IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 1, Ver. II (Jan – Feb. 2015), PP 40-45
www.iosrjournals.org

Recovery of Data in Cluster Computing By Using Fault Tolerant Mechanisms

Balne Sridevi
Assistant Professor, Bajaj Institute Of Technology & Sciences,
Department Of Computer Science and Engineering, Warangal, AP, INDIA

Abstract: *Many applications executing on cluster architectures consisting of several number of computers create several problems with reliability. The problems are system crashes, code failures and network problems etc. This paper using Fault tolerant mechanisms to recover the data in cluster computing. The two Rollback Recovery protocols[5] are Theft Induced check pointing protocol (TIC) and systematic event logging protocol TIC protocol is used to recover the information during crash, the information is send to client (receiver) by using SEL protocol. It is proven that the complexity of protocol is very small and the information lost by crashed process is recovered here.*
Section 1: describes the main objectives of TIC&SEL protocols
Section 2: describes the background for grid or cluster computing
Section 3: describes the definition of checkpoints & different types of checkpoints
Section 4: gives the details about work stealing or task stealing of different tasks
Section 5: describes an algorithm, which is mainly used for retrieving information from database server and creates backup of db
Section 6: gives the experimental results
Section 7: gives concluded the different techniques
Keywords: *Cluster computing, rollback, recovery, message logging. Event logging, check pointing, local technique, Forced technique, Global technique*.

I. Introduction

Large Applications executing on cluster architectures several computational nodes, inter connection networks, mass storage with reliability. This paper presents two fault tolerant techniques to overcome the problems with reliability. The two fault tolerant techniques are TIC [10] and SEL [10] protocols we implemented KAAPI (Kernel Adaptive Asynchronous Parallel Interface) by taking 'n' number of clients. This KAAPI [1] environment gives the general relationship between the processors and processes. Each process has its own stack (or) queue. The main objective of TIC protocol is to store the information in periodic time intervals and that information will send to receiver by using SEL protocol. Thus two protocols are introduced in sections 4.1

II. Background

Several Fault tolerant techniques are introduced to overcome the problem of reliability, node failure and the need for dynamic configuration over extensive runtime. The recovery means that it depends on redundancy we have several redundancy principles i.e. information, time and spatial. The information redundancy depends on stable storage method ,the other two redundancies are not suitable for cluster (or) Grid [20] computing.

III. Different Approaches

Based on stable storage we have two approaches

1. Logging based approaches.
2. Check pointing based approaches.

1. Logging based approaches:
In this approach, there is a difficulty with message logging [6] due to time and spatial redundancy.

2. Check pointing based approaches:
In this approach, the state of computations from first check point to last check point (or) last storage is saved.

3.1 Definition of Check point:

A check point is used to save the state of computation from starting point to ending point. It is based on saved states are based on time intervals.

Check pointing [15] can be categorized into two ways:
1. Co-coordinated check pointing.
2. Un Co-coordinated check pointing.

1. Co-coordinated check
Pointing:
It is used to construct the consistent global state (or) storage.

2. Un Co-coordinated check
Pointing:
It is used to save the state independently and achieves the consistent global storage in the recovery stage.

3.2 Creation of Checkpoints:
Checkpoints can be created in two ways:
1. At specific time intervals (or) check pointing periods.
2. Work stealing (or) task stealing.

If we consider the check pointing periods, these are called local checkpoints.
If we consider the checkpoints w.r. to work stealing (or) task stealing are called forced checkpoints.
The TIC protocol with respect to forced and local checkpoints shown in fig 1. The creation of checkpoints can be initiated by.1) Work stealing, 2) at specific check pointing periods. We will first describe the protocol [8] with respect to work stealing, since it is the cause of only communication b/w processes.

IV. Work Stealing

It is the only mechanism for distributing the workload constituting and primary mechanism for load distributes is based on scheduling algorithm called work stealing.

The principle is simple i.e., when a process becomes idle it tries to steal work from another processes called victim. The initiating process is called thief [2].
The checkpoints resulting from work stealing are called forced checkpoints. The TIC protocol w.r. to forced checkpoints is defined in figure 1.

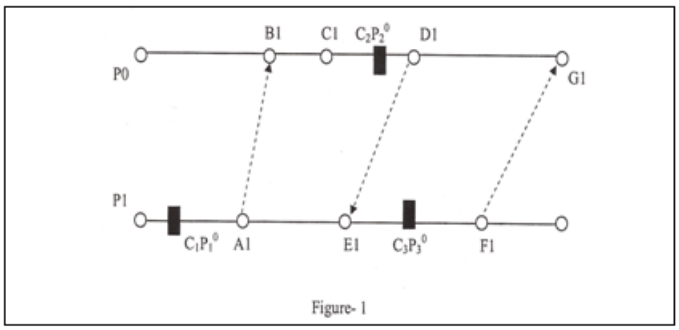


Figure- 1

4.1 Local Check pointing:
By applying this techniques, we can recover the crashed messages from starting point to last storage (or) stable storage[16].For this purpose we need two protocols.1) Theft Induced check pointing, 2) Systematic Event legging.

