Study and Design of Global Snapshot Compilation Protocols for Rollback-Recovery in Mobile Distributed System

Author(s): 1Khan Rais Abdul Hamid, 2Dr. Parveen Kumar
1Research scholar, 2Professor, Dept. of CSE, NIMS University, Jaipur, India

Abstract: Checkpoint is characterized as an assigned place in a program at which ordinary process is intruded on particularly to protect the status data important to permit resumption of handling at a later time. A conveyed framework is an accumulation of free elements that participate to tackle an issue that can’t be separately comprehended. A versatile figural framework is a dispersed framework where some of procedures are running on portable hosts (MHs). The presence of versatile hubs in an appropriated framework presents new issues that need legitimate dealing with while outlining a checkpointing calculation for such frameworks. These issues are portability, detachments, limited power source, helpless against physical harm, absence of stable stockpiling and so forth. As of late, more consideration has been paid to giving checkpointing conventions to portable frameworks. Least process composed checkpointing is an alluring way to deal with present adaptation to internal failure in portable appropriated frameworks straightforwardly. This approach is without domino, requires at most two recovery_points of a procedure on stable stockpiling, and powers just a base number of procedures to recovery_point. In any case, it requires additional synchronization messages, hindering of the basic calculation or taking some futile recovery_points. In this paper, we complete the writing review of some Minimum-process Coordinated Checkpointing Algorithms for Mobile Computing Systems.

Key words: Fault tolerance, checkpointing, message logging, independent checkpointing, consistent global state, domino effect, coordinated checkpointing and mobile systems.

1. Introduction

Mobile Hosts (MHs) are increasingly becoming common in distributed systems due to their availability, cost, and mobile connectivity. An MH is a computer that may retain its connectivity with the rest of the distributed system through a wireless network while on move. An MH communicates with the other nodes of the distributed system via a special node called mobile support station (MSS). A “cell” is a geographical area around an MSS in which it can support an MH. An MSS has both wired and wireless links and it acts as an interface between the static network and a part of the mobile network. Static nodes are connected by a high speed wired network [1, 4, 14, 18, 20, 26].

A recovery_point is a local state of a process saved on the stable storage. In a distributed system, since the processes in the system do not share memory, a global state of the system is defined as a set of local states, one from each process. The state of channels corresponding to a global state is the set of messages sent but not yet received. A global state is said to be “consistent” if it contains no orphan message; i.e., a message whose receive event is recorded, but its send event is lost. To recover from a failure, the system restarts its execution from the previous consistent global state saved on the stable storage during fault-free execution. This saves all the computation done up to the last checkpointed state and only the computation done thereafter needs to be redone [5, 9].

In coordinated or synchronous checkpointing, processes take recovery_points in such a manner that the resulting global state is consistent. Mostly it follows the two-phase commit structure. In the first phase, processes take tentative recovery_points, and in the second phase, these are made permanent. The main advantage is that only one permanent recovery_point and at most one tentative recovery_point is required to be stored. In the case of a fault, processes rollback to the last checkpointed state [2, 3, 4, 5, 7, 26]. In this paper, we carry out the literature survey of some Minimum-process Coordinated Checkpointing Algorithms for Mobile Computing Systems.

2. Coordinated Checkpointing Algorithms for Mobile Computing Systems.

Lai and Yang’s[16] Worldwide depiction calculation for non-FIFO systems depends on two perceptions on the part of a marker in a FIFO system. The principal perception is that a marker guarantees that condition C2 is fulfilled for LSi and LSj when the previews are recorded at procedures pi and pj separately. The Lai-Yang calculation satisfies this part of a marker in a non-FIFO system by utilizing a shading plan on calculation communications that functions as follows: Every procedure is at first white and turns red while taking a depiction. What might as well be called the “marker sending rule” is executed when a procedure turns red. Each communication sent by a white (red) procedure is hued white (red). Hence a white (red) communication is a communication that was sent before (after) the sender of that communication recorded its neighborhood recovery-point. Every white procedure takes its preview whenever the timing is ideal, yet no later than the moment it gets a red communication.

