Deep Learning Based Optimization in Wireless Network

Lu Liu*, Yu Cheng*, Lin Cai*, Sheng Zhou† and Zhisheng Niu†
*Department of Electrical and Computer Engineering, Illinois Institute of Technology, USA
Email: lliu41@hawk.iit.edu; {cheng,lincai}@iit.edu
†Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, P. R. China
Email: {sheng.zhou,niuzhs}@tsinghua.edu.cn

***Abstract*—With the development of wireless networks, the scale of network optimization problems is growing correspondingly. While algorithms have been designed to reduce complexity in solving these problems under given size, the approach of directly reducing the size of problem has not received much attention. This motivates us to investigate an innovative approach to reduce problem scale while maintaining the optimality of solution. Through analysis on the optimization solutions, we discover that part of the elements may not be involved in the solution, such as unscheduled links in the flow constrained optimization problem. The observation indicates that it is possible to reduce problem scale without affecting the solution by excluding the unused links from problem formulation. In order to identify the link usage before solving the problem, we exploit deep learning to find the latent relationship between flow information and link usage in optimal solution. Based on this, we further predict whether a link will be scheduled through link evaluation and eliminate unused link from formulation to reduce problem size. Numerical results demonstrate that the proposed method can reduce computation cost by at least 50% without affecting optimality, thus greatly improve the efficiency of solving large scale network optimization problems.**

***Index Terms*—Optimization; Machine learning; Deep learning**

I. Introduction

Nowadays, wireless networks are rapidly growing in coverage and size. At the same time, multi-layer or heterogeneous network structure, involvement of various network resources, and exploitation of new transmission techniques further increase the complexity of network model [1], [2]. In order to improve and optimize network performance under such complex network models, one critical issue faced by researchers is the scale of problem, which has been greatly enlarged by the size and structure of networks. Therefore, developing efficient algorithms in solving the large scale problems becomes an emerging task for future networks.

Problem decomposition and distributed solution are the main approaches for dealing with the large scale optimization problems. Delay column generation (DCG) [1], [3], [4] has been exploited as a decomposition technique in solving optimization problems. However, in these works, DCG is not able to reduce the number of constraints, which still results in considerable time consumption in solving the decomposed problem. Game based distributed approach is suitable for solving the problems where a large number of users/players share or compete for network resources [5]. Since the players take actions in parallel and in a distributed manner where each one only needs to deal with a small scale problem locally, the original problem can be solved efficiently. But due to the introduction of price of anarchy, optimality of solution may not be always guaranteed in the game approaches. On the other hand, distributed algorithms such as reinforced learning [6] and dual decomposition [7] are able to provide optimal solution, but usually suffer from long convergence time, especially when the problem scale is large. Better convergence performance can be achieved with alternating direction method of multipliers (ADMM) [8], which decomposes the large scale problem into small sub-problems to facilitate parallel computation. However, ADMM requires the optimization problem to be in specified structure, which limits its application.

As discussed above, the existing algorithms focus on reducing the complexity when solving the problem. But if the scale of the problem formulation remains the same, the computation cost incurred by problem size still exists. This motivates us to take a novel approach of reducing the problem size itself, i.e., formulating an optimization problem with smaller size while solving it yields the same solution as that of the original problem. By analyzing the solutions of flow-constrained optimization problems in our previous work [1], we observe that many links are not involved in the optimal solution. This observation indicates that it is possible to cut off part of the elements (in our case, links) to reduce problem size without changing the optimal solution. Then the goal is to decide which elements can be excluded before solving the problem. In other words, we need to find a method to predict or evaluate the usefulness of links in the solution.

Link evaluation based optimizations have been discussed in [9] and [10], but the methods proposed in these works are heuristic approaches and cannot be well adapted to changing flow demands. Therefore the goal is to find an adaptive method to perform link evaluation under different flow input, or to find the relationships between link values and flow demands. To this end, we adopt deep learning to perform link evaluation, which is capable of extracting highly abstracted features from input and finding latent relationships between input and output [11]. Application of deep learning in networking mostly focus on data analysis or processing [12], detection [13], localization [14] and classification [15]. Deep learning based network

978-1-4673-8999-0/17/$31.00 ©2017 IEEE

optimization has only been discussed in a few works such as [16], but in limited scenarios. Applying deep learning to improve the efficiency in solving generic network optimization problems still remains unexploited, which is to be addressed in this paper. To the best of our knowledge, this is the first work that exploits learning based approach in improving performance of wireless network optimization.

