A MOBILITY-BASED CHECKPOINTING PROTOCOL FOR MOBILE COMPUTING SYSTEM

Suparna Biswas[1] and Sarmistha Neogy[2]

[1]Department of Computer Science & Engineering, West Bengal University of Technology, Kolkata, India
mailtosuparna@gmail.com
[2] Department of Computer Science & Engineering, Jadavpur University, Kolkata, India
sarmisthaneogy@gmail.com

ABSTRACT

A new checkpointing and failure recovery algorithm for mobile computing system is proposed here. Mobile hosts save checkpoints based on mobility and movement patterns.Movement patterns considered here are of three types – i) Intercell movement pattern ii) combination movement pattern ii) Intracell movement pattern. Mobile hosts save checkpoints when number of hand-off exceeds a predefined hand-off threshold value. Disconnection is a frequent phenomenon and is of two types: i) planned disconnection ii) unplanned disconnection.Hence mobile hosts save two types of checkpoints - i) permanent checkpoint based on hand-off threshold value covering unplanned disconnection ii) migration checkpoint covering planned disconnection. Hand-off threshold is a function mobility rate , movement pattern, message passing frequency and failure rate.

KEYWORDS

Checkpointing, movement pattern, intercell, intracell, combination, mobility, hand-off, migration checkpoint

1. INTRODUCTION

A mobile computing system is a distributed system where some of the processes run on mobile hosts (MHs) moving over the network and a few fixed hosts (MSS) act as access points to communicate with MHs. Presence of the following characteristics distinguish between distributed system and mobile computing systems:

- Limited Bandwidth
- Limited and vulnerable MH local storage
- Frequent disconnection/connection
- Limited power
- Cost to locate MHs

Computing potential of these systems is often hampered by their susceptibility to failures. Checkpointing and Rollback Recovery is an efficient technique for providing fault tolerance to distributed as well as mobile computing systems. Mobility and frequent disconnections of MHs due to hand-off or failure is inherent in MCS. This feature of MCS and its effect on checkpointing is the prime focus of our proposed technique. Traditional checkpointing algorithms are periodic. But periodic checkpointing is not suitable for a system with mobile hosts. This is because depending on movement pattern the number of hand-offs may be more in one checkpoint interval than the other. This may result in uneven recovery time upon failures of MHs. Hence system's reliability becomes unpredictable. Checkpointing based on movement pattern and mobility rate of MHs may cause delay in checkpointing. This motivates us to introduce a concept of migration checkpoint. An MH upon saving migration checkpoint, sends it attached with migration message to its current MSS

before disconnection. The latest MSS of disconnected MH participates in checkpointing with m_checkpoint hiding the fact that the MH is still disconnected. During checkpointing participating MHs are barred only from receiving execution message as it will change list of dependent MHs in current checkpoint interval.

The rest of the paper is structured as follows: Section 2 describes system model & preliminary assumptions. Section 3 discusses some of the related works and our observations. Section 4 elaborates data structures and notations used. Section 5 explains proposed checkpointing scheme, basic ideas and describes the algorithm. Section 6 gives Necessary correctness proofs. Section 7 elaborates simulation and performance analysis. In Section 8 we conclude our work.

2. SYSTEM MODEL

Mobile computing systems generally consists of n MHs and m MSSs, n>>m. MHs are connected through wireless network and MSSs are connected through wired network. Communication links connecting MHs & MSSs are assumed to be FIFO. Messages take arbitrary but finite amount of time during transmission. There are no synchronized clocks or shared memory among nodes. Two types of messages are hereby assumed: i) Execution Messages(m_e) – generated based on computational work of processes and ii) Coordination Messages(m_c) – generated to coordinate the checkpointing activity. Two types of checkpoints are saved: i) Migration Checkpoint (M_checkpoint) – saved before planned disconnection of MHs ii) Permanent checkpoints. More than one process may try to initiate checkpointing but only one process can have the privilege over the others depending on some criteria.

2.1 PRELIMINARY ASSUMPTIONS

i) MHs do not have global clock and do not share memory
ii) During checkpointing, MHs are barred from receiving m_es. Because if an MH receives m_e, then sender become dependent on receiver. Checkpoint coordination will fail.
iii) Proposed algorithm is a combination of logging and checkpointing techniques.
iv) During checkpoint interval, messages sent, received are saved into log file.
v) MHs refresh log files to control its size so that log file search does not incur overhead.
vi) Proposed algorithm is non-blocking i.e. MHs can compute, send messages during checkpointing.

3. RELATED WORKS

Prakash and Singhal describe in [19] a checkpointing algorithm for Mobile Conputing System. Checkpoint collection is synchronous and non-blocking. A minimum number of nodes are forced to take checkpoints. Each MH maintains a dependence vector. MHs maintains causal relationships through message. This scheme reduces energy consumption by powering down individual components during periods of low activity.

In [9] T.Park et.al has presented an efficient movement based recovery scheme. This scheme is a combination of message logging and independent checkpointing. Main feature of this algorithm is that a host carrying its information to the nearby MSS can recover instantly in case of a failure. To enhance failure-free execution, concept of a 'certain range' is introduced. An MH moving inside a range , recovery information remains in host MSS otherwise it moves recovery information to nearby MSS. Though recovery is ensured, failure-free execution cost increases. Due to this out of range concept overheads due to transfer of checkpoint from one MSS to another MSS increases many fold. In this scheme two movement-based schemes are suggested-distance based and frequency based. Distance based scheme focuses on the distance between mh_i and the MSS carrying its latest checkpoint. Frequency based scheme concerns the number of hand-offs to limit cost of collection of logs in different cites in case of recovery.

Sapna E. George [4]et.al describes a checkpointing and logging scheme based on mobility of MHs. A checkpoint is saved when hand-off count exceeds a predefined optimum threshold. Optimum threshold is decided as a function of MH's mobility rate, failure rate and log arrival rate. Recovery probability is calculated and recovery cost is minimized in this scheme.