Hence, when a white procedure gets a red communication, it records its neighborhood depiction before preparing the communication. This guarantees no communication sent by a procedure subsequent to recording its neighborhood preview is prepared by the destination procedure before the destination records its nearby depiction. Along these lines, an express marker communication is not required in this calculation and the “marker” is piggybacked on calculation communications utilizing a shading plan.

The second perception is that the marker illuminates process pj of the estimation of (send (mij) send (mij) ∈LSj) so that the condition of the channel Cij can be figured as travel (LSi, LSj). The Lai-Yang calculation satisfies this part of the marker in the accompanying way: Every white...
procedure records a past filled with all white communications sent or got by it along every channel. At the point when a procedure turns red, it sends these histories alongside its preview to the initiator procedure that gathers the worldwide depiction. The initiator procedure assesses travel (LSi, LSj) to register the condition of a channel Cij as given underneath:

SCij=white communications sent by pi on Cij – white communications got by pj on Cij = [mij] send (mij) ∈ LSi – [mij] rec (mij) ∈ LSj] Condition C2 holds on the grounds that a red communication is excluded in the depiction of the beneficiary procedure and a channel state is the distinction of two arrangements of white communications. Condition C1 holds on the grounds that a white communication mij is incorporated into the depiction of procedure Pj if Pj gets mij before taking its preview. Something else, Mij is incorporated into the condition of channel Cij.

Silva and Silva [53] designed the methodology which did not speak with others amid the past depiction aggregation interim don't have to record new previews. The above talked about conventions plan to diminish the working cost associated with synchronized depiction assemblage. Studies are performed with a specific end goal to diminish the synchronization correspondences, decrease the quantity of procedures to preview and to make the conventions non-nosy. Singhal [20, 2008] were first to give particular procedure non-nosy facilitated depiction accumulation convention for MDS. Be that as it may, their method may prompt irregularities [9]. In [9, 1998], it was demonstrated that there does not exist a specific procedure non-blocking facilitated depiction assemblage calculation. Henceforth in specific procedure composed preview aggregation calculations, some interruption of the procedures records place [3], or some futile depictions are taken [7].

Kim and Park procedure [19] outlined an enhanced plan for adaptation to non-critical failure amid check guiding, it makes a sub-tree for new preview. In the event that none of the procedures on which it transitively depends, fizzes, the reliable recovery line is a progressed for the submitted procedure along these lines complete prematurely end of the preview can be abstained from amid recuperation, an improved plan of Koo and Toueg by taking provisional depiction, recuperation is through conditional depiction and confer.

Neves et al. [33] Outlined an approximately synchronized preview arrangement strategy that takes out the working expense of synchronization. In this methodology it is accepted that the timekeepers at the procedures are approximately synchronized. Inexactly synchronized tickers can actuate the neighborhood depictions at all the procedures roughly in the meantime without a controller. In the wake of recording a depiction, a procedure sits tight for a period, which is aggregate of most noteworthy time to see a disappointment of different procedure in the system and the most astounding dissimilarity between tickers. It is comprehended that all depictions worried to a particular coordination session have been considered without the need of sharing any correspondence. On the off chance that a disappointment experienced, it is recognized inside the specific time and the strategy is deserted.

Parkan & Singhal Procedure built up a low overhead depiction accumulation and recuperation calculation for distributed applications in MCS that need the necessity of MCS, past calculations were static dist-systyet in present calculations not very many are static hubs. The creator has given qualities of good calculation, 1) calculation ought to be non nosy and effective, 2) ought to have min loss of recuperation from hub failure, 3) ought to have low accessible data transfer capacities 4) low vitality utilization 5) ought to adapt to changing topology of the system because of versatility. It was the principal calculation to consolidate these two methodologies i.e. least process and non-blocking. All the more particularly, it drives just a base number of procedures to take previews and does not hinder the fundamental calculation amid depiction. Prakash Singhal calculation constraints just piece of procedures to take depictions, the csn of some procedures might be obsolete, and will most likely be unable to dodge irregularities. It endeavors to take care of this issue by having every procedure keeps up a cluster to spare the csn, where csn[i] has been the normal csn of Pi. Note that Pi's csn[i] might be not quite the same as Pj's csn[j] if there is no correspondence amongst Pi and Pj for a few preview interims. By utilizing csn and the initiator recognizable proof number, they assert that their non-blocking calculation can dodge irregularities and Mob-Hosts the quantity of previews amid depiction. It was found that the calculation may prompt inconsistencies[9]. An aftereffect of the effectiveness and non nosiness criteria is that the depiction initiator does not from the earlier know the character of the considerable number of hubs that will take part in the preview accumulation. This raises the issue of productive end identification of the preview gathering process.