In this paper, we develop the learning based approach on a typical optimization problem in wireless networks, which is demand constrained energy minimization problem in generic multi-dimensional networks. Results from our previous work [1], [3], [4] indicate that it may not be necessary to include all the links into formulation since some of them will not be used in the optimal solution. With training data generated from off-line pre-computations of sample problems, we adopt deep learning algorithm to find the relationship between multi-commodity flow demand information and link usage in optimal solution. Based on the learning result, we then predict whether a link will be scheduled for the target problem with link evaluation. Then links that are not likely to be scheduled will be excluded from the problem formulation to reduce problem size. Numerical results demonstrate that the proposed method is able to reduce the computation time to at most 50% compared with that of the original problem, without degrading the optimality of solution. Our contributions are summarized as follows:

1) We analyze the solution of network optimization problems, and discover that it is possible to reduce problem size by limiting number of elements (in this case, links) involved in the formulation without sacrificing optimality;
2) We exploit deep learning to investigate the latent relationship between flow demand and link usage, and propose a learning based method for link evaluation and problem size reduction;
3) Through numerical results, we demonstrate that the proposed method can reduce computation cost in solving optimization problems by at leat a half while maintain the optimality of solution, thus greatly increase the efficiency of solving large scale optimization problems.

The remainder of this paper is organized as follows. Section II describes the system model and formulation of the optimization problem, as well as the analysis on problem size. Section III introduces the proposed learning based method for problem size reduction. Performance evaluations are presented in Section IV. Finally, Section V gives the conclusion remarks.

II. Problem Formulation and Analysis

A. System Model

We consider a generic wireless network with set of nodes $\mathcal{N}$. A node can set up a transmission link to another one if the latter is within its transmission range. Denote the set of commodity flows in the network as $\mathcal{C}$, where each flow is specified by a source-destination node pair ($\{n^s, n^d\}$) and a pre-defined flow demand. The problem is to fulfill the flow demand with least amount of energy consumption through network configuration, resource allocation and transmission scheduling. Particularly, in a network with multi-dimensional resource space such as multi-radio multi-channel (MRMC) network, the problem is to obtain a joint solution on radio/channel assignment, routing and link scheduling.

We exploit the multi-dimensional modeling technique proposed in our previous work [1] to facilitate linear programming of the joint optimization problem. Each transmission link can be mapped to several virtual links (tuple-links as defined in [1]), differentiated by the transmitter radio, receiver radio and transmission channel. For example, in a network with 4 channels and each node equipped with 2 radio interfaces, a physical link (directional) will be mapped to 16 virtual links. Denote the set of all the virtual links in the network as $\mathcal{L}$.

Simultaneous transmissions of multiple virtual links may incur different types of interference or conflict to each other. If two links sharing the same radios are scheduled for transmission at the same time, then neither of them can work due to radio conflict. For two links with no radio conflict but working in the same channel, co-channel interference will be incurred, which affects the SINR at receiver side. In addition, if two links with no radio conflict are working in different channels, there will not be any mutual influence. Based on the above definitions, we can obtain the SINR at the receiver of virtual link i as

$$\mathrm{SINR}_i = \begin{cases} 0, & \text{if } i \text{ has radio conflict} \\ & \quad \text{with other active links;} \\ \frac{h_i p_i}{\sum\limits_{j \in I_i} h_{ji} p_j + \sigma^2}, & \text{otherwise.} \end{cases}$$

where h_i is the channel gain of link i, h_{ji} is the interference coefficient from link j to link i, p_i is the transmit power of link i, I_i is the set of virtual links working on the same channel as that of link i, and σ^2 is the random noise. Further, according to the Shannon-Hartley equation, the achievable transmission rate of link i can be expressed as

$$r_i = \mathrm{BW} \log_2(1 + \mathrm{SINR}_i) \tag{1}$$

where BW denotes the bandwidth.

With the above modeling, it can be seen that if the status (on or off) of all the virtual links are given, the transmission rate of links as well as the total power consumption in the network can be obtained. Based on this, we define a possible status of all the virtual links in the network as a transmission pattern, and the optimization problem can be transformed to a scheduling problem over all the patterns (denoted as $\mathcal{A}$). The solution of this scheduling problem can also imply routing and radio/channel assignment, which means we are able to obtain the desired joint solution.