Cao and Singhal presents in [10] a non-blocking coordinated checkpointing algorithm with the concept of "Mutable Checkpoint" which is neither temporary nor permanent and can be converted to temporary checkpoint or discarded later and can be saved anywhere. In this scheme MHs save a disconnection checkpoint before any type of disconnection .This checkpoint is converted to permanent checkpoint or discarded later. In this scheme only dependent processes are forced to take checkpoints.

4. BASIC IDEA OF PROPOSED PROTOCOL

Mobility and hand-off of mobile hosts are considered in checkpointing and recovery protocol. But none of existing algorithms related to our work have considered movement-pattern of mobile hosts. Movement pattern of MHs may be of three types: i) intercell ii) intracell iii) combination of the two. It is our observation that hand-off is not only dependent on mobility rate but also on movement pattern. If an MH moves in intercell movement pattern hand-off rate will be proportional with mobility rate hence checkpointing should be hand-off based. If an MH moves in intracell movement pattern no hand-off will occur hence hand-off based checkpointing will not serve the purpose. Periodic checkpointing will work in such cases where interval of checkpoints will be chosen based on average failure rate of mobile hosts. Checkpoint interval is inversely proportional to failure rate . If an MH moves in combination movement pattern checkpointing will be hand-off based but hand-off count will reach hand-off threshold after much longer time than in intercell movement pattern. Another important observation is that MHs disconnect in planned or unplanned way. Planned disconnection is of longer duration and less frequent. Unplanned disconnection is of shorter duration and frequent. In [10], MHs save a disconnection checkpoint before any type of disconnections. We find that checkpointing based on hand-off covers unplanned disconnection due to insufficient radio cover. Other reasons for unplanned disconnection are not considered here. Hence in our scheme MHs save a checkpoint termed as "migration checkpoint"independently before planned disconnection. This modification reduces number of checkpoints to be saved forcefully by disconnections which are sudden. Thus overheads due to save checkpoints are reduced and memory, bandwidth utilizations are optimized. These observations lead us to design a checkpinting protocol that will be a combination of movement based checkpointing and periodic checkpointing based on movement patterns of mobile hosts.

5. DATA STRUCTURES AND NOTATIONS

CMSS = Current MSS of an MH
CoMSS = Coordinator MSS of checkpoint protocol
old_MSS = MSS that an MH leaves due to hand-off
new_MSS = MSS to which an MH joins after hand-off
T_count = an integer variable to count time
H_count = an integer variable to count hand-off
n= number of MHs
m=number of MSSs
T_w = waiting time
MH_i , i = o.....n n no. of MHs
$csn_{i,j}$ = Checkpoint sequence number
i = 0.....n
j = 0.....n
chkpt = checkpoint
m_e = execution message
m_c = coordination message

h_T =hand-off threshold
mp =movement pattern
$combination_{(p\%,q\%)}$= during a checkpoint interval MH moves p% intercell and q% intracell
intercell = MH moves across cells during a checkpoint interval
intracell = MH moves within a cell during a checkpoint interval
interval = periodic checkpoint interval for MHs moving in intracell movement pattern
dis_pointer = <t, MH_id>
D_flag = 0, MH connected
= 1, MH disconnected
P_dis = 0, unplanned disconnection
= 1, planned disconnection
GCCS [ ] = Global Consistent Checkpoint set

MH_Structure:

MH_id, i = 0.....n
CMSS_id, j = 0.....m
H_count = 0
T_count = 0
H_T = K
T_w =0
T_{w_max} =$T_{recovery}$
MH_D [ ]
$csn_{i,j}$, i = 0....n
j = 0....n

MSS_Structure:

MSS_id, j = 0.....m
CMH_list [ ]
log_file
chkpt
GCCS [ ]

Log_file
at sender **:**

Receiver_MH_id
chkpt_interval
m_e

checkpoint_file:

MH_id
status
Data
chkpt_interval

5.1. Fundamental Protocol Scheme

In a mobile computing system all the MHs are connected to MSSs by assumption. MHs move in any possible direction with a fixed mobility rate in the system assumed here. As MHs move hand-off will occur. If hand-off count exceeds predefined threshold value, it initiates checkpoint protocol, saves a temporary checkpoint and sends checkpoint initiation message to its current MSS. Current MSS now coordinates checkpointing. MSS_{co} forwards checkpoint request message to all MHs dependant on initiator MH during current checkpoint interval. MH_Ds save temporary checkpoint and send reply to $MSS_{co.}$ MSS_{co} converts MH_i's temporary checkpoint to permanent checkpoint and forwards reply all MH_Ds. MH_Ds convert temporary checkpoint to permanent checkpoint and send commit message to $MSS_{co.}$

5.2 Enhanced Proposed Protocol Scheme

Following events may happen during the basic checkpointing protocol being executed:

Event: MH moving within a cell (intracell movement) will never initiate checkpointing in the above mentioned checkpointing scheme.

Solution: Each MH maintains a local timer. Depending on the application and the system, MH will initiate checkpoint protocol after a suitable time interval..

