flops, respectively. It should be also noted that, the solution of global and local problems could be different because the null-space of H are not necessarily equal to the null space of H_is.

$$\hat{X}^G = H^{\dagger} Y = (H^T (H H^T)^{-1}) Y \tag{4}$$

$$\hat{X}_i = H_i^{\dagger} Y_i = \left(H_i^T (H_i H_i^T)^{-1}\right) Y_i \quad \text{for } i = 1, ..., L \tag{5}$$

B. Partition Design in MDFE Process

The accuracy of the ultimate estimate $\hat{X}_P^F$ depends on the construction of the local observation matrix H_i $(i = 1, ..., L)$ that is determined by the design of the set I_i in partition P. However, in reality, it is very hard to *directly* relate the ultimate performance of the sub-space inference technique to the construction of H_i [31]. In practice, by accepting the unavoidable sacrifice in performance, other metrics are utilized to measure the estimation performance as a function of some characteristics of the sub-space observation matrix [18] [19] [31] [5]. To achieve the best possible performance using such a metric, the design of partition P can be formulated as an integer optimization problem which is an NP-hard problem [32]. To simplify this problem and enhance the performance of MDFE, we first introduce an auxiliary metric in Theorem 1, and then a pseudo-optimal partitioning algorithm is developed to design partition P. In Theorem 1, we show that the Mean Square Error (MSE) for both global and MDFE estimates can be respectively bounded proportional to:

$$SCN := \sqrt{n}\kappa_2(H) \quad \text{and} \quad SCN_i := \sqrt{n_i}\kappa_2(H_i) \tag{6}$$

where $\kappa_2(H)$ denotes the Condition Number (CN) of the corresponding matrix. Our main goal in computing this upper bound is to show that the ultimate accuracy of estimation technique is proportional with the condition number of the observation matrix and the number of unknowns. This is intuitive since solving UDLI problems with an observation matrix with a lower condition number, and with less number of unknowns can potentially produce more accurate and stable solutions [33].

Consequently, partition P can be designed by Alg. 1 where metric SCN_i is used to measure the effectiveness of sub-spaces. Comparing to greedy partitioning algorithm in [18] (where it only considers metric $\kappa_2(H)$), Alg. 1 can provide a better estimation accuracy in the majority of iterations,

Algorithm 1 : Partitioning Algorithm

Initialization: $I = \{1, ..., m\}$ and $i = 1$
while $i \leq L$ **do**
- Construct I_i by sequentially choosing K rows of H with lowest $\sqrt{n_i}\kappa_2(H_i)$
- Set $I = I \backslash I_i$ and $i = i + 1$
end while

as it considers the multiplication of both related metrics in accuracy, that is, $\kappa_2(H)$ and n; detailed derivation and further discussion can be found in [21]. This algorithm starts from the first row of H and sequentially chooses the row that minimize the SCN of the sub-matrix. This continues to complete the first sub-space I_1 with K rows where $K := \lceil m/L \rceil$, and L is appropriately designed [18]. After removing these K rows from H, the algorithm repeats from the beginning. Also, the fundamental search strategy in this algorithm is a pseudo-optimal scheme and, of course, more effective search strategies can be used for the design of partition P using the same metric. In particular, heuristics algorithms can be utilized to solve the NP-Hard partition design problem. For this purpose, the cost function of a well designed heuristic algorithm, such as genetic algorithm [34] or particle swarm algorithm [35], not only can appropriately target the same metric in Alg. 1 but also it can target the ultimate estimation accuracy, as we have shown in [36].