Cao and Singhal [9] Composed particular procedure interruption based strategy for preview gathering in MDS. In this method, interruption time is significantly compressed regarding [23]. Direct conditions of each procedure is held in a bit cluster of length n for n forms. Originator process gets the immediate reliance vectors of all procedures and discovers least set. At that point, the preview advance is sent alongside the base set to all procedures. Amid the period, a procedure sends its reliance vector to the originator procedure and catches the base set, it keeps on being in the interruption time frame. A procedure records its depiction on the off chance that it is in the base set.

In Cao-Singhal Non-intrusive Recovery-pointProcedure [7], they set up that no min-process non-blocking calculation exists. There are two rules in plotting very much sorted out composed preview calculations. To begin with is to relax up the non-blocking stipulation while keeping the min-process property in place. The other is to loosen up the min-process condition while keeping the non-blocking property set up. The new issues in versatile figuring system, for example, low data transfer capacity of remote channel, high pursuit cost, and restricted battery life, set forward that the planned depiction calculation ought to be a min-process calculation. Accordingly, Cao and Singhal built up a calculation that unwinds the min-process condition. In this plan, to outline productive preview calculations for versatile figuring systems, they presented the idea of changeable depiction, which is neither a conditional preview nor a perpetual preview, Mutable depictions can be put away anyplace, e.g., the primary memory or nearby plate of portable hosts(Mob-Hosts).

Such calculations depend on the two-stage submit plan and spare two sorts of depictions on the steady stockpiling: conditional and lasting.
In the main stage, the initiator catches a conditional depiction and powers all applicable procedures to record speculative previews. Every procedure illuminates the initiator whether it succeeded in catching a conditional preview. At the point when the initiator infers that all important procedures have gainfully caught provisional depictions, it demands them to make their conditional previews changeless; else, it requests that they desert them. A procedure, on accepting the communication from the initiator, demonstrations likewise. A non-blocking depiction calculation does not require any procedure to put off its fundamental calculation. At the point when procedures don't concede their calculations, it is likely for a procedure to get a calculation communication from another procedure which is as of now running in another preview interim. In the event that this circumstance is not fittingly took care of, it might bring about a discrepancy [38]. In their calculation, originator, say Pin, sends the preview solicitation to any procedure, say Pj, just if Pin gets m from Pj in the present CI. Pj takes its conditional preview if Pj has sent m to Pin in the present CI; generally, Pj presumes that the depiction solicitation is a futile one. Correspondingly, when Pj takes its speculative depiction, it engenders the preview solicitation to different procedures. This procedure is proceeded till the depiction demand achieves all the procedures on which the initiator transitively depends and a preview tree is framed. Amid preview, if Pi gets m from Pj such that Pj has gotten some depiction in the present start before sending m, Pi might be compelled to take a preview, called variable preview. On the off chance that Pi is not in the base set, its changeable preview is pointless and is disposed of on confer. The enormous information structure MR[] is likewise appended with the preview solicitations to diminish the quantity of pointless depiction demands. The reaction from every procedure is sent straightforwardly to initiator.

By using mutable checkpoints:

1. Number of irrelevant checkpoints is reduced
2. Overhead of taking mutable checkpoints is negligible

3. Hybrid of Minimum-process and All-Process Coordinated Checkpointing Algorithms for Mobile Computing Systems.

In minimum-process checkpointing, some processes, having low communication activity, may not be included in the minimum set for several recovery_point initiatives and thus may not advance their recovery line for a long time. In the case of a recovery after a fault, this may lead to their rollback to far earlier checkpointed state and the loss of computation at such processes may be exceedingly high. In all-process checkpointing, recovery line is advanced for each process after every global recovery_point but the checkpointing overhead may be exceedingly high, especially in mobile environments due to frequent recovery_points. MHs utilize the stable storage at the MSSs to store checkpoints of the MHs. Thus, to balance the checkpointing overhead and the loss of computation on recovery, a hybrid checkpointing algorithm for mobile distributed systems is proposed, where an all-process recovery_point is taken after certain number of minimum-process recovery_points.

