Fog and Edge Computing – Smart Waste Management System

Balazs Barcza
National College of Ireland
Dublin, Ireland
x19190638@student.ncirl.ie

1 Abstract

The amount of rubbish produced in cities is increasing daily as a result of the fast urbanization and population growth. Waste is produced by both people and other living things. Everywhere there is life and people, trash will be produced. In this work, it is suggested to construct an edge AI-based trash management system using Internet of Things (IoT) and AI technologies. The suggested solution is an inventive one that contributes to increasing waste management effectiveness while minimizing the need for physical work. Using clever garbage collection and management strategies can contribute to the creation of healthy environments. I give a tutorial on how to simulate real-time case studies for intelligent waste management systems using the iFogSim toolbox.

Keywords:

edge computing; modeling; iFogSim simulation tools; Smart Waste Management System

2 Introduction & Methodology

Fog computing is the latest architecture that functions as a layer between the cloud and the Internet of Things (IoT) worlds. Providing services will be provided to the network edge directly. IoT devices are being utilized for storing, processing, collecting, and transferring massive amounts of data created by new applications. IoT : The Internet of Things (IoT) is a network of physical objects "things" embedded with sensors, software, and other technologies to connect and transfer data with other devices and systems over the internet. While engaging with the massive amount of produced data, IoT interacts with applications that are time-sensitive and have demands for location-aware, high capacity processing, and low energy consumption. Fog computing is a decentralized computing infrastructure in which data, compute, storage and applications are located somewhere between the data source and the cloud. Like edge computing, fog computing brings the advantages and power of the cloud closer to where data is created and acted upon. Cities are being turned into smart cities that offer services for smart living, smart education, smart environment, and more as a result of the advancement of information and communications technology. Despite the numerous advantages of smart city innovations, many difficulties still need to be addressed because rising urbanization is driving a demand for smart cities solutions. The management of garbage has become one of the most pressing concerns of the day as a result of the modern society's rapid expansion and rising waste production. The traditional method, which is incompatible with contemporary technologies, involves manually collecting and monitoring rubbish from dumpsters. This requires more time, money, and labor. The current adoption of more intelligent waste management systems is a result of the development of Internet of Things (IoT) and Cloud technologies. However, when they adopt a centralized cloud computing paradigm, the bulk of present waste management systems could experience significant service delays and high bandwidth costs. By transforming centralized cloud computing systems into distributed edge computing systems, the server can be placed closer to the data sources. This provides many benefits, such as quick reaction times, efficient network bandwidth use, ability to do time-sensitive procedures, etc. The paper offers a garbage management system for smart cities based on edge AI as a remedy for the drawbacks of the centralized cloud computing paradigm. I provide a cloud-based trash management system that streamlines the disposal procedure. The cloud server and the smart trash cans are linked. The cloud server receives the data on the waste level after a predetermined amount of time. No action is taken until the waste level reaches the threshold value and the trash is collected, unless the trash can generates an alarm indicating that the threshold level has been reached.

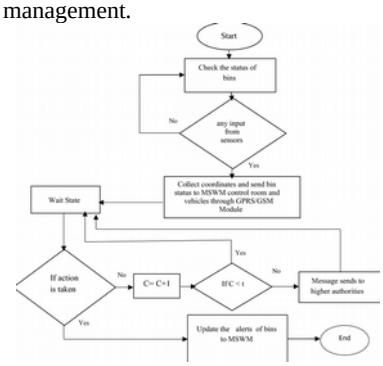
1. figure waste management system architecture [1

For the Smart Waste Management System, built-in classes for creating fog nodes, sensors, and actuators are available in Building Simulation with iFogSim.

- FogDevice: This class provides a constructor to create the fog devices and to define the hardware properties of the fog devices i.e. node name
- Sensor: By using the sensor class, we create IoT devices in iFogSim. Whilecreating the sensors, we define the gateway device id and setup link latency. Gateway device is the device with whom the sensor is attached and any devices, such as a router, fog node, or a proxy can serve as the Gateway. Setup link latency is the latency time to create a connection between the sensor and fog device.
- Actuator: Actuator class allows to create the objects in iFogSim that are used to display the output or any information.

3 Application requirements

There are various fog computing simulation tools available, each with its own set of methods and features. The fog infrastructure is made up of a collection of devices that don't have any computing capabilities and a collection of resources that provide compute power, storage capacity, and other services, all of which are coupled to form a complicated system. Clouds have provided users with elastic and on-demand storage, computing, and network management services throughout the last few decades. Cloud computing has a number of challenges, including security, information privacy, high computation latency, cost management, and resource management.

