# CPN Modeling and Performance Analysis of Hierarchical Fault Management System

Swati Singhal*, Heman Pathak
*Department of Computer Scinece,*
*Kanya Gurukul Campus, Dehradun, Uttarakhand, India*
*aggarwalswati37@gmail.com, hemanp@rediffmail.com

***Received:*** *29.11.2018,* ***Accepted:*** *14.12.2018*

## Abstract

A mobile agent is a composition of computer software and data which is able to migrate from one computer to another autonomously and continue its execution on the destination computer. During its life cycle, a mobile agent may be lost or blocked due to failure of host machine, failed system components or link failure during migration. To tolerate the faults during life cycle of mobile agents, a Hierarchical Fault Management System for mobile multi user system had been proposed earlier.

This paper gives the description of Colored Petri Net modeling of Hierarchical Fault Management System. Assumptions for the modeling and parameters for the analysis have also been identified and reported. Model has been used to evaluate the performance of Hierarchical Fault Management System in presence of various faults and given in form of graphs for their min, max and average values. Data has also been collected for different fault rate of Host, Mobile Agent System, mobile agent and link but only one fault handle at a time. Simulation results for different fault cases show that Hierarchical Fault Management System gives expected results and Trip Time and Network Overhead varies as fault rate of mobile agent, mobile agent system, host, link increases.

**Keywords -** Mobile Agents, Multi Agent System, Fault Tolerance, Colored Petri Net Tool.

## Introduction

Any software or hardware component in a distributed system may be subject to failures. A particular failing component (e.g. agent, link or machine) may stop a MA from proceeding with its execution and block its execution. The person or application that has configured the agent, finds that its agent has not returned in a realistic time. In a synchronous system such as the internet, it is impracticable to notice properly whether the agent has failed or whether it is purely slow (de Assis Silva and Popescu-Zeletin, 1998). MA execution can only proceed if the failed machine and/or the agent improve. If the agent owner is another application, then it may be blocked as well. Blocking is thus undesirable in MA execution and mechanisms that prevent blocking are needed. The ability of a system or component to continue normal operation despite the presence of hardware or software faults is known as fault tolerance (Pathak *et al.,* 2011).

The problem of blocking is a fundamental issue in fault tolerance and can be addressed by masking the occurrence of failures in a system from the user. A mechanism is required to tolerate the faults mentioned so that the MA finishes the assigned tasks within the prescribed time limit. Some of the existing Mobile Agent System (MAS) provide some mechanisms to tolerate some of the faults in MA life cycle ( Kajorth *et al.,* 1997; Tardo and Luis, 1996; Zimmermann, 1995; Gray, 1995; Gray *et al.,* 1998). To tolerate the faults during life cycle of MAs, a Hierarchical Fault Management System (HFMS) for mobile Multi User System (MUS) has been proposed in the paper (Pathak and Singhal, 2017). In order to tolerate different types of faults and to minimize the overhead of fault management, the fault diagnostic scheme has been distributed into two layers. At the lowest layer, a thread based approach has been used to watch MA and other components of the MAS to detect faults and at the highest layer;

Agent Transfer Protocol (ATP) has been used to ensure the fault tolerant migration of MA in the global and local network.

**System Model**

In the proposed HFMS an effort has been prepared to offer solutions to bear all kinds of faults. In HFMS every network has its personal network fault management system to improve the failed host and MAS within the network, so these routines are not discussed in the paper. No link fails lastingly. Link failure is momentary and failed links are recovered automatically by the network management system.

**Components of Model**

Different kinds of faults are detected and tolerated at different layers of HFMS. Global Service Provider (GSP), Local Service Provider (LSP) and Personal Service Provider (PSP) works at the different layers. All three components have their different role to handle fault in HFMS.

**Role of GSP in HLMS**

Apart from its key role to receive and pass MA to designated host, it is also responsible to insure fault free migration of MAs in LAN and Global Network (GN) by implementing the ATP discussed in (Pathak and Singhal, 2017). It also instructs other components i.e. Checkpoint Manager (CM) and Host Manager (HM) installed at router. Every time when a MA enters in a network, GSP instruct CM to save the MA and its execution state to Local Shared Storage Space (LSSS). This checkpoint data is used to recover the MAs, lost by failure local links, MAS or its executing host. This software routine is responsible to detect host failure. Each host of the network periodically sends a heartbeat message “I am alive” (say) to Host Manager (HM). In case HM does not receive a message from a host, it suspects the host failure. To confirm host failure, it uses ping command. In case of host failure, it adds the failed host to the list of failed host and informs GSP about this fault.