Theorem 1: Assume LNE is the sub-space estimation technique and let X, $\hat{X}^G$ and $\hat{X}^F_P$ respectively represent the true unknown vector, its global estimate (using Eq.(4)) and its MDFE estimate (using Eq.(3)) for some partition P and fusion weights $\{\omega_i^F\}_{i=1}^L$ where $\hat{X}_i$ is computed using Eq.(5). Then, the Mean Square Errors (MSE) of global and MDFE estimates are bounded as: (please refer to [21] for complete proof).

$$\begin{aligned} \left\| \hat{X}^G - X \right\|_2 &\leq \|X\|_2 + \sqrt{n}\kappa_2(H) \|X\|_2 \\ \left\| \hat{X}^F_P - X \right\|_2 &\leq \|X\|_2 + \sum_{i=1}^{L} \left(\sqrt{n_i}\kappa_2(H_i) \|X_i\|_2 \right) \end{aligned} \tag{7}$$

C. Fusion Process in MDFE

The fusion process F plays an important role in the MDFE for providing accurate estimates, and it is performed by computing the convex combination of local descriptions where different weighting functions, with low computation overhead [18], are applied to local estimates. The fusion function ω_{SCN}, chooses x_j from the sub-space with the lowest SCN, and fusion function ω_{Avg} computes the average of the observed x_j's estimated at different sub-spaces. The optimal fusion weight ω_{Opt} is computed by the following supervised procedure [20] [21]. First, apply MDFE framework on the training data set $X_{Test} := \{X_1, ..., X_t, ..., X_{T_0}\}$ where column vector $X_t \in \mathbb{R}^n$ for $t = 1, ..., T_0$ and T_0 is small compare to the size of the data (e.g. $\frac{T_0}{T}$ =0.1). Then for each t in $t = 1, ..., T_0$, generate LNE matrix $\hat{X}_t = \left[\hat{X}_{t_1}, ..., \hat{X}_{t_L} \right]$ with size $n \times L$ (unobservable entries in each sub-space are zero), containing L local descriptions. Next for each t, calculate the optimal fusion weight using Eq.(8), where ω_t is an $n \times L$ matrix that equalizes local descriptions, and $\mathcal{RS}$ and $.*$ denote row-sum and componentwise multiplication operators, respectively. The optimal weight ω_{Opt} (an $n \times L$ matrix) is then computed using Eq.(9). More effective and efficient fusion methods for MDFE can also be implemented using information fusion techniques [37] [38].

$GE = \frac{1}{T}\sum_{t=1}^{T} \frac{\left\|X(t) - \hat{X}^G(t)\right\|_2}{\|X(t)\|_2}$,	$FE_w = \frac{1}{T}\sum_{t=1}^{T} \frac{\left\|X(t) - \hat{X}^F_w(t)\right\|_2}{\|X(t)\|_2}$, $Gain_w =$
$\Delta_p = 100 \times \frac{O(mn^2) - max(\{O(m_i n_i^2)\}_{i=1}^L)}{O(mn^2)}$,	$\Delta_s = 100 \times \frac{O(mn^2) - \sum_{i=1}^L O(m_i n_i^2)}{O(mn^2)}$

TABLE I: Performance evaluation criteria.

$$\begin{aligned} \omega(:,:,t) = & \min_{\omega_t} \left\| X_t - \mathcal{RS}\left(\omega_t . * \hat{X}_t\right) \right\|_2 \\ & \text{s.t. } \omega_t \geq 0 \text{ and } \{\mathcal{RS}\left(\omega_t(j,:) = 1\right)\}_{j=1}^n. \end{aligned} \tag{8}$$

$$\omega_{Opt}^{ij} = \frac{1}{T_0}\sum_{t=1}^{T_0} \omega(j,i,t) \quad \text{for } i = 1, ..., L, \text{and} \quad j = 1, ..., n \tag{9}$$