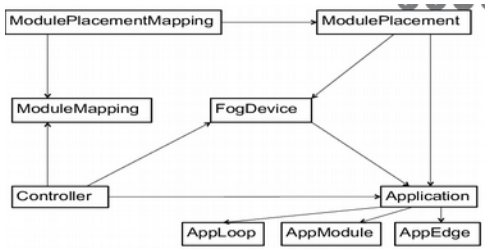


2.figure waste management system workflow [2]

Fog computing uses the benefits of the cloud environment to increase location awareness and workload latency by installing layered fog nodes between edge IoT devices and cloud data centers. With real-world infrastructure, it's hard to verify fog computing's usefulness and scalability. Because a single point of failure or mistake during computation might lead to erroneous results and waste of resources. Creating fog technologies and techniques necessitates a significant investment in resources such as connectivity, storage, and computing power, all of which are costly. A suitable solution is to use a simulation tool to create a realistic virtual world. Simulators are used to simulate real-world scenarios that are difficult to implement due to their complexity. In this case study I will use the iFogim tool.

- **iFogSim:**

This is an open source fog computing, edge computing, and IoT toolkit for modeling and simulating fog computing, IoT, and edge computing networks. The resource management methodologies in iFogSim may be adjusted to the research field. It is a fork of CloudSim, a well-known cloud simulator. When measuring the impact of resource management, it is a Java-based application that takes into account compute latency, energy consumption, network traffic, and cost.



5.figure Main iFogSim java classes and their relationships [3]

This architecture supports three primary services.

Power Management: Unlike cloud computing, fog nodes are limited in terms of storage, processing capabilities, and power consumption. It's crucial to keep track of and regulate the power usage of fog devices.

Resource Management: The whole architecture is designed to allow fog resources to be provisioned for a variety of IoT applications while ensuring great service quality.

Monitoring Component: This element of the architecture is in charge of tracking the status and performance of the application.

Available: https://github.com/Cloudslab/iFogSim

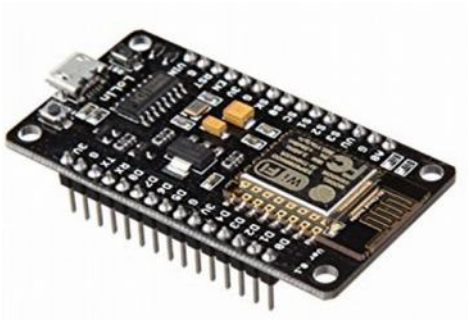
Smart Bin & Sensors

In this case study, my choice is Mr. Fill bin. Mr. Fill Ultra 120l is a smart, solar-powered, compacting waste or recycling station. This compacting version could hold 5 to 10 times as much trash as a standard garbage bin. It features sensors that monitor and record how filled it is and how much collection is being done. It gathers solar energy for waste compression and real-time status reporting.

6 Figure Mr. Fill Ultra 120l smart bin [4]

It has a 4G connection communication. Because I did not find the internet what type of sensors, It has, I have picked up Ultrasonic sensor HC-SR04 and Node MCU, temperature and humidity sensor. In my project, I am using those sensors, but I could integrate more sensors into the smart bin. Smart bin could have GPS, WIFI sensor, camera, metal detector, motion sensor , microbiology sensor, smoke detector, sound detector and weight detector etc.

The Node MCU is an open source Internet of Things (IoT) platform device that comprises of the ESP8266 module with a 32-bit ARM CPU, support for WiFi networks and 4g connection, and built-in 128 K Bytes flash memory, USB Port to power (5 volts), programming, and input pins with 1.8 V and 10 digital GPIOs pins.



7 figure Node MCU [5]

The ultrasonic sensor (HC-SR04) is a reasonably priced sensor that provides non-contact measurement capabilities from 2 cm to 400 cm with range accuracy up to 3 mm. It has an ultrasonic transmitter, a receiver, and a control circuit.

8 figure Ultrasonic sensor HC-SR04 [6]

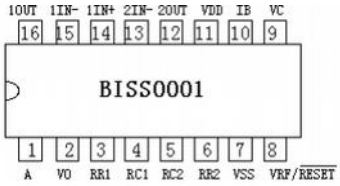
GND, VCC, Trig (Trigger), and Echo (Receive) are the other four pins (Ground). High frequency sound waves, which move through the air at the speed of sound, are occasionally released by the HC-SR04. A sensor is built into each garbage bin to notify the user when trash is at present level. Based on the distance traveled and the time required to return after reflection, we determine the amount of trash in the bin.

DHT22: The digital temperature and humidity sensor uses application-specific modules to capture technology and digital temperature and humidity sensor technology to ensure that products have high reliability and excellent long-term stability. It is a calibrated digital signal output temperature and humidity combined sensor.

- 3 to 5V power and I/O
- No more than 0.5 Hz sampling rate (once every 2 seconds)

9. figure DHT22 temperature-humidity sensor [7]

I used a PIR motion sensor to detect how many times the client open the smart bin. BISS0001 is a PIR (passive infra-red) controller, using analog mixing digital design technique and manufactures by CMOS process.

