Energy Efficient Task Scheduling and Data Allocation Strategy in Heterogeneous Environment with Real Time Constraints

Rakesh D. More
Assistant Professor
Late G. N. Sapkal College of Engineering,
Anjaneri, Nasik
University of Pune

ABSTRACT

In this paper, we concentrate on the problem of high Energy consumption in heterogeneous environment regarding data allocation scheme in distributed database and task scheduling policies in real-time system. We will focus on soft real time system application having if any deadline is missed then time penalty is occurred. Here we will combine both shared memory as well as in-memory database concepts in heterogeneous database environment. Basically in our paper we will concentrate on try to perform task within deadline having various type of databases responsible to store data. An important problem is how to assign processors to real-time application tasks, obvious here basically concentrate on real time scheduling strategy. As per studied that algorithm concentrate on EDF algorithm having preemptive nature. When higher priority transaction currently going to allocate processor and execute that transaction but another transaction is most needful to execute primary so using preemptive scheduling policy try to overcome that type of problem. Here global index is maintain into the local cache that is responsible to fast search particular server where that result is available and try to give response within time constraint and the total system energy consumption can be minimized. In this paper concentrate on replicas of databases due to this when any specific system crash then able to restore that database.

Keywords

Distributed System, Shared Memory DB, In-Memory DB, Global Index, RTOS, RT Scheduling Algorithm.

1. INTRODUCTION

Survey of literature is the mandatory phase to be carried out in every filed before starting the actual work. Advantage of conducting this phase is one can easily understand the existing system with its advantages and disadvantages. Then only one can proceed towards the proposed system.

Here what I am doing is first of all studying the heterogeneous environment and various issues come under the same point. Then study of various architecture styles for representing those systems which will be present in our network with their various aspects along with the suitable topological structure.

A data allocation and scheduling strategy will be very important in any kind of computing whether it may be a cloud or grid computing or even distributed computing.

Study of real time system with the following points will make my proposed system stronger and those points are as follows [1]:

1.1 Real time v/s General purpose system

General Purpose System:

Many non-real-time operating systems also provide similar machine oriented kernel level facilities. The important differentiation in general purpose operating systems and real-time operating systems is the need for " deterministic type " timing behavior in the RTOS. Generally, deterministic type real time operating system facilitate consume only known and expected amounts of time [2][3]. In theory, these service times could be expressed as mathematical formulas having category is Hard as well as soft Real time services [4]. These formulas must be strictly algebraic and not include any random timing constraint regarding fixed value. Various parts in timing facility produce random delays in application software and could then make the application randomly miss real-time deadlines – a scenario clearly unacceptable for a real-time embedded system.

General-computing system are non-real-time operating systems having non-deterministic state manner behaviour. It does not required fixed time interval for produce any transaction into application software and thus cause slow responsiveness of an application at unpredicted period. Developer team of a non-real-time operating system for the algebraic formula describing the timing behavior of one of its services (such as sending a message having different task execution), in future unable to get an algebraic formula regarding time constraints. There are various kind of Real time operating systems are avail in market whose necessary timing constraints primarily [5].

1.2 Real Time System:

Real-time systems are defined as those systems in which the correctness of the system depends not on the logical result of transaction, but on the basis timing constraints for any application at which the final computation are produced [6]. If the timing constraints of the application structure are not properly meet then obvious system crash is occurred. Hence, it is necessary situation that that the time is important factor of the system. promising timing activity necessary that the system be *predictable*. Predictability means that when a task is activated it should be possible to determine its completion time with certainty. It is also desirable that the system attain a high degree of utilization while satisfying the timing constraints of the system.

A real-time application is normally composed of multiple tasks with different levels of transaction activity found. Deadline missing is not applicable in a real-time system, soft real-time tasks could miss some deadlines and the system could still work correctly. However, missing some deadlines for soft real-time tasks will lead to paying penalties. On the other hand, *hard* realtime compulsory follow timing constraints, if not, unwanted or unexpected computational results will be produced in the system. There exists another group of real-time tasks, namely *firm real-time tasks*, which are such that the sooner they finish their computations before their deadlines, the more rewards they gain[7].

