南京郵電大學
Nanjing University of Posts and Telecommunications

Offloading Algorithms for Maximizing Inference Accuracy on Edge Device in an Edge Intelligence System

Andrea Fresa and Jaya Prakash Champati

IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 34, NO. 7, JULY 2023

汇报人：汤耀杰
汇报时间：2023.9.23

目录

CONTENTS

Introduction

随着边缘计算的出现，边缘设备（ED）和边缘服务器（ES）之间的作业卸载问题在过去受到了广泛的关注。由于越来越多的应用程序对来自EDs数据样本进行机器学习（ML）推理，我们通过考虑以下新的方面研究卸载推理任务：

1)与典型的计算任务相比，推理任务的处理时间取决于ML模型的大小；

2)最近提出的深度神经网络（DNNs）为资源受限的设备提供了通过权衡推理精度来缩小模型大小的选择；

Toward this end, we formulate the following problem: *given n data samples at time zero, find a schedule that offloads a partition of the jobs to the ES and assigns the remaining jobs to m models on the ED, such that the total accuracy is maximized and the makespan is within a time constraint T*. Solution to

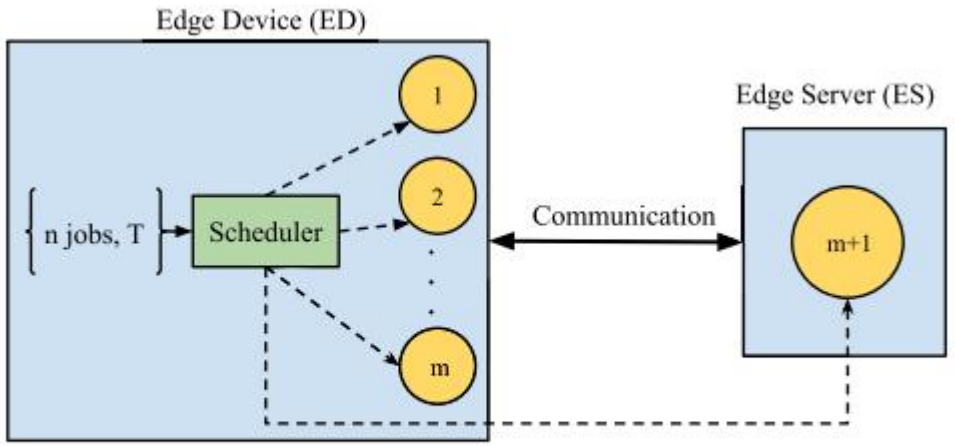

Fig. 1. Scheduling inference jobs between an ED and an ES.

Introduction

Since the true accuracy provided by a model for a given data sample can only be inferred after the job is executed, for analytical tractability, we consider the average test accuracy of the model is its accuracy for any data sample. Given the processing and communication times of the jobs, we formulate the problem as an Integer Linear Program (ILP). We note that the ILP is agnostic to the actual ML models used on the ED and the ES. Different ML models on the ED may correspond to different instantiations of the same DNN with different hyper-parameter values (cf. [8]), or they may correspond to different ML algorithms such as linear regression, SVMs, DNNs, etc., although in this work we focus on the former setting.

Solving the formulated ILP is challenging due to the following reasons. Partitioning the set of jobs between the ED and the ES is related to scheduling jobs on parallel machines [11], and assigning the jobs to the models on the ED is related to the knapsack problem [12], both are known to be NP-hard. A special case of our problem, where the ED has a single model ($m = 1$), is the Generalized Assignment Problem (GAP) with two machines [13]. GAP is known to be APX-hard,[1] and the best-known approximation algorithm provides a solution that has maskespan at most $2T$ [14]. However, the algorithms for GAP and their performance guarantees are not directly applicable to our problem due to the additional aspect that, on the ED there are multiple models to choose from. We propose a novel algorithm that solves a Linear Programming relaxation (LP-relaxation) of the ILP, uses a counting argument to bound the number of fractional solutions, and used a simple rule to round the fractional solution.

Introduction

Our main contributions are summarized below[2]:

- We formulate the total accuracy maximization problem subject to a constraint T on the makespan as an ILP. Noting that the ILP is NP-hard, we propose an approximation algorithm Accuracy Maximization using LP-Relaxation and Rounding (AMR^2) to solve it. The runtime of AMR^2 is $O(n^3(m+1)^3)$.
- We prove that the total accuracy achieved by AMR^2 is at most a small constant (less than 1), lower than the optimum total accuracy, and its makespan is at most $2T$. We also extend AMR^2 for the case of K ESs and prove performance bounds. A salient feature of AMR^2 is that it is asymptotically optimal.
- For the case of identical jobs, i.e., the data samples are identical, we propose an optimal algorithm Accuracy Maximization using Dynamic Programming (AMDP), which does a greedy packing for the ES and solves a Cardinality Constrained Knapsack Problem (CCKP) for job assignments on the ED. We implement AMDP and demonstrate that its runtime is much lower compared to AMR^2.
- As proof of concept, we perform experiments using a Raspberry Pi, equipped with MobileNets, and a server, equipped with ResNet50, that are connected over a Local Area Network (LAN). The MobileNets and ResNet50 are trained for classifying images from the ImageNet dataset. We estimate processing and communication times for different sizes of images and implemented AMR^2 and a greedy algorithm, Greedy-RRA, on Raspberry Pi. Our results indicate that the total test accuracy achieved by AMR^2 is close to that of the LP-relaxed solution, and its total true accuracy is, on average, 40% higher than that of the greedy algorithm. We also implemented AMR^2 and Greedy-RRA for the case of K ESs. We find that as the number of ESs increases the total accuracy increases but the total accuracy gain provided by AMR^2 over Greedy-RRA decreases.

Related Works

A. Offloading and ML Inference Jobs

and how to offload them to an ES [2]. The objectives that were considered for optimizing the offloading decision are, 1) minimize job execution times, see for example [17], [18], [19], [20], 2) minimize the energy of the ED spent in computing and/or transmitting the jobs, subject to a constraint on the execution delay, see for example [21], [22], [23] and [24], and 3) decide scheduling strategy to guarantee statistical QoS for jobs where unreliable communication channels between EDs and ESs are considered, see for example [25]. However, the above works consider generic computation jobs, and the aspect of accuracy, which is relevant for the case of inference jobs, has not been considered.