B. Problem Formulation

The pattern based scheduling problem is to assign active time t_a for each pattern $a \in \mathcal{A}$, with the total scheduling time bounded by one normalized slot

$$\sum_{a \in \mathcal{A}} t_a \leq 1 \tag{2}$$

To minimize the total energy consumption in the network, the objective function will be formulated by summing up energy consumption of all the patterns, which is

$$\min \quad E = \sum_{a \in \mathcal{A}} P_a t_a \tag{3}$$

where P_a is the total power consumption of pattern a, which can be obtained by $P_a = \sum_{i \in \mathcal{L}} p_{i,a}$. $p_{i,a}$ is the transmit power of virtual link i in pattern a, which is 0 if i is in off state and p_i otherwise.

Denote the flow of commodity c on virtual link i as $f_{i,c}$. Since the sum flow of all commodities on a virtual link is limited by the link capacity, which is determined by the transmission rate and transmission time. Based on this, we can obtain the following constraint

$$f_i = \sum_{c \in \mathcal{C}} f_{i,c} \leq \sum_{a \in \mathcal{A}} r_{i,a} t_a \tag{4}$$

where $r_{i,a}$ is the transmission rate of i in pattern a.

In addition, considering the flow balance, we have the following constraint for each internal node n

$$\sum_{i \in \mathcal{L}_{n+}} f_{i,c} = \sum_{j \in \mathcal{L}_{n-}} f_{j,c}, \quad \forall c \in \mathcal{C}, \forall n \in \mathcal{N} \setminus \{n_c^s, n_c^d\} \tag{5}$$

where $\mathcal{L}_{n+}$ denotes the set of incoming virtual links to node n and $\mathcal{L}_{n-}$ denotes the set of outgoing links from n.

Similarly, at a source or destination node, we have

$$\sum_{i \in \mathcal{L}_{n_c^s -}} f_{i,c} = \sum_{j \in \mathcal{L}_{n_c^d +}} f_{j,c} \geq d_c, \quad \forall c \in \mathcal{C} \tag{6}$$

where d_c is the flow demand of commodity c.

Therefore the optimization problem will be formulated as

$$\begin{aligned} \min_{\{f_{i,c}, t_a\}} \quad & \text{Objective (3)} \\ \textit{s.t.} \quad & \text{Constraint (2),(4),(5)} \\ & f_{i,c} \geq 0, \quad \forall i \in \mathcal{L}, \forall c \in \mathcal{C} \qquad (7) \\ & t_a \geq 0, \quad \forall a \in \mathcal{A} \qquad (8) \end{aligned}$$

The above formulation follows linear programming (LP) problem, therefore can be solved with any LP algorithms. Considering the feature of the problem, we exploit delay column generation (DCG) method, as in our previous work [1].

C. Analysis on Problem Scale and Solution

The size of the optimization problem is related to the dimension of optimization variables and the number of constraints. In our case, it is mainly determined by the number of virtual links ($|\mathcal{L}|$) and the number of patterns ($|\mathcal{A}|$). It can be seen that the number of patterns is also determined by $|\mathcal{L}|$. Although with DCG method, we do not have to involve all the $|\mathcal{A}|$ patterns into the problem formulation, we still need at least $|\mathcal{L}|$ patterns to obtain an initial feasible solution. As a result, $|\mathcal{L}|$ is the most critical parameter affecting the scale of the optimization problem.

Based on the analysis, the problem size can be reduced if less links are involved into the formulation. From solving such type of optimization problems, we discover that it is possible to ignore some virtual links in formulating and solving the optimization problem while the solution is not severely degraded. In fact, in the final solution of the optimization problem, many virtual links will not be scheduled, or carry only a small amount of flow traffic. For example, Fig. 1 shows the histogram of flow traffic (percentage of the total traffic in the network) on all the 1420 virtual links in the optimal solution of a sample problem. It can be observed that the flow demands are fulfilled by a small number of virtual links and most links remain idle. Considering that virtual links of the same physical link are equivalent in the solution, we further investigate the flow distribution on physical links, as shown in Fig. 2. Again, the figure shows that only some of the links are involved in transmissions. There can be several reasons leading to this result. For example, if a link is far away from all the source-destination pairs and is unlikely to be covered by any path, it will probably be not scheduled. In addition, if two links share similar locations but one has better quality (in terms of transmission rate) than the other, then the one with the worse quality will less likely to be scheduled.