A strategy is proposed to optimize the size of the csn. In order to address different checkpointing intervals, he replaced integer csn with k-bit CI. Integer csn is monotonically increasing, each time a process takes its recovery_point, it increments its csn by 1. k-bit CI is used to serve the purpose of integer csn. The value of k can be fine-tuned.

The minimum-process checkpointing algorithm is based on keeping track of direct dependancies of processes. Initiator process collects the direct dependency vectors of all processes, computes minimum set, and sends the recovery_point request along with the minimum set to all processes. In this way, blocking time has been significantly reduced as compared to [9].

During the period, when a process sends its dependency set to the initiator and receives the minimum set, may receive some messages, which may alter its dependency set, and may add new members to the already computed minimum set. In order to keep the computed minimum set intact and to avoid useless recovery_points, he proposed to block the processes for this period.

6. CONCLUSION

A survey of the literate on checkpointing algorithms for mobile distributed systems shows that a large number of papers have been published. We have reviewed and compared different approaches to checkpointing in mobile distributed systems with respect to a set of properties including the assumption of piecewise determinism, performance overhead, storage overhead, ease of output commit, ease of garbage collection, ease of recovery, useless checkpointing, low energy consumptions.

REFERENCES

[1] Acharya A. and Badrinath B. R., “Checkpointing Distributed Applications on Mobile Computers,” Proceedings of the 3rd International Conference on Parallel and Distributed Information Systems, pp. 73-80, September 1994.

[2] Cao G. and Singhal M., “On coordinated checkpointing in Distributed Systems”, IEEE Transactions on Parallel and Distributed Systems, vol. 9, no.12, pp. 1213-1225, Dec 1998.

[3] Cao G. and Singhal M., “On the Impossibility of Min-process Non-blocking Checkpointing and an Efficient Checkpointing Algorithm for Mobile Computing Systems,” Proceedings of International Conference on Parallel Processing, pp. 37-44, August 1998.

[4] Cao G. and Singhal M., “Mutable Checkpoints: A New Checkpointing Approach for Mobile Computing systems,” IEEE Transaction On Parallel and Distributed Systems, vol. 12, no. 2, pp. 157-172, February 2001.

[5] Chandy K. M. and Lamport L., “Distributed Snapshots: Determining Global State of Distributed Systems,” ACM Transaction on Computing Systems, vol. 3, No. 1, pp. 63-75, February 1985.

[6] Elnozahy E.N., Alvisi L., Wang Y.M. and Johnson D.B., “A Survey of Rollback-Recovery Protocols in Message-Passing Systems,” ACM Computing Surveys, vol. 34, no. 3, pp. 375-408, 2002.

[7] Elnozahy E.N., Johnson D.B. and Zwaenepoel W., “The Performance of Consistent Checkpointing,” Proceedings of the 11th Symposium on Reliable Distributed Systems, pp. 39-47, October 1992.
[8] Hélary J. M., Mostefaoui A. and Raynal M., “Communication-Induced Determination of Consistent Snapshots,” Proceedings of the 28th International Symposium on Fault-Tolerant Computing, pp. 208-217, June 1998.

[9] Koo R. and Toueg S., “Checkpointing and Roll-Back Recovery for Distributed Systems,” IEEE Trans. on Software Engineering, vol. 13, no. 1, pp. 23-31, January 1987.

[10] J.L. Kim, T. Park, “An efficient Protocol for checkpointing Recovery in Distributed Systems,” IEEE Trans. Parallel and Distributed Systems, pp. 955-960, Aug. 1993.

[11] L. Kumar, M. Misra, R.C. Joshi, “Checkpointing in Distributed Computing Systems” Book Chapter “Concurrency in Dependable Computing”, pp. 273-92, 2002.