Recently, a few works considered the problem of maximizing accuracy for inference jobs on the ED [26], [27], [28]. In [26], the authors studied the problem of maximizing the accuracy within a deadline for each frame of a video analytics application. They do not consider offloading to the edge and their solution is tailored to the DNNs that use early exits [9]. A similar job a the mobile device. A heuristic solution was proposed in [28] for offloading inference jobs for maximizing inference accuracy subject to a maximum energy constraint. In contrast to the above works, we consider multiple models on the ED and provide performance guarantees for AMR^2. The authors in [29] studied the minimization of the energy consumption on the EDs by offloading inference jobs to Cloudlets. In [30], the authors proposed a novel approach for reducing communication costs while offloading images. This work proposes the computation of a discrimination matrix P, which is computed on the Cloud, that is then used by the ESs. When an ED offloads an image X to an ES, the ES computes the product $P^T X$ that determines the features that are relevant to be offloaded to the cloud. This approach reduces the communication cost to offload the image to the cloud and also increases the accuracy of the inference result. In [31], the authors studied the properties of DNN networks in order to increase the parallelism in computation. Specifically,

Related Works

B. Job Scheduling

As noted in Section I, our problem is related to the knapsack problem [12]. To see this, note that if it is not feasible to schedule on the ES and all jobs have to be assigned to the ED, then maximizing the total accuracy is equivalent to maximizing profit, and the constraint T is equivalent to the capacity of knapsack. In this case, our problem turns out to be a generalization of the CCKP [32]. Another special case of our problem, where the ED has only a single model, can be formulated as a GAP [13], [33], with two machines. In GAP, n jobs (or items) have to be assigned to r machines (or knapsacks). Each job-machine pair is characterized by two parameters: processing time and cost. The objective is to minimize the total cost subject to a time constraint T on the makespan. It is known that GAP is APX-hard [34].

In their seminal work [14], the authors proposed an algorithm for GAP that achieves minimum total cost and has makespan at most $2T$. Their method involves solving a sequence of LP feasibility problems, in order to tackle the processing times that are greater than T, and compute the minimum total cost using bisection search. Their algorithm can also be used for solving a related extension of GAP, where the cost of scheduling a job on a machine increases linearly with decrease in the processing time of the job. In comparison to this setting, the accuracies (equivalent to negative costs) are not linearly related to the processing times of the jobs and thus the proposed method in [14] is not directly applicable to the problem at hand. Our proposed algorithm AMR^2 is different from their method in that it does not require to solve LP feasibility problems and the use of bisection search. Further, we prove the performance bounds using a different analysis technique which is based on a counting argument for the LP-relaxation and solving a sub-problem of the ILP.

System Model

A. ML Models and Accuracy

Consider an ED and an ES connected over a network and the ED enlists the help of the ES for computation offloading. At time zero, n inference jobs, each representing the processing requirement of a data sample on a pre-trained ML model, are available to a scheduler at the ED. Let j denote the job index and $J = \{1, 2, \ldots, n\}$ denote the set of job indices.

Let $a_i \in [0, 1]$ denote the top-1 accuracy of model i. Since we do not know if a job is classified correctly by a model without first processing it on that model, for analytical tractability, we consider that the accuracy of a model i for any job is a_i. Note that assigning a job to a model with higher top-1 accuracy increases its probability of correct classification. WLOG, we assume that $a_1 \leq a_2 \leq \ldots \leq a_m$, and also assume that the model $m+1$ is a state-of-the-art model with a higher top-1 accuracy than the models on the ED, i.e., $a_m \leq a_{m+1}$. In the sequel, the

B. Processing and Communication Times

The processing time of job j on model $i \in M \backslash \{m+1\}$ is denoted by p_{ij}, and on model $m+1$ it is denoted by $p'_{(m+1)j}$. In several applications, the data samples may need pre-processing before they are input to the ML model. For example, in computer vision tasks, images require pre-processing and the time required for pre-processing varies with the size of the image [35]. In our experiments with the images from the ImageNet dataset, the pre-processing stage only involves reshaping the images to input to the DNN models. Let τ_{ij} denote the pre-processing time of job j on model i. We consider the pre-processing times are part of the processing times defined above.

Let c_j denote the communication time for offloading job j. It is determined by the data size of the job, i.e., the size of the data sample in bits, and the data rate of the connection between the ED and the ES. Given $p'_{(m+1)j}$ and c_j, the *total time* to process job j on the ES, denoted by $p_{(m+1)j}$, is given by $p_{(m+1)j} = c_j + p'_{(m+1)j}$. We deliberately use similar notation for the processing

System Model

C. Optimization Problem

Let x_{ij} denote a binary variable such that $x_{ij} = 1$, if the scheduler assigns job j to model i, and $x_{ij} = 0$, otherwise. Note that, if $x_{(m+1)j} = 1$, then job j is offloaded to the ES. Therefore, a schedule is determined by the matrix $\mathbf{x} = [x_{ij}]$. We impose the following constraints on $\mathbf{x}$:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} p_{ij}x_{ij} \leq T \tag{1}$$

$$\sum_{j=1}^{n} p_{(m+1)j}x_{(m+1)j} \leq T \tag{2}$$

$$\sum_{i=1}^{m+1} x_{ij} = 1, \forall j \in J \tag{3}$$

$$x_{i,j} \in \{0,1\}, \forall i \in M, \forall j \in J, \tag{4}$$

where constraints (1) and (2) ensure that the total processing times on the ED and the ES, respectively, are within T, and thus the makespan is within T. Constraints in (3) imply that each job is assigned to only one model and no job should be left unassigned, and (4) are integer constraints. We are interested in the following accuracy maximization problem $\mathcal{P}$:

$$\underset{\mathbf{x}}{\text{maximize}} \quad A = \sum_{i=1}^{m+1}\sum_{j=1}^{n} a_i x_{ij}$$

$$\text{subject to} \quad (1), (2), (3), \text{ and } (4).$$

Note that $\mathcal{P}$ is an ILP. We will show later that a special case of $\mathcal{P}$ reduces to CCKP, which is NP-hard, and thus $\mathcal{P}$ is NP-hard. Let A^* denote the optimal total accuracy for $\mathcal{P}$.

Accuracy Maximization Using LP-Relaxation And Rounding(AMR²)

A. LP-Relaxation

Given $\mathcal{P}$, we proceed with solving the LP-relaxation of $\mathcal{P}$, where the integer constraints in (4) are replaced using the following non-negative constraints:

$$x_{ij} \geq 0, \forall i \in M \text{ and } \forall j \in J. \tag{5}$$

Note that, the constraints $x_{ij} \leq 1$ are not required as this is ensured by the constraints in (3). Let the matrix $\bar{\mathbf{x}} = [\bar{x}_{ij}]$ and A^*_{LP} denote the schedule and the total accuracy, respectively, output by the LP-relaxation. The LP-relaxed solution provides an upper bound on the total accuracy achieved by an optimal schedule, and thus we have $A^*_{\text{LP}} \geq A^*$.

Note that the solution to the LP-relaxation may contain x_{ij} values that are fractional, and the rounding procedure is critical to proving the performance bounds. To design a rounding procedure, we first refer to a key result in [36], where the author studied the problem of assigning N jobs to K parallel machines with the objective of minimizing the makespan. For the LP-relaxation of this problem, the author presented the following counting argument: there exists an optimal basic solution in which there can be at most $K - 1$ *fractional jobs*, i.e., the jobs that are divided between machines, and all the other jobs are fully assigned. Further, the simplex algorithm outputs such a

Accuracy Maximization Using LP-Relaxation And Rounding(AMR²)

主要是单纯形法的求解原理

Lemma 1: For the LP-relaxation of $\mathcal{P}$, there exists an optimal basic solution with at most two fractional jobs.