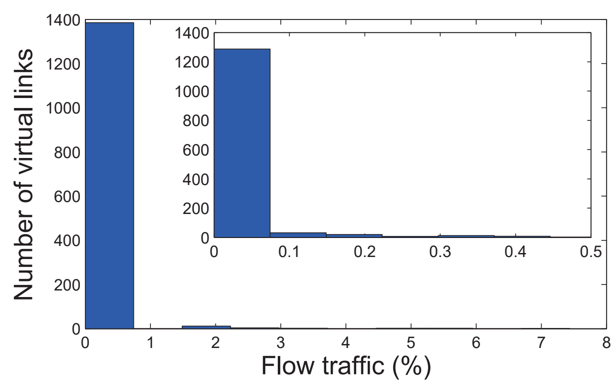


Fig. 1. Virtual link flow distribution.

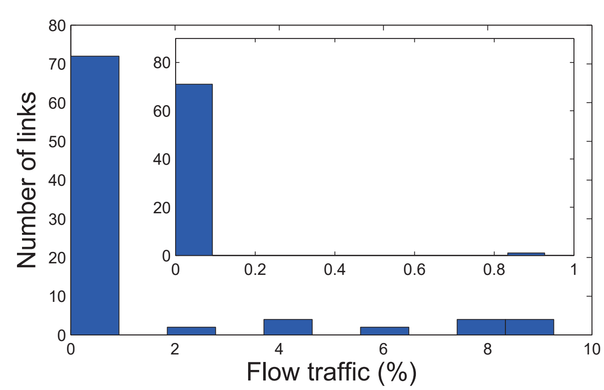


Fig. 2. Physical link flow distribution.

According to the observation of results, if the unused links are excluded from the formulation, the optimization solution can remain unchanged. This will enhance the efficiency of solving optimization problems for reduced computation time and storage cost. However, the observations of which links are unused are done after the problem is solved, which provide no help to solving the problem. Therefore, we need to decide which links can be excluded before solving the problem.

For a fixed topology (physical locations of nodes) of the network, the main factor that determines whether a link will be scheduled or not is the flow information, i.e., which nodes are the source and destination of each commodity and how much is the flow demand. Therefore the problem becomes: given the flow information, evaluate the usefulness of all the links. Since all the virtual links corresponding to the same physical link are equivalent, the evaluation can be performed on physical links. In other words, we need to find the relationship between input flows and the values of physical links.

Generally, evaluating a link based on flow information is not straight forward, since the value of a link is affected by many factors, such as the link quality, the location and the interference among links, which are indirectly affected by the flow information. Therefore there is no explicit expression on the relationship. One approach is to leverage the knowledge obtained from solutions of sample computations, based on which we can predict the value of links for a new problem, i.e., new input flow information. This motivates us to exploit machine learning techniques to predict link values, which is to be elaborated in the next section.

III. Deep Learning Based Link Evaluation

The relationship between input (flow information) and output (link evaluation) in our case is highly complicated, and there is no explicit characterization on how input will influence output. To infer the latent relationship in such a complex structure, we exploit deep learning to uncover the relationships.

A. Deep Learning

Deep learning is a learning method that can extract high level abstractions from input data through a deep structure consisting of multiple processing layers. One of the typical deep learning structure is deep belief net (DBN), which is proposed by G. Hinton in [17]. As shown in Fig. 3, DBN is constructed by multiple hidden layer (stochastic hidden cause) and one visible layer (input data), where different layers are connected by symmetric weights and the state or value of each unit is determined as a function of the values of connected units and the corresponding weights. The training of DBN is to infer the states of all the unobserved variables in hidden layers and adjust the interacting weights between units such that the network is more likely to generate the observed data in visible layer. After training, the top layer can be viewed as the abstracted information (features) from the input data. DBN itself is an unsupervised learning model, which means it does not rely on labels of training samples and can learn

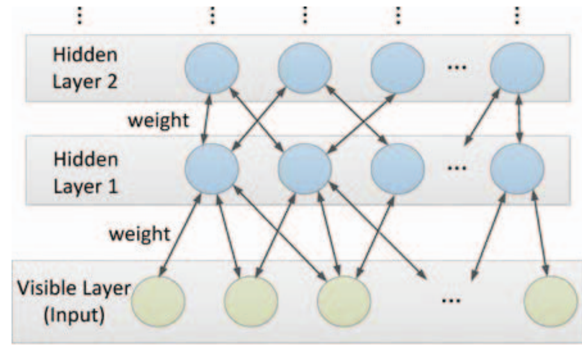


Fig. 3. DBN structure.