1.3 Periodic task, Aperiodic task and Sporadic task

Periodic tasks are real-time tasks which are activated (released) regularly at fixed rates (periods). Normally, periodic tasks have a constraint which indicates that instances of them must execute once per period p. Aperiodic tasks are real-time tasks which are activated irregularly at some unknown and possibly unbounded rate of the event. Actually timing constraint task follow the deadline as per specifying in the system

Sporadic tasks are real-time tasks which are activated irregularly with some known bounded rate of the event. The bounded rate is characterize by a minimum inter-arrival period for the transaction completion..

An aperiodic task has a deadline by which it must start or finish, or it may have a constraint on both start and finish times are necessary. a periodic type task, a period means once per period P or exactly P units separately. For example, a radar that tracks flights produces data at a fixed rate [8].

.As there exist three different kinds of task. I will focus on Soft real time system concentrate on to solve the sporadic task.

1.4 Scheduling algorithms which comes under the real time system like EDF, RM: RM Scheduling:

Each process is assigned a (unique) priority based on its period; the shorter the period, the higher the priority I.e, for two processes i and j, This assignment is optimal in the sense that if any process set can be scheduled (using pre-emptive priority-based scheduling) with a fixed-priority then the process set can also be scheduled easily with a rate monotonic assignment scheme

Note, priority 1 is the lowest (least) priority.

1.5 EDF Scheduling:

The runnable processes are executed in the order determined by the absolute deadlines of the processes The next process to run being the one with the shortest (nearest) deadline Although it is usual to know the relative deadlines of each process (e.g. 25ms after release), the absolute deadlines are computed at run time and hence the scheme is described as dynamic. It can be preemptive or non-preemptive scheduling technique .

1.6 Preemptive/Non-preemptive tasks:

In some real-time operating system task can be preempted if another task of higher priority becomes ready for execution purpose. Other side the task execution of a non-preemptive should be completed without interruption, once it is started Preparation of global index and keeping that index either at one system or at all systems Searching the require data or system and executing that query / transaction in real time fashion As I am saying real time constraints there is need to store the global index or data in main memory and executing the query on or before the deadline.

As there exist number of scheduling algorithms RM, EDF, Preemptive, Non-preemptive & priority driven etc. But here considering execution of query in our distributed heterogeneous environment assumption is that query should get execute within the specified deadline. Therefore we can go for EDF Scheduling algorithm which is totally based on deadline assign to that specific query[9].

1.7 Multiprocessor/Single processor systems

The number of the available processors is one of the main factors in deciding how to schedule a real-time task. In multiprocessor systems, the scheduling algorithms should prevent simultaneous access to shared resources and devices. Additionally, the best strategy to reduce the communication cost should be provided.

1.8 Why Real Time constraint?

The purpose of a real-time system is to have a physical effect within a chosen time-frame. Naturally, a real-time system consists of a controlling system and a controlled system[10]. The controlling system cooperates with its environment based on information available about the environment.

On a real-time system which controls a device or process, sensors will provide readings or observations at periodic intervals and the computer must respond by sending signals to actuators. In all cases, there will be a time bound within which the response should be delivered which is obvious thing in real time system. The ability of the computer to meet these demands depends on its capacity to perform the necessary computations in the given time.

The same thing which I want to achieve is that if a global transaction is initiated in the heterogeneous environment that should be completed on or before the specified deadline.

The meaning of applying real time constraints is that with the help of real time scheduling algorithms a developing the proposed system. If a number of events occur, the computer will need to schedule the task so that each response is provided within the required time bounds[11][12]. In this case the system lacks sufficient resources; a system with unlimited resources and capable of processing at infinite speed could satisfy any such timing related queries.

Everything should be managed in the real time fashion. Searching is need to conducted in the faster manner i.e. when any transaction is initiated at any server that server first of all decompose and will check whether required data is present locally or can it be a global transaction. If it is global transaction then identification of desired servers where the required data is present and assigning the query / sub query to those respective server and when those servers will computes the transaction then finally the merged results will be given back to end users.

1.9 Real time scheduler:
The another thing which needs to be consider is the introduction of real time scheduler because it is one of the most important module which will handle all the tasks / queries or processes in our distributed system. Therefore we will introduce real time scheduling algorithm in our system.

1.10 Deciding the replication strategy:
In distributed computing many things we have to consider like replication which is nothing but keeping the copies of data at multiple servers or sites and because of this approach the major advantage to get is that "availability of data" even one of the system crashes or downs. Therefore I can think about the complete or selective replication strategy.