Proof: Since LP-relaxation of $\mathcal{P}$ has $n+2$ constraints, apart from the non-negative constraints in (5), one can show using LP theory that there exists an optimal basic solution with $n+2$ basic variables that may take positive values and all the non-basic variables take value zero. Under such an optimal basic solution, for the n constraints in (3) to be satisfied, at least one positive basic variable should belong to each of those n constraints. The remaining 2 basic variables may belong to at most two equations. This implies that at least $n-2$ equations should have exactly one positive basic variable whose value should be 1 in order to satisfy the constraint. Therefore, there can be at most two equations with multiple basic variables whose values are in (0, 1), and the two jobs that correspond to these equations are the fractional jobs. □

$$\sum_{i=1}^{m}\sum_{j=1}^{n} p_{ij}x_{ij} \leq T \tag{1}$$

$$\sum_{j=1}^{n} p_{(m+1)j}x_{(m+1)j} \leq T \tag{2}$$

$$\sum_{i=1}^{m+1} x_{ij} = 1,\ \forall j \in J \tag{3}$$

$$x_{ij} \geq 0, \forall i \in M \text{ and } \forall j \in J. \tag{5}$$

Accuracy Maximization Using LP-Relaxation And Rounding(AMR²)

这边为了方便后面的证明，将非整数解单独拿出来定义；

Given the basic optimal solution to the LP-relaxation, the result in Lemma 1 reduces the rounding procedure to assigning at most two fractional jobs. WLOG, we re-index the jobs and refer to the fractional jobs by job 1 and job 2. We define the set $I = J \backslash \{1, 2\}$ and refer to the assignment of I under $\bar{\mathbf{x}}$ as the *integer solution of the LP-relaxation*. We define:

$$P_1 = \sum_{i=1}^{m} \sum_{j \in I} p_{i,j} \bar{x}_{i,j}, \tag{6}$$

$$P_2 = \sum_{j \in I} p_{m+1,j} \bar{x}_{m+1,j}. \tag{7}$$

With a slight abuse in notation, we use i_j and k_j to denote the indices of the machines on which the fractional job $j \in \{1, 2\}$ is scheduled. We have

$$\bar{x}_{i_1} + \bar{x}_{k_1} = 1, \tag{8}$$

$$\bar{x}_{i_2} + \bar{x}_{k_2} = 1. \tag{9}$$

Accuracy Maximization Using LP-Relaxation And Rounding(AMR²)

Algorithm 1: AMR^2.

Input: p_{ij}, for all $i \in M$ and $j \in J$.
Solve the LP-relaxation of $\mathcal{P}$.
if One fractional job **then**
  if $P_2 + p_{m+1,1} \leq 2T$ **then**
    Assign job 1 to model $m+1$
  else
    Assign job 1 to model
    $\arg\max_{i \in M \setminus \{m+1\}} \{a_i : p_{i1} + P_1 \leq 2T\}$.
  end if
end if
if Two fractional jobs **then**
  for all $j \in \{1, 2\}$ **do**
    if $\bar{x}_{i_j} > \bar{x}_{k_j}$ **then**
      $x^{\dagger}_{i_j} = 1$
    else
      $x^{\dagger}_{k_j} = 1$
    end if
  end for
end if
Output: Assignment matrix $\mathbf{x}^{\dagger}$ and total accuracy $A^{\dagger}$

The main steps of AMR² are summarized in Algorithm 1. In the first step, AMR² solves the LP-relaxation. In the second step, if there is one fractional job, it is assigned to model with largest accuracy such that the makespan does not exceed $2T$. If there are two fractional jobs, we use the simple rounding rule and assign the job to the model on which it has a higher fraction. Though the algorithm is not sophisticated, we will later see that proving the performance bounds is involved. We use $\mathbf{x}^{\dagger}$ and $A^{\dagger}$ to denote the schedule and the total accuracy, respectively, output by AMR².

Computational Complexity: The computational complexity of solving an LP with l variables is $O(l^3)$ (cf. [37]). In the LP-relaxation, the number of variables are $n(m+1)$ and thus its runtime is $O(n^3(m+1)^3)$. The rounding technique has negligible complexity when compared to the complexity of solving the LP-relaxation. In conclusion, the runtime is $O(n^3(m+1)^3)$.

In this section, we analyse AMR² and present a $2T$ bound for its makespan and show that its total accuracy is at most $a_{m+1} - a_1$ lower than the optimal accuracy.

Accuracy Maximization Using LP-Relaxation And Rounding(AMR^2)

Theorem 1: If $\mathcal{P}$ is feasible, then the makespan of the system under AMR^2 is at most 2T.

Proof: For the case of one fractional job, the result follows by the construct of AMR^2. For the case of two fractional jobs, based on the fractional job assignment output by the LP-relaxation solution we consider three cases and for each case, we consider sub-cases based on the schedule of AMR2. For the proof, we assume (10) and (11) are true. The proof steps are similar for other cases.

$$\bar{x}_{k_1} < \bar{x}_{i_1}, \tag{10}$$

$$\bar{x}_{i_2} < \bar{x}_{k_2}. \tag{11}$$

Case 1: Both jobs are assigned as fractional on the ED, i.e., i_1 and k_2 are in $\{1, 2, \ldots, m\}$. Clearly, in this case the makespan on the ES is at most T, same as that given in the LP-relaxed solution. Suppose that after rounding, the completion time on the ED under AMR^2 violates $2T$, i.e.,

$$P_1 + p_{i_1} + p_{k_2} > 2T. \tag{12}$$

From the LP-relaxed solution, we obtain

$$T - P_1 = p_{i_1}\bar{x}_{i_1} + p_{k_1}\bar{x}_{k_1} + p_{i_2}\bar{x}_{i_2} + p_{k_2}\bar{x}_{k_2}. \tag{13}$$

Substituting (13) in (12), we obtain

$$p_{i_1} + p_{k_2} > T + p_{i_1}\bar{x}_{i_1} + p_{k_1}\bar{x}_{k_1} + p_{i_2}\bar{x}_{i_2} + p_{k_2}\bar{x}_{k_2}. \tag{14}$$

Using (8) and (9) in (14), we obtain

$$p_{i_1}\bar{x}_{k_1} + p_{k_2}\bar{x}_{i_2} - p_{k_1}\bar{x}_{k_1} - p_{i_2}\bar{x}_{i_2} > T. \tag{15}$$

$$p_{i_1}\bar{x}_{k_1} + p_{k_2}\bar{x}_{i_2} + p_{k_1}\bar{x}_{k_1} + p_{i_2}\bar{x}_{i_2} > T. \tag{16}$$

Given (10) and (11), (16) should hold if we substitute $\bar{x}_{i_1}$ in place of $\bar{x}_{k_1}$ and $\bar{x}_{k_2}$ in place of $\bar{x}_{i_2}$, i.e.,

$$p_{i_1}\bar{x}_{i_1} + p_{k_1}\bar{x}_{k_1} + p_{i_2}\bar{x}_{i_2} + p_{k_2}\bar{x}_{k_2} > T. \tag{17}$$

However, the left hand side (LHS) of (17) is equal to $T - P_1$ (cf. (13)), which is smaller than T. Therefore, (15) is false and by contradiction $P_1 + p_{i_1} + p_{k_2} \leq 2T$ is true.