4.1.1 Theft Induced Check pointing & Systematic Event logging (TIC and SEL):
It is responsible for getting (or) retrieving information from crashed [14] processes (or) crashed systems.
The SEL is helpful for retrieving data (or) information using periodic time intervals.
1. In figure [1] consider two processes P0 & P1.
2. A process P1 is created on stable resource [4].
3. If it gets a process P0 that has a task to be stolen called theft task Tth, stealing a task form process P0. Before executing Tth, the process P1 checkpoints its state as $C_1P_1^0$.
4. Event A1 is the Execution of task Tth, which sends a theft request to P0.
5. Event B1 is the response of Tth by P0.
6. Between events B1 and C1, it recognizes a task Ts and identifies it as stolen by P1.
7. Between events B1 and C1 victim P0 is in difficult section w.r. to operations (theft operations).

8. Between events C1 and D1, it requests and forces a checkpoint to reflect the theft. At this timeP0 becomes a victim.
9. Event D1 gets sending Ts to P1.
10. Event E1 is the response of stolen task [7] Ts from P0 thief P1 initiatives entries for two tasks Ts1 and Ts2 in its linear structure steak shown in fig. 2.

Entries for two tasks Ts1 and Ts2.

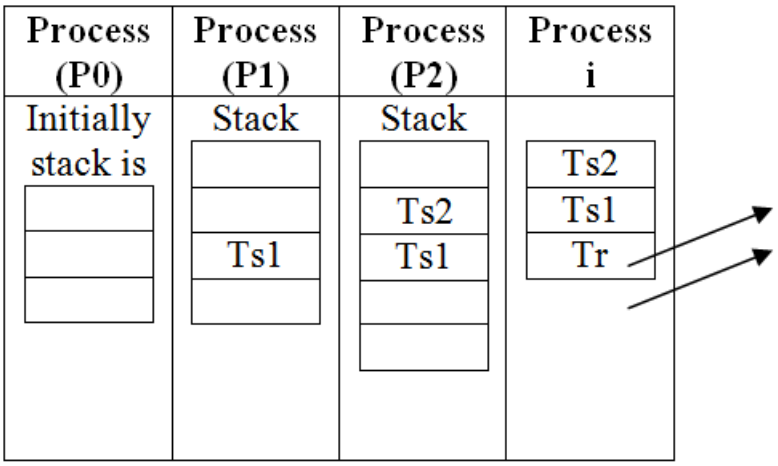


Figure- 2

Where the task Tr is changed with giving the experimental results of execution of Ts1 and Ts2 to P0 and becomes ready when Ts is completed.

11. When P1 is completed, the execution of Ts1 and Ts2 to P0 in event F1.
12. Event G1 is the response of P0.

V. Algorithm

Step 1: **What is the Recovery [5]?**
Retrieve the information (or) data from database server.
It is based on redundancies i.e., time, spatial (or) information.

Step 2: **Data base Recovery?**
Here the database is recovered by using the following approaches.
1) Log base Recovery [3].
2) First phase commit protocol [13].
3) Second phase commit protocol [13].
4) Third phase commit protocol. [13]

Step 3: If there is any problem with network or system crashes how can we recover (or) retrieve the data from the crashed nodes?
Step 4: To overcome this problem, we have two fault tolerant mechanisms.
1) Theft Induced check pointing protocol (TIC).
2) Systematic Event logging Protocol (SEL)

5.1. TIC:

The TIC is responsible for getting information from the crashed (or) process system. The main objective of TIC is to allow rollback of only crashed process. A process can be rolled back to its last checkpoint. Under TIC protocol, the faulty processes can be rolled back, TIC maintains global consistent [9] state of execution.

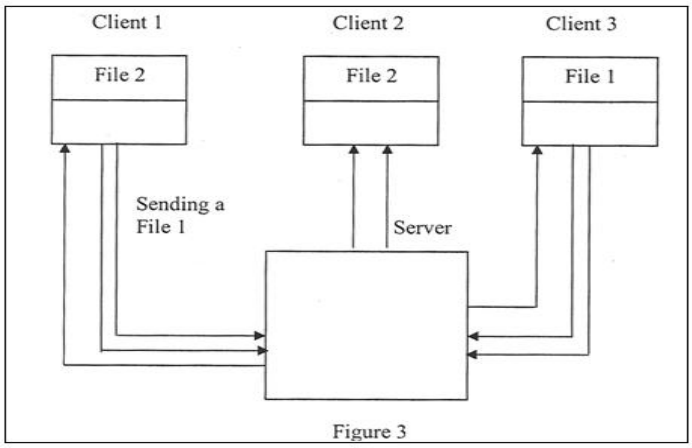


Figure 3

5.2. SEL:

This protocol is helpful for retrieving information using periodic time intervals and send to client with the help of SEL. SEL was derived from log-based method. The motivation for SEL is to reduce the amount of computation. In SEL, only the events relevant for the construction of data flow graph [8] are logged.