IV. The Performance Evaluation of MDFE

To evaluate the performance of MDFE framework, the well-known 14-Node Tier-1 PoP Topology [18] [8] is considered. The routing matrix H of this network is a sparse and binary matrix of size 50×182. We consider synthetics traffic traces, including a set of T vectors $\{X(t)\}_{t=1}^T$ where $T = 250$ and for each t, $X(t)$ is an $n \times 1$ vector. Synthetic traffic traces are generated using three different distributions [8]: 1) Uniform distribution where $x_j \sim U(100, 500)$; 2) Gaussian distribution where $x_j \sim \mathcal{N}(\mu_j, 40)$ (where $\mu_j \sim U[100, 500]$); 3) Poisson distribution where $x_j \sim Pois(\lambda_j)$, $\lambda_j \sim U[100, 500]$. Moreover, two more realistic traffic distributions known as: 4) Log-Normal distribution where $x_j \sim Log - Normal(\mu, \sigma)$ with mean μ =100 and σ =2, and 5) Weibull distribution where $x_j \sim Weibull(\lambda_w, k_w)$ with scale parameter λ_w =170 and shape parameter k_w =1, are considered [39]. Moreover, in [21], the performance of MDFE framework on Abilene and Geant networks have been evaluated.

The performance of the MDFE is evaluated using various criteria that are introduced in Table I where $\hat{X}^G(t)$ denotes the global estimate and $\hat{X}^F_w(t)$ denotes the MDFE estimate resulted from MDFE framework. The subscription w denotes the type of the fusion function used in the MDFE process, which are defined in Section III-C and it could be SCN, Avg, and Opt. Quantities GE and FE_w respectively measure the accuracy of both global and MDFE estimates in the normalized L_2 sense. Also, $Gain_w$ quantifies the performance improvement using our MDFE framework compared with global estimation case.

Parallel and sequential processing gains (Δ_p and Δ_s) measure the reduction in computational complexity using the MDFE structure where the computational complexity of LNE is approximated as: $C(m, n) = mn^2$ [18]. Note that the sequential processing gain can also be an indication of the reduction in required processing power using the MDFE framework.

Distribution	GE	FE_{Avg}	FE_{SCN}	FE_{Opt}	$G_{Avg}\%$	$G_{SCN}\%$	$G_{Opt}\%$	Δ_p	Δ_s
Uniform	0.4271	0.3604	0.3763	0.3531	15.62	11.89	17.33		
Gaussian	0.4552	0.4075	0.4412	0.3508	10.48	3.07	22.94		
Poisson	0.4630	0.3819	0.3966	0.3266	17.52	14.34	29.46	94.89	82.78
Log-Normal	0.4240	0.3482	0.3668	0.3357	17.87	13.49	20.83		
Weibull	0.4337	0.3726	0.3882	0.3660	14.08	10.49	15.61		
Uniform	0.4271	0.3536	0.3645	0.3473	17.21	14.66	18.68		
Gaussian	0.4552	0.3995	0.4272	0.3596	12.24	6.15	21.00		
Poisson	0.4630	0.3664	0.3817	0.3241	20.86	17.55	30.00	97.87	90.75
Log-Normal	0.4240	0.3394	0.3524	0.3274	19.95	16.88	22.78		
Weibull	0.4337	0.3675	0.3787	0.3611	15.26	12.68	16.74		

TABLE II: The performance of TM estimation with MDFE for $L = 5$, and $L = 7$ (rows in gray) of the 14-PoP network [18]. Please refer to [21] to see the results on Abilene and Geant networks for different values of L.

A. MDFE Applications (1): TM Estimation

The performance of MDFE framework on TM estimation for different distributions of input traffic is shown Table II. It indicates that the MDFE framework with least norm estimation technique can noticeably enhance the estimation accuracy compared with the global LNE. For the majority of traffic distributions, this improvement is obtained over almost all iterations, as we have shown in [21]. Among these, the estimation precision of the optimal fusion technique [20] is higher than heuristic fusion techniques, which can be simply applied without using learning data. The SCN fusion method is of particular importance in distributed implementation of MDFE framework [18], where it can significantly reduce the communication costs and/or delays. This table also shows that the MDFE framework is able to remarkably enhance the processing gains. These high processing gains are obtained where communication delays are negligible in comparison with processing times that is compatible with the architecture of today's multi-core processors. The high processing gain in the sequential MDFE case (Δ_s), where local descriptions are sequentially produced, indicates that the MDFE framework can also reduce the required processing power due to the fact that the processing power is a function of the complexity of the process.