Accuracy Maximization Using LP-Relaxation And Rounding(AMR^2)

Case 2: One job is assigned as fractional between the ED and the ES and the other job is assigned as fractional between two models on the ED. WLOG, we consider job 1 is assigned to models on the ED and job 2 is assigned between the ED and the ES. We consider the following sub-cases.

Case 2a: Job 2 is scheduled on the ES and from (11) we must have $k_2 = m + 1$. From the LP-relaxed solution, we have

Case 2b: Job 2 is scheduled on the ED. When using the basic solution of the LP problem, the completion time of the ED is

Case 3: Both jobs are assigned as fractional between ED and ES by the LP-relaxation. We have three different sub cases based on the fractional assignation of the LP-relaxation problem.

Case 3a: Both jobs are scheduled on the ES. The completion time of the ES, when considering the basic solution $\bar{x}$ is:

Case 3b: One job is scheduled on the ED and the other on the ES. The completion time of the ED under the LP-relaxed solution satisfies

Case 3c: Both jobs are scheduled on the ED. We claim

Accuracy Maximization Using LP-Relaxation And Rounding(AMR^2)

Theorem 2: The difference between the optimal total accuracy A^* and $A^\dagger$, the total accuracy achieved by AMR^2, is upper bounded by $a_{m+1} - a_1$.

Proof: Since $A^* \leq A^*_{LP}$, we prove the result with respect to A^*_{LP}. WLOG, we consider that i_1 and i_2 as the indices of the models with lower accuracy, respectively, for jobs 1 and 2, and k_1 and k_2 are the index of the models which provide higher accuracy. To prove the performance bound we distinguish the following three cases.

Case 1: $\bar{x}_{k_1} \geq \frac{1}{2}, \bar{x}_{k_2} \geq \frac{1}{2}$. In this case, AMR^2 will schedule job 1 on model k_1 and job 2 on model k_2. The contribution of the following jobs to the $A^\dagger$ is $a_{k_1} + a_{k_2}$. The contribution of the same jobs to the optimal solution A^*_{LP} is: $a_{i_1}\bar{x}_{i_1} + a_{k_1}\bar{x}_{k_1} + a_{i_2}\bar{x}_{i_2} + a_{k_2}\bar{x}_{k_2}$. However, $a_{k_1} > a_{i_1}$ and $a_{k_2} > a_{i_2}$: thus it is trivial that $A^\dagger > A^*$.

Case 2: $\bar{x}_{k_1} \geq \frac{1}{2}$ and $\bar{x}_{k_2} < \frac{1}{2}$. Here, AMR^2 schedules job 1 on k_1 and job 2 on i_2.

$$\begin{aligned} A^*_{LP} - A^\dagger &= a_{i_1}\bar{x}_{i_1} + a_{k_1}\bar{x}_{k_1} + a_{i_2}\bar{x}_{i_2} + a_{k_2}\bar{x}_{k_2} - a_{i_2} - a_{k_1} \\ &= a_{i_1}\bar{x}_{i_1} + a_{i_2}(\bar{x}_{i_2} - 1) + a_{k_1}(\bar{x}_{k_1} - 1) + a_{k_2}\bar{x}_{k_2} \\ &= a_{i_1}\bar{x}_{i_1} - a_{i_2}\bar{x}_{k_2} - a_{k_1}\bar{x}_{i_1} + a_{k_2}\bar{x}_{k_2} \\ &= \bar{x}_{i_1}(a_{i_1} - a_{k_1}) + \bar{x}_{k_2}(a_{k_2} - a_{i_2}). \end{aligned}$$

Substituting $\bar{x}_{k_1} \geq \frac{1}{2}$ and $\bar{x}_{k_2} < \frac{1}{2}$ in the above equation we have $a_{i_1} - a_{k_1} < 0$. In conclusion

$$A^*_{LP} - A^\dagger \leq \frac{1}{2}(a_{k_2} - a_{i_2}). \tag{39}$$

Proof for the case $\bar{x}_{k_1} < \frac{1}{2}, \bar{x}_{k_2} \geq \frac{1}{2}$ is similar to Case 2.

Case 3: $\bar{x}_{k_1} < \frac{1}{2}$ and $\bar{x}_{k_2} < \frac{1}{2}$. AMR^2 schedules job 1 on i_1, and job 2 on i_2.

$$\begin{aligned} A^*_{LP} - A^\dagger &= a_{i_1}\bar{x}_{i_1} + a_{k_1}\bar{x}_{k_1} + a_{i_2}\bar{x}_{i_2} + a_{k_2}\bar{x}_{k_2} - a_{i_1} - a_{i_2} \\ &= \bar{x}_{k_1}(a_{k_1} - a_{i_1}) + \bar{x}_{k_2}(a_{k_2} - a_{i_2}) \leq a_{m+1} - a_1. \end{aligned}$$

In the last equation above, we used $\bar{x}_{k_1} < \frac{1}{2}$ and $\bar{x}_{k_2} < \frac{1}{2}$, and the fact that $a_{k_1} - a_{i_1}$ and $a_{k_2} - a_{i_2}$ are upper bounded by $a_{m+1} - a_1$. □

Accuracy Maximization Using LP-Relaxation And Rounding(AMR^2)

From Theorem 2, the accuracy ratio between AMR^2 and the optimal schedule is bounded as follows:

$$\frac{A^\dagger}{A^*} \leq 1 + \frac{a_{m+1} - a_1}{A^*}.$$

Note that $a_{m+1} - a_1 < 1$ and A^* grows as $O(n)$, where n is the number of jobs. To see the latter, note that $A^* \geq na_1$ since the total accuracy for n jobs cannot be lower than na_1. Thus, $\frac{A^\dagger}{A^*}$ goes to 1 as n goes to infinity. Thus, AMR^2 is asymptotically optimal.

Corollary 1: If the processing times of all jobs on the ES are at most T, then $A^* \leq A^\dagger$.

Accuracy Maximization using Dynamic Programming (AMDP)

In this section, we consider the problem $\mathcal{P}_{\mathrm{I}}$, a special case of $\mathcal{P}$ where the jobs are identical, i.e., $p_{ij} = p_i$, for all models $i \in M$. We present Accuracy Maximization using Dynamic Programming (AMDP) for $\mathcal{P}_{\mathrm{I}}$. The formulation for $\mathcal{P}_{\mathrm{I}}$ is given below.

$$\underset{\mathbf{x}}{\text{maximize}} \quad \sum_{i=1}^{m+1} a_i \sum_{j=1}^{n} x_{ij}$$

$$\text{subject to} \quad \sum_{i=1}^{m} p_i \sum_{j=1}^{n} x_{ij} \leq T \tag{40}$$

$$p_{m+1} \sum_{j\in 1}^{n} x_{(m+1)j} \leq T \tag{41}$$

$$\sum_{i=1}^{m+1} x_{ij} = 1, \quad \forall j \in J \tag{42}$$

$$x_{ij} \in \{0,1\}, \quad \forall i \in M, \forall j \in J. \tag{43}$$

Lemma 2: Under an optimal schedule, the number of jobs assigned to the ES is given by $n_c = \lfloor \frac{T}{p_{m+1}} \rfloor$.