VI. Experimental Results

6.1. Applications and Platform description:

The basic performance and complexity of TIC and SEL protocols were practically determined for QAP [19] i.e., "Quadratic Assignment Problem", which was initialized and parallelized in an environment called KAAPI that is "Kernel for Adaptive, Asynchronous parallel Interface". KAAPI environment consists of processes shown in fig.4.This environment contains different processors; it can be implemented as a C++ library. A processor contains one (or) more processes.

KAAPI processor model

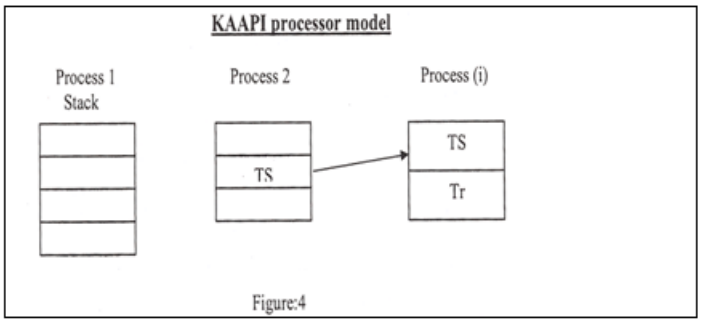


Figure:4

6.2. Life Cycle of a task in KAAPI execution model:

At task Initialization (or) task creation, the task enters into a state created, this time the task is pushed onto stack. When all input data of particular task is available, the task enters into a state called ready state. A ready task is on the top of stack that is it can be executed from stack and deleted. A task which is in ready state can be stolen, the particular task enters into stolen state on present process becomes as victim [11] when the task completed and particular state is to be deleted. If a task is to be stolen, the nearly created task (or) process utilizes the same model in the above figure.

6.3. Fault- Free executions:

6.3.1 Satin approach:

The complexity of protocols in fault free executions and executions containing faults

6.3.1.1: Execution times of Protocols in fault free executions:

It shows the execution times of applications for various protocols in the absence of faults. Two observations can be made here.

1) The application scales with number of processes for all protocols.
2) There is a very little difference between execution times of protocols for the same no. of processors. Here the complexity is the total amount of data associated with the protocol is sent to stable storage. This

complexity is affected by total size and number of messages. Here in this approach, TIC complexity is very small. It is shown in fig.5.

6.3.2. Execution times with faults:

First we measure the cost incurred by computation last due to faults and complexity of protocols. Here for each protocol.

$$\frac{TP_1^{\text{with fault}} - TP_1^1}{TP_1^1}$$

We observed here, if the number of faults increased, the execution time grows (or) increased linearly. There is a complexity (or) difficulty with Rollback but do not contain the complexity of check pointing (or) logging of events.

Execution times of Protocol

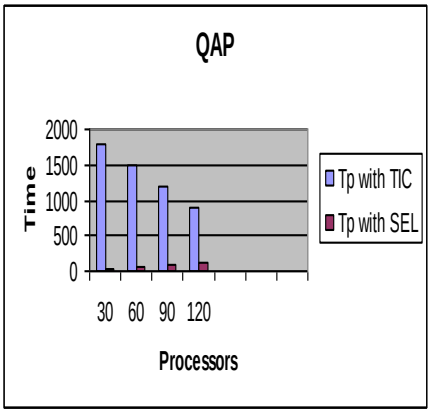


Figure- 5

6.4. Local, Global and Forced Checkpoints:

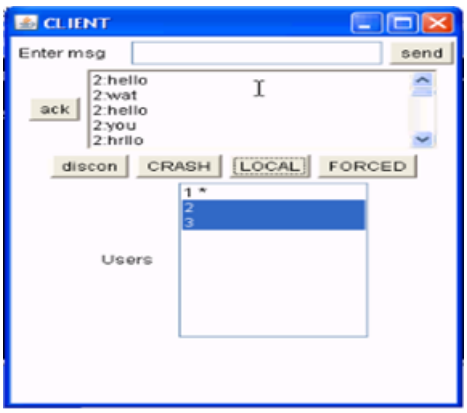
After a crash. It checks the failure data in database.

Ex: Serialization [3] (keep the conversational state) It maintains same state of execution.