**Role of LSP in HLMS**

LSP is the server at the middle layer, installed on each host of the network. It is responsible to communicate with GSP and informing about MAS failure. It also participates in implementing ATP and to update Agent Table (AT) and initiate recovery procedure in case of failure of MA and MAS.

**Role of PSP in HLMS**

PSP is the server at the lowest layer, installed on each host of the network. It monitors the MAS as well as all agents running at the host by maintaining threads for each. In case the MA fails or MAS crashes due to any reason, it informs itsLSP about the failure.

**CPN Modelling of HLMS**

CPNs are formally equivalent to traditional PNs; however, the richer notation makes it possible to model interactions in CPNs where it would be impractical to do so with PNs. Tokens move from one place to another through transitions. Transitions allow tokens to pass if all input arcs are enabled. Tokens entering from multiple places may be merged at transitions. Tokens leaving transitions may be duplicated to multiple destination places. CPNs may be organized in hierarchical fashion to allow reuse and top-down or bottom-up development (Jensen, 1983; Pathak, 2010; Jensen, 1981; Jensen, 1997; Vila *at al.*, 2007; Yakovlev, 2000; Adamski *et al.*, 2000; Ratzer *et al.*, 2003). The modeling of HFMS has been done using CPN tool. System model for HFMS is same as discussed in (Pathak and Singhal, 2017). Some of the components are also same as discussed in (Pathak and Singhal, 2017). The correctness, liveliness and fairness of the model has also been verified and reported. The different components of HFMS are Router, HostId, MA, UA, Packet, AT and LT. All these components have already been explained in the (Pathak and Singhal, 2017).

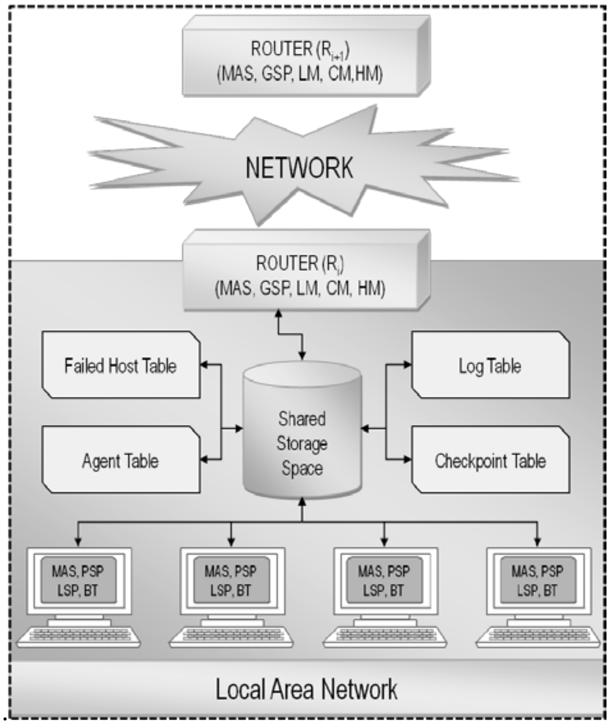

**Figure 1:** Architecture of Hierarchical Fault Management System

**Network Setup**

The Network setup used to experiment with CPN model of HLMS, CBHSA and HFMS is shown in Figure 2. The network is an interconnection of two networks, where each network contains nine hosts.

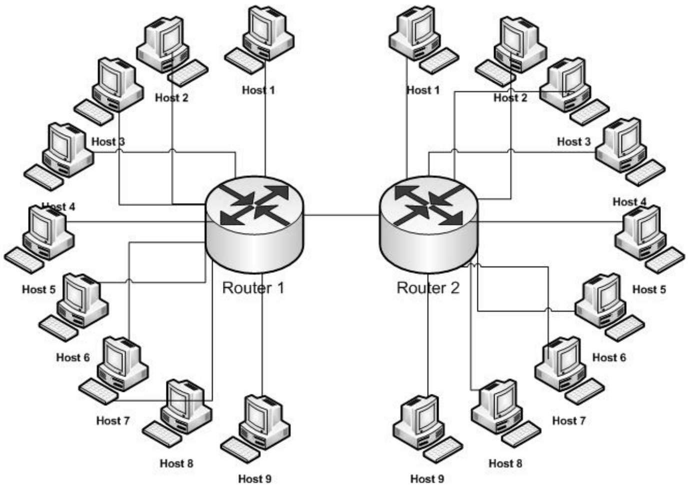

**Figure 2:** Network Setup for experimentation

### Performance Evaluation of HFMS

Before going to discuss performance analysis of HFMS, Experimental set up of the model has been made and parameters for performance analysis have been identified.