In summary, the MDFE framework with partitioning Alg. 1 can significantly speed up the process of computation without compromising the accuracy of solution in the majority of cases. In fact, comparing to our greedy partitioning algorithms in [18], the MDFE framework with partitioning Alg. 1 can achieve higher improvement ratio, defined as the percentage of iterations in which MDFE outperforms the estimation accuracy of centralized/global LNE. This is mainly due to the use of both the condition number of observation matrix ($\kappa_2(H)$) and the number of unknowns (n) in partitioning Alg. 1, as in Theorem 1 we have shown the upper bound error is proportional to $\sqrt{n}\kappa_2(H)$. Please refer to [21], [18] and [20] for more details and further results where we have shown the compatibility of the MDFE frameworks on Abilene and Geant networks, and analyzed the performance of MDFE for different values of L and in the presence of noise and failure.

B. MDFE Applications (2): Network Anomaly Detection for Cyber Security

To deal with influx of new threats in large-scale and complex computer networks, along with increasing attack surface that arise with the proliferation of Internet-of-Things (IoT) devices, the ability to detect anomalous behaviors (e.g., attacks, fraud) in a timely manner is crucial. The MDFE framework can be applied to network anomaly detection to speed up the process without compromising accuracy. For this purpose, we model the dynamic of the network's flows as [26]:

$$X(t+1) = CX(t) + W(t) \ , \ Y(t) = HX(t) + V(t) \qquad (10)$$

where $W(t)$ and $V(t)$ are respectively process and measurement noises which are modeled as WGN [14], [26]. Also, matrix C (with size $n \times n$) models network's flow dynamics and matrix H (with size $m \times n$) is the routing matrix.

In the presence of anomaly, the set of network flows $X(t)$ are contaminated with the output of a two states Markov-Chain (MC) with silent and active states, denoted by S and A, respectively. The transition probability from state S to A is denoted by P_{SA} and the transition probability from state A to S is denoted by P_{AS}. In the silent state, there is no anomaly and $x_i(t)$ remains unchanged. Assuming that the MC is in active state in the interval $\Delta = [t_1, t_2]$, then $x_i(\Delta)$ is corrupted as $x_i(\Delta) = \alpha \frac{x_i(t_1)+x_i(t_2)}{2}$ where α is a constant, controlling the power of the anomaly.

Anomaly detection is accomplished using the method introduced in [26] where in the first step a Kalman filter is used to filter out the normal traffic. This is done by comparing our future predictions of the traffic matrix state to an estimate of the actual traffic that is made using more recent measurement data than those used for prediction. In the second step the residual filtered process (denoted by $\tau(t)$ in [26]) is then examined for anomalies (please refer to [26] for details).

In our model, the existence of an anomaly in the residual process is recognized by a falling-edge followed by a rising-edge. In the global case, the anomaly detection is performed by comparing the residual process $\tau(t)$ with a particular threshold. In our MDFE framework, the residual processes $\{\tau_i(t)\}_{i=}^{L}$ from sub-spaces are fused by computing the average of residual processes using ω_{Avg} [21], that is, $\tau_{Avg}(t) = \oplus_{i=1}^{L} \omega_{Avg_i} \tau_i(t)$. Accordingly, the anomaly detection is performed by comparing the residual process $\tau_{Avg}(t)$ with an appropriate threshold. In both cases, we experimentally set the threshold to get the best achievable performances.