We define $n_l = n - n_c$. Since the jobs are identical, without loss of generality, we assign the last n_c jobs to the ES. We are now only required to compute the optimal assignment for jobs $j \in \{1, \ldots, n_l\}$ to the models 1 to m on the edge device. Thus, given Lemma 2, solving $\mathcal{P}_{\mathrm{I}}$ is reduced to solving the following problem $\mathcal{P}_{\mathrm{I}}'$:

$$\underset{\mathbf{x}}{\text{maximize}} \quad \sum_{i=1}^{m} a_i \sum_{j=1}^{n_l} x_{ij}$$

$$\text{subject to} \quad \sum_{i=1}^{m} p_i \sum_{j=1}^{n_l} x_{ij} \leq T, \tag{44}$$

$$\sum_{i=1}^{m} x_{ij} = 1, \quad \forall j \in \{1, \ldots, n_l\}$$

$$x_{ij} \in \{0,1\}, \forall i \in M \backslash \{m+1\}, \forall j \in \{1, \ldots, n_l\}. \tag{45}$$

Accuracy Maximization using Dynamic Programming (AMDP)

We do a variable change to formulate the CCKP. Let r denote an index taking values from $\{1,\ldots,mn_l\}$. We define new variables z_r, accuracies $\bar{a}_r$, and processing times $\bar{p}_r$ as follows: for $i \in M\backslash\{m+1\}$ and $j \in \{1,\ldots,n_l\}$,

$$z_r = \{x_{ij} : r = j + (i-1)n_l\},$$

$$\bar{a}_r = a_i,\ (i-1)n_l \le r < in_l,$$

$$\bar{p}_r = p_i,\ (i-1)n_l \le r < in_l.$$

The CCKP using $\{z_r : 1 \le r \le mn_l\}$ as the decision variables is stated below.

$$\underset{\{z_r\}}{\text{maximize}} \quad \sum_{i=1}^{mn_l} \bar{a}_r z_r$$

$$\text{subject to} \quad \sum_{r=1}^{mn_l} \bar{p}_r z_r \le T, \tag{46}$$

$$\sum_{r=1}^{mn_l} z_r = n_l, \tag{47}$$

$$z_r \in \{0,1\},\ \forall r \in \{1,\ldots,mn_l\}. \tag{48}$$

Let $\{z_r^* : 1 \le r \le mn_l\}$ denote an optimal solution for CCKP. The CCKP can be interpreted as follows. We have mn_l items, where each item represents a model and there are n_l copies of the same model. Since the jobs are identical, the problem reduces to the number of times a model is selected, equivalent to the number of jobs assigned to it, such that all jobs are assigned. In the following lemma we state that solving CCKP results in an optimal solution for $\mathcal{P}_1'$.

Algorithm 2: AMDP.

1: $n_l = n - \lfloor \frac{T}{p_{m+1}} \rfloor$
2: Assign the jobs $j \in \{n_l+1,\ldots,n\}$ to the ES
3: Solve the CCKP for $\{z_r^*\}$ using the DP algorithm
4: Assignment for remaining jobs:
$x_{ij}^* = \{z_r^* : r = j+(i-1)n_l\}$ for all $i \in M\backslash\{m+1\}$, and $j \in \{1,\ldots,n_l\}$.

Accuracy Maximization using Dynamic Programming (AMDP)

Lemma 3: The solution $x^*_{ij} = \{z^*_r : r = j + (i-1)n_l\}$ is an optimal solution for $\mathcal{P}_\mathrm{I}'$.

Proof: By construction, $\mathcal{P}_\mathrm{I}'$ and CCKP have one-to-one mapping between the decision variables, have equivalent objective functions and constraints in (44) and (46). They only differ in the constraints (45) and (47). We note that (47) is equivalent to

$$\sum_{j=1}^{n_l}\sum_{i=1}^{m} x_{ij} = n_l. \tag{49}$$

Let $\mathcal{P}_\mathrm{I}^\ddagger$ denote the problem $\mathcal{P}_\mathrm{I}'$ with the constraint (45) replaced by (49). From the above observations, $\mathcal{P}_\mathrm{I}^\ddagger$ is equivalent to CCKP, and thus it is sufficient to show that an optimal solution $\{x^\ddagger_{ij}\}$ for $\mathcal{P}_\mathrm{I}^\ddagger$ is optimal for $\mathcal{P}_\mathrm{I}'$. Since (49) is a relaxation of the constraint in (45), the optimal objective value of $\mathcal{P}_\mathrm{I}^\ddagger$ should be at least the optimal objective value of $\mathcal{P}_\mathrm{I}'$. On the other hand, given $\{x^\ddagger_{ij}\}$, consider the assignment where for each model i, we assign $\sum_{j=1}^{n_l} x^\ddagger_{ij}$ jobs to it. Given that the jobs are identical, and from (49), all the n_l jobs will be assigned exactly once to some model. Thus, this assignment is feasible for $\mathcal{P}_\mathrm{I}'$ and objective value under this assignment will be equal to the optimal objective value of $\mathcal{P}_\mathrm{I}^\ddagger$. Thus, $\{x^\ddagger_{ij}\}$ is also an optimal solution for $\mathcal{P}_\mathrm{I}'$.

The summarize the main steps of the algorithm. Let s, k, and τ denote positive integers. We define $y_s(\tau, k)$ as the maximum accuracy that can be achieved by selecting items from the set $\{1, \ldots, s\}$, where $s \leq mn_l$, given a time constraint τ ($\leq T$) and the number of items to be selected are k ($\leq n_l$).

$$y_s(\tau,k) = \max\left\{\sum_{r=1}^{s}\bar{a}_r z_r \,\middle|\, \sum_{r=1}^{s}\bar{p}_r z_r \leq \tau, \sum_{r=1}^{j} z_r = k, z_r \in \{0,1\}\right\} \tag{50}$$

The DP iterations are given below:

$$y_s(\tau,k) = \begin{cases} y_{s-1}(\tau,k) & \text{if } \bar{p}_s \geq \tau \\ \max\{y_s(\tau-\bar{p}_s, k-1)+\bar{a}_s, y_{s-1}(\tau,k)\} & \text{otherwise.} \end{cases}$$

We compute the solution for $y_s(T, n_l)$, where $s = mn_l$.

The computational complexity of the DP algorithm is $O(mnT)$ and AMDP has the same computational complexity.

Extension Of AMR^2

Inference jobs can be executed locally on one of the m ML models, or remotely on one of the K ESs. Again, our objective is to maximize the total inference accuracy subject to a constraint T on makespan. Let the index set for the ML models is $M' = \{1, .., m, m+1, \ldots, m+K\}$, where $m+1$ to $m+K$ denote the models on the ESs. We formulate the following constraints:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} p_{ij}x_{ij} \leq T \tag{51}$$

$$\sum_{j=1}^{n} p_{ij}x_{ij} \leq T, \forall i = \{m+1, .., m+K\} \tag{52}$$

$$\sum_{i=1}^{m+K} x_{ij} = 1, \forall j \in J \tag{53}$$

$$x_{ij} \in \{0, 1\}, \forall i \in M', \forall j \in J, \tag{54}$$

are interested in the following accuracy maximization problem $\mathcal{P}_K$:

$$\underset{\mathbf{x}}{\text{maximize}} \quad A = \sum_{i=1}^{m+K}\sum_{j=1}^{n} a_i x_{ij}$$

$$\text{subject to} \quad (51), (52), (53), \text{ and } (54).$$

Algorithm 3: AMR^2 for K ESs.