### Experimental Set Up For HFMS

Performance analysis of HFMS has been done on the basis of simulation results obtained from its CPN model. Timed CPN has been used to model HFMS. Values have been assigned to some of the time based parameters before simulation starts.

Table 1 shows the time assign to some of the transitions of CPN model of HFMS.

**Table 1:** Time Values Declaration For Different

| Time Variable Declaration | Value Declaration |
|---|---|
| Local Table access count | 10 time units |
| Global Table access count | 20 time units |
| MA Global Migration Count | 100 time units |
| MA Local Migration Count | 50 time units |
| Local acknowledgement Count | 10 time units |
| Global acknowledgement Count | **20 time units** |

### Parameters for Performance Analysis

Before using the model to collect results, it needs to be setup for analysis and parameters also need to be identified for which model is to be used. Parameters identified for analysis are defined and discussed here (Pathak and Aggarwal, 2016).

TRIP TIME (TT)
When security algorithm is applied, Trip time of MA is:

$$TT=CT+(MT+ET)*n+C$$

Here C is a constant that model other factors that may delay MA’s execution. ET is the execution time.
NETWORK OVERHEAD (ND)
ND is function of the following-

$$ND=fun(LMC*a, LTO*b, GMC*c, GTO*d)$$

Here LMC is the local MA migration count, GMC is the global MA migration count, LTO is the count of local table accessing, and GTO is the count of global table accessing. Here a, b, c and d are the weights based on size of packet/message and type of links.

**Table 2:** Data Used for Performance analysis of HFMS

| **MA Failure Rate** | **10** | **20** | **30** | **40** | **50** |
|---|---|---|---|---|---|
| **Trip Time** | | | | | |
| **Minimum** | 6.555 | 3.963 | 3.200 | 3.160 | 3.142 |
| **Average** | 15.42195 | 9.00009 | 6.25164 | 4.56974 | 3.73803 |
| **Maximum** | 29.115 | 16.204 | 11.597 | 10.916 | 6.736 |

| | | | | | |
|---|---|---|---|---|---|
| **Network Overhead** | | | | | |
| **Local** | 1296.875 | 974.625 | 799.75 | 800.875 | 772.375 |
| **Global** | 840 | 572.5 | 352.5 | 500 | 425 |
| **Total** | 2136.88 | 1547.13 | 1152.25 | 1300.88 | 1197.38 |

**Table 3:** Data Used for Performance analysis of HFMS

| **Host Failure Rate** | **10** | **20** | **30** | **40** | **50** |
|---|---|---|---|---|---|
| **Trip Time** | | | | | |
| **Minimum** | 98.391 | 133.915 | 199.034 | 397.583 | 947.998 |
| **Average** | 145.5632 | 181.6082 | 343.0405 | 848.2202 | 1747.822 |
| **Maximum** | 195.573 | 270.349 | 641.354 | 1370.594 | 2752.494 |
| **Network Overhead** | | | | | |
| **Local** | 10805 | 17394.5 | 20486 | 22703.5 | 24876 |
| **Global** | 4530 | 4560 | 4980 | 5220 | 5300 |
| **Total** | 15335 | 21954.5 | 25466 | 27923.5 | 30176 |

**Performance of HFMS in Presence of Faults**

This section of the paper observes and analyses the performance of HFMS in presence of faults. Proposed approach generated various faults in the CPN model of HFMS by changing the failure rate and then measures its performance in terms of TT and ND. In a real system, many faults may occur simultaneously, but for performance measurement one fault at a time has been handled. For each case, number of MA is 10 and itinerary size is fixed i.e. 10 and it includes both local and global hosts. Experiments are repeated 1000 times and minimum, maximum and average cases are reported.