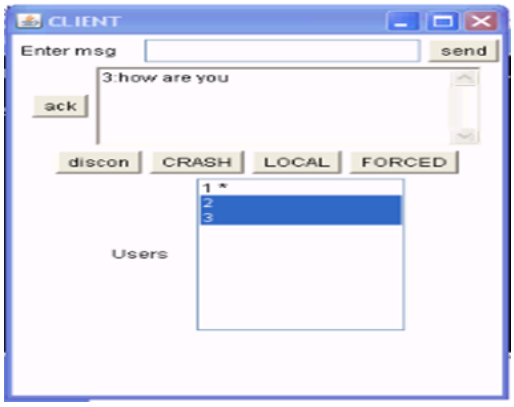
All the files are stored using Queue (or) stack. For example server socket will opening the new connectives. How can we invoke the thread? By using data send start ( ) it invokes run () functions. TIC protocol is responsible for getting information from the crashed process (or) system.

SEL protocol is helpful for retrieving information using periodic time intervals. By applying forced techniques, we will display the last message the experimental result is shown in fig.6.TIC and SEL are the fault tolerant mechanisms. The main objective of TIC is to allow logging events [18] for tasks are their additions and deletions.

After crash, by applying local technique, the messages from first point to last storage are displayed like



By applying forced technique we will get the result like



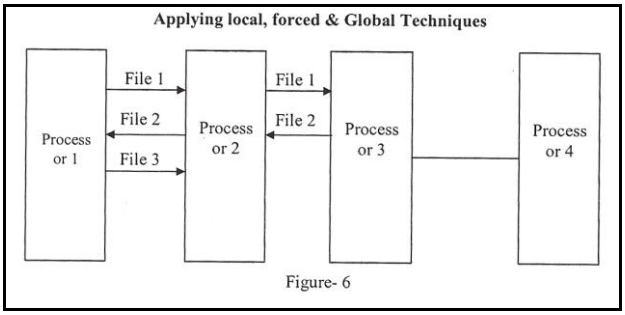


Figure- 6

If processor 2 is crashed here then the information of processor 2 is lost. To recover this information we need local and faced check points.

By applying local technique, we will get the information from file 1 to file 3.If you want to display file [1] information by applying global technique. To display the information about last file we can apply forced technique. Here I described a new technique called Global technique. By applying this we can get any message in the middle. By applying this technique, we can display desired message.

VII. Conclusion

If there is 'n' number of processors, suppose if a processor [1] wants to send information to processor [2], processor [3] etc., similarly if a processor [2] sends information to P1, P3, and P4……. Pn.If P2 is suddenly crashed here, the information is lost on P2. We have different protocols and techniques to recover the messages from starting point to ending point. By applying local techniques, we can display the message from first to last stable storage. By applying forced techniques [12], we can display the last message. By applying Global techniques, we can display our own desired messages.

References

[1]. Ivan Stojnenvic, "Hand book of wireless Networks and Mobile computing".
[2]. Elsevier, "Computer Architecture".
[3]. Arun K.Pujari, "Data Mining techniques".
[4]. D.K. Pradhan, "Fault tolerant computer system design".
[5]. R. Baldoni, "A communicative Individual check pointing protect that ensures Rollback- Dependency tractability".
[6]. L. Alvisi and K. Mazullo, "Message logging: pessimistic, optimistic, casual and optimal".
[7]. V. Strumpen, "Potable and fault- tolerant S/w systems".
[8]. D.curen," A survey of simulation in sensor networks"University Of Binghamaton project report for subject CS580.
[9]. L.Alvisi, "A survey of Rollback- Recovery Protocols in message- passing systems".
[10]. A Krings, S.Jafar, "Theft Induced check pointing for reconfigurable Data flow applications".
[11]. Narner P.E. and Knight J.C. security monitoring, Visualization and System survivability IEEE/SEI.
[12]. Blattner M.M. and Glinert E.P. multimodal Integration.
[13]. Kaye, J. Making scents: Aromatic output for HCI.
[14]. Axelsson S. Visualization for Intrusion detection.
[15]. G. Stellner, "Co-check check pointing and process Migration for MPI.
[16]. IEEE 802.15.4a Standard. Wireless MAC and PHY Specifications for Low – Rate Wireless Personal Area Networks.IEEE: Piscataway, NJ, USA, 2006.
[17]. R. Strom and S. Yemeni, "Optimistic Recovery in Distributed systems".
[18]. J.Slic, B. Robic and T. Ungerer, "Asynchrony is parallel computing from dataflow to multithreading" progress in computer search.
[19]. F.Bande, D.Coromel, C. Delb and L.Henrio, "A hybrid message logging CIC protocol for constrained checkpointability.
[20]. A large scale Nation- wide infrastructure for Grid Research Gird- 5000.