Event: If any MH disconnects during checkpointing

Solution: MH_i itself, any MH_D or any other MHs in the system may be disconnected from the network. MSS of the disconnected MH saves a pointer variable dis_pointer defined by two paramers:

dis_pointer = ‹ t, MH_id ›, t = time instant when an MH disconnects from MSS
MH_id = ID of that particular MH

- **Planned disconnection:** To save energy can go to planned disconnection. MH saves m_checkpoint, forwards it attached with disconnection message to MSS_c before disconnection.
- **Unplanned disconnection:** Due to movement of MHs from one MSS to another MSS or insufficient radio cover an MH can go to unplanned disconnection. Hand-off based checkpointing scheme described here takes care of these sudden, temporay and frequent events
- **Failure:** Due to network failure or any other reason an MH may fail or crash. Initial effect is that the MH will not be connected to any MSS. After a certain time interval this disconnected MH will be treated as failed MH and a recovery operation will be performed. How the time interval will be calculated or how the two events failure and unplanned disconnection will be distinguished? Answer is as follows:
  MH send reconnection message to a new MSS that broadcasts a 'hello' message defined as: hello (t_{hellow_S}, MH_id) to all MSSs. MSSs receive hello, save t_{hellow_R} and convert t_{hellow_S} to their own local time, $t_{hellow\_S\_local}$. MSSs compare MH_id with that saved in dis_pointer. If any MSS find a match in MH_id, it calculates disconnected time period as follows:
  $T_{disconnection} = (t_{hellow\_S\_local} - t)$
  If ($T_{disconnection} \leq T_{dis\_unplanned\_max}$) = = TRUE
  Disconnection==unplanned disconnection;
  else
  Disconnection = = failure;
  Recovery ():

Working Example: Let there be 5 MHs executing and communicating via message passing. MH_0 moves intercell,receives 4 execution messages from others and sends 1 execution message to another MH as shown in figure 2. In MH_0's dependence list there are MH_1, MH_3, MH_4, MH_5. At the time instant of point p shown in figure 1, hand-off count of MH_0 exceeds hand-off threshold value and initiates checkpointing. Checkpoint request message is forwarded to the dependent MHs through current MSS. Dependent MHs that are connected save temporary checkpoint. This is true for MH_1, MH_4, MH_5 but MH_3 is disconnected. Now consider following cases:

Case1: MH_3 does not save migration checkpoint- initiator delays checkpointing process till MH_3 reconnects at point d, saves temporary checkpoint and forwards to initiator.

Case2: MH_3 saves migration checkpoint , forwards to current MSS before disconnection. MSS on behalf of it participates in checkpointing. Checkpointing is not delayed.

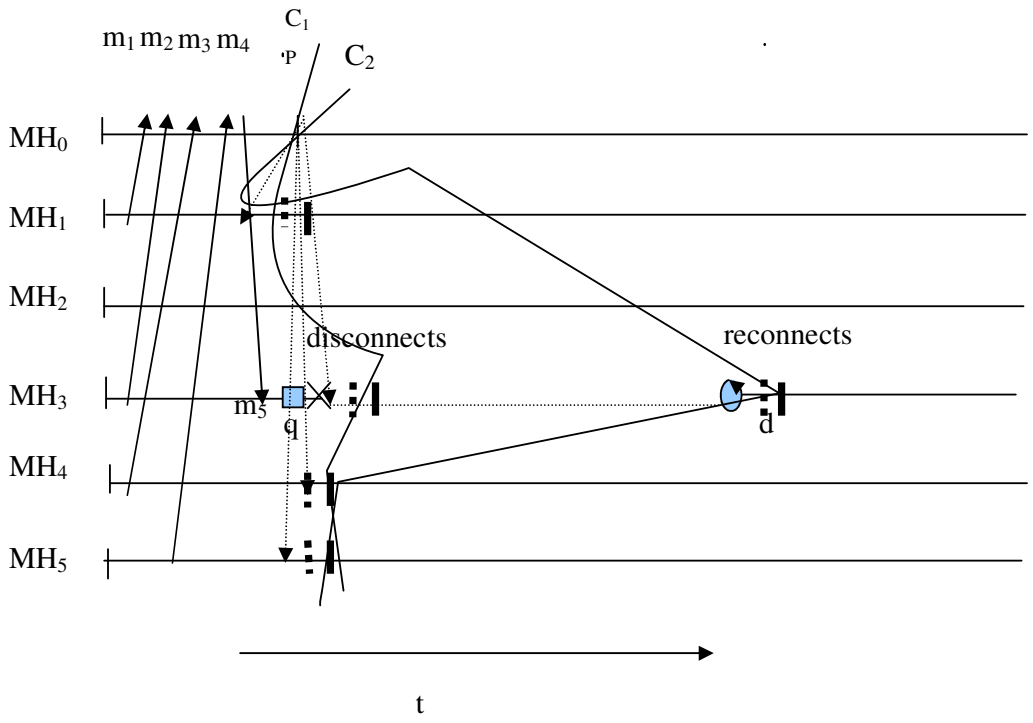


Figure 1 : Working Example of proposed checkpointing Scheme

Time duration when MHs are disconnected

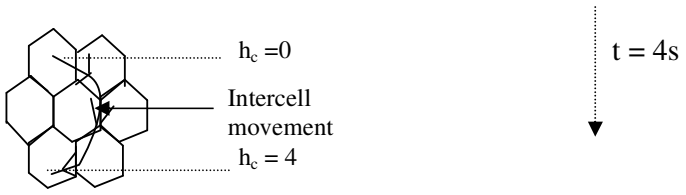
Time duration when MHs are connected

5.3. Basic Ideas:

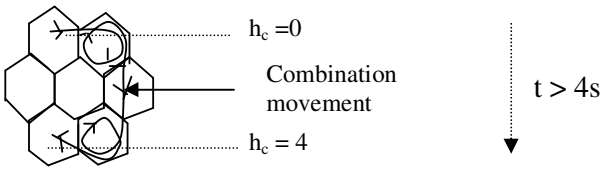
Some of the basic ideas used here are explained below:

a) Movement Pattern:

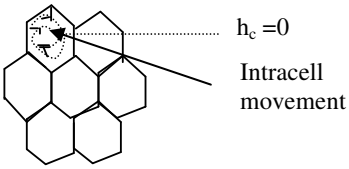
1. **Intercell Movement Pattern:** If a MH moves across cells, movement pattern is intercell.