Table 2 and Table 3 give the tables used for TT and ND for different cases of HFMS. Various such cases are listed below-

*Case 1: TT vs. MA Failure Rate*

This experiment shows the effect of MA failure rate on MA TT. Failure rate is the frequency with which a MA fails during execution. In HFMS a failed MA cannot be recovered and it leads premature termination of MA. Failed agents do not complete their itinerary so total TT of experiment will decrease as MA failure rate increases. Figure 2 shows the graph between TT and MA failure rate.

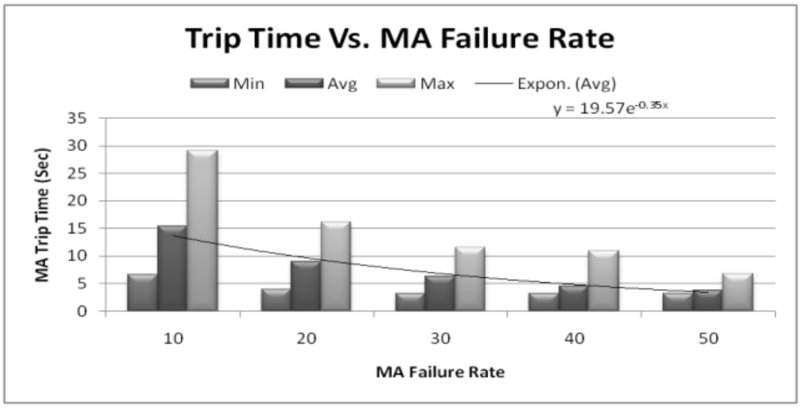

**Figure 2:** Trip Time vs. Increasing MA Failure Rate

It is clear from the graph that TT decreases exponentially with the increasing MA failure rate because of premature termination of MA.

*Case 2: ND vs. MA Failure Rate*

Figure 3 below shows the graph between ND and MA failure rate. It is clear from the graph that ND decreases with increased MA failure rate due to premature termination of MA.

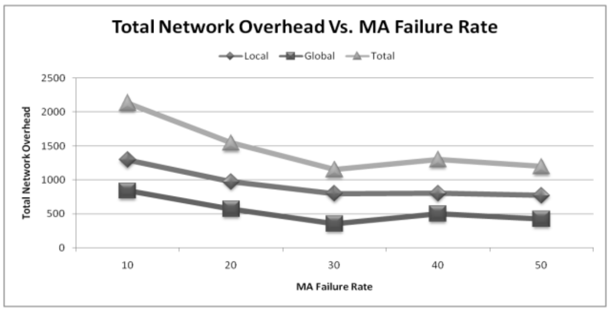

**Figure 3:** ND vs. Increasing MA Failure Rate

*Case 3: TT vs. Place Failure Rate*

This experiment shows the effect of place failure on MA TT. In HFMS, place failure delayed execution of MA.

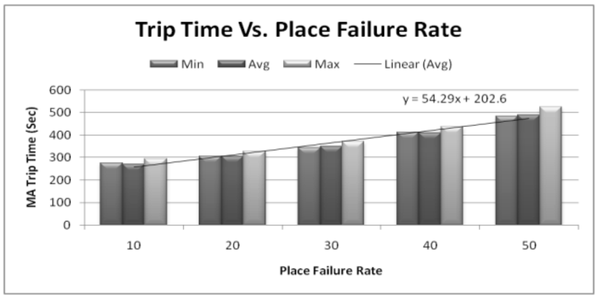

**Figure 4:** TT vs. Increasing Place Failure Rate

Figure 4 shows the graph between TT and place failure rate for min, max and average cases. It is clear from the graph that TT increases linearly as place failure rate increases.

*Case 4: ND vs. Place Failure Rate*

Since place failure does not introduce any local or global transmission, its recovery is independent of ND. It is clear from the graph shown in Figure 5 that ND is constant for both local and global migration for increasing place failure rate.

