Scientific Journal of Impact Factor (SJIF) : 3.134

ISSN (Print) : 2348-6406
ISSN (Online): 2348-4470

International Journal of Advance Engineering and Research Development

A STUDY ON PROCESS MIGRATION ALGORITHMS

Samarth Parikh[1], Drashti Patel[2], Vatsal Shah[3]

[1]*IT Dept., B.V.M Engineering College, samarthparikh30@gmail.com*
[2]*IT Dept., B.V.M Engineering College, drashtipatel6576@gmail.com*
[3]*IT Dept., B.V.M Engineering College, vatshal.shah@bvmengineering.ac.in*

Abstract: *Process migration is the act of transferring an active process between two machines. Today in the real world while process is under execution it may happens that the process gets stacked in between. If this happens for more than one process then there should be some mechanism that helps process to precede further .Here comes the idea of LOAD BALANCING using PROCESS MIGRATION and restoring the process from the point it left off on the selected destination node. We use process migration to control over the load sharing, availability of long process, utilizing some special resources*.

Keyword: *Migration, Address space, Process, kernel, execution*

I. INTRODUCTION

The distributed system allows the transfer of the process between the various nodes. This movement of the executing process from one to another node is known as process migration. The Migration of the process requires a transfer of the kernel data relating to process and address space. Once the sufficient amount of the data has been sent, execution may resume on the destination host. But, the order of the information transfer varies system to system and algorithm used by the system. We can transfer process from one node to another by different process migration algorithm. Process can use the resources that arerequiringcompleting its task at the destination node. After using the resources at the destination node it may migrate back to the Source node. Migration algorithm may suffer from the following problems: Time lag between decision to migrate and resumption on the destination host, Residual dependency.

II. PROCESS MIGRATION ALGORITHMS

2.1 Eager Copy:

The steps taken in the Eager copy:

1. Decision is taken for migration
2. Execution of the process in source node is suspended
3. ENTIRE ADDRESS SPACE is transfer, including the kernel data like open files or directory etc.
4. Process is reconstructed on the destination host
5. Execution begins at destination host

Execution of the process will not resume until all the address space of the process is transferred to destination node [1].
Figure shows the graphical view of the Eager copy algorithm:

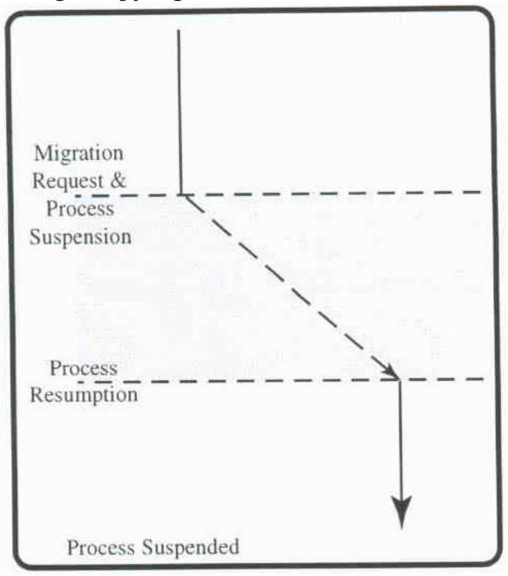

Figure 1. Eager Copy **[1]**

After all the address space has been transfer the process at the source node destroys by normal process destruction algorithm. It is most common algorithm used by, as it has straight forward implementation. This algorithm has one drawback, the time lag between process suspension and process resumption which depend on the size of the process.
Disadvantage: time delay between process suspension and resumption

2.2 Lazy copy:
Eager copy transfer all the address space to the destination node before resumption of the process, in contrast this algorithm transfer only minimal necessary subset of the address space require to start execution of the process.
The order of task in the Lazy copy algorithm is as follows:

1. Decision is taken for migration
2. Execution of the process in source node is suspended
3. The MINIMAL NECESSARY SUBSET OF PROCESS' STATE is transfer
4. Process is reconstructed on the destination host
5. Execution begins at destination host