To evaluate the performance, we consider the 14-PoP network in [18]. The flows are generated according to Eq.(10) [26] where, for simplicity and according to [14], matrix C is modeled as $C = I_n$. The link-load measurement process $Y(t)$ is a noisy process with SNR=43 dB [1]. Here, to consider a harder scenario for anomaly detection, we assume that α=2. To apply our MDFE framework, we use Alg. 1 to partition the set of observations into $L = 7$ sub-spaces where the estimation in sub-spaces is performed by applying the Kalman Filter on each sub-problem.

The performance and reliability of network anomaly detection process are evaluated by computing the probability of detection (P_d) and the probability of false alarm (P_{fa}),

(π_A,P_{AS})	$\pi_A = 0.05$	$\pi_A = 0.1$	$\pi_A = 0.15$	$\pi_A = 0.2$
$P_{AS} = 0.1$	(0.81,0.84)	(0.68,0.76)	(0.63,0.71)	(0.60,0.68)
$P_{AS} = 0.2$	(0.70,0.84)	(0.60,0.75)	(0.57,0.69)	(0.56,0.67)
$P_{AS} = 0.3$	(0.63,0.82)	(0.57,0.71)	(0.55,0.67)	(0.53,0.64)
$P_{AS} = 0.4$	(0.61,0.79)	(0.53,0.68)	(0.52,0.64)	(0.51,0.63)
$P_{AS} = 0.5$	(0.56,0.75)	(0.51,0.66)	(0.50,0.63)	(0.49,0.61)

TABLE III: The pair of probability of detections (P_d^G,P_d^{FE}).

(π_A,P_{AS})	$\pi_A = 0.05$	$\pi_A = 0.1$	$\pi_A = 0.15$	$\pi_A = 0.2$
$P_{AS} = 0.1$	(0.33,0.31)	(0.36,0.34)	(0.40,0.36)	(0.40,0.37)
$P_{AS} = 0.2$	(0.39,0.33)	(0.41,0.35)	(0.41,0.36)	(0.42,0.37)
$P_{AS} = 0.3$	(0.40,0.33)	(0.41,0.34)	(0.42,0.35)	(0.41,0.35)
$P_{AS} = 0.4$	(0.42,0.34)	(0.40,0.33)	(0.41,0.33)	(0.41,0.34)
$P_{AS} = 0.5$	(0.39,0.31)	(0.39,0.31)	(0.39,0.33)	(0.39,0.33)

TABLE IV: The pair of probability of false alarms (P_{fa}^G,P_{fa}^{FE}).

defined in Eq.(11). These two metrics are evaluated through a Monte-Carlo simulation and their averages are shown in Table III and Table IV. In this table, $\pi_A = \frac{P_{SA}}{P_{SA}+P_{AS}}$ denotes the long term proportion of time spent in state A where π_A changes from 0.05 to 0.2, which is a reasonable range in networking applications [40]. Having, π_A and P_{AS}, we can determine P_{SA} as $P_{SA} = P_{AS}\frac{\pi_A}{1-\pi_A}$. Tables III and IV show the performance of our MDFE framework in comparison with the global case where superscripts G and FE denote the performance metrics computed in global and MDFE cases, respectively. It is clear that by applying MDFE, the detection performance is improved. This improvement in detection performance is achieved while the computational complexity is significantly reduced because smaller-size problems can be solved in parallel.

$$\begin{aligned} P_d &= \frac{1}{n}\sum_{j=1}^{n} Pr\left(\text{correct anomaly detection for } j^{th} \text{ flow}\right) \\ P_{fa} &= \frac{1}{n}\sum_{j=1}^{n} Pr\left(\text{detecting anomalies when there are not (for } j^{th} \text{ flow)}\right) \end{aligned} \tag{11}$$

It is interesting to note that, as it was expected, by increasing π_A and P_{AS} (or equivalently in the presence of more-frequent short-bursty anomalies) the detection performance is degraded. Among these, the probability of detection is decreased more noticeably, that is, it is more difficult to detect an anomaly with the rapid transitions between two silent and active states. This behavior is reasonable and it shows that short-bursty anomalies/attacks are more difficult to detect. Accordingly, it is of particular importance in network monitoring to use more powerful anomaly detection techniques for detecting anomalies with more-frequent short-bursty behaviors.

