Biomedical Research 2016; Special Issue: S267-S270 *ISSN 0970-938X*
www.biomedres.info

Blending finite automata with threading for parallel and multi-device data fusion for a new context-aware cyber physical system application framework.

Veeramuthu Venkatesh[1*], Vaithiyanathan V[1], Pethuru Raj[2]

[1]School of Computing, SASTRA University, India

[2]Cloud Infrastructure Architect, IBM, Bangalore, India

Abstract

Proffering context awareness facilities to end-users is a long-term objective of Ambient Intelligence (AmI). A context-aware system has to know the context information and understand what exactly is happening in a particular environment in order to build and provide distinct capabilities to the users and the occupants of the environment. However there are multiple challenges in precisely capturing the context or activity details from sensors and actuators deployed in the environment. There are possibilities for aggregating wrong data due to various reasons at different levels (sensor, network, etc.). Several technologies and mathematical concepts are being recommended to substantially enhance the quality of the sensor/actuator data so that the decisions being derived out of the data collected are right. The concept of fuzzy finite automata is used by most of the researchers to solve these issues (trustworthiness and timeliness of the sensor data). But it has turned out to be a complex affair. In this paper, we have empowered Deterministic Finite Automata (DFA) with threading to enable correct decision-making out of data from multiple sensors/devices in any environment. In short, considering the multiplicity and heterogeneity of devices in our personal as well as professional environments, DFA with the parallelization capability is going to be the novelty and game-changer for precision-centric context-aware computing. The suggested method is verified and validated through a freely obtainable data set and the results obtained prove the simplicity and the correctness of our solution approach.

Keywords: Internet of things (IoT), Cyber physical system (CPS), Smarter environments, Cloud, Big data analytics, Finite automata, Context-aware computing.

Accepted on August 31, 2016

Introduction

Undoubtedly it is going to be the cloud-enabled [1], service-oriented [2] and knowledge era. The objective of this work is to give every service-providing, brokering, and consuming component whether it is a software application or a hardware component in any environment has to be knowledge-enabled in order to be correct and applicable for people in that environment. There is a bevy of pioneering edge, connectivity and fusion technologies emerging and evolving fast to smoothly facilitate the requirements of capturing and bringing forth context knowledge in time. Any kind of futuristic services (business, social, mobile, embedded, wearable, cloud, etc.) have to take the aspect of context-awareness into consideration in order to be distinctive in their capabilities and competencies. The services need to be embedded and enabled with the capability of gaining and using context information in time to be adaptive in their operations, offerings and outputs automata [3,4] is an extensible model for appropriately representing data and control flows towards automating the realization of context-aware applications for smarter environments [5,6] such as smart hospitals, homes, hotels, etc.

In this paper, we explain a new framework for quickly and easily composing context-aware [7] applications. The gist of the framework is to capture and fine-tune data from different devices, store them in a database system, then choosing the relevant data, sending them in.

The System Architecture

The architecture diagram given in Figure 1 provides a detailed view of how multi-threaded DFA shown in Figure 2 actually works in a smarter environment. Before getting into the architecture details, let us see the definition of DFA.

Deterministic finite automata definition

DFA- Deterministic Finite Automata

DFA M is defined as M which has 5 tuple (Q, Σ, δ, q0, F)

Q: a finite set of states

Σ: a finite set of input symbols

δ: a transition function (δ: Q × ∑ → Q) is used to map a current state into next state upon an input symbol.

q0: a start state

Layer 1 comprises of the actual environments itself and also has sensors to keep track of the environment. Only after occurrence of any event it is preceded to next layer. The Layer 2 consists of two parts database and data fusion. The data fusion is used to convert sensor values into transition values and then it is stored in database The required values of a context is retrieved from the database and the input is send to a multi-threaded DFA (Layer 3). The result of DFA is either accepted or rejected. This result is then sent to the Layer 4 where results from different threads are collected in queue. The result is then passed on to the event checker to effect a decision and output is produced. In case of any disruption in the sensor (or) environment is detected in event checker flow then passes to proactive adapter in Layer 5 which checks for the reason of disruption. The data so calculated is passed to environment adjuster which produces signals so that actuators can correct the sensors.