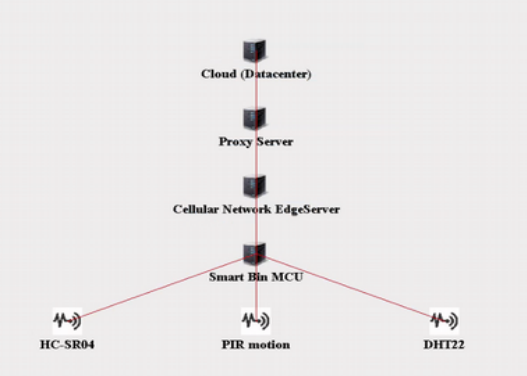


10. figure BISS0001 Micro Power PIR Motion Detector IC [8]

The smart waste bins are connected to the Node MCU. The node is connected to fog server via a router device. One of the crucial components of the sensor node's and the system's overall topology's design is wireless communication. Numerous technologies exist that offer features like high bandwidth (Wifi), long range (4G/GSM/CDMA), low power (Bluetooth low energy (BLE), or mesh networks (ZigBee). In this case study, I can choose a long-range wide area network (LoRaWAN) or 4G network solutions. It is not a good choice for Bluetooth and WiFi because it is short-range. (my choose is 4G). Practically 4G downlink and uplink rates of 5 to 12 Mbps and 2 to 5 Mbps, respectively, have been attained. Few 4G network providers have also achieved peak downlink speeds of approximately 35-50 Mbps.

Devices	MIPS	RAM (GB)	Up/Down Link (Mbps)	Level	P. Busy (W)	P. Idle (W)
Cloud	75000	64	1000	0	16*103	16*83.25
Proxy Server	7000	32	500	1	107.339	83.43
Cellular Network (edgeserver1,edgeserver2)	7000	32	35	2	107.339	83.43
Node MCU	7000	4	20	3	87.53	82.44
Sensors	DHT22	PIR motion sensor	HC-SR04			

11. Figure Simulating Fog Computing Applications using iFogSim Toolkit



12. figure iFogSim topology

4. iFogSim building simulation for an intelligent waste management system

Before we want to build a simulation we have to be sure we have a few programs preinstalled.

- Git (to download the iFogsim)
- Java 8 (to run the simulation)
- Eclipse (programming interface)

It this topic we will discuss the simulation and code. Next step we have to build our simulation in iFogSim.

1. The first make a new class in org.fog.test.perfeval package. (*WasteManangment.java*)
2. We have to import a few packages to run the simulation
   import org.fog.entities.FogDevice;
   import org.fog.entities.Sensor;
3. We can build the topology to make visual our simulation
4. We have to create the main class where we run the simulation and set up how many areas and bin per areas to create.
   static int numOfAreas = 1;
   static int numOfBinPerArea = 4;
5. We can set up do we want to cloud instance
   private static boolean CLOUD = false;
6. Create a Arraylist to save the sensors and FogDivice
   static List<FogDevice> fogDevices = new ArrayList<FogDevice>();
   static List<Sensor> sensors = new ArrayList<Sensor>();
7. We have to create an application instance and pass appId
   Application application = createApplication(appId, broker.getId());
8. start and stop the simulation
   CloudSim.startSimulation();
   CloudSim.stopSimulation();
9. Creates the fog devices in the physical topology of the simulation.

```
private static void createFogDevices(int userId, String appId) {
        FogDevice cloud = createFogDevice("cloud", 44800, 40000, 100, 10000, 0, 0.01, 16*103, 16*83.25);
        cloud.setParentId(-1);
        fogDevices.add(cloud);
        FogDevice proxy = createFogDevice("proxy-server", 2800, 4000, 10000, 10000, 1, 0.0, 107.339, 83.4333);
        proxy.setParentId(cloud.getId());
        proxy.setUplinkLatency(100); // latency of connection between proxy server and cloud is 100 ms
        fogDevices.add(proxy);
        for(int i=0;i<numOfAreas;i++){
                cellularNetwork(i+"", userId, appId, proxy.getId());
        }
```