Though this approach will have some drawbacks such as storage space requirements will be more.

2. PROPOSED SYSTEM
2.1 Hybrid Approach
Conceptual framework for any system is nothing but identification of various methods, algorithms and identification of modules required to implement the said proposed system.

In this case what I am proposing as shown in figure 1 is the decreasing energy/power consumption in distributed environment by maintaining real time constraints.

Day to day continuously dealing with data warehouse concept which is nothing but a collection of logically separated databases.

As we know number of end users is accessing this system to perform certain manipulations for analysis as well as strategic planning purpose. As this system contains historical as well as current data.

As we are saying this system may be logically separated among the 2/3 servers. We can prepare a global Index structure for this large volume of data set. Will determine whether this transaction/query can be executed locally or globally

Whenever any transaction will be initiated at any system then system first up all will check the index and then will determine whether this transaction is locally or globally.

Time required to execute this transaction whether it may be local or global can be reduce by applying some real time constraints in which idea is to reduce time required to execute the transaction which in turns will reduce the power consumptions and hence energy efficiency can be increase in this distributed environment. Therefore i wish to propose the system which will satisfies real time constraints even if some sort of time penalty can be introduce.

As our emphasis is on soft real time system in this case a bit more time complexity can be acceptable so that I will try to implement this proposed system in RT fashion.

Therefore main focus will be on scheduling all the processes/transaction/queries which will occur in distributed environment in RT fashion. For the same purpose RT scheduler will play an important role an along with this many other modules will contribute towards the implementation of RT application. In this application our achievement will be to reduce the latency as compare to general purpose system.

Our intension is to reduce time required to execute transaction by maintaining real time constraints. We can use Main memory to store large volume of database which is known as In-Memory database.

If store required data on secondary storage then definitely time required to fetch data from secondary storage and execute that particular transaction will be more. Therefore idea is to introduce in-memory database which will reduce noticeable time after that I can prepare a common global cache where hold recently used data for faster execution of query. Addition to this transaction manager, Transaction co-ordinator, Recovery manager, and Buffer manager will be the additional modules.

Heterogeneous environment there are maximum chances to have different DBMS and different softwares installed on individual system therefore data migration will be a challenging task among the two or more different DBMSs. We can implement RT Converter at individual site which will be useful at the time of data integration and normalization into the different databases. RT converter will be responsible to convert Data from one DBMS to the other DBMS by maintaining real time constraint.

Addition to this for reducing the time install RTOS on every system.

As above mentioned with respect to replication point, we can go for having replication strategy in our network which will be very useful for increasing the parallelism. So that can increase transaction throughput in our system in RT fashion. This type of strategy can be very useful in case of if certain node fails and requesting node instead of waiting can fetch required data from other systems. Therefore low maintenance and low movement of data will be observed in our network At the time of modification process of replicas which can handle by introducing Locking mechanism which is nothing but acquiring locks on certain data objects so that another process will not able to access this data object until reflection will be there on all the replicas. Challenging task in this case is updating of replicas after modifying the primary copy or one of the replica. Therefore tentatively as this is my survey of literature having following two protocols:

1. Majority protocol
2. Read one, write all (Available)

As the name suggest majority protocol after modifying the primary or one of the replica system will broadcast one message to the systems where replica of that data object will resides. System will perform updation if more than one half systems will reply with "Ready" message. This is known as majority protocol. In this case within one cycle majority replicas will be updated and later on updation will be carried out on those systems which were not participated in the updation process.

Contrast to this there is another protocol which says Read any one copy and modify it after that will update other available replicas manually. Therefore focus on majority protocol in our network.