1: **Input**: p_{ij}, for all $i \in M'$ and $j \in J$.
2: Solve the LP-relaxation of $\mathcal{P}_K$.
3: The fractional job are collected in F.
4: **for all** $j \in F$ **do**
5: **if** $\bar{x}_{ij} > \bar{x}_{kj}$ **then**
6: $x^{\dagger}_{ij} = 1$
7: **else**
8: $x^{\dagger}_{kj} = 1$
9: **end if**
10: **end for**
11: **Output**: Assignment matrix $\mathbf{x}^{\dagger}$ and total accuracy $A^{\dagger}$

Extension Of AMR^2

Lemma 4: For the LP-relaxation of $\mathcal{P}_K$, there exists an optimal basic solution with at most $K+1$ fractional jobs.

Theorem 4: For the case of K edge servers, AMR^2 provides a makespan of at most $2T$.

Theorem 5: Assuming a_{m+K} and a_1 are the highest and the lowest accuracy, respectively, the total accuracy achieved by an optimal schedule is at most $(K+1)\frac{(a_{m+K}-a_1)}{2}$ higher than the total accuracy achieved by the extended AMR^2 (Algorithm 3), i.e., $A^* - A^\dagger \leq (K+1)\frac{(a_{m+K}-a_1)}{2}$.

From Theorem 5, we have

$$\frac{A^\dagger}{A^*} \leq 1 + \frac{(K+1)(a_{m+K}-a_1)}{2A^*}.$$

Noting that K is a constant, $a_{m+K} - a_1 < 1$, and A^* grows as $O(n)$, we see that AMR^2 is asymptotically optimal for the case of K ESs.

Experimental Results

In this section, we first present the experimental setup. We then present the implementation details for estimating the processing and communication times. As explained in Section II, the aspect of multiple models on the ED has not been considered in computation offloading literature and there are no existing algorithms that are applicable for the problem at hand for a performance comparison. Therefore, we present the performance comparison between AMR^2 and a baseline Greedy Round Robin Algorithm (Greedy-RRA). Given the list of jobs, Greedy-RRA offloads them from the start of the list to the ES until the constraint T is met. The remaining jobs are assigned in a round robin fashion to the models on the ED until the constraint T is met. Any further remaining jobs are assigned to model 1. Note that Greedy-RRA solution may violate the time constraint T and its runtime is $O(n)$. Finally, we also demonstrate the performance of the extended AMR^2 for the case of multiple ESs.

For the experimental setup, we chose image classification as the ML application due to its prevalence. Nevertheless, we emphasize that our system model and the proposed algorithm apply to other ML applications.

Our experimental setup comprises a Raspberry Pi device (the ED) and a local server (the ES) that are connected and located in the same LAN. Raspberry Pi has 4 cores, 1.5 GHz CPU frequency, and 4 GB RAM, with the operating system Raspbian 10, while the server has 512 cores, 1.4 GHz CPU frequency, and 504 GB RAM, with the operating system Debian 11. All the functions on Raspberry Pi and on the server are implemented using Python 3. We used HTTP protocol to offload images from Raspberry Pi to the ES and implemented HTTP Client and Server using Requests and Flask, respectively. The LP-relaxation problem of P is solved using PuLP library on the ED. The data samples are images from the ImageNet dataset for which we use DNN models for inference. On Raspberry Pi we import, from the TensorFlow Lite library, two pre-trained MobileNets corresponding to two values 0.25 and 0.75 for the hyperparameter α, which is a width multiplier for the DNN [38].

size of the DNN model. On the ES, we import a pre-trained ResNet50 model [6] from the Tensorflow library. The ResNet50

Experimental Results

TABLE I
TEST ACCURACIES OF THE CONSIDERED DNN MODELS [4]

Model	Top-1 Accuracy
MobileNet $\alpha = 0.25$ (model 1)	0.395
MobileNet $\alpha = 0.75$ (model 2)	0.559
ResNet50 (model 3)	0.771

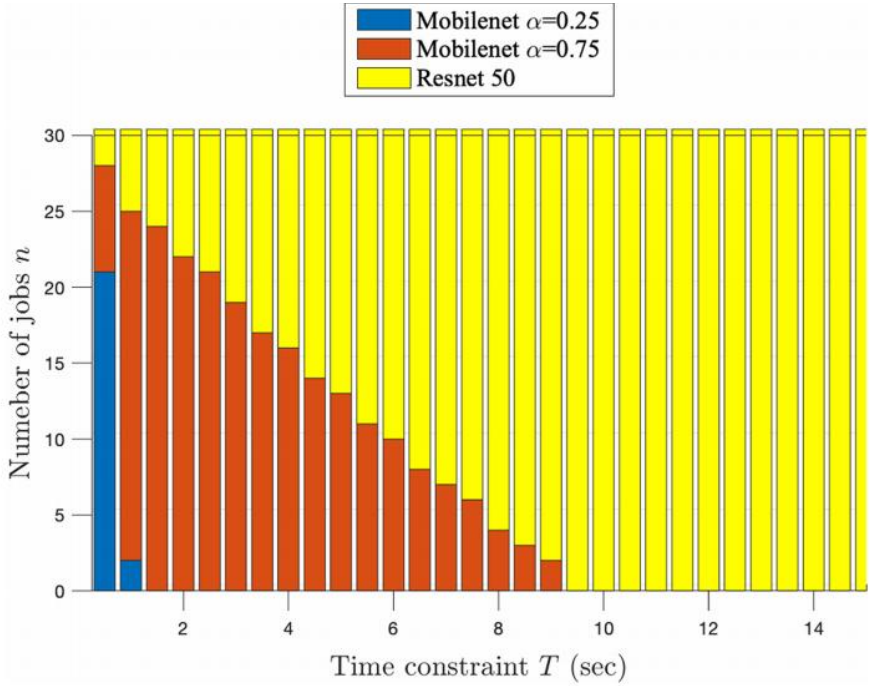

Fig. 4. Job assignment under AMR^2 for varying T.