2. **Combination Movement Pattern:**If a MH moves across different cells as well as within a cell movement pattern is combination

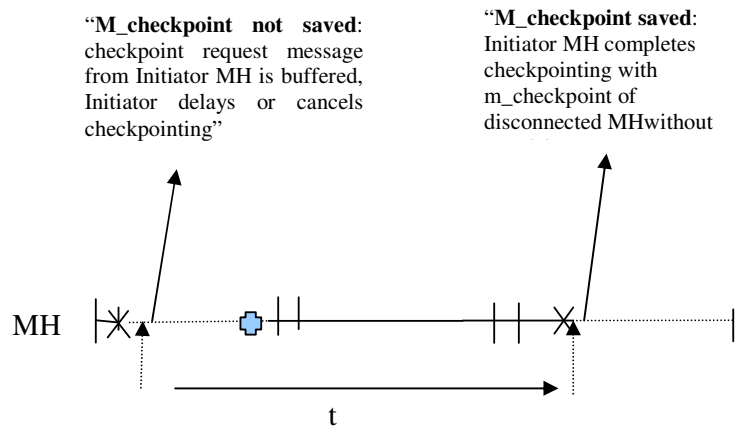


3. **Intracell Movement Pattern**: If a MH moves only within a single cell movement pattern is intracell.



b) Disconnection and Failure: An MH is not connected to the network in both the cases – i) MH disconnects ii) MH fails. Then how to distinguish these two events? Before finding answer, what is the need to find that? Obvious fact is that a disconnected MH reconnects after some time interval but a failed MH never reconnects automatically until a recovery operation is performed. Disconnection is of two types – i) planned disconnection: Most likely to last considerably longer than unplanned ones and less frequent. This is modeled using a uniform distribution with 180s minimum and 300s maximum disconnected time[16]. Mobile hosts save migration checkpoint before planned disconnection so that fault tolerance of the system does not get reduced. ii) unplanned disconnection considered here may be caused due to unwanted interference and loss of radio cover. Unplanned disconnection is less frequent and of longer duration. Unplanned disconnection time has a uniform distribution with 10 seconds minimum and 15 seconds maximum[16]. Unplanned disconnections and failure both are sudden events without any prior information. Then how to distinguish? In MSS a flag is set at the instant of sudden disconnection, time count starts in local timer. If MH reconnects within 15 seconds , the event is unplanned disconnection MH failure. MSS calls for recovery operation.

c) Migration Checkpoint: An MH saves a temporary checkpoint independently before planned disconnection and sends it to current MSS attached to disconnection message. This temporary checkpoint is termed as migration checkpoint. Planned disconnection duration is much longer, 180s -300s. During this period if any checkpoint request message comes from any other MH for the disconnected MH, the MSS with its migration checkpoint participates in checkpointing on behalf of the MH without delaying the process.



disconnects

reconnects

checkpoint request message

connected period

Disconnected period

6. ALGORITHM

/* MHs initiates checkpoint protocol, logs are saved in log files, updates dependent MH list if any execution message is received */

```
checkpoint initiation() {
H_count = T_count =Tw= 0;
D_flag = 0, HT = k;
if (mp ==“intercell || combi”) {
        if (MH sends me)
                log();
        else
                if(MH receives me)
                        update MHD_list();
                else

                        if (MH hand-offs)
                        {
                                hand_off();
                                H_count =H _count+1;
                        }
                        if (H_count > HT) {
```

```
                        checkpoint algorithm();
                        H_count = 0;
                        }
}

else
        if(MH moving "intracell" ){
                T_count++;
                if (MH sends m_e)
                        log();
                else
                        if(MH receives m_e)
                                update MH_D_list();
                        else
                                if(T_count > checkpoint interval){
                                        checkpoint algorithm();
                                        T_count = 0;
}

        }
```

/* receiver MH id of sent execution message, current checkpoint interval, sent execution message are saved in log file */

```
log(){
        FILE *fp;
        fp = fopen (log_msg, "w");
        fprintf (fp, "%d%d%s", MH_R_id, chkpt_intv, m_e);
        fclose(fp);
        }
```

/* updates list of dependent MHs*/

```
update MH_D_list(){
        for ( i=0; i<n_dep; i++){
                MH_D_list[ i ] =  sender MH_id;
```

/* coordinates checkpointing algorithm that includes hand-off , planned and unplanned disconnection, failure recovery of MH_s */

```
checkpoint algortihm()
{
        MH takes checkpoint;
        csn = csn+1;
        MH forwards checkpoint, csn, MH_id to CMSS;
        CoMSS=CMSS;
        CoMSS forwards chkpt_req to MH_Ds;
        if ((D_flag = =0) ∀ MH_D){
                save checkpoint() ;
                checkpoint coordination();
        }
        else
                if (p_dis = =1 )
                planned disconnection ();
                else
                        if ((T_disconnection ≤ T dis _ unplanned _ max ) = = TRUE){
```

```
                        disconnection = unplanned;
                        wait(); }
                else
                        failure recovery();

        }return;
/*Current MSS of initiator MH coordinated checkpointing*/
checkpoint coordination()
{
        MH_Ds forward acks to CoMSS;
        CoMSS forwards m_c to save permanent chkpt to MH_Ds;
        MH_Ds converts status 'temporary checkpoint' to 'permanent checkpoints';
        MH_Ds sends chkpt, csn  to CoMSS ;
        CoMSS saves checkpoint fps ;
        GCCS [k] = {csn_{p,q}, p = q = 0.....n
        return;
}
planned disconnection()
{
        MH saves m_checkpoint;
         forwards it attached with disconnection message to MSS_c ;
        return;

/* checkpoints are saved in a file */
save checkpoint()
{
        FILE *fp;
        fp = fopen (chkpt, "w");
        fprintf (fp, "%d%s%f%d", MH_id, status, data,chkpt_intv)
        fclose(fp);
}return;

/* MH moves one MSS to another MSS*/
hand-off()
{
        sends leave_msg, M_chkpt to CMSS;
        sends join_msg, CMSS_id to new MSS;
        CMSS = new MSS;
        Old MSS = CMSS;
        MH_list [j] = MH_id ; j=0.....n
}return;

/* MH recovers after failure */
failure Recovery()
{
        failed MH reconnects to any MSS arbitratrily ;
        That MSS broadcasts its through all MSSs;
        MSS that finds a match with past MH list,  with latest checkpoint of failed MH sends latest
        checkpoint to the MSS;
        MH starts execution from the state saved in latest checkpoint;
}return;
END
```

6.1. Correctness Proof

Theorem 1: Proposed Checkpointing protocol optimizes recovery cost and reduces probability of data loss.