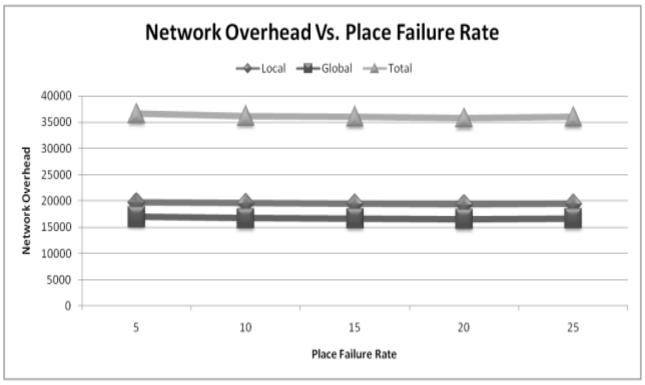

**Figure 5:** ND vs. Increasing Place Failure Rate

*Case 5: TT vs. MAS Failure Rate*

This experiment shows the effect of MAS failure rate on MA TT. During the MA execution, the MAS may crash and all the MAs executing on it are lost. All MAs executed of failed MAS are recovered and submitted to another host. It has been assumed that an alternate host is always available in the same network for the execution of MA, which was executed on the crashed MAS. Recovery of failed agents will consume time and increase TT. Figure 6 shows the graph between TT and MAS failure rate for min, max and average percentage. It is clear from the graph that TT increases exponentially as MAS failure rate increases.

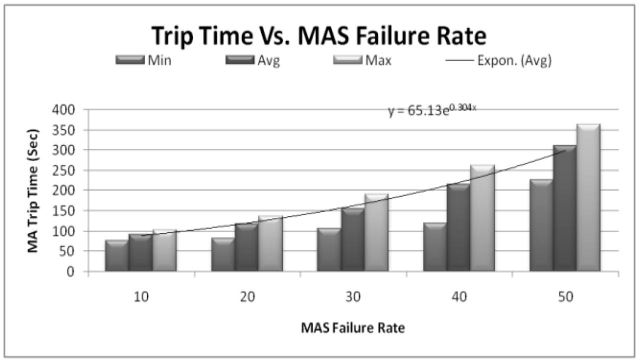

**Figure 6:** TT vs. Increasing MAS Failure Rate

*Case 6: ND vs. MAS Failure Rate*

In case of MAS failure, failed agents are recovered locally so it requires various local migrations but no additional global migrations are required. Figure 7 and Figure 8 below shows the graph between ND and MAS failure rate for local and global movements. It is clear from the graph that ND increases as MAS failure rate increase for local migration but MAS failure has no effect on global migration overhead.

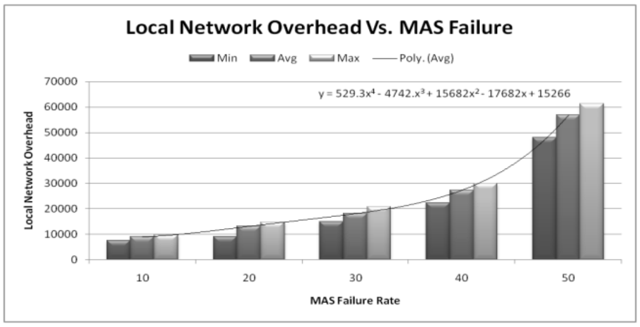

**Figure 7:** Local ND vs. Increasing MAS Failure Rate

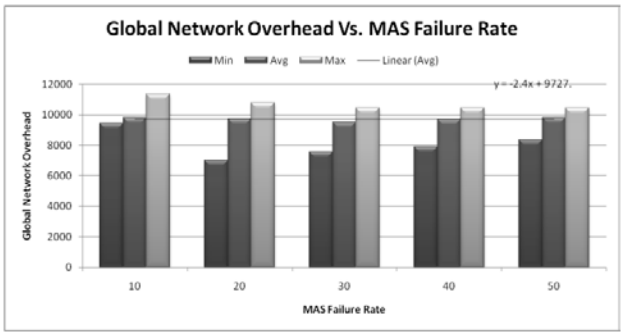

**Figure 8:** Global ND vs. Increasing MAS Failure Rate

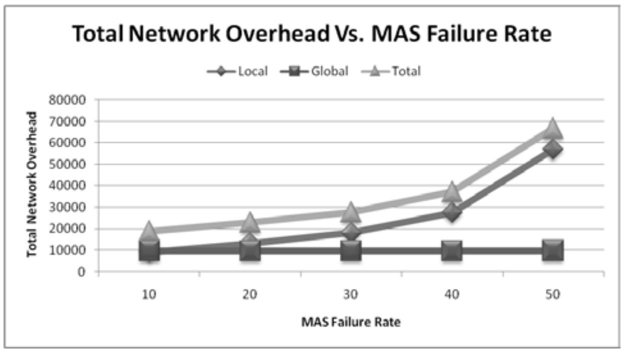

**Figure 9:** ND vs. Increasing MAS Failure Rate

*Case 7: Trip Time vs. Host Failure Rate*

During the execution of MA, the host machine may go down and all MAs hosted by it are lost. Host failure is tolerated in the same way as system failure so the performance is expected to be similar as in case of system failure. Unlike system failure, where a fault is detected by a thread, host failure is detected by HM on router.

