Fault Tolerant Techniques in Mobile Grid Computing: A Survey

Amit Savyanavar¹, Pranav Ghate²

¹, ²Department of Computer and engineering, MITCOE, Kothrud, Pune University, Pune, India

Abstract: Computational grid systems are the systems, where a task is distributed in various geographically distributed nodes. An important property of the grid system is Fault tolerance; it is a property of the system which enables a system to operate properly in case of failures. Checkpointing and replication are the two commonly used techniques for fault tolerance. Furthermore, these techniques mitigate the lost work, they commonly migrate a snapshot of the latest correct state of the failed node or share information between the resources to ensure consistency between them. However, these techniques are not full proof as they may introduce some kind of runtime overheads. Increasing fault tolerance means making your system more reliable and available to the users. It is also essential to satisfy the QoS requirement of the computational grids. Reliability and availability are the two greatest challenges in grid due to the unreliable nature of its resources.

Keywords: Fault tolerance, Computational Grids, Checkpointing, mobile grid systems, QoS.

1. Introduction

A system that coordinates resources which are not bound to a centralized control using some general purpose protocols and some significant Quality of Service (QoS) is called a grid. They consist of nodes that are geographically distributed to execute a specific task. The probability of failure increases with increase of grid components or load. This may give rise to process failure, node failures, machine crashes, link failure etc. Together giving rise to failure to complete the job in given deadline, denial of services violating service level agreements. Consequently, expected QoS degrades.

Grid systems are widely used in the areas like academics, industry etc. A new kind of grids called Mobile Grids is receiving growing heed and is becoming an important part in computational grids. It consists of mobile nodes which provide computing resources and facilitate user access to the grid. Similarly mobile grids can aid in utilization of any unutilized resources on the device. On the contrary they provide a chance for mobile nodes to save their own resources and use the resources of the grid.

While executing a task consider a failure occurs, that is a component fails, then previous work will be lost and the application has to be restarted and continue from scratch resulting in time and resource wastage. We need a technique which can prevent or detect these faults. The property of the grid which enables the system to continue operating smoothly while the failure has occurred is termed as Fault Tolerance. A complementary but a different approach to increase the dependability of the system is Fault prevention. This property includes techniques like inspecting the nodes or the resources and eliminates the cause by which a failure or a fault may occur.

When do we say a failure has occurred? It is the time where the system behaves differently than the specified behaviour. The seed of occurrence of a failure is called an error, and a fault is a root cause of a failure. Nevertheless a fault may not certainly generate an error, regardless the same fault may evolve in multiple failures. Likewise a single error may cause multiple failures.

The paper is organised as follows, Section 2 briefly describes some of the recent fault tolerance technique. Section 3 describes some of the findings and Section 4 presents the conclusion.

2. Fault Tolerance Methodologies

As we know fault tolerance is an important property of mobile grid systems, as faults which may occur through dynamic links, less reliable links etc may cause fatal node failure intern failure of the whole system. However some techniques or methods are used to overcome these faults. Some of them are listed below.

2.1 Reliability Arrangements

Paul J Darby III and nian-Feng Tzeng have proposed a reliability middleware driven checkpointing arrangement [1] where the nodes act as providers and consumers. An arrangement is stable when a consumer or provider does not prefer any other provider or consumer, respectively, to its current partner in the relationship. The relationship is defined as:

\[
\text{MH}_i \rightarrow \text{MH}_j
\]

Where, \(\text{MH}_i\) is a mobile host working as a provider and \(\text{MH}_j\) is a mobile host working as a dedicated Consumer. Above relationship is defined as, each provider in the arrangement has a dedicated consumer. If stable arrangement which is formed say, \([\text{I}]\), holds the relationship \(\text{R}_e \geq \text{R}_d\) where \(\text{R}_e\) is the reliability of some checkpointing arrangement and \(\text{R}_d\) is the reliability of the stable checkpointing arrangement. This arrangement is called \([\text{I}]\). The reliability is found using: \([12]\)

\[
\text{R}[\text{I}] = \prod_i \left[1 - (1 - \text{R}_a) (1 - \text{R}_b)e^\lambda\right]
\]

Where,

\([\text{I}]\) – is some unique checkpointing arrangement. Which is a directed graph consisting of a unique collection \(\text{MH}_\text{e} \rightarrow \text{MH}_\text{i}\) relationship on \(\text{MH}_\text{e}\), where
Relationships.

The pairing gain is calculated by: \[12\]

Two nodes having highest pairing gain amongst two nodes in the system are derived using a Fuzzy rule system. Stable storage and battery level together decide the fitness of fuzzy set. Memory capacity and crisp values in the range of (FRS), where the weights are availability of stable storage each relationship is checked. And a pair is formed of those two nodes having highest pairing gain amongst two relationships.

The pairing gain is calculated by: \[12\]

\[ \phi \]

Where,

\[ \Delta t \] - Connectivity of host C, i.e, \[ \phi_{c} \]

\[ \Delta t_{a} \] - Connectivity of host B, \[ \phi_{b} \]

\[ \Delta t_{b} \] - Connectivity of host B, \[ \phi_{b} \]

\[ \Delta t_{c} \] - Minimum checkpoint data transmission interval.

\[ \Delta t_{a} = \Delta t_{b} = \Delta t_{c} \]

\[ \Delta t_{a} \] - Minimum checkpoint data transmission time.

\[ \Delta t_{b} \] - Minimum recovery data transmission time.

