# CSE 535 : Mobile Offloading Project

**Gokul Vijayaraghavan**
Arizona State University
gvijaya5@asu.edu
1222645060

**Aparokshith Rao**
Arizona State University
asrao13@asu.edu
1218314175

**Adithya Natarajan**
Arizona State University
anatar17@asu.edu
1221965901

**Sundaravadivel Chandrasekaran Ponmudi**
Arizona State University
sundarav@asu.edu
1222352703

## ABSTRACT

**The project focuses on creating a distributed computing infrastructure using only mobile devices which are connected via wifi. The system uses a master-slave architecture, that is the model comprises of master application in one device and slave application in multiple devices. For this project to demonstrate distributed computing we use matrix multiplication. The master shares the matrix with slaves which do the multiplication and send back the results. The master then combines the results and gives us the output. In addition to the demonstration we have also done comparison between with and without distribution.**

### Keywords

Distributed computing; matrix multiplication; mobile computing; mobile offloading; master-slave operation

## INTRODUCTION

There has been tremendous growth in the computing field. Today we can do the tasks done by a desktop within our hands with the help of mobile devices. But there is not much power in the mobile devices as most of them are battery dependent and restricting its usage.In this project we try to solve this by using the concept of offloading. That basic idea of this concept is to share the workload among the devices so that the time taken and computational budget needed is low. To showcase this we have taken the idea of matrix multiplication. The system does not need any powerful chipsets or high speed wifi.Thus reducing the requirements and time consumed.

## PROJECT SETUP

Our architecture consists of one master application and three slave applications.

### Master Setup

| | |
|---|---|
| Device model: | Pixel 4a |
| OS: | Android 11 |

### Slave Setup

**Slave 1:**

| | |
|---|---|
| Device model: | Moto G8+ |
| OS: | Android 10 |

**Slave 2:**

| | |
|---|---|
| Device model: | B100DL |
| OS: | Android 9 |

**Slave 3:**

| | |
|---|---|
| Device model: | B100DL |
| OS: | Android 9 |

The application needs the following permissions from devices before it can start

- ACCESS_WIFI_STATE
- ACCESS_NETWORK_STATE
- INTERNET
- WRITE_EXTERNAL_STORAGE
- READ_EXTERNAL_STORAGE
- MANAGE_EXTERNAL_STORAGE
- ACCESS_FINE_LOCATION
- ACCESS_COARSE_LOCATION
- ACCESS_BACKGROUND_LOCATION

## METHODOLOGY

The system uses distributed computing as its backbone. In distributed computing the master takes all the input and identifies the tasks to be done, then the work is split by the master to different devices called slave. In this project for connecting the devices we use wifi. Once the devices are connected, the slaves share their information such as battery level, latitute and longitude. Over here we set a threshold value with which we filter if the slave is capable for connection or not. For battery level if the slaves batery is less than 25% we drop it. We calculate the distance between the master and slave using the information obtained during connection and we use the eucledian distance formula. If the distance is above 150 units the slave connection is rejected.

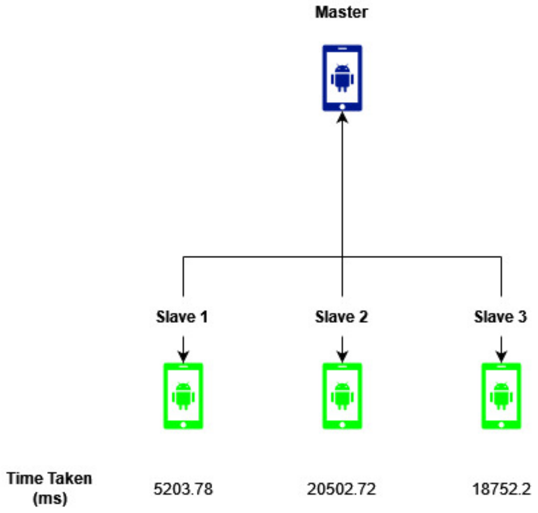


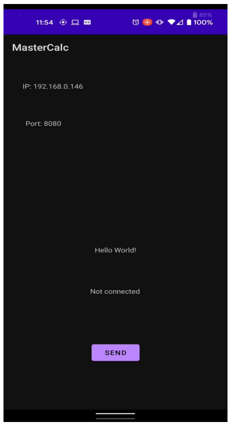
Fig.1 Architecture

For example we have chosen the task of matrix multiplication. The master takes input of two 50 x 50 matrices A and B and shares it with the slaves. Here we chose round robin fashion for sharing the matrix, that is the first matrix A is sent in whole to all the slave devices and the matrix B is sent columnwise. This means if there  are four slaves the first column is sent to first slave, the 2nd column to the 2nd device and so on, when the number of column goes over the number of devices it starts again from the 1st.

## IMPLEMENTATION

The user interface of both the master and slave devices have been shown below. For this project all the devices must be connected in the same wifi network.