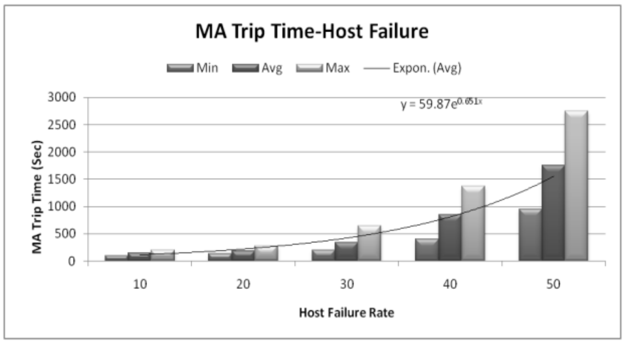

**Figure 10:** TT vs. Increasing Host Failure Rate

Figure 10 shows the graph between TT and Host failure rate for min, max and average cases. It is clear from the graph that TT increases as host failure rate increases with exponential time.

*Case 8: ND vs. Host Failure Rate*

Figure 11 and Figure 12 below shows the graph between ND vs. host failure rate for local and global movements.

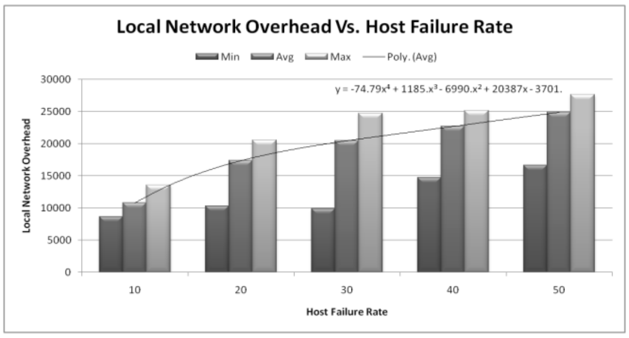

**Figure 11:** Local ND vs. Increasing Host Failure Rate

It is clear from the graph that ND increases as host failure rate increase for local migration. But global migration overhead is independent of host failure.

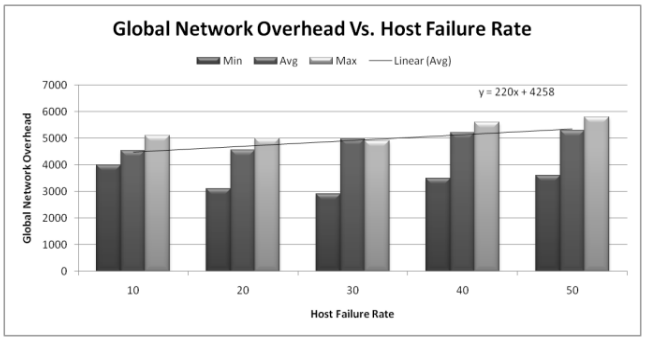

**Figure 12:** Global ND vs. Increasing Host Failure Rate

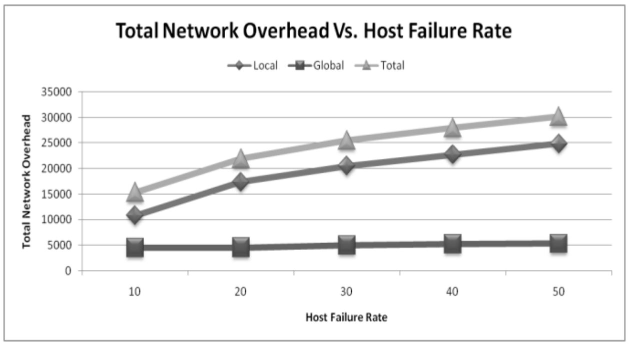

**Figure 13:** ND vs. Increasing Host Failure Rate

*Case 9: TT vs. Link Failure Rate*

This experiment shows the effect of global link failure rate on MA TT. A global link failure is tolerated by HFMS by implementing global ATP. Although ATP is able to tolerate link failure but it delayed the execution of MA. Since failure is detected only after waiting time is over, so delay is significant. Figure 14 shows the graph between TT vs. global link failure rate for min, max and average case. It is clear from the graph that TT increases as link failure rate increases exponentially.