Figure 2 shows the gain in the performance of anomaly detection using the MDFE framework where GP_d and GP_{fa} are respectively defined as: $GP_d = 100 \times \frac{P_d^{FE}-P_d^G}{P_d^G}$ and $GP_{fa} = 100 \times \frac{P_{fa}^G-P_{fa}^{FE}}{P_{fa}^G}$. Although these gains are positive for different values of π_A and P_{AS}; however, for a fix π_A the gain is higher for larger values of P_{AS}. Hence, the MDFE framework can improve the detection performance in the presence of more-frequent short-bursty anomalies, which is of particular importance in network monitoring applications. Such a gain is achieved by fusing local estimates which can significantly reduce the effect of noise, and it can be increased

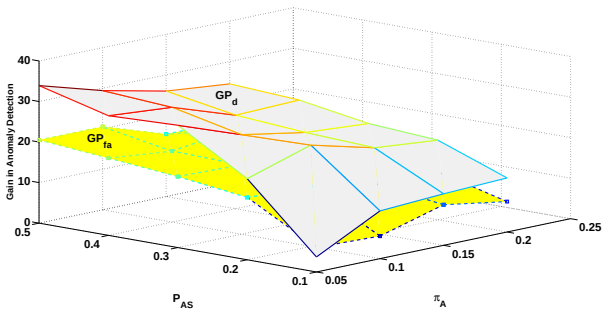

Fig. 2: Gain in anomaly detection using MDFE

by computing the optimal fusion weights [20].

V. MDFE and Computational Intelligence

The MDFE framework can be applied to a variety of emerging applications in Computational Intelligence (CI) that can be formulated as UDLI problems. Demand estimation in smart grids, urban traffic estimation, traffic intensity estimation and detection in vehicular networks, and network anomaly detection are the examples of large-scale applications [24] [25] [26] [27] where the MDFE framework can be applied to speed up the process of applying highly complex computational intelligence algorithms, and provide a timely estimate of unknowns without compromising the accuracy of solution. In this regard, MDFE is complementary to a variety of CI methods by reducing the associated computational cost, a commonly perceived bottleneck for large-scale CI applications.

In networking applications, for example in TME using neural networks [10], the MDFE framework can partition a large-scale TME problem into multiple sub-problems where smaller-size neural networks can be quickly trained to provide sub-space estimate of unknown TMs. Furthermore, the fusion operation in MDFE can be implemented using advanced data fusion techniques [37] [38]. For example a neural network can be trained to optimally fuse local estimates. Accordingly, an alternative accurate estimate of TMs can be quickly obtained.

In addition, the MDFE framework can be used to speed-up the implementation of Evolutionary Optimization Algorithms (EOA), such as Genetic Algorithm (GA), where a near-optimal solution to a large-scale optimization problem can be obtained by running the principles of natural evolution on populations of solution representations (e.g. chromosomes in GA) over the large number of iterations. For example in solving large-scale NI problems, the MDFE framework can remarkably reduce the complexity of evaluating the target fitness function (as the main highly complex component of EOAs), and accordingly, speed-up the the process of converging to a near-optimal solution in large-scale evolutionary optimization problems [36]. Moreover, evolutionary algorithms can be utilized to solve the NP-hard partition design problem in MDFE. For this purpose, the cost function of a well designed heuristic algorithm, such as genetic algorithm [34], can appropriately target the same metric in Alg. 1 to design the partition P. Furthermore, EOAs can directly target the ultimate estimation accuracy (e.g. FE_w in Table I) to design a more effective

partition P in the MDFE framework with higher estimation accuracy [36].