[12] L. Kumar, M. Misra, R.C. Joshi, “Low overhead optimal checkpointing for mobile distributed systems” Proceedings. 19th IEEE International Conference on Data Engineering, pp 686 – 88, 2003.

[13] L. Lamport, “Time, clocks and ordering of events in a distributed system” Comm. ACM, vol.21, no.7, pp. 558-565, July 1978.

[14] Ni, W., S. Vrbsky and S. Ray, “Pitfalls in Distributed Nonblocking Checkpointing”, Journal of Interconnection Networks, Vol. 1 No. 5, pp. 47-78, March 2004.

[15] Parveen Kumar, Lalit Kumar, R K Chauhan, V K Gupta “A Non-Intrusive Minimum Process Synchronous Checkpointing Protocol for Mobile Distributed Systems” Proceedings of IEEE ICPWC-2005, January 2005.

[16] Kumar, P., “A Low-Cost Hybrid Coordinated Checkpointing Protocol for Mobile Distributed Systems”, Mobile Information Systems pp 13-32, Vol. 4, No. 1., 2007.

[17] Pradhan D.K., Krishana P.P. and Vaidya N.H., “Recoverable Mobile Environment: Design and Trade-off Analysis,” Proceedings 26th International Symposium on Fault-Tolerant Computing, pp. 16-25, 1996.

[18] Pradhan D.K. and Vaidya N., “Roll-forward Checkpointing Scheme: Concurrent Retry with Non-dedicated Spares,” Proceedings of the IEEE Workshop on Fault-Tolerant Parallel and Distributed Systems, pp. 166-174, July 1992.

[19] Pushpendra Singh, Gilbert Cabillic, “A Checkpointing Algorithm for Mobile Computing Environment”, LNCS, No. 2775, pp 65-74, 2003.

[20] Prakash R. and Sinha M., “Low-Cost Checkpointing and Failure Recovery in Mobile Computing Systems,” IEEE Transaction On Parallel and Distributed Systems, vol. 7, no. 10, pp. 1035-1048, October 1996.

[21] Lalit Kumar Awasthi, P.Kumar, “A Synchronous Checkpointing Protocol for Mobile Distributed Systems: Probabilistic Approach” International Journal of Information and Computer Security, Vol.1, No.3 pp 298-314, 2007.

[22] Parveen Kumar, Preeti Gupta, Anil Kumar Solanki, “An Efficient Minimum-process Checkpointing Scheme for Non-Deterministic Mobile Distributed Systems”, CIIT International Journal of Networking and Communication Engineering”, July 2010.

[23] Nuno Neves and W. Kent Fuchs. “Adaptive Recovery for Mobile Environments”, in Proc.IEEE High-Assurance Systems Engineering Workshop, October 21-22, 1996, pp.134-

[24] Higaki H. and Takizawa M., “Checkpoint-recovery Protocol for Reliable Mobile Systems,” Trans. of Information processing Japan, vol. 40, no.1, pp. 236-244, Jan. 1999.

[25] Pushpendra Singh, Gilbert Cabillic, “A Checkpointing Algorithm for Mobile Computing Environment”, LNCS, No. 2775, pp 65-74, 2003.

[26] Sunil Kumar, R K Chauhan, Parveen Kumar, “A Minimum-process Coordinated Checkpointing Protocol for Mobile Computing Systems”, International Journal of Foundations of Computer science, Vol. 19, No. 4, pp 1015-1038 (2008).

[27] Parveen Kumar, Rachit Garg, “Soft Checkpointing Based Hybrid Synchronous Checkpointing Protocol for Mobile Distributed Systems”, International Journal of Distributed Systems and Technologies”, 2(1), 1-13, January-March 2011.

[28] Ruchi Tuli, Parveen Kumar, “New Paradigms in Checkpoint Processing and Recovery Techniques for Mobile Distributed Systems,”, Wimon-2011

[29] Ruchi Tuli, Parveen Kumar, “The Design and Performance of a checkpointing scheme for mobile ad hoc networks”, PDC/TA-2011