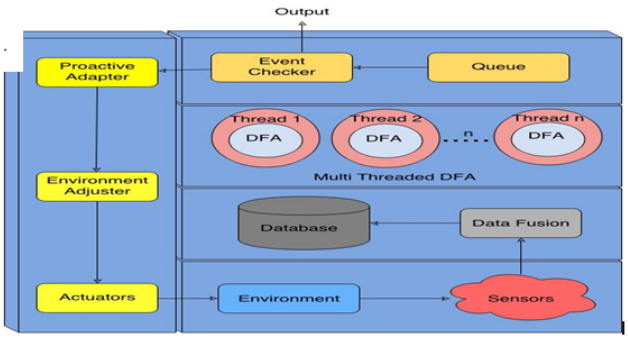

Figure 1. *Five layered multi-threaded DFA architecture.*

A. Sample use case

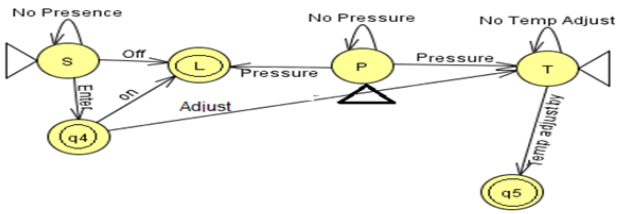

Figure 2. *Multi-threaded DFA.*

- S-Security camera
- L-Lights
- P-Pressure sensor
- T-Temperature sensor
- q4 and q5-temporary state

Implementation Overview

As an example to illustrate the concept, we take smart home as the scenario. The taken scenario is further split into context. A scenario has N input sensors. As an example in our case smart home is further divided into various contexts such as Hall, Bedroom, and Kitchen etc. To illustrate the concept we restrict our discussion to one particular context i.e. Hall context.

In this context we have 4 sensors namely light, pressure, temperature, surveillance sensors. The surveillance sensors should work with security camera and sends input to a commonly stored database. The sensors value stored depends on user permission or is compared with previously stored value in database. Similarly pressure sensors detects and sends the data if a person sits on couch, the light sensor should detect the current light ambience in the room and should correct itself by synchronizing with the values from pressure sensors. The temperature sensor should detect the current temperature value and accordingly activity recognition should take appropriate value by either switching on air conditioner or fan or room heater.

The Decision-Making Method

The decision making process is supported by a 'DFA'. The sensor values are converted into transition values after comparing with publicly available data set. The n sensors data is classified into m groups that perform similar action. The m groups are given as inputs to DFA having m threads to execute in parallel. Sensors which are appropriate are taken from a pool of sensors available in a given scenario using a modified context attribute matching.

The algorithms that we have used are mentioned below:

Algorithm 1: Multi-threaded DFA

Input: y_k input Sensor transition values

Output: output string;

1: For each input from ‘m’ groups of sensor data

2: Create a thread x_i

3. Pass y_m to each x_i

4: DFA result=DFA (y_m) from each x_i

5: Push values into queue (mid, DFA result)

6: End for

7: Context=pop_queue ()

8: Output=call event checker (context)

9: if (output>0)

10: Return output

11: else

12: Call proactive_adpater (context)

13: Return 'Error occurred'

14: End

Algorithm 2: DFA

Input: y_k the input string;

Output: result;

1: For each x_i for a given q_j define transition function into transition list.

2: Loop till y_k is empty

3: if (q_j, y_l) is defined in transition then

4: Move from q_j to q_m

5: if q_j is in the set of final states

6: Set result=1;

7. else

8: Set result=0;

9: Return result

10: End

In the Algorithm 1, the function call to event checker will take the present context into account and tells what all the threads that were not accepted are. The next call to the function proactive_adapter gives the reason for what those threads are not accepted i.e. it provides the value of sensors that made those threads unacceptable.

Database Module

The proposed framework has a dedicated database module in order to query Simple Object Access Protocol (SOAP) messages of various components and appliances that are installed in home-based environment. This component has two different groups of classes called contented components and information purveyor mechanisms. The data-provider component assists in statistics rescues and updates. Designers can use the assembly, command, and information reader objects to directly operate data. In this component, all communication includes information exchange, and addressed the common information exchange restriction by using Extended Mark-up Language (XML) as its payload information format.

Figure 3 show the result of implementing Algorithms 1 and 2 in java. In that Figure, the total sensors are the sensors that are installed in the environment. The sensors used are the one that are shortlisted according to attribute matching algorithm and finally to actuators are the final events that are triggered.

