Khalid Alsubhi, Zuhaib Imtiaz, Ayesha Raana , M. Usman Ashraf, Babur Hayat, 2020. MEACC: an energy-efficient framework for smart devices using cloud computing systems. *Frontiers of Information Technology & Electronic Engineering*, 21(6):917-930. https://doi.org/10.1631/FITEE.1900198



# MEACC: an energy-efficient framework for smart devices using cloud computing systems

**Key words:** Offloading; Smart devices; Cloud computing; Mobile computing; Power consumption

Corresponding author: M. Usman Ashraf
E-mail: usman.ashraf@skt.umt.edu.pk
ORCID: https://orcid.org/0000-0001-7341-8625

# Motivation

- The smart phone has been adopted as an appropriate and probably the optimum tool for communication using voice, text, and video. The use of the smart mobile phone is now expanding in business applications as well.
- However, despite the growing demand for smart phone use and its applications, these devices have fallen short of passable resources like computational capacity and battery power. At the user end, there is a trade-off between mobile devices that use many resources and resource-constrained mobile devices.
- To mitigate this problem, different energy augmentation techniques have been proposed. These techniques have targeted the resource constraints in smart phones.

# Main idea

- In this study, the communication cost and execution cost are evaluated for a file to be offloaded to cloud servers. While calculating the execution cost, different attributes are considered including CPU usage, random access memory (RAM) usage, and battery consumption for file execution.
- In contrast, the communication cost includes the response time, i.e., the time taken by any file to be offloaded to the cloud and give a response.

# Main idea (Cont'd)

- We propose a new approach named mobile energy augmentation using cloud computing (MEACC) to make job offloading decisions.
- MEACC contains two algorithms, one for making offloading decisions and the other for file computation.
- The offloading decision algorithm is the baseline for determining a final decision to offload any task or not.
- The file computation supporting algorithm is used to complete the profiler log information; it reads the inputted file word by word and provides a total word count for the file.

# Method: Algorithm 1

**Algorithm 1** Offloading decision

**Input:** TextFile
**Output:** LocalExecutionCost, CommunicationCost, ExecutionDecision(Local/Cloud)

```
1   ProcedureTakeDecision(Textfile)
2   FS←TextFileSize
3   Threshold←GetDeviceInfo
4   RR←EstimateRequiredResource
5   if Threshold[index]<RR[] then
6       if (Wi-Fi==true; CC+Const<LEC) then
7           return offload
8       else
9           return NotOffload
10      end if
11  else
12      return NotOffload
13  end if
```

The offloading decision algorithm takes a text file as the input and decides whether to offload the execution to the cloud based on two parameters:

- Local execution cost
- Communication cost

# Method: Algorithm 2

**Algorithm 2** File computation

```
Input: TextFile
Output: TotalWords, TotalCount, ExecutionTime
1   ProcedureTakeDecision(Textfile)
2   LF←Loadtxtfile
3   while (LF.readLine()=Null) do
4      Text←readLine
5      Array[]←Text.Split()
6      Array.sort(Array[index])
7      for (i=0; i<ArrayLenght; i=i+1) do
8         for (j=i+1; j<ArrayLength; j=j+1) do
9            if word=true then
10              count++
11              Words∪word
12              TotalCount∪count
13           else
14              break
15           end if
16        end for
17     end for
18  end while
19  return TotalWords, TotalCount, ExecutionTime
```

This algorithm is to complete the profiler log information. The algorithm reads the inputted file word by word, and determines the total number of words in the file. After file read is completed, the execution time to complete this task is displayed on the dashboard.

# Major results

- Our algorithm can calculate the estimated mobile resources required for file execution. Based on the calculation results, the algorithm decides whether to offload its execution.