- **Step 1:** We install master application in master nodes and slave applcation in slave nodes.



IP: 192.168.0.146

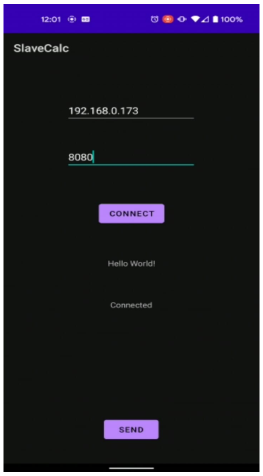
Port: 8080

Hello World!

Not connected

Fig.2 Master application MasterCalc and Slave application SlaveCalc

- **Step 2:** The required permissions are given as said in project setup.
- **Step 3:** The discovery in the master application starts. Now we connect the devices by entering the Master ip address and Master’s port that is usually 8080.



192.168.0.173

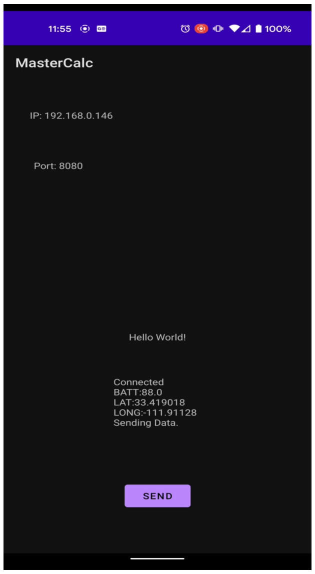
8080

Hello World!

Connected

Fig.3 SlaveCalc is connected

- **Step 4:** Once it is done, the slave is connected to the master and sends its battery and location information to the master device.



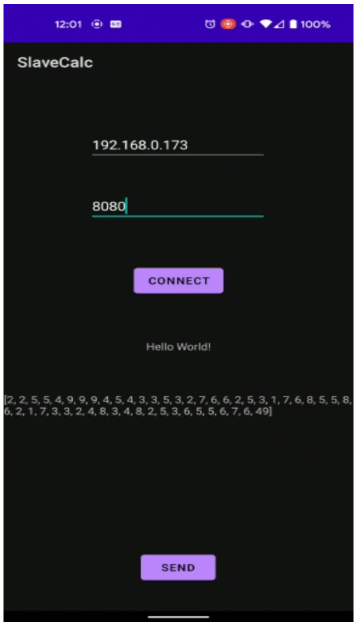
```
IP: 192.168.0.146

Port: 8080

Connected
BATT:88.0
LAT:33.419018
LONG:-111.91128
Sending Data.
```

Fig.4 MasterCalc gets the required details

- **Step 5:** We then send the matrix from MasterCalc to the SlaveCalc by pressing the send button.



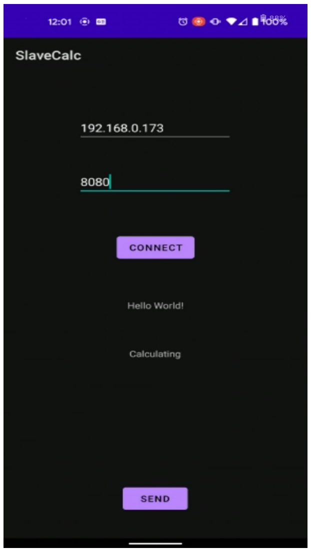
```
192.168.0.173

8080

[2, 2, 5, 5, 4, 9, 9, 9, 4, 5, 4, 3, 3, 5, 3, 2, 7, 6, 6, 2, 5, 3, 1, 7, 6, 8, 5, 5, 8,
6, 2, 1, 7, 3, 3, 2, 4, 8, 3, 4, 8, 2, 5, 3, 6, 5, 5, 6, 7, 6, 49]
```

Fig.5 SlaveCalc receives the matrix

- **Step 6:** At this point the slave devices have whole of Matrix A and respective columns of matrix B.
- **Step 7:** This step we calculate the matrix in the slave device and send it back .



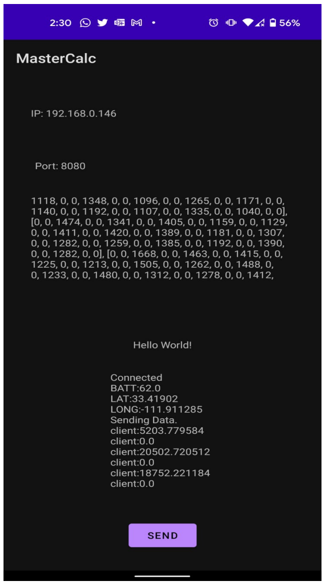
```
192.168.0.173

8080

Calculating
```