Proof: The above theorem is divided into following two lemmas by which the theorem is proved.

Lemma 1: Proposed checkpointing algorithm optimizes recovery cost

Proof: If a MH fails, its recovery from latest saved checkpoint includes two cost components: searching cost of last saved checkpoint and transferring it to the MSS where the MH will recover . Hand-off based checkpointing reduces searching cost because number of MSSs to be searched for information recovery is fixed by threshold value of hand-off count.

Let, $h_T = k$

number of MSSs to be searched = k

Searching time of k number of MSSs = k unit

Recovery time is almost constant as data transfer through high speed wired network takes finite amount of time which is almost constant

Hence, recovery cost is linear over time irrespective of mobility rate of MH

Lemma 2: Checkpointing protocol reduces bulk amount of data loss of an MH moving in intracell movement pattern.

Proof: In case of an MH moving in intracell movement pattern hand-off count does not change but time count changes. Hence a time interval is calculated based on failure rate of a particular application or system where checkpointing algorithm is implemented.

Thus theorem1 is proved.

Theorem 2: The Proposed algorithm ensures consistent global checkpointing

lemma 1: No orphan or lost message is generated by the technique

Proof: Checkpointing algorithm involves only dependant MHs in a particular checkpoint interval. Moreover if any MH_D disconnects after a while of saving a checkpoint then possibility of orphan or lost message gets eliminated.

Lemma 2: Coordinator MSS of a checkpointing process does not fail. Hence failure recovery is absolutely possible as Coordinator MSS saves the set of global consistent checkpoint set.

Thus theorem2 is proved.

Theorem 3: Proposed algorithm is domino effect free

Proof: Mobile hosts save checkpoints based on hand-off or time interval. Coordinated checkpoint process is followed here and only dependent MHs during current checkpoint interval are forced to take checkpoints. Hence in case of failure only depending MHs are forced to rollback only upto latest saved checkpoints belong to latest global consistent checkpoint set.

Theorem 4: Checkpoint interval varies depending on movement pattern of MHs for constant mobility rate.

Lemma : An MH moving intercell initiates checkpointing faster than an MH moving in any other movement pattern.

Proof: Let m_o = 1 cell/unit time, $h_T = k$

if (mp = intercell)

$\{(h_{count} > h_T) = = True\}$ after (100 / intercell)*k unit time = (100/100)*k unit time = k unit time

Hence $MH_{(mp = intercell)}$ initiates checkpointing at an interval of k unit time

if (mp = combi && combi = 50-50)

$\{(h_{count} > h_T) = = True\}$ after (100 / intercell)*k unit time = (100/50)*k unit time = 2k unit time

This proves the Lemma and Theorem 4.

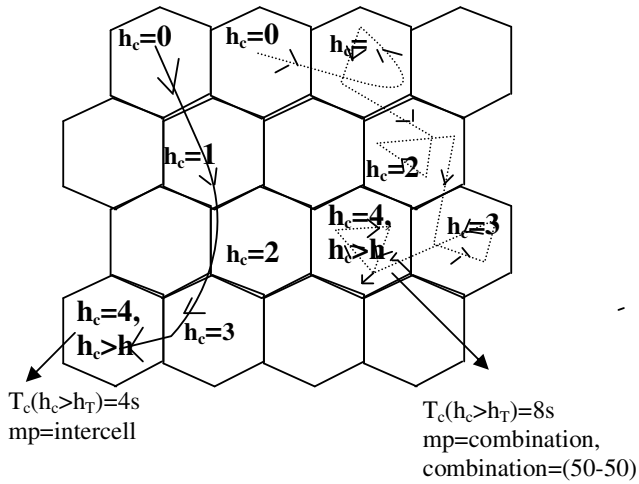


Figure 2 : Movement patterns of Mobile Hosts

7. PERFORMANCE ANALYSIS

7.1 Relation of Movement Pattern and Hand-off Threshold

We have observed that h_T is a function of the following parameters:
$h_T = f(m_o, mp, msg\_passing\ freq, failure\ rate)$
and hand-off is a function of :
$hand\text{-}off = f(m_o, mp)$
Let m_o = 1 cell / unit, hand-off rate = 1cell /unit, $h_T = k$, send_msg. freq. = 1/unit

if(mp = intercell)

($h_{count} > h_T$) is true after {(100/50)*k}unit = k unit =checkpoint interval
logs are scattered in k different MSSs.
msg_log(send) = k unit

if(mp = combi$_{(p\%,q\%)}$)

($h_{count} > h_T$) will be true after {(100/intercell)*h_T}unit = checkpoint interval

Case 1: **If (p=q=50%)**
Checkpoint interval=2 k unit
logs scattered in 2k different MSSs.
Number of logs = 2k

Case 2: **If (p=75%,q=25%)**
Checkpoint interval = 1.33k unit
logs scattered in 1.33k different MSSs.
Number of logs = 1.33k