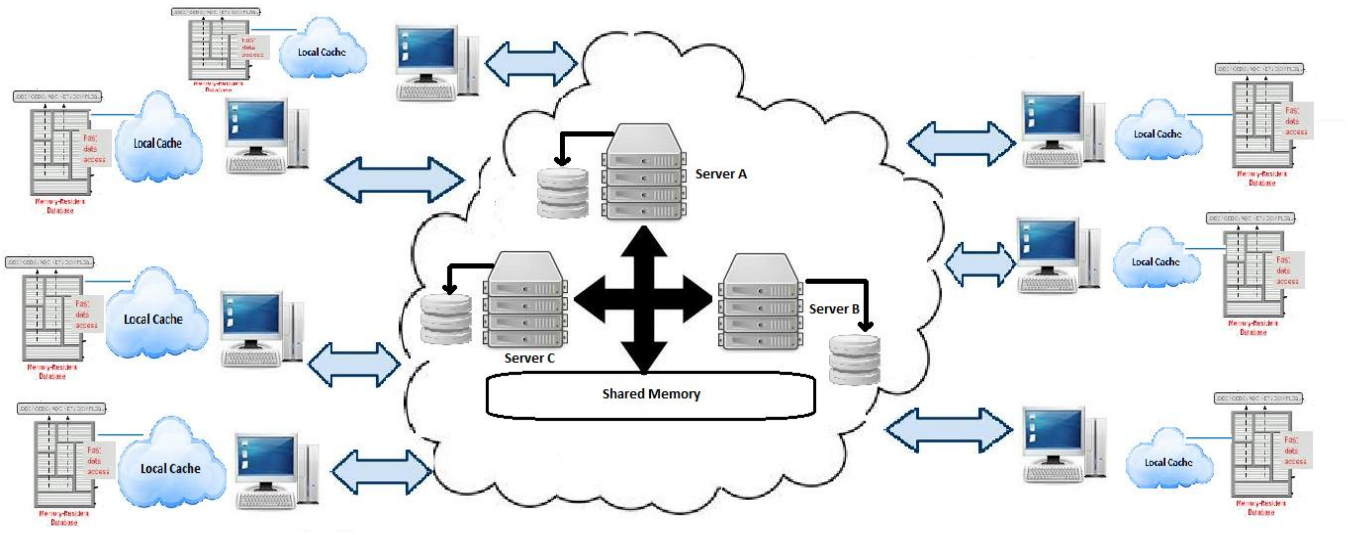

Fig 1: Architecture of Hybrid approach to resolved soft real time system application.

As we are thinking about the real time system and assumption is such that task or transaction should get executed on or before the deadline. This I am trying to achieve by proposing the above mentioned architecture style which is distributed. As it is distributed but having heterogeneous in nature systems may comprises of various hardware as well as DBMS installed on them. Therefore in such type of architecture task should get executed on or before specified deadline which is difficult to achieve.

Therefore we can go for the replication strategy in our heterogeneous environment

at servers side so task can retrieve the required data within the time. Same thing we can describe more deeply such as if a transaction will initiated at certain client that system first of all will check in its "global index" which contains information about the data stored in every system then if the required data is locally present then system first of all check its own "main memory" as we can introduce the "In-memory database" which is nothing but string the data in main memory if "yes" the within very less time task will get executed else system will move towards the "Local Cache" if data is present in the same area or not and if not then will retrieve data from the secondary system. If such case doesn't exists i.e. data is present at some other server or client will transfer the query to that server or client. Here what we have to think is the "busy" mode of that system i.e. if it is busy with other task so in this case one more solution is to divert that task to one of the server as shown in the figure. That server will execute task and return the result and we will same in-memory database strategy for those servers also.

We can some database management system softwares like "raimma Data base management system" which is one of the solutions to real time applications.

Raima Database Manager can be divided up so that part of the database resides in main memory while the rest of the database leftovers on-disk this is called as hybrid database system model. We know that data may be constantly changed when repeatedly apply modification and repeatedly accessed is placed in main memory while more consistent

Vast amount of data is stored on secondary storage storage device.

It is ideal for programming interoperating systems of networked and distributed applications and data such as those found in large amount for different application systems. Various configuration of systems provide developers a wide variety of powerful programming options and functionality.

The RDM architecture allows for configurations that support parallel operations to increase data throughput efficiency Currently in market focus on multicore operating system for faster execution of task. RDM database system is responsible to increase throughput.

As we are saying this is heterogeneous environment and number of client as well as server will try to retrieve data from each other the important point here is to "**Upadation"** which can be solved by introducing such type of software which Implement read-only transactions you can read a virtual snapshot of your embedded database while it is being concurrently updated. Again one more considerable point is **"Security"** which can be solved using such type of software which provides an implementation of the AES/Rijndael encryption cipher with care for 128 bit, 192 bit, and 256 bit.

2.2 Single Node Description:

As per above shown figure, we are seen components are including in single node for performing soft Real Time operation within deadline. Here each component having special functionality.