Fig.6 SlaveCalc calculates and sends back the matrix

- **Step 8:** The last step where the master receives the parts, arranges them and displays out.



```
IP: 192.168.0.146

Port: 8080

1118, 0, 0, 1348, 0, 0, 1096, 0, 0, 1265, 0, 0, 1171, 0, 0,
1140, 0, 0, 1192, 0, 0, 1107, 0, 0, 1335, 0, 0, 1040, 0, 0],
[0, 0, 1474, 0, 0, 1341, 0, 0, 1405, 0, 0, 1159, 0, 0, 1129,
0, 0, 1411, 0, 0, 1420, 0, 0, 1389, 0, 0, 1181, 0, 0, 1307,
0, 0, 1282, 0, 0, 1259, 0, 0, 1385, 0, 0, 1192, 0, 0, 1390,
0, 0, 1282, 0, 0], [0, 0, 1668, 0, 0, 1463, 0, 0, 1415, 0, 0,
1225, 0, 0, 1213, 0, 0, 1505, 0, 0, 1262, 0, 0, 1488, 0,
0, 1233, 0, 0, 1480, 0, 0, 1312, 0, 0, 1278, 0, 0, 1412,

Connected
BATT:62.0
LAT:33.41902
LONG:-111.911285
Sending Data.
client:5203.779584
client:0.0
client:20502.720512
client:0.0
client:18752.221184
client:0.0
```

Fig.7 MasterCalc gets the matrices and displays with time taken and battery level

The Fig.1 shows the time taken and battery consumed by each client and the total time consumed is the maximum by the slaves and it is 20502.72 ms.

- **Step 9:** For identifying the failure or disconnection in slave we initialize all the response matrix with 0 in master, if the corresponding row remains the same that is 0 without any change, means the particular slave failed.

## OBSERVATION

The main thing we observed here is that the master takes 54781.25 ms which is almost 250% of the time taken by offloading that is 20502.72 ms. For unbiased comparison we used the same type of multiplication for master multiplication as in the offloading.

## LIMITATION

The major limitation we found was security issues. This project is safe when the devices are in a trusted location or, it can be dangerous when used in public locations. Compared with wifi connection, bluetooth can be safer but it can slow down the model by a significant level. For further work we may involve encryption for the message. The range of wifi is restricted to 658ft in this project and we can use a better and powerful connection for future works. The last part of the operation where the master combines the matrixes takes a lot of time compared to other tasks.

## COMPLETION OF TASKS

| S.No | Task | Assignee |
|---|---|---|
| 1. | Developing a Master mobile application that can be used to start a program that collects battery levels from mobile phones and lists it in a file | Sundar - Adithya |
| 2. | Developing a service discovery application, that sends queries to available mobile phones in close proximity through Bluetooth or WiFi to request participation | Aparokshith - Gokul |
| 3. | Developing a dispatcher application that choses a mobile phone among the ones that accepted the request based on matching some requirements. Requirements can be minimum battery level, and location proximity | Sundar - Aparokshith |
| 4. | Sending requests to start battery monitoring application on the slave side | Adithya - Gokul |
| 5. | Developing a slave application that can receive a request from a master through Bluetooth or WiFi | Aparokshith - Adithya |
| 6. | Adding a feature so that application can also monitor the battery level and current location and send it back to the master if the user decides to consent. | Sundar-Adithya |
| 7. | Extending the above feature for periodic monitoring | Sundar - Gokul |
| 8-12 | Using the framework for matrix multiplication using distributed computing | Aparokshith - Sundar |
| 13. | Identifying any error or failure in the slave client | Gokul - Adithya |
| 14. | Estimating execution time of the Matrix Multiplication if done only on the Master | Adithya - Sundar |
| 15. | Estimating execution time If done using the distributed approach with no failure | Aparokshith - Gokul |

| | | |
|---|---|---|
| 17. | Estimating power consumption of Master and Slave nodes without distributed computation | Adithya-Gokul |
| 18. | Estimating power consumption of Master and Slave nodes with distributed computation | Gokul - Aparokshith |

## CONCLUSION

The idea of this project is to reduce computing time when there is a computational budget. We were able to successfully achieve this with the help of mobile offloading. In this project we only use wifi and GPS and a few mobile devices. With the help of matrix multiplication we can see the reduction in the time taken. An implementation of this offloading concept on high performance machines can help us with time and resource demanding tasks.

## ACKNOWLEDGMENTS

We would like to take this opportunity to thank Dr. Ayan Banerjee for his constant support throughout the project.

## REFERENCES

[1]https://www.youtube.com/watch?v=nw627o-8Fok&list=PLFh8wpMiEi88SIJ-PnJjDxktry4lgBtN3

[2]https://developers.google.com/nearby

[3]https://developer.android.com/guide/topics/connectivity/wifip2p

[4] S. Sou, "Mobile Data Offloading With Policy and Charging Control in 3GPP Core Network," in *IEEE Transactions on Vehicular Technology*, vol. 62, no. 7, pp. 3481-3486, Sept. 2013, doi: 10.1109/TVT.2013.2255899.