Case 3: **If (p=25%, q=75%)**
Checkpoint interval= 4k unit
logs scattered in 4k different MSSs.
Number of logs = 4k

if(mp=intracell)

$$\text{checkpoint interval} \propto \frac{1}{failurerateofMH}$$

Relationship derived between movement pattern and hand-off threshold value helps to chose an optimum value of h_T so that checkpoint saved are not so close, not so far as well as recovery cost is minimized. Above relationship clearly shows that for different movement patterns of mobile hosts checkpoint interval will vary from k unit to 4k units and more. Hence k value can not be a high value. If mobility rate of mobile hosts is high, k should be of high value to reduce closeness of

checkpoints. If mobile hosts communicates with high frequency of message passing, k should be kept low and vice versa.
Performance analysis is done in the following system environment**.**
Let us define a mobile computing system with following specifications:
Probability that the application will fail is
$1-(1-\lambda R)^n$ …….(i)
Λ = failure rate, R = total execution time of an application without any fault
If a fault occurs, let total running time of an application = $R^{'}$
$R^{'} > R$
Let, checkpoint interval = c
In the system each two processors are grouped together
Hence, total n/2 number of buddies is there in the system
As in [5], the probability of an unrecoverable error during the execution is
$1-(1-\lambda^2 R^{'} C)^{n/2}$ …….(ii)
MTBF (M) = 20 years, n=5000, R = 400 hours, $\lambda = 1/M = 5.71 * 10^{-6}$ / hour
If $h_T = k$, C= k if MH is moving in intercell movement pattern
Probability of an unrecoverable error = $1-(1- \lambda^2*3R*k)$, mp = intercell
$= 1-(1- \lambda^2*3R*2k)$, mp = combination (50-50)
$=1-(1- \lambda^2*3R*1.33k)$, mp = combination (75-25)
$=1-(1- \lambda^2*3R*4k)$, mp = combination (25-75)
$= 1-(1- \lambda^2*3R*avg.k)$, mp = intracell

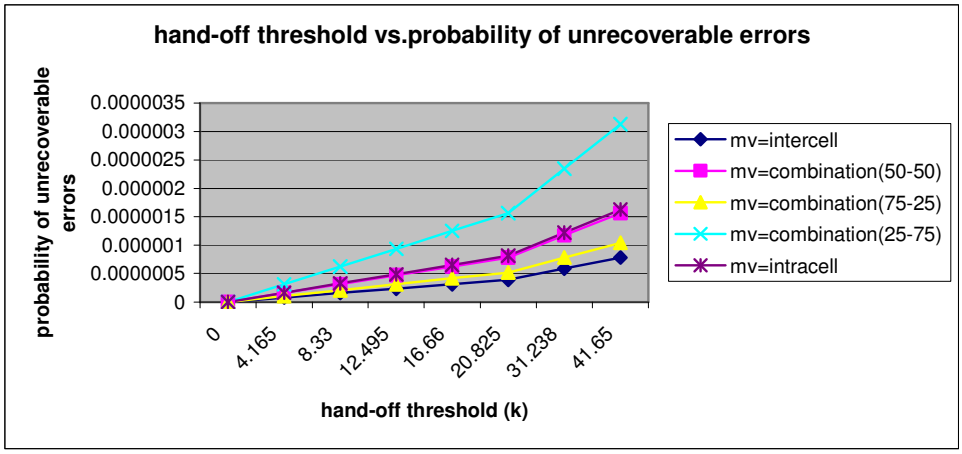


Figure 3 Probability of unrecoverable errors vs. hand-off threshold (k) for different movement patterns and constant mobility rate.

Probability of unrecoverable errors increases with hand-off threshold. For a constant value of hand-off threshold, probability of unrecoverable errors varies with different type of movement patterns. Probability of unrecoverable errors is proportional to checkpoint interval for a particular movement pattern. When hand-off threshold is constant, checkpoint interval of an MH moving in intercell movement pattern
is more than that of an MH moving in any type of combination movement pattern. Similarly for constant hand-off threshold value, checkpoint interval of an MH moving in combination movement pattern with higher '%' of intercell movement is more than that of an MH moving in any other type of combination movement pattern. For simplicity checkpoint interval of an MH moving in intracell movement pattern is chosen to be that of an MH moving in combination movement pattern with lowest '%' of intercell movement.

Our next study finds relation ship between total checkpointing time and hand-off threshold value.
Case1**:** MH undergoes planned disconnection, its MSSc participates in checkpointing on behalf of MH with its saved m_checkpoint. Hence checkpointing is transparent to planned disconnection of MHs.
Case 2: m_checkpoint not saved, checkpointing gets haulted from 180s-300s.
Total Checkpointing time (Tc)=d*m_checkpoint transfer time+ k*log retrieval time+disconnection time
d = number of planned disconnections during a checkpoint interval, assumed to be 1
k = number of logs
In ref. of [4], values of different components of Tc are taken as follows:
M_checkpoint transfer time = 0.08s
log retrieval time = 0.002s
disconnection time = 180s (min.) – 300s (max)
Case 1: Tc = k*.002s
Case 2: Tc(min) = 0.08+k*0.002+180
Tc(max) = 0.08+k*0.002+300