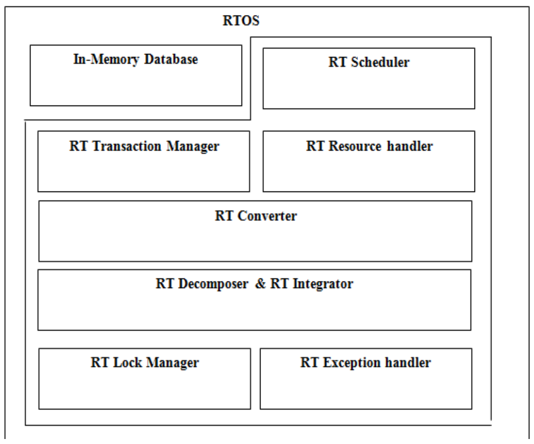

Fig 2: Client system configuration having RTOS.

3. FLOW OF PROPOSED SYSTEM:

1. Distributed environment will initialise all the systems
2. Those systems may be Heterogeneous client $P_c = \{P_0, P_1, \ldots, P_n\}$
3. Heterogeneous server $P_s = \{P_0, P_1, \ldots, P_n\}$
4. If any client initialises a request at any system, system will first of all identifies whether it is local or global
5. That Transaction request can be treated as set $R_{eq} = \{R_0, R_1, R_2, \ldots, R_n\}$
6. As we are introducing individual Local Memory to every system those sets can be considered as $M = \{M_0, M_1, \ldots, M_n\}$
7. Every system will prepared its own local as well as global index
8. After getting the transaction system will check its own index and then determines transaction can be executed locally or globally.
9. This global index will be quite similar inverted index structure in which system will easily identifies the correct location of data where data resides.
10. If feels cannot perform locally will divert that transaction to the respective server by identifying the global index
11. But in this case focus should be on checking whether that system is under loaded or not
12. Before conducting that transaction locally or globally need arises to acquire the LOCK on the data object required to perform the successful execution of that query.
13. LOCK MANAGER at individual site is responsible to grant the LOCK on certain data object to avoid the confusion.
14. One more thing is important that is use of RT Decomposer and Integrator
15. As our focus will be on maximizing the transaction throughput RT Decomposer will decompose the transaction into multiple sub parts and with this same approach final result can be merged.
16. Similarly transaction coordinator will be one of the module which will Handle all the transactions.
17. RT scheduler which is responsible to schedule all the transaction based on the DEADLINE of each task i.e. EDF strategy will be applicable.
18. Resource availability is major issue focus should be on identifying the available resources as well as those resources which are not available.
19. A resource pool can be helpful to hold all the available resources.
20. In this case replication strategy can be used which will improves the transaction throughput because data availability.
21. Space complexity will be more if we are using the replication strategy but it is not one of the constraints for our proposed system.
22. We can think upon the preemptive scheduling in which forcefully we can terminate the current transaction though which is not acceptable in real time system but as our focus is on soft real time system we can successfully execute the transaction which is having near deadline.
23. Cache memory will play important role which will avoid the maximum time required to execute the transaction
24. We can introduce the In-memory database which is different than the traditional database.
25. In this type of database we will hold the data in main memory with higher maximum scalability
26. So the time required to fetch the data from secondary disk and then loading it into the main memory for execution of a transaction will be more
27. Cache will hold the most recently used data which can be immediately satisfies the data requirement for current transaction.
28. Our emphasis should be on reducing the Time complexity for the proposed system.
29. So this In-memory database will reduce such time and it is helpful to increase energy utilisation.
30. Hence sequence will be in our distributed environment will such that system will first of check main memory then cache memory and still if required data not found
31. Then will divert the transaction to the respective server.
32. But again with respect to above point need arises to check whether that system is under loaded or over loaded.
33. If in our environment any system (server) fails it will be possible to recover that system by introducing the recovering technique we will reinitiate the transaction.
34. Our emphasis should be on having the state full nature for every system.
35. With respect to above mentioned point LOG MANAGER can be prepared which will be useful during system recovery.
36. In all these above mentioned points there is need to focus on the remote procedure call mechanism OR remote method invocation which is very important in distributed computing.
37. Inter process communication will be important area which needs a keen attention
38. As we are saying we are building the soft real time system in which deadline should not missed so we need to avoid the PIPELINING mechanism.
39. Pipelining mechanism will create some problems with system and the important delay also.