In Fig. 4, we examine the number of jobs assigned to different models under AMR^2. Observe that as T increases the number of jobs assigned to larger models increases. Also, note that MobileNet $\alpha = 0.25$ is only being used when T is small. In all

Experimental Results

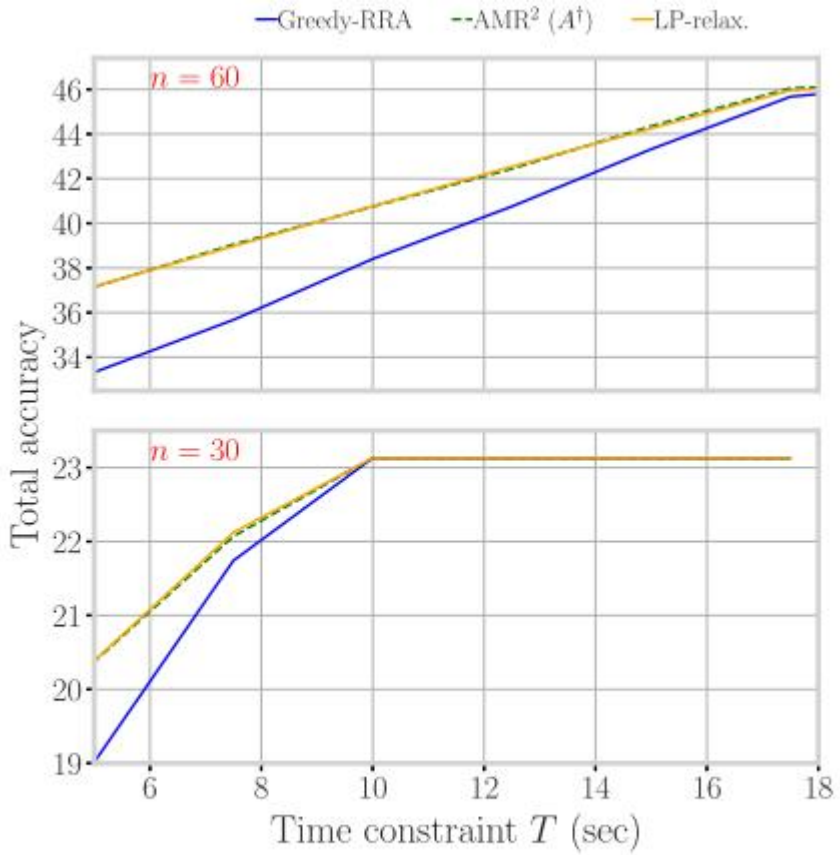

Fig. 5. Total accuracy varying T for $n = (30, 60)$.

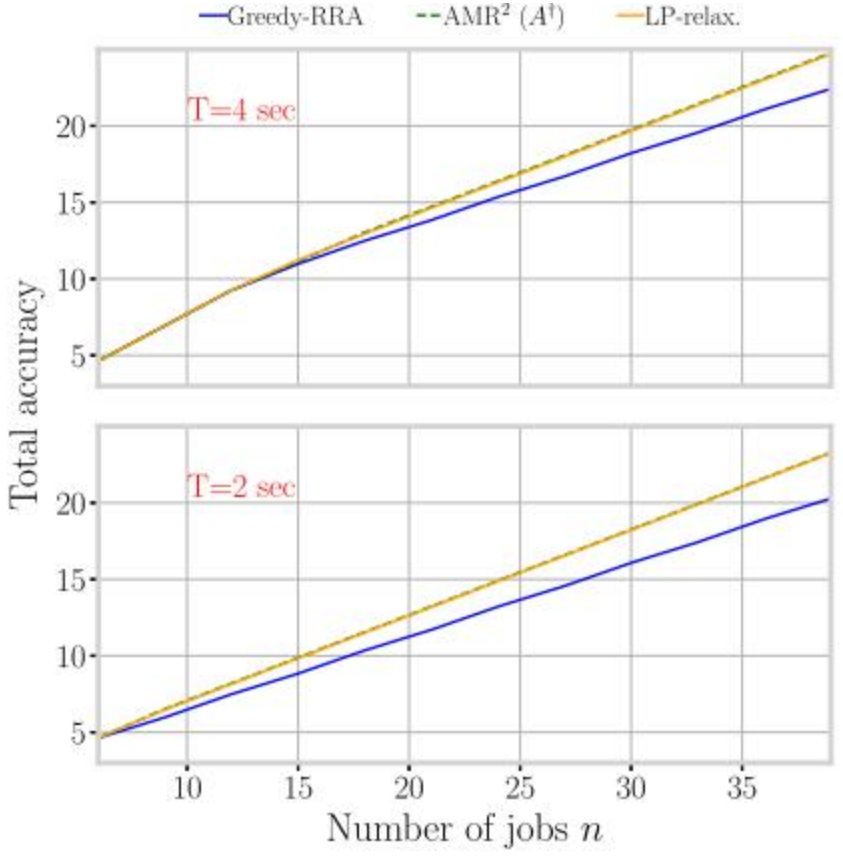

Fig. 6. Total accuracy varying n with $T = (2, 4)$ sec.

From Fig. 5, we observe that AMR^2 always has higher total accuracy than Greedy-RRA with a percentage gain between 20–60% averaging at 40%, but the percentage gains are lower at smaller T. The latter fact is also confirmed in Fig. 6 when $T =$ 2 sec. This is expected, because for $T = 2$ sec, not many jobs can be offloaded to the server as the processing times are around 0.3 seconds. For $T = 4$ sec we see significant gains of around 40–50%.

Experimental Results

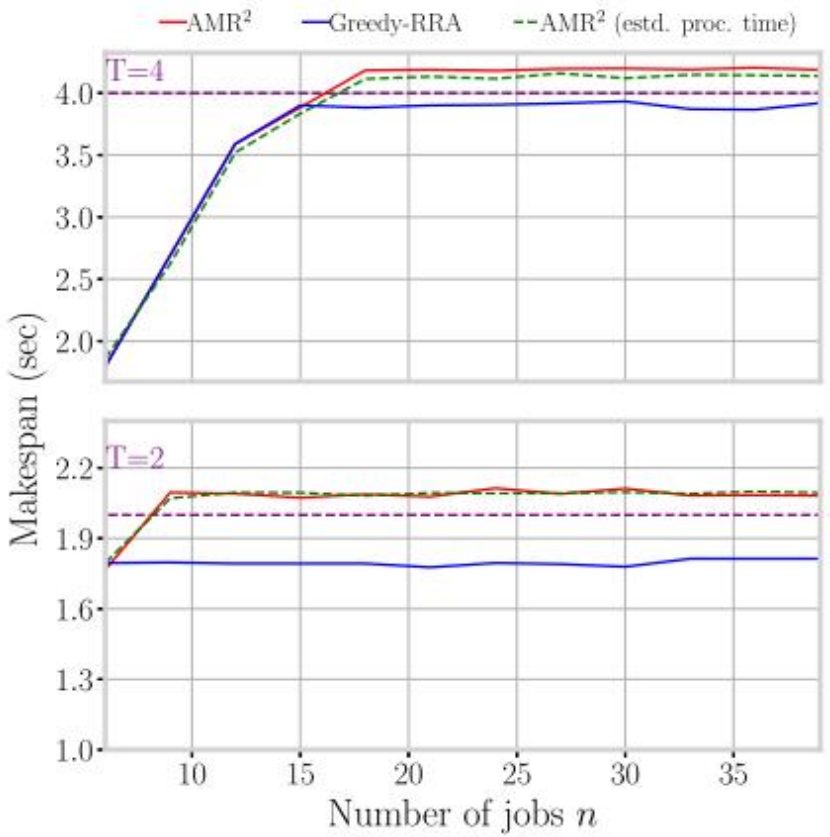

Fig. 7. Makespan under AMR^2 and Greedy-RRA for varying n, and $T = 2$ sec and $T = 4$ sec.

processing times are small. For $T = 4$, AMR^2 violates T for $n \geq 17$, but then it saturates at a makespan with a maximum percentage of violation of 15%. This is expected because from Lemma 1 there cannot be more than two fractional jobs irrespective of n value and thus, the constraint violation due to the reassignment of the fractional jobs do not increase beyond $n = 30$. This saturation effect can also be observed for $T = 2$. In this case, the percentage of violation under AMR^2 is higher because the processing times on the server are comparable to $T = 2$ sec and reassigning a fractional job to the server results in a higher percentage of violation.