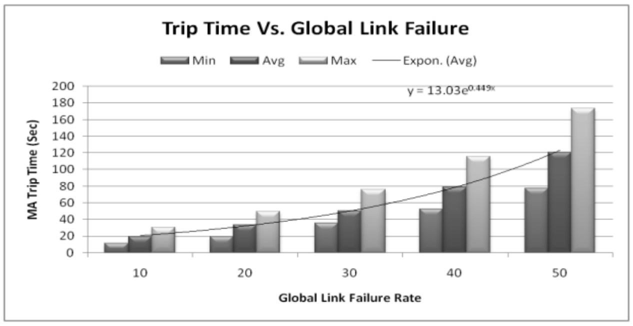

**Figure 14:** TT vs. Increasing Global Link Failure Rate

*Case 10: ND vs. Link Failure Rate*

Global ATP requires the retransmission of probe, acknowledgement, and MA using global links so it increases the Global ND.

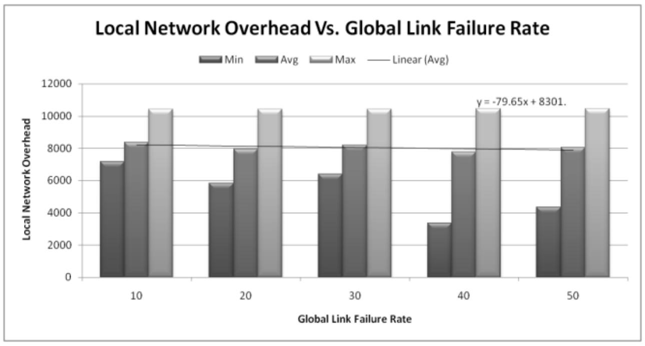

**Figure 15:** Local ND vs. Increasing Global Link Failure Rate

Locally only LT is require to be concern so local ND is independent of link failure. Figure 16 and Figure 17 below shows the graph between ND and link failure for local and global migration. It is clear from the graph that Global ND increase exponentially as global link failure rate increases but local ND is almost constant for global link failure.

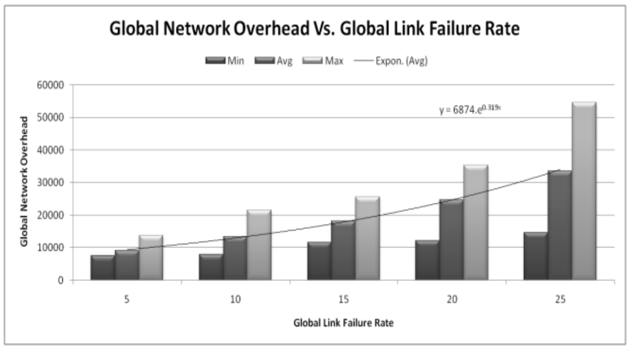

**Figure 16:** Global ND vs. Increasing Global Link Failure Rate

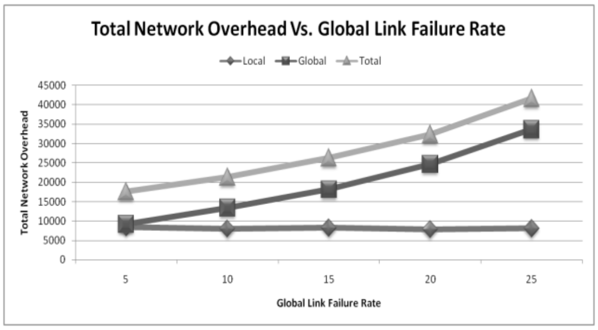

**Figure 17:** ND vs. Increasing Global Link Failure Rate

## Conclusion and Future Work

HFMS is a complete framework that tolerates different faults such as failure of MA, MAS, host and links. Blocking problems have also been addressed and solutions have been proposed. Check-pointing and ATP are the main core idea in HLMS to tolerate host/MAS and link failure. Simulation results using CPN shows that HFMS tolerates different faults (one at a time) without delaying or increasing the ND for low failure rate. There are significant delay and increased overhead for high fault rate. For low failure rate, the survivability of MA in HFMS is ensured and it is able to achieve tolerance without increasing ND or time delay substantially. If host/system failure rate increases, then the MA may be blocked. This blocking may be avoided by providing a Ping method. In this method it first checks whether a host is available or not. In case of link failure, if an alternative list of hosts is defined in its itinerary and also, if the order of the itinerary is not fixed, the MA can visit some other host in its itinerary and may try to visit the disconnected host latter when at least one of the links resumes.