the features from input data. However, it can also be used to supervised learning for classification or discrimination, by adding the layer of labels on top of the trained DBN and applying back propagation (BP) fine-tuning, as shown in Fig. 4. BP fine-tuning algorithm will adjust the values of units and weights to match each input sample to its corresponding label by minimizing the misclassification error. The DBN

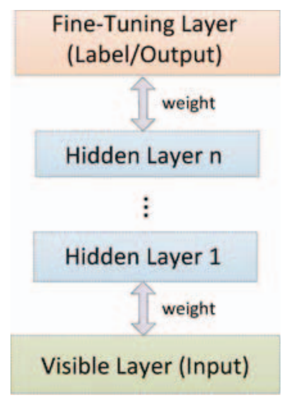


Fig. 4. DBN with BP fine-tuning.

based supervised learning algorithm can be summarized as follows. Based on training samples, DBN and BP fine-tuning process learns the values of units in hidden layers and the interacting weights. Then replacing the visible layer by a new input sample (test sample), the network can generate an output as the classification result of the input.

B. Link Evaluation

In order to apply the learning algorithm to our link evaluation problem, we first define the input and output layers. The input flow information is expressed as a flow demand vector $\mathbf{x}$ as the input layer. $\mathbf{x}$ is an $N \times 1$ vector with each entry $\{x_n\}$ defined as

$$x_n = \begin{cases} d_c & \text{if } n = n_c^s \\ -d_c & \text{if } n = n_c^d \\ 0 & \text{otherwise} \end{cases}$$

The previously mentioned classification model is indeed constructing the relationship between input data and output labels. This differs from our case where the output is continuous values of link evaluations. In fact, the raw output of the learning algorithm is given as continuous values over all the labels, which indicate the probabilities that the input sample belongs to different categories, and the largest one is chosen as the classification result. In order to apply the learning model to our problem, we directly take the raw output as the evaluation of links, which is a vector with a length equal to the number of links in the network.

Based on the input and output definitions specified for our problem, we generate the training data set as pairs of flow demand vector (input) and link evaluations (output) obtained from sample computations. Particularly, each training sample is obtained by solving the optimization problem formulated with a randomly generated flow demand vector, and recording the amount of flow on all the links in optimal solution as the link evaluation vector. The computation will be repeated to build the training data set. Then this data set is used for training the DBN based learning algorithm.

The training process will assign values for all the interacting weights in the learning structure. With this result, we may input the new flow demand vector from the problem we are going to solve. Then the output of the learning structure will provide the evaluation vector on all the links. The result can be viewed as an estimation on whether a link would be used under the new flow demand vector, indicated by the values in the vector. Based on the evaluation, we can keep the links with higher values and exclude those with very small values. Then the optimization problem will be formulated with only the links that are kept. Reducing the number of links will also reduce the number of virtual links at the same degree. Generally, smaller number of links will lead to less optimization variables and constraints, which reduces the computation time and storage cost. Examples of the cost reduction will be shown in the next section.

Remark: The proposed approach is not limited to the formulation presented in this paper. For other types of optimization problems in wireless network, the proposed algorithm can be applied by replacing link evaluations with the corresponding network elements in formulation and following similar procedures for training and implementation.

IV. Performance Evaluation

In this section, we present numerical results of the proposed deep learning based optimization. The optimization is performed in a network with 25 randomly deployed nodes, where each node is equipped with one or multiple radio interfaces. Transmissions can take place on multiple non-overlapping channels. We consider the topology of nodes are fixed, while the flow information changes over time. The goal is to learn the link values through off-line training such thatthe optimization problem can be solved in reduced time under new flow demand.

We first construct the training set by solving pre-generated optimization problems, where each case is formulated with randomly generated flow demand vector and solved with DCG based decomposition algorithm. Solving each problem will provide one pair of input (flow vector) and output (link values), and the data pairs are used as training samples for the learning algorithm. The performance of the leaning process is tested with test samples (newly generated flow vectors). We input the test flow vectors to the learning structure to get the learning results, which are the link value vectors estimated by the learning algorithm. The learning result is then compared with the true result obtained from solving the optimization problem. A sample comparison is shown in Fig. 5.