10. Create Sensors and latency o connection between the smart bin and router

```
// LEVEL 3 Smart bin + Sensors
private static FogDevice addSmartBin(String id, int userId, String appId, int parentId){
        FogDevice smartBin = createFogDevice("MCU-"+id, 500, 1000, 10000, 10000, 3, 0, 87.53, 82.44);
        smartBin.setParentId(parentId);
        System.out.println("ultrasonicSensor");
        Sensor ultrasonicSensor = new Sensor("HC-SR04"+id, "ULTRASONICSENSOR", userId, appId, new DeterministicDistribution(5));
        sensors.add(ultrasonicSensor);
        ultrasonicSensor.setGatewayDeviceId(smartBin.getId());
        ultrasonicSensor.setLatency(1.0);  // latency of connection between bin (sensor) and the parent Smart Camera is 1 ms

        System.out.println("motionSensor");
        Sensor motionSensor = new Sensor("PIR-motion-sensor"+id, "MOTIONSENSOR", userId, appId, new DeterministicDistribution(5))
        sensors.add(motionSensor);
        motionSensor.setGatewayDeviceId(smartBin.getId());
        motionSensor.setLatency(1.0);  // latency of connection between PTZ Control and the parent Smart Camera is 1 ms

        System.out.println("temperature");
        Sensor temperature = new Sensor("DHT22"+id, "TEMPERATURE", userId, appId, new DeterministicDistribution(5)); // inter-tra
        sensors.add(temperature);
        temperature.setGatewayDeviceId(smartBin.getId());
        temperature.setLatency(1.0);  // latency of connection between bin (sensor) and the parent Smart Camera is 1 ms

        return smartBin;
```

11. Create a fog device
    @param nodeName name of the device to be used in simulation
    @param mips MIPS
    @param ram RAM
    @param upBw uplink bandwidth
    @param downBw downlink bandwidth
    @param level hierarchy level of the device
    @param ratePerMips cost rate per MIPS used
    @param busyPower
    @param idlePower
12. create host device
    String arch = "x86"; // system architecture
    String os = "Linux"; // operating system
    String vmm = "Xen";
    double time_zone = 10.0; // time zone this resource located
    double cost = 3.0; // the cost of using processing in this resource
    double costPerMem = 0.05; // the cost of using memory in this resource
    double costPerStorage = 0.001; // the cost of using storage in this
13. Function to create the Intelligent Surveillance application in the DDF model.
    @param appId unique identifier of the application
    @param userId identifier of the user of the application
    @return

5. Evaluation

Compared to the cloud-based version, the fog-based solution has low latency and little network utilization. The outcomes of network consumption and latency in the fog environment, as well as the outcomes of cloud-based deployment. In environments where great performance is required in real-time, latency must be decreased. Fog computing has the advantage of avoiding frequent requests to the cloud and doing calculations at the network's edge to provide a speedy response back to the client device and reduce latency.

```
=========================================
TEMPERATURE ---> 0.5286326618088424
MOTIONSENSOR ---> 0.5286326618088424
ULTRASONICSENSOR ---> 0.5286326618088424
=========================================
Cloud : Energy Consumed = 2832533.333333333
Proxy-server : Energy Consumed = 166866.59999999995
EdgeServer-0 : Energy Consumed = 166866.59999999995
MCU-0-0 : Energy Consumed = 166357.89055422207
MCU-0-1 : Energy Consumed = 166357.89055422207
MCU-0-2 : Energy Consumed = 166357.89055422207
MCU-0-3 : Energy Consumed = 166357.89055422207
Cost of execution in cloud = 40000.0
Total network usage = 0.0
```

13. figure simulation result with cloud usage

```
=========================================
TUPLE CPU EXECUTION DELAY
=========================================
=========================================
Cloud : Energy Consumed = 3087440.0
Proxy-server : Energy Consumed = 166866.5999999999
EdgeServer-0 : Energy Consumed = 166866.5999999999
MCU-0-0 : Energy Consumed = 164880.0
MCU-0-1 : Energy Consumed = 164880.0
MCU-0-2 : Energy Consumed = 164880.0
MCU-0-3 : Energy Consumed = 164880.0
Cost of execution in cloud = 100500.0
Total network usage = 78920.0
```

14. figure simulation result without cloud usage

6. Conclusion

Due to their 24-hour operation, smart waste management systems are more efficient than traditional technologies. Since less physical work is required, smart waste management systems will also have lower operating costs than traditional waste management systems. Low latency responses between the data source, the station, and the centralized control station can be achieved by distributing edge-based control stations across operation regions.

7. Links

GitHub: https://github.com/Balays33/Fog-and-Edge-Computing--Smart_Waste_Management_System.git

YouTube: https://youtu.be/UmtY1jmvGvo

Reference:

[1] https://mycity360.co.il/smart-waste-management-system/

[2] https://link.springer.com/article/10.1007/s13369-020-04637-w

[3] https://www.researchgate.net/figure/Main-iFogSim-java-classes-and-their-relationships_fig1_328366480 S.H. Shah Newaz

[4] https://www.mr-fill.com/brochures/brochures-display/

[5] https://www.electronicscomp.com/nodemcu-esp8266-wifi-development-board

[6] https://www.alibaba.com/product-detail/HC-SR04-to-world-Ultrasonic-Wave_62133514008.html

[7] https://iotstore.al/en/home/7-dht22-temperature-humidity-sensor.html

[8]https://cdn-learn.adafruit.com/assets/assets/000/010/133/original/BISS0001.pdf