VI. Conclusion

We developed a new algorithm for partitioning large-scale UDLI problems into multiple smaller sub-problems, based on our theoretical upper bound for the least square estimation error. We showed that by applying our MDFE framework, the computational complexity of solving a large-scale UDLI problem is significantly reduced without compromising the accuracy of solution. This is of particular importance where the computational complexity is the main bottleneck in applying effective machine learning/intelligence algorithms in large-scale CI problems. To show the effectiveness of MDFE framework, its performance was evaluated on network TM estimation and anomaly detection.

References

[1] Q. Zhao, Z. Ge, J. Wang, and J. Xu, "Robust traffic matrix estimation with imperfect information: Making use of multiple data sources," *ACM-SIGMETRICS*, 2006.

[2] L. Bai and J. Choi in *Low Complexity MIMO Detection*, Springer, 2012.

[3] K. Batenburg and W. Kosters, "A neural network approach to real-time discrete tomography," *Int. Workshop on Combinatorial Image Analysis (IWCIA)*, 2006.

[4] S. Mehrjoo, A. Sarrafzadeh, and M. Mehrjoo, "Swarm intelligent compressive routing in wireless sensor networks," *Wiley, An international journal on computer intelligence*, 2014.

[5] R. Aster, B. Borchers, and C. Thurber in *Parameter Estimation and Inverse Problems*, Academic Press, 2012.

[6] Y. C. Eldar and G. Kutyniok in *Compressed Sensing: Theory and Applications*, Cambridge Univ. Press, 2012.

[7] R. M, Y. Zhang, W. Willinger, and L. Qiu, "Spatio-temporal compressive sensing and internet traffic matrices," *IEEE/ACM Transactions on Networking*, vol. 20, pp. 662–676, 2012.

[8] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic matrix estimation:existing techniques and new directions," *ACM SIGCOMM*, 2002.

[9] J. Cao, D. Davis, S. Wiel, and B. Yu, "Time varying network tomography: Router link data," *Journal of American Statistical Association*, vol. 95, pp. 1063–1075, 2000.

[10] H. Zhoua, L. Tanb, Q. Zengc, and C. Wua, "Traffic matrix estimation: A neural network approach with extended input and expectation maximization iteration," *ELSVIER, Journal of Network and Computer App.*, vol. 60, pp. 220–232, 2016.

[11] A. Omidvar and H. S. Shahhoseini, "Intelligent ip traffic matrix estimation by neural network and genetic algorithm," *2011 IEEE 7th International Symposium on Intelligent Signal Processing (WISP)*, 2011.

[12] K. H. Reddy and P. Chakroborty, "A fuzzy inference based assignment algorithm to estimate o-d matrix from link volume counts," *Comput., Environ., and Urban Systems*, vol. 22, pp. 409–423, 1998.

[13] J. Yi, S. Fengjun, Z. Yang, and L. Linhao, "A ga approach for traffic matrix estimation," *2nd IEEE Int. Conf. on Broadband Network and Multimedia Technology (IC-BNMT '09)*, 2009.

[14] A. Nucci and K. Papagiannaki in *Design, Measurement and Management of Large-Scale IP Networks*, Cambridge Univ. Press, 2009.

[15] M. Malboubi, L. Wang, C.-N. Chuah, and P. Sharma, "Intelligent sdn based traffic (de)aggregation and measurement paradigm (istamp)," *IEEE INFOCOM*, 2014.

[16] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *Journal of American Statistical Association*, vol. 91, pp. 365–377, 1996.

[17] G. Huang, A. Lall, C.-N. Chuah, and J. Xu, "Uncovering global icebergs in distributed streams: Results and implications," *J. Network Syst. Manage.*, vol. 19, no. 1, pp. 84–110, 2011.

[18] M. Malboubi, C. Vu, C.-N. Chuah, and P. Sharma, "Decentralizing network inference problems with multiple-description fusion estimation (mdfe)," *IEEE/ACM Transactions on Networking*, vol. 24, pp. 2539–2552, August, 2016.