Figure shows the graphical view of the lazy copy algorithm:

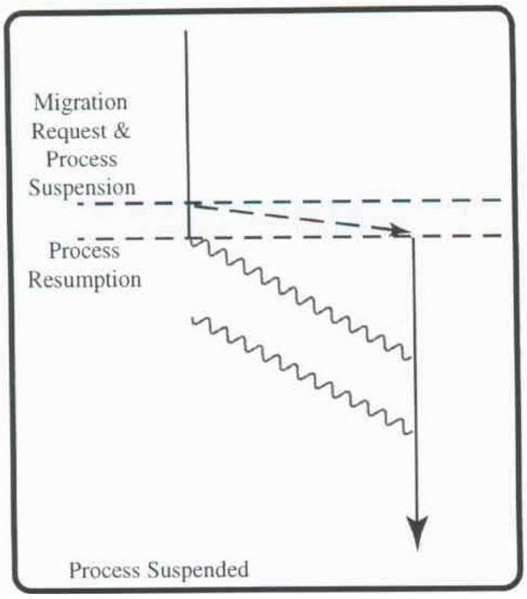

***Figure 2. Lazy Copy*[1]**

Once the process resumes on the remote node, reference may be made by the process to state information which still resides on the source node. Process is suspended for some time until page reference by the process will be received. It has some more delay than local page fault has. So the continuous execution of the process depends upon both Source and destination node, which is called RESIDUAL DEPENDENCY [4].
Benefits of Lazy copy:
- Increase speed in resumption of process [1].
- Less traffic in network [1].
Disadvantage: Residual dependency

2.3 Pre-copy:
In contrast with eager copy and lazy copy, this algorithm did not suspend the process until, all the address space of the process transferred from the source host to the destination host.

1. The order of task in the pre copy algorithm is as follows:
2. Decision is taken for migration
3. Address space of the process is transferred.
4. Parallel with this transfer, execution of the process is continuing on the source host.
5. When address space transfer is completed, execution of the process in source node is suspended.
6. Data altered after the initial transfer is sent again
7. Process reconstructed and execution start.

Figure shows the graphical view of the pre copy algorithm:

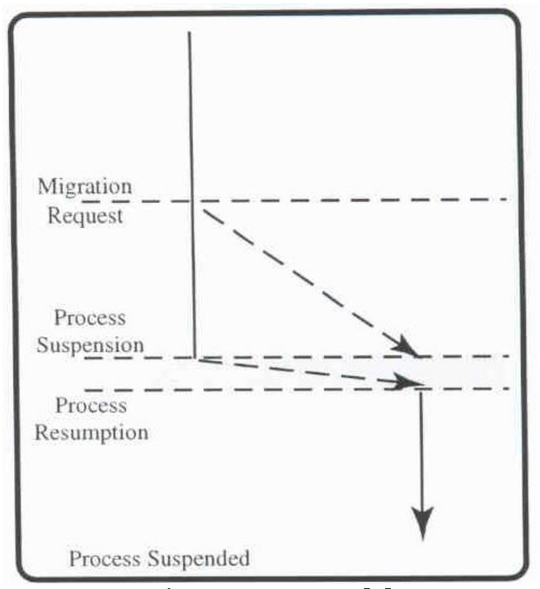

***Figure 3. Pre Copy*[1]**

There is no way to know whether the particular page will be access by the process after transferring has been done or not. That means information must be kept on the destination host until process migration is completed. It avoids large time lag between process suspension and resumption.
Disadvantage: Transfer some data twice.

2.4 Flushing algorithm:

It depends upon operating system's implementation of virtual memory [1]. The backing storage for the virtual memory is implemented by files. These files are stored by the network server and equally accessible to every other node.
The order of task in the flushing algorithm is as follows:

1. Execution of the process in source node is suspended
2. All dirty pages are flushed to the file server.
3. Kernel data is packaged and transferred to the destination host.
4. Process rebuilt on the destination node.
5. Resumes execution with pages retrieved from the file server.