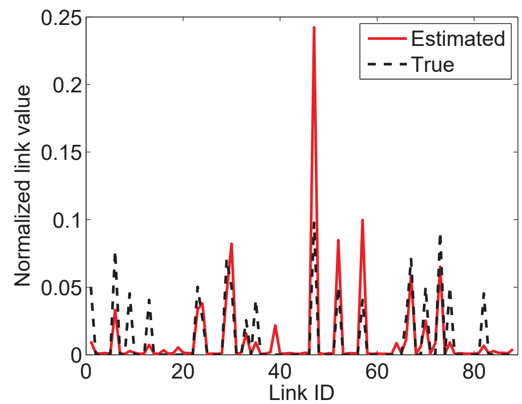


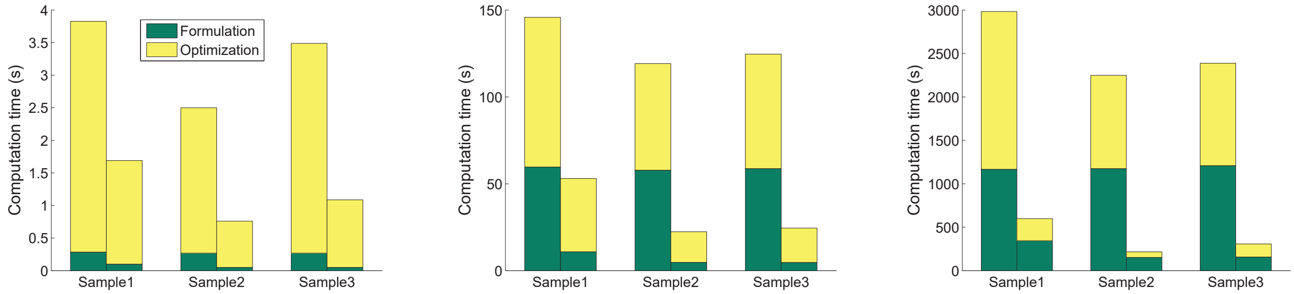
Fig. 5. Estimated link evaluation from learning result.

A link with high value indicates that it will likely be scheduled, while a link with very low value indicates it may not be used in the optimal solution. From Fig. 5 we can observe that, the learning process is able to detect most of the high valued links, since the learning results also show relatively high values for such links.

With the estimated link values obtained from learning results, we further apply a threshold to the values such that links with lower values than threshold will be eliminated from the formulation to reduce the optimization problem scale. The threshold should be properly set such that the difference between results of reduced size problem and original problem is no larger than 3%[1]. The computation time (including time consumptions in formulation and optimization) of original problem and reduced size problem(s) are shown in Fig. 6 and 7, where Fig. 6 shows the results at different percentages of remaining links (obtained by varying threshold), and Fig. 7 shows the results under different network configurations.

As shown in Fig. 6 and 7, excluding unused links from formulation can efficiently reduce the computation time. In most cases of the test samples, the number of links can be reduced by half. Generally, reducing the number of links by half leads to a much smaller scale problem with at most half of the original size, as well as the computation time. Moreover, with the time consumed in link setup (for multi-dimensional modeling) and problem formulation taken into consideration, the actual computation time is further saved. Fig. 7 also

[1]In fact, in most cases, the results of reduced size problem and original problem are the same



(a) 1 radio per node, 1 channel.

(b) 2 radios per node, 4 channels.

(c) 3 radios per node, 8 channels.

Fig. 7. Computation time under different network configurations.

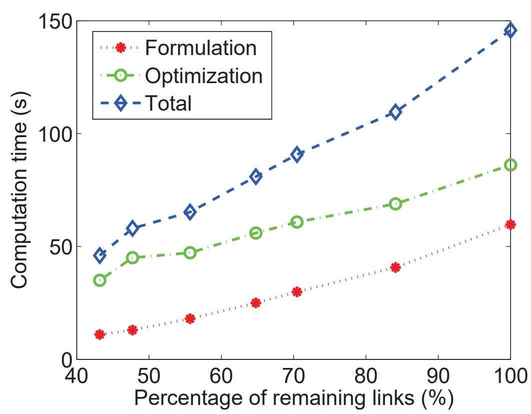


Fig. 6. Computation time at different portions of remaining links.

indicates that the cost reduction will be more significant as the scale of original problem grows. Thus the learning based algorithm is promising in improving the efficiency of solving large scale optimization problems.