We need to achieve the ACID properties in our proposed system.

4. ADVANTAGES OF SYSTEM

1. As this architecture comprises In-Memory Database & Shared memory database architecture styles so it will be more advantageous to execute the task in real time fashion.
2. As we know in the real time system focus should be on deadline i.e. task should get executed on or before the specified deadline, my proposed system will try to achieve the same by giving responses without missing the deadline so this system will be suitable for those application where even a short deadline missed will not cost a lot i. e implementation of soft real time applications.
3. This architecture also solved Data loss or server crash issue by storing replicas at different server locations.
4. In this architecture concentrate on representation of database accessing from different databases having heterogeneous environment.
5. This architecture focus on energy efficient task scheduling by fast accessing using "global Index" concept which stored in local cache of every system for fast searching & assigning task.

5. CONCLUSION

In This paper, we will developed one real time energy efficient strategy having fast execution of transaction model within deadline, thus it can able to solved soft real time task and Firm real time task. In this proposed architecture we introduce combination of in-Memory Database towards every system as well as Shared Memory Database in Server side of Distributed System due to this we can achieve the properties of high availability of Database, Replicas are responsible to provide data when crashed due to some system or network or disaster issues. We also focus on searching strategy which can support for fast searching using Global Index into the local cache of every system. Here mainly focus on EDF algorithm for task scheduling in real time operating system.

6. ACKNOWLEDGMENTS

M.E. in Computer Science and Engineering from SSBT's COE & T, Bambhori, Jalgaon affiliated to North Maharashtra University, India. His research area includes Intrusion Detection System, Real Time System and Distributed Database System.

Presently he is working as Assistant Professor with the Department of Computer Engineering, Late G. N. Sapkal College of Engineering, Anjaneri, Nashik.

7. REFERENCES

[1] Yan Wang, Kenli Li, Hao Chen, Ligang He, and Keqin Li, "Energy Aware Data Allocation & Task Scheduling on Heterogeneous Multiprocessor Systems with Time Constraints", IEEE Transactions on Emerging Topics in Computing, VOL. X, NO. Y, MONTH 2014.

[2] http://belhob.wordpress.com/2007/10/page/4/

[3] http://en.wikibooks.org/wiki/Microprocessor_Design/Real-Time_Operating_System

[4] http://www.queryhome.com/15864/what-is-the-major-difference-between-normal-os-and-rtos#.U0-mJKIm2tk

[5] http://rakeshvarimalla.blogspot.in/2008/08/difference-between-rtos-and-os.html

[6] Prof. Kasim M. Al-Aubidy, "Classification of Real Time Systems",Computer Engineering Department, Philadelphia University, Summer semester,2011.

[7] Arezou Mohammadi," Scheduling Algorithms for Real-Time Systems", Queen's University, Kingston, Ontario, Canada, April 2009.

[8] Arezou Mohammadi & Selim G. Akl, "Scheduling Algorithm for Real Time Systems", Technical report No-2005-499, School of Computing, Queen's University, Kingston, Ontario,Canada,July 15,2005.

[9] Arezou Mohammadi & Selim G. Akl, "Number of Processors for Scheduling a Set of Real-Time Tasks: Upper and Lower Bounds", Technical Report Number 2007-535, School of Computing, Queen's University, Kingston, Ontario, Canada K7L 3N6, June 13, 2007.

[10] Jan Lindstrom Solid , "Real Time Database Systems", an IBM Company It¨alahdenkatu 22 B 00210 Helsinki, Finland March 25, 2008.

[11] http://www.ukessays.com/essays/computerscience/embedded-hardware-operating-systems-omputer-science essay.php

[12] Himanshu Poddar, Jagtar Singh, ,G.S. Sekhon," Dynamic Scheduling for Hard Real-Time Multiprocessor Systems ", Proceedings of National Conference on Challenges & Opportunities in Information Technology (COIT-2007) RIMT-IET, Mandi Gobindgarh. March 23,2007.

8. AUTHOR'S PROFILE

Rakesh D. More received B.E. Degree in Computer Engineering in 2010 from SKNCOE,(Sinhgad Institute), Pune affiliated to University of Pune, India.