Experimental Results

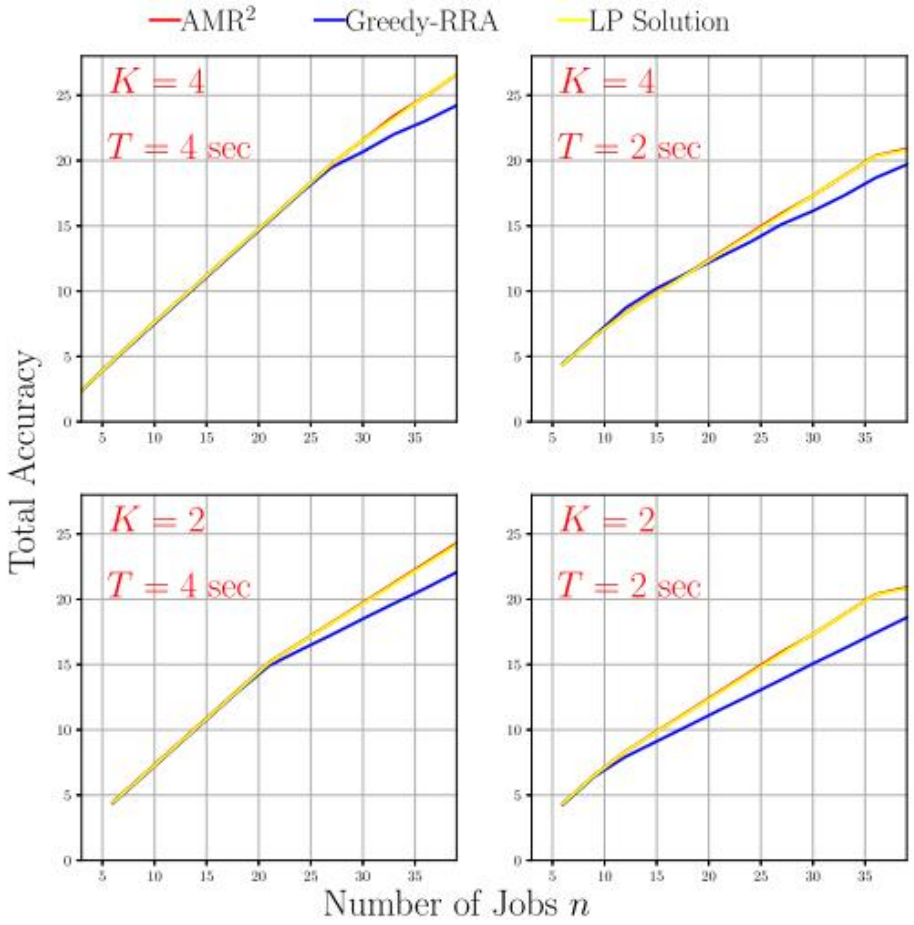

Fig. 8. Total Accuracy under AMR^2 and Greedy-RRA for varying n, and $T = 2$ sec and $T = 4$ sec. On the left in the case of 2 ESs, on the right with 4 ESs.

assignment by including k ESs. From Fig. 8, we observe that as the time constraint increases, the difference between the total accuracy achieved by AMR^2 and Greedy-RRA decreases.

time T on them. Also, as the number of edge servers increases the total accuracy increases. For example, for $T = 2.0$ sec, the total accuracy achieved by AMR^2 increases by 20% for $K = 4$ when compared to $K = 2$. Furthermore, observe that for $K = 2$ we achieve a total accuracy of 21 for 30 jobs, but for $K = 4$ we reach the same accuracy with 27 jobs. Thus, the average accuracy per job increases as k increases, we observe similar trends as in

the case of $K = 1$ for increasing T. The figures are not presented due to redundancy.

Experimental Results

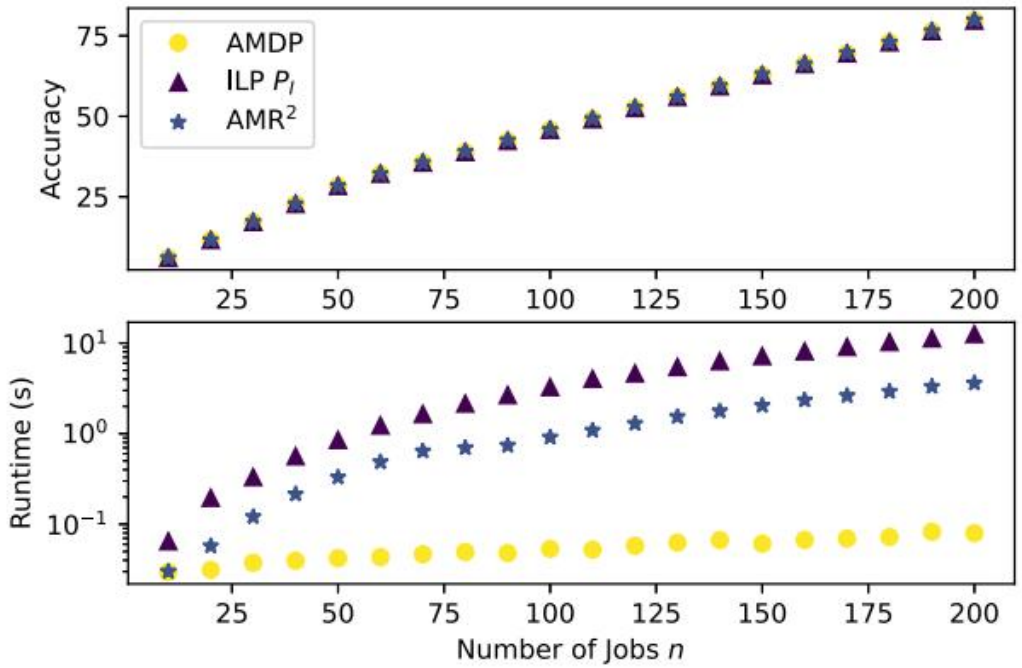

Fig. 9. Comparison between the solution of AMDP, an ILP solver which solves $\mathcal{P}_I$, and AMR^2.

In Fig. 9, we present this comparison by varying the number of jobs between 10 to 200. Observe from the upper sub-figure that AMDP reaches the same solution as the ILP solver. From the lower sub-figure, observe that the runtime of AMDP is much lower (by more than an order of magnitude) when compared to the runtime of the ILP solver, which increases approximately linearly (note that the figure is in log scale) with the number of jobs. Interestingly, we found that AMR^2 also provides an optimal solution for these problem instances as there are no fractional jobs to be rounded. However, the advantage of using AMDP is that it has a lower runtime (an order of magnitude less) than that AMR^2.

南京邮电大学先进网络与经济实验室

谢谢！

计算机学院 软件学院
网络空间安全学院
School of Computer Science

BDSIP