V. Conclusion

In this paper, we have proposed a deep learning based method in reducing problem scale for wireless network optimizations. Based on the observation that omitting unscheduled links can reduce problem size without affecting the solution, we have performed prediction on link usage in solution based on the learning results from off-line pre-computed training data. With the prediction, we have excluded unused links to re-formulate the optimization problem into smaller scale. In addition, we have presented numerical results, which demonstrated that with the proposed learning based method, optimization problems can be solved with at most half of the original cost while preserving the optimality of solution. Therefore the proposed approach is promising in improving efficiency of solving large scale optimization problems in wireless networks.

Acknowledgment

This work was supported in part by the NSF under grants CNS-1053777, CNS-1320736, ECCS-1610874, and Career award ECCS-1554576, and by National Natural Science Foundation of China under grant 61628107.

References

[1] L. Liu, X. Cao, W. Shen, Y. Cheng, and L. Cai, "Dafee: A decomposed approach for energy efficient networking in multi-radio multi-channel wireless networks," in *Proc. IEEE INFOCOM*, 2016, pp. 1–9.

[2] K. K. Mensah, R. Chai, D. Bilibashi, and F. Gao, "Energy efficiency based joint cell selection and power allocation scheme for hetnets," *Digital Communications and Networks*, vol. 2, no. 4, pp. 184–190, 2016.

[3] L. Liu, X. Cao, Y. Cheng, L. Du, W. Song, and Y. Wang, "Energy-efficient capacity optimization in wireless networks," in *Proc. IEEE INFOCOM*, 2014, pp. 1384–1392.

[4] Y. Cheng, X. Cao, X. S. Shen, D. M. Shila, and H. Li, "A systematic study of the delayed column generation method for optimizing wireless networks," in *Proc. ACM MobiHoc*, 2014, pp. 23–32.

[5] P. Semasinghe, E. Hossain, and K. Zhu, "An evolutionary game for distributed resource allocation in self-organizing small cells," *IEEE Trans. Mobile Comput.*, vol. 14, no. 2, pp. 274–287, 2015.

[6] N. Morozs, T. Clarke, and D. Grace, "Distributed heuristically accelerated q-learning for robust cognitive spectrum management in lte cellular systems," *IEEE Trans. Mobile Comput.*, vol. 15, no. 4, pp. 817–825, 2016.

[7] H. Zhang, C. Jiang, N. C. Beaulieu, X. Chu, X. Wang, and T. Q. Quek, "Resource allocation for cognitive small cell networks: A cooperative bargaining game theoretic approach," *IEEE Trans. Wireless Commun.*, vol. 14, no. 6, pp. 3481–3493, 2015.

[8] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[9] H. Li, Y. Cheng, C. Zhou, and W. Zhuang, "Minimizing end-to-end delay: A novel routing metric for multi-radio wireless mesh networks," in *Proc. IEEE INFOCOM*, 2009, pp. 46–54.

[10] H. Li, Y. Cheng, C. Zhou, and P. Wan, "Multi-dimensional conflict graph based computing for optimal capacity in mr-mc wireless networks," in *Proc. IEEE ICDCS*, 2010, pp. 774–783.

[11] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[12] X.-W. Chen and X. Lin, "Big data deep learning: challenges and perspectives," *IEEE Access*, vol. 2, pp. 514–525, 2014.

[13] B. Dong and X. Wang, "Comparison deep learning method to traditional methods using for network intrusion detection," in *Proc. IEEE ICCSN*, 2016, pp. 581–585.

[14] W. Zhang, K. Liu, W. Zhang, Y. Zhang, and J. Gu, "Wi-fi positioning based on deep learning," in *Proc. IEEE ICIA*, 2014, pp. 1176–1179.

[15] N. T. Nguyen, R. Zheng, and Z. Han, "On identifying primary user emulation attacks in cognitive radio systems using nonparametric bayesian classification," *IEEE Trans. Signal Process.*, vol. 60, no. 3, pp. 1432–1445, 2012.

[16] M. A. Wijaya, K. Fukawa, and H. Suzuki, "Intercell-interference cancellation and neural network transmit power optimization for mimo channels," in *Proc. IEEE VTC Fall*, 2015, pp. 1–5.

[17] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.