Performance Evaluation

With the help of multi-threading in DFA, one can achieve high throughput as it really takes less execution time when compared to a normal DFA. It is because there is no parallel execution in DFA, whereas in Multi-threaded Deterministic Finite Automata (MT-DFA) each thread is executing its own context. Figure 4 gives comparative performance levels of both DFA and MT-DFA.

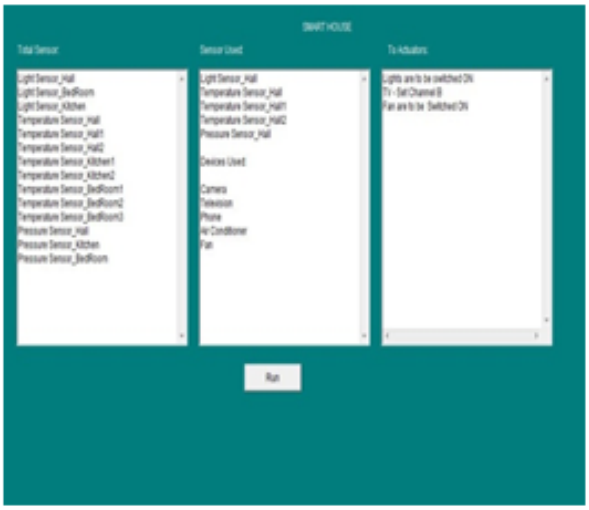

Figure 3. *Multi-threaded DFA based decision making process.*

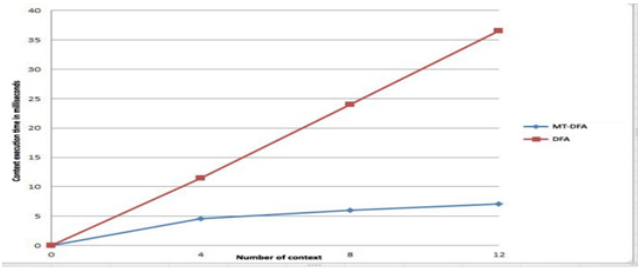

Figure 4. *Number of context vs. execution.*

Conclusion and Future Enhancements

Next-generation systems need to take the fast-evolving context information in order to be distinct in their operations, offerings and outputs. There are a bevy of approaches, algorithms and architectures by worldwide researchers in order to intrinsically and insightfully enable software applications, applications and infrastructures to be situation-aware. However, based on the real-world experiences, it is clear that capturing context details precisely and feeding them to our everyday systems in time is beset with challenges. However for the ensuing era of smarter and sophisticated systems, extracting and embedding context knowledge into our everyday devices, appliances, machines, instruments, equipment, wares, utensils, gateways, gadgets and gizmos emerges as the mandatory thing. In this paper, we have blended the proven multi-threading concept with the finite automata construct in order to parallelize the process of fusing distributed and different device data in order to speed up the task of extracting context information. We have simulated our proposal and shown that our idea works well in fulfilling the requirements.

References

1. Kim KI, Su YB, Dong CL, Chang SC, Hun JL, Kyu CL. Cloud-based gaming service platform supporting multiple devices. ETRI J 2013; 35: 960-968.

2. Wang H, Wang S. Ontological map of service oriented architecture for shared services management. Expert Sys Appl 2014; 41: 2362-2371.
3. Bar-Cohen Y. Biologically inspired intelligent robots using artificial muscles. International Conference on MEMS, NANO and Smart Systems, 2003.
4. Ramakrishnan M, Balasubramanian S. Parallel communicating extended finite Automata systems communicating by states. Int J Comp Eng Technol 2010; 1: 166-179.
5. Kaushik G, Sairam K, Nandakrish R, Venkatesh V. Competent smart car parking: An OSGi approach. J Artific Intel 2013; 6: 43-51.
6. Pethururaj CH, Devi AJ. Green technologies for the energy-optimized clouds. Asian J Res Soc Sci Hum 2016; 6: 165-178.
7. Xuansong L, Xianping T, Jian L. Improving the quality of context-aware applications-an activity oriented context approach quality software. IEEE Int Conf Qual Softw 2013; 173-182.

*Correspondence to

Venkatesh V

School of Computing

SASTRA University

India