Figure shows the graphical view of the flushing algorithm:

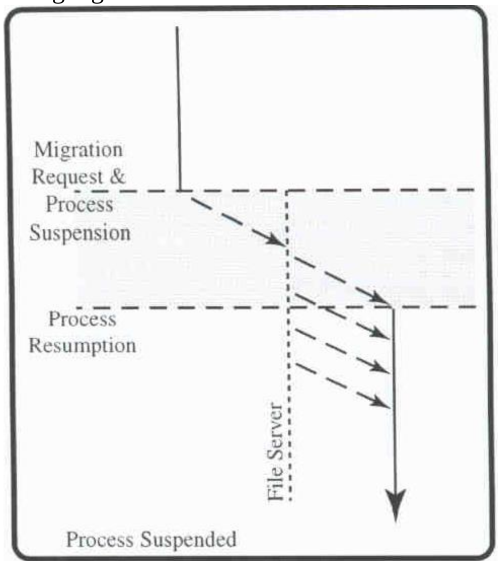

***Figure 4. Flushing*[1]**

Flushing avoids all residual dependencies and reduces the time lag between processes. Page fault can be resolved by the file server.

2.5 Post copy algorithm:

The order of task in the post copy algorithm is as follows:

1. Decision is taken for migration

2. Execution of the process in source node is suspended
3. The MINIMAL NECESSARY SUBSET OF PROCESS' STATE is transfer.
4. Process is reconstructed on the destination host
5. Execution begun at destination host.
6. Parallel to this execution source node continuously sends the remaining data to the destination node.

Figure shows the graphical view of the post copy algorithm:

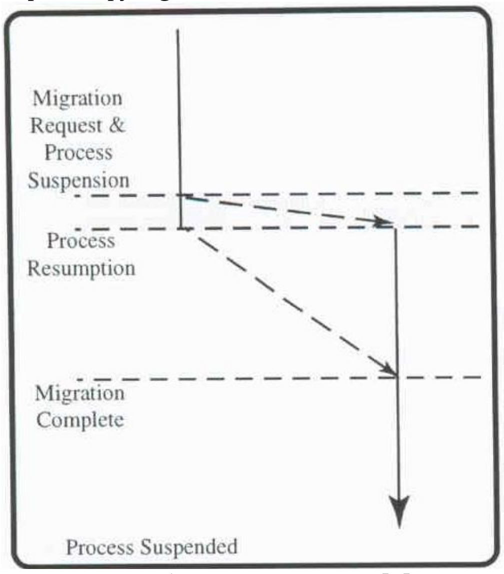

***Figure 5. Post copy* [1]**

III. ADVANTAGES OF PROCESS MIGRATION

1. Load balancing
2. Reliability of the system
3. Less traffic network
4. Resource utilization.

IV. CONCLUSION

This paper describes different process migration algorithms. Disadvantage of eager copy, the time delay, is overcome through lazy copy, and it has also some disadvantages like residual dependency which is solved in pre copy algorithm.

By examining different process migration algorithm, it is seen that post copy algorithm is the most efficient algorithm. It transfer the minimal portion of the address space for resumption of the process and once the process start execution, parallel with it remaining state is transfer. This algorithm decreased the time delay between process suspension and resumption, which is maximum in eager copy. Post copy solves the problem of residual dependency that occurs in lazy copy.

REFERENCES

[1]Michael Richmond, Michael Hitchens, "A new process migration algorithm", Technical Report 509, September 1996.

[2] Dejan S. Milojicic,Fred Douglis, Yves Paindaveine, Richard Wheeler, Songnian Zhou, "Process Migration", ACM Computing Surveys, Vol. 32, No. 3, September 2000.

[3] Amirreza Zarrabi, “A GENERIC PROCESS MIGRATION ALGORITHM”, International Journal of Distributed and Parallel Systems (IJDPS) Vol.3, No.5, September 2012.

[4]Douglis, F., Osterhout, “Transparent process Migration Design Alternatives and the Sprite Implementation“, 21(8),pp. 757-786, August 1991.