[19] M. Malboubi, C. Vu, C.-N. Chuah, and P. Sharma, "Decentralizing network inference problems with multiple-description fusion estimation (mdfe)," *IEEE INFOCOM*, April, 2013.

[20] M. Malboubi, C. Vu, C.-N. Chuah, and P. Sharma, "Compressive sensing network inferrence with multiple description fusion estimation," *IEEE GLOBECOM*, 2013.

[21] M. Malboubi, J. Garrison, C.-N. Chuah, and P. Sharma, "Parallelizing under-determined inverse problems with applications in network inference and anomaly detection," tech. rep., 2016. At: https://www.dropbox.com/s/66qiywc3qwkji3c/MDFE2ndJNLTechRep.pdf?dl=0.

[22] Y. Qiao, Z. Hu, and J. Luo, "Efficient traffic matrix estimation for data center networks," *IFIP Networking Conf.*, 2013.

[23] Z. Hu, Y. Qiao, and J. Luo, "Atme: Accurate traffic matrix estimation in both public and private datacenter networks," *IEEE Transactions on Cloud Computing*, May, 2016.

[24] R. Ranganatha, R. Qiu, Z. Hu, S. Hou, M. Pazos-Revilla, G. Zheng, Z. Chen, and N. Guo, "Cognitive radio for smart grid: Theory, algorithms, and security," *International Journal of Digital Multimedia Broadcasting*, 2011.

[25] H. Kamal, M. Picone, and M. Amoretti, "A survey and taxonomy of urban traffic management: Towards vehicular networks," *CoRR abs/1409.4388*, 2014.

[26] A. Soule, K. Salamatian, and N. Taft, "Combining filtering & statistical methods for anomaly detection," *ACM SIGCOMM*, 2005.

[27] W. Wang, D. Lu, X. Zhou, B. Zhang, and J. Mu, "Statistical wavelet-based anomaly detection in big data with compressive sensing," *EURASIP Journal on Wireless Communications and Networking*, vol. 269, 2013.

[28] M. Roughan, "A case study of the accuracy of snmp measurements," *JECE*, vol. 2010, pp. 33:1–33:7, Jan. 2010.

[29] P. Goransson and C. Black in *Software Defined Networks: A Comprehensive Approach*, Cambridge University Press, 2014.

[30] T. Hastier, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations*. New York: Springer-Verlag, 2009.

[31] M. Elad, "Optimized projections for compressed sensing," *IEEE Trans. on Signal Proc.*, vol. 55(12), pp. 5695–5702, 2007.

[32] S. Vries and R. Vohra, "Combinatorial auctions: A survey," *Journal on Computing*, vol. 15, pp. 284–309, 2003.

[33] S. Kay in *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice-Hal, 1993.

[34] P.-C. Chu and J. Beasley, "A genetic algorithm for the set partitioning problem," 1995.

[35] G. Anders, F. Siefert, and W. Reif, "A particle swarm optimizer for solving the set partitioning problem in the presence of partitioning constraints," *Int. Conf. on Agents and Artificial Intelligence (ICAART)*, 2015.

[36] M. Malboubi, Y. Gong, W. Xiong, C. Chuah, and P. Sharma, "Software defined network inference with passive/active evolutionary-optimal probing (sniper)," *IEEE, ICCCN*, 2015.

[37] J. Raol in *Data Fusion Mathematics: Theory and Practice*, CRC Press, 2015.

[38] Y. Zheng, "Methodologies for cross-domain data fusion: An overview," *IEEE Trans. on Big Data*, vol. 1, pp. 16–34, 2015.

[39] M. Crovella and B. Krishnamurthy in *Internet Measurement: Infrastructure, Traffic and Applications*, Wiley, 2006.

[40] K. Papagiannaki, N. Taft, S. Bhattacharyya, P. Thiran, K. Salamatian, and C. Diot, "A pragmatic definition of elephants in internet backbone traffic," 2002.