Experiment 8

Aim : Implementation of Deadlock Detection Algorithm

Theory:

A deadlock is a condition in a system where a set of processes (or threads) have requests for resources that can never be satisfied. Essentially, a process cannot proceed because it needs to obtain a resource held by another process but it itself is holding a resource that the other process needs. More formally, Coffman defined four conditions have to be met for a deadlock to occur in a system:

1. **Mutual exclusion** A resource can be held by at most one process.
2. **Hold and wait** Processes that already hold resources can wait for another resource.
3. **Non-preemption** A resource, once granted, cannot be taken away.
4. **Circular wait** Two or more processes are waiting for resources held by one of the other processes.

Chandy-Misra-Hass Detection Algorithm

This is considered an *edge-chasing*, *probe-based* algorithm.

It is also considered one of the best deadlock detection algorithms for distributed systems.

If a process makes a request for a resource which fails or times out, the process generates a *probe message* and sends it to each of the processes holding one or more of its requested resources.

Each probe message contains the following information:

- the *id* of the process that is blocked *(the one that initiates the probe message)*;
- the *id* of the process is sending this particular version of the probe message; and
- the *id* of the process that should receive this probe message.

When a process receives a probe message, it checks to see if it is also waiting for resources.

If not, it is currently using the needed resource and will eventually finish and release the resource.

If it *is* waiting for resources, it passes on the probe message to all processes it knows to be holding resources it has itself requested.

The process first modifies the probe message, changing the sender and receiver ids.

If a process receives a probe message that it recognizes as having initiated, it knows there is a cycle in the system and thus, **deadlock**.

ProgramsChandy-Misra-Hass Detection Algorithm

```
import java.util.Scanner;
public class DeadlockDetection
{
```

```
	static int n;
	static boolean nodes[][];
	public static void main(String[] args)
	{
		Scanner src = new Scanner(System.in);
		System.out.print("Enter The Number Of Nodes : ");
		n = src.nextInt();
		nodes = new boolean[n][n];
		System.out.println("Enter The Links (1 If Link Exists Else Any Number) : ");
		for (int i = 0; i < n; i++)
		{
			for (int j = 0; j < n; j++)
			{
				if (i != j)
				{
					System.out.print("\tLink " + (i + 1) + "-->" + (j + 1) + ": ");
					nodes[i][j] = src.nextInt() == 1;
				}
			}
		}
		int origin = 0, source = 0, dest = -1, i = 0, noProbes = -1, nextOrigin;
		System.out.println("\nGenerated Probes (Origin, Source, Destination) : ");
		do
		{
			noProbes++;
			System.out.println("-------------------------------------------------");
			System.out.println("Probe With Process P" + origin + " As Origin : ");
			nextOrigin = getNextNode(origin);
			for (i