Fault tolerant execution of a MA has certain overheads and can slow down its execution. Not every application of MA is of the same importance and may require limited fault tolerance or no fault tolerance at all. For some applications such as a simple query, fast result is more important than fault tolerant execution. It would be better if the user, the agent platform, or the application itself, is able to decide if and when fault tolerance is to be incorporated and to what extent. The inclusion of a fault tolerance mechanism for one application should not have any influence on other running applications, so that the concurrent execution of both fault tolerant and non-fault-tolerant applications is possible.

## References

Adamski, M. A., Karatkevich, A., and Wegrzyn M. 2005. *Design of Embedded Control Systems, Springer.*

de Assis Silva, F. and Popescu-Zeletin, R. 1998. An approach for providing MA fault tolerance, *In K. Rothermel and F. Hohl, editors, MAs, Proceedings of the Second International Workshop, MA'98", LNCS 1477, Springer Verlag*, pp. 14-25.

Gray, R. S 1995. AgentTcl: A transportable agent System, *in Proceedings of the CIKM Workshop on Intelligent Agents, Fourth International Conference on Information and Knowledge Management (CIKM 95), Baltimore, Maryland*.

Gray, R. S., Kotz, D., Cybenko, G., and Rus, D. 1998. D'Agents: Security in a multiple language, mobile agent system, *in G. Vigna (ed), Mobile Agents and Security, LNCS 1419, Springer-Verlag*, pp. 154-187.

Jensen, K. 1981. Coloured Petri Nets and the Invariant Method. *Theoretical Computer Science* 14, 317–336.

Jensen, K. 1983. High-level Petri Nets. *In: A. Pagnoni, G. Rozenberg (eds.): "Applications and Theory of Petri Nets", Informatik-Fachberichte Springer-Verlag,* Vol. 66, 166–180.

Jensen, K. 1997. Coloured Petri Nets. *Basic Concepts, Analysis Methods and Practical Use", Volume 1, 2 and 3, Monographs in Theoretical Computer Science, Springer-Verlag.*

Kajorth, G., Lange D.B., and Oshima, M. 1997. A Security Model for Aglets. *IEEE Internet Computing*. 1(4): 68-77.

Pathak, H. 2010. Fault Tolerant Execution of Mobile Agent Systems" Thesis Submitted In Gurukula Kangri University Haridwar.

Pathak, H., and Singhal, S. 2016. Performance Analysis of Hierarchical Location Management Scheme to Locate Mobile Agents. *International Journal of Advanced Research in Computer and Communication Engineering*. 5(3), 757-762.

Pathak, H., Gard, K., and Nipur, 2011. Three Layered Hierarchical Fault Tolerance Protocol for Mobile Agent System. *International Journal of Scientific & Engineering Research.* 2(1), 19-25.

Ratzer, A. V., Wells, L., Lassen, H.M. Laursen, M., Qvortrup, J.F., Stissing, M.S., Westergaard, M., Christensen, S., and Jensen, K. 2003. CPN Tools for Editing, Simulating, and Analysing Coloured Petri Nets, *in ICATPN 2003, LNCS 2679, Springer-Verlag Berlin Heidelberg,* pp. 450–462.

Singhal, S., and Pathak, H. 2017. A New Hierarchical Fault Management System (HFMS) of Mobile-Multi Agents. *International Journal on Computer Science and Engineering.* 9(3), 83-93.

Tardo, J., and Luis V. 1996. Mobile Agent Security and Telescript," *in Proceedings of 41st IEEE International Computer Conference (IEEE COMPCON Spring '96), San Jose, CA*, pp. 58-63.

Vila, X., Schuster, A., and Riera, A. 2007. Security for a multi-agent system based on jade. *Computers & Security,* 26,391-400.

Yakovlev, A., Gomes, L., and Lavagno, L. 2000. *Hardware Design and Petri Nets. Springer.*

Zimmermann, P. R. 1995. *The Official PGP User's Guide, The MIT Press.*