In [1] Parmeet Kumar and Pritee Parwekar has proposed a fuzzy rule based fault tolerance in mobile grids where the checkpointing arrangement is derived without accessing any wired network or a static host. In this paper each provider gets a request message from a consumer which is in the proximity range of the provider. If the provider already has a dedicated consumer then gain of each relationship is checked. And a pair is formed of those two nodes having highest pairing gain amongst two relationships.

The proposed protocol constructs a limited size rule base of size \[ n^{2} \] where \( n \) is number of values used for ruling. In this case those are High, Low and Intermediate. So it will have a rule size of \( 3^{2} \).

2.2 Checkpointing Algorithms

In [21] an antecedent graph approach for checkpointing is proposed which Antecedent Graphs (AGs) are checkpointed onto a stable storage periodically so that the size of the graph piggybacked on the message does not reduce the efficiency. We see how the graphs are formed considering 3 mobile agents. In the beginning each agent is at the state \( S_{A0} \), \( S_{B0} \), \( S_{C0} \) respectively. The intervals are determined by the messages that are received. When agent B receives message \( m1 \) from C, the deterministic interval and AG of state interval \( S_{Ag} \) provides information about the previous condition.

Ex. Formation of AG of graph for agent A. message \( m2 \) is received by agent A from agent B. A combines the antecedence graph received from B to its own graph for the formation of the event \( S_{B1} \). B sends a message \( m0 \) to C. After this, the message \( m1 \) is received by B from C; with the difference of antecedence graph of C (if a message from C has been previously received). B combines the AG received from C to its own graph for the formation of the event \( S_{B1} \). C receives a message \( m0 \) from B for the formation of the event \( S_{C1} \). After this, the message \( m1 \) is sent to B from C. After this, C receives the message \( m3 \) from A, with the antecedence graph of A. C combines the antecedence graph received from A to its own graph for the formation of the event \( S_{C2} \).

They have also used a parallel checkpointing algorithm where each checkpoint is given a unique sequence number, which increases monotonically. When a node sends a request it attaches checkpoint request and numeric weight value in parallel the dependent agents (DA) make AGs of events which occur during the checkpointing stage. However all the agents specified in the AG receives the inquiry message from checkpointing agent (CA). If DA agrees then CA sends the weight indicating positive response to starting agent (SA). If the SA receives such responses from all DAs then it checkpoints the info in Base station (BS) and informs other respectively. The maximum length graph for these agents is then constructed and stored in stable storage by BA.

Jack Dongarra, Thomas Herault and Yves Robert in [20] revisited a Double Checkpointing and proposed a Triple checkpointing algorithm. In double checkpointing to avoid crash of the whole application by a single failure local checkpoints are replicated. This algorithm states that once checkpointing is done locally and again the same checkpoint.
is replicated remotely. In case of application failures the checkpointed data may be recovers from two points, therefore if the node fails the data can be recovers from local or a paired node increasing the reliability. The non blocking algorithm is given below:

- Periodic checkpoints are taken, with period \( P = \delta + \theta + \sigma \).

\( \delta \) – Period of checkpointing locally and no application work is performed.

\( \theta \) – Period of checkpointing remotely. Each node checkpoints remotely, that is it exchanges its checkpointed data with its paired node.

\( \sigma \) – Period of application executing at full speed.

The weight, which is the amount of checkpointing data, is given by \( W = (\theta - \phi) + \sigma = P - \delta - \phi \)

Where,

\( \phi \) - is some communication overhead.

In triple checkpointing algorithm the nodes triples rather than pairs. It is more reliable when a fixed amount of memory is available for checkpointing. When the failure occurs the local checkpoint is lost and the checkpoint has to be recovered from its buddy. Triple checkpointing algorithm works just the same as double checkpointing but here we add another secondary node for checkpointing one more time. This reduces the overhead occurred in double checkpointing. Here 1st node checkpoints on primary node 2 and secondary node 3. Similarly node 2 has 2 buddies’ node 1 and node 3 and node 3 has again 2 buddies’ node 1 and node 2.

3. Findings

Here we find that in check pointing arrangements the reliability middleware is used to maintain the reliability of the check pointing arrangement where as in other paper fuzzy logic is used to do the same. Reliability middleware makes a stable pairs where as other algorithm makes the arrangement depending on fuzzy rules both are way better then Random Checkpoint arrangement (RCA) algorithm, shown below.

Figure 1: Non blocking Checkpointing Algorithm

Figure 2: RCA vs. Reliability driven middleware algorithm

In antecedent graph approach a base agent is used which increases the dependability and if the base agent fails the whole system crashes. Where as in double checkpointing there is no base agent and nodes depend on the neighboring buddy. Here the checkpoints are saved on neighbor. However they generate decent communication and other overheads which may create some problems. Whereas triple checkpointing has one more node where the data is checkpointed and reduces these overheads.

4. Conclusion

From above survey we can conclude that ReD middleware and fuzzy algorithms are efficient and reliable then the RCA. Also, the checkpointing algorithms which does not include the base agent are more reliable then the algorithms including the base agent.

5. Future Scope

In future we may merge these techniques and try to make a hybrid algorithm which provides a reliable checkpointing arrangement, which is decentralized and with low communicational overhead.

References