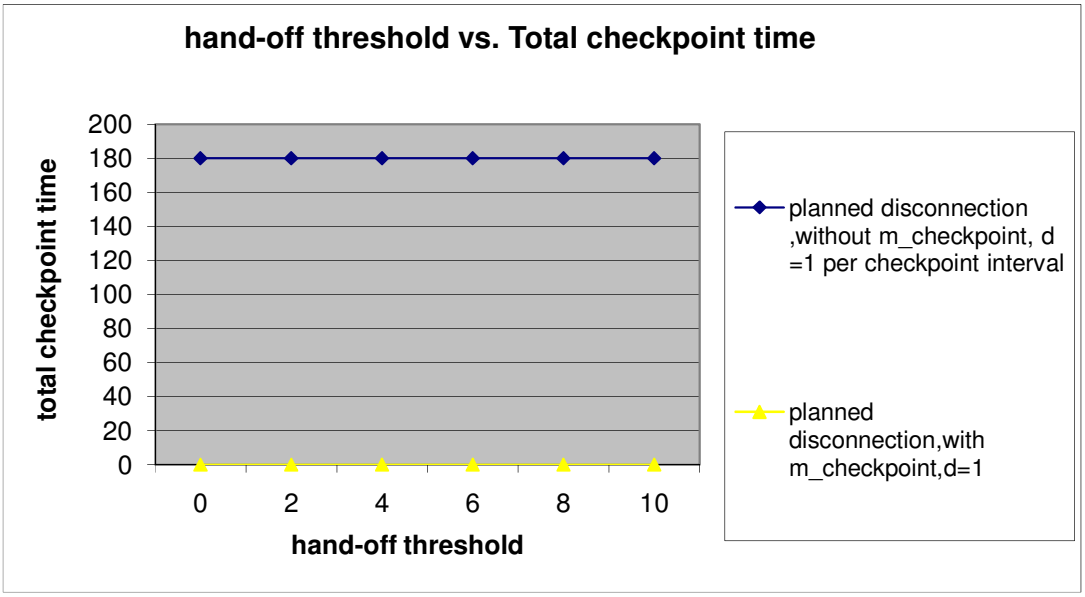


Figure 4: Total checkpoint time vs. hand-off threshold

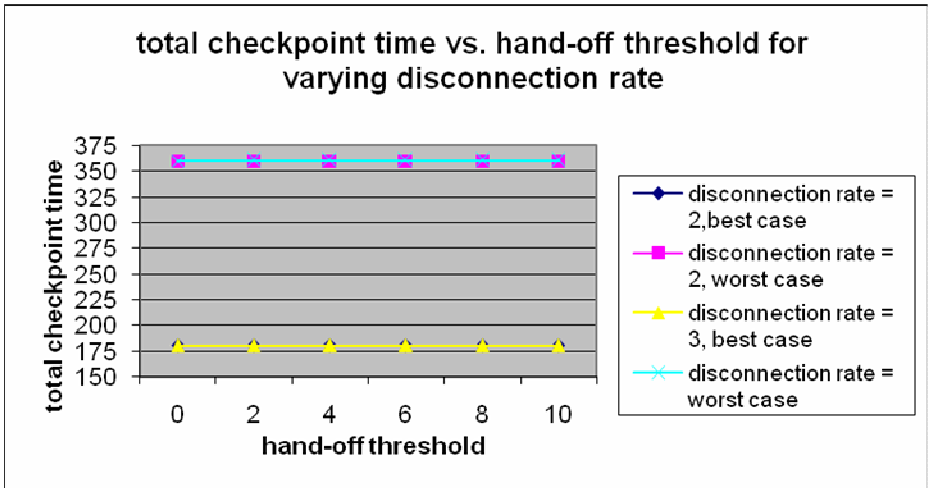


Figure 5: Total checkpointing time vs. hand-off threshold for disconnection rate more than one.

Best Case: more than one MH disconnects and reconnects at same time instant
Worst Case: MHs disconnects and reconnects one after another

Proposed algorithm ensures that MHs with any type of movement patterns, different mobility rates, different hand-off threshold values save checkpoint**.**
Fig. 1 proves this claim. In [4] checkpoint is saved only based of movement of MHs across cells. Hence MHs moving within a cell is not static but according, their work never will initiate checkpointing. Moreover in [4] all MHs save checkpoint independently. Hence no probability of getting a checkpoint request from any other MH but that is possible in our proposed scheme as coordinated checkpointing algorithm is chosen to save checkpoints.

In [4], recovery probability F_r = prob{T_r^i <= T} = $\sum_{k=1}^{2M+1} P_k\{1-e-\theta_k T\}$ ……(i)

For an MH moving in intracell movement pattern, M=0,

θ_k =0

putting these values in above equation , F_r = 0
Reason behind this is that this MH never saves checkpoint hence recovery probability upon failure is 0.
In proposed scheme, $\theta_k > 0$, $\forall$ MHs, as explained in sec. 7.1, hence $F_r \neq 0$.

Perfomance enhancement in terms of overhead opimizations:
i) **number of coordination messages (m_c) :**
Checkpoint initiator process forwards checkpoint request messages to only dependant MHs in current checkpoint interval.
ii) **MHs save migration checkpoint before planned disconnection only**
if hand-off threshold =k, mobility rate = 1 cell/unit time, checkpoint interval = k unit time. Hence number of unplanned disconnections due to hand-off during a checkpoint interval = k. If m_checkpoint is saved before all disconnections then overheads calculated in ref. to [2], are as follows:

a) k*.32unit time=0.32k unit time will be required to load these checkpoints through wireless channel in MSSs [2]
b) storage overhead = 1MB*k = k MB
c) Stable storage access = k times.
d) Poor bandwidth utilization

8. CONCLUSIONS

Desigining a fault tolerant system is always difficult. Fault tolerance using checkpoints in a Mobile Computing System imposes more challenges because of some unique characteristics of mobile hosts. Proposed checkpointing algorithm is a complete one in comparison with other relevant works because it is designed based not only on mobility and hand-off of MHs but movement patterns are also considered. Unike others, MHs moving within a cell is checkpointed exclusively. Hence, our checkpointing scheme is stronger from the point of view of failure recovery. Disconnection of MHs is a frequent phenomenon which may delay checkpointing. Hence the concept of migration checkpoint is introduced before planned disconnection so that checkpointing can be completed without any dealy resulting enhanced fault tolerance in the proposed scheme.

References

[1] Sunil K. Gupta, R.K.Chauhan, P. Kumar, (April 2008), "Backward Error Recovery Protocols in Distributed Mobile Systems: A Survey", Journal of Theoretical and Applied Information Technology, vol 4.