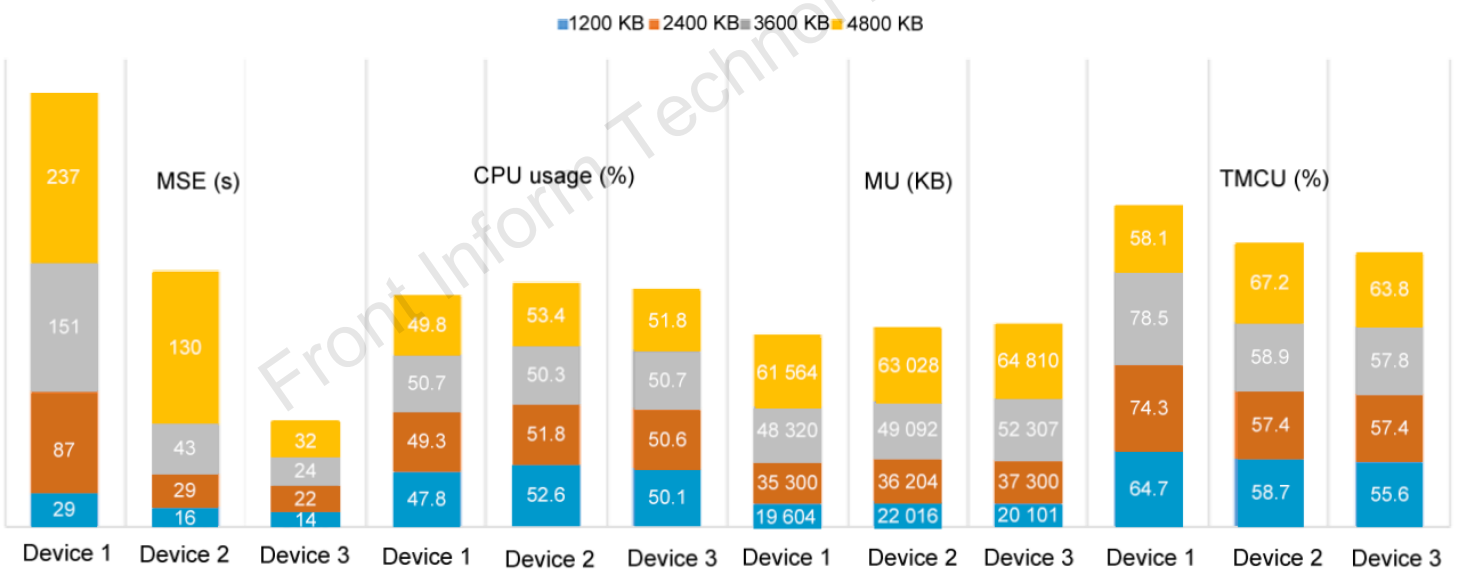


**Fig. 4 Resource consumption on mobile devices**
MSE: mobile side execution time; MU: memory usage; TMCU: total mobile CPU usage

# Major results (Cont'd)

- Compared with Monica's algorithm and mobile capabilities augmentation using cloud computing (MCACC), our algorithm takes less file execution time on the mobile side.

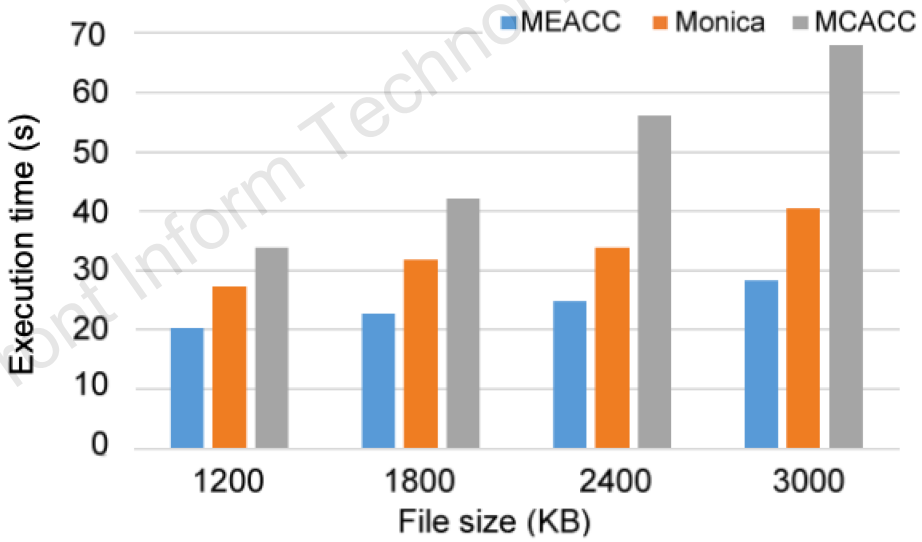


**Fig. 11 MEACC vs. Monica execution in the cloud with communication cost**

# Conclusions

- A decision-making approach called MEACC is proposed to augment the battery life of smart mobile phones by making the offloading decision for resource-intensive tasks. On the cloud side, communication cost was calculated based on the Internet speed.
- The MEACC approach improved the offloading decision-making process by implementing an impartially precise execution cost prediction method that was evaluated by the Android application AnotherMonitor.

# Conclusions (Cont'd)

- The quantified execution cost was compared to the communication cost of offloading by evaluating the efficiency. MEACC outperformed the other selected models, the Monica algorithm and MCACC.
- The MEACC approach includes communication cost calculation and the overhead of uploading/downloading the task for its complete processing, which makes it more scalable for multiple Internet service providers.