[2] C Men, Zhenoeng Xu, and D.Wang, (2007)"An Efficient Handoff Strategy for Mobile Computing Checkpoint System"EUC 2007, LNCS 4808, pp. 410–421.

[3] S Biswas Saha, S Neogy,(2007), "A Low Overhead Checkpointing Scheme for Mobile Computing Systems," , 15th International Conference on Advanced Computing and Communications, adcom, pp.700-705.

[4] Sapna E. George,Ing-Ray Chen,Ying Jin, (2006)"*Movement-Based Checkpointing and Logging for Recovery in Mobile Computing Systems", MobiDE,* 51-58

[5] Gengbin Zheng, Chao Huang, Laxmikant V. Kale, (April 2006)"Performance Evaluation of Automatic Checkpoint-based Fault Tolerance for AMPI and Cham++", ACM SIGOPS Operating Systems Review, Vol.40, Issue 2 , pages: 90-99, ISSN: 0163 - 5980

[6] F.Quaglia, B.Ciciani, R.Baldoni,(2006)"Checkpointing Protocols in Distributed Systems with Mobile Hosts: a Performance analysis",Workshop on Fault-Tolerant Parallel and Distributed Systems, pages 742-755.

[7] A.Agbaria, William H.sanders, (2004)"Distributed Snapshots for Mobile Computing systems", Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communicarions (Percom '04), pp.1-10.

[8] Ting – Yao Jiang, Qing – Hua Li,(2003) "An Efficient Recovery Scheme for Mobile Computing System"International Conference on Machine Learning and Cybernetics, vol.4, pages : 2031- 2036

[9] Taesoon Park, Namyoon Woo, Heon Y. Yeom,(2003) "An Efficient recovery scheme for fault-tolerant mobile computing systems", Future Generation Computer System, 19(1): 37-53

[10] G.Cao, M.Singhal,(Feb 2001), "Mutable Checkpoints: A New Checkpointing Approach for Mobile Computing Systems", IEEE Transactions on Parallel and Distributed system, vol.12, Issue 2, pages: 157-172, ISSN: 1045-9219

[11] E.N. (Mootaz) Elnozahy, L.Alvisi, Yi-min Wang and David B. Johnson, *(September 2002), "A Survey of Rollback Recovery Protocol in Message Passing System"ACM Comput. Surv., Vol. 34, No. 3. pp. 375-408.*

[12] Chen-Min Lin, Chyi-Ren Dow, (2001)"Efficient Checkpoint based Failure Recovery Techniques in Mobile Computing Systems", Journal of Information Science and Engineering vol.17, 549-573.

[13] R.C.Gass , B.Gupta, (2001)"An EfficientCheckpointing Scheme for Mobile Computing Systems", European Simulation Symposium, Oct 18-20, pp.1-6.

[14] H.Higaki, M.Takizawa, (1999)"Checkpoint-recovery protocol for Reliable Mobile Systems", Transactions of Information Proceeding Japan, vol.40, no.1, pp.236-244.

[15] B.Yao, Kuo-Feng Ssu, W.Kent Fuchs, (1999), "Message Logging in Mobile Computing", International Symposium on Fault-Tolerant Computing, pages 150-157.

[16] Djamel H.Sadok, Judith Kelner and Carlos de Maorais Cordeiro,(1999), "Disconnection protocol support in mobile access", Journal of the Brazilian Computer Society, vol.5, Print ISSN 0104 – 6500

[17] Guohong Cao, Mukesh Singhal,(1998) "Low-Cost Checkpointing with Mutable Checkpoints in Mobile Computing Systems", 18th International Conference on Distributed ComputingSystems, P.464, May 26-29.

[18] M.Satyanarayanan,(1996) *"Fundamental Challenges in Mobile Computing"*, PODC '96: Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing, pp. 1-7.

[19] R.Prakash, M.Singhal, (1996) "Low Cost Checkpointing and Failure Recovery in Mobile Computing Systems", IEEE Transacrions on Parallel and Distrinuted Systems, VOL. 7, NO. IO, OCTOBER 1996

[20] B.R.Badrinath, A.Acharya, (June 1994) T. liminski "Structuring Distributed Algorithm for Mobile Hosts", Proc. 14th Int.conf. Distributed Computing Systems.

[21] George H. Forman and John Zahorjan,(April,1994)"*The Challenges of Mobile Computing"*, IEEE Computer Journal, vol.27, no.4, pp.38-47,

[22] Arup Acharya, B.R.Badrinath, (October 1994),"Checkpointing distributed applications on mobile computers, Proceedings of the 3rd International Conference on parallel and distributed information systems, p.73-80 , Autin, Texas, United States.

[23] R.Koo, S.Toueg, (1987) "Checkpointing and Rollback-Recovery for Distributed Systems", IEEE Transactions on software Engineering, Vol.SE-13, No.1.

[24] K.M.Chandy, L.Lamport, (Feb 1985) "Distributed snapshots: Dtermining Global States of Distributed Systems", ACM Transactions Computer Systems, Vol.3, no.1, pp.63-75.

[25] *Mobile Communications(2nd Edition), by* Jochen Schiller, Publisher: Addison Wesley

Authors

Short Biography:

Suparna Biswas: She obtained her B.Tech in Electronics & Communication Engineering from University of Kalyani and M.E. in Software Engineering from Jadavpur University. She is a faculty in the Dept. of Computer Science & Engg., West Bengal University of Technology. Her areas of research interests are Fault Tolerant Mobile Computing, Software Engineering.

Sarmistha Neogy: Obtained her B. E. in Computer Science & Engg., M. E. in Computer Science & Engg. and Ph.D. (Engg.) from Jadavpur University and is currently teaching in the Dept. of Computer Science & Engg., Jadavpur University. Her research interests include Fault tolerance, Distributed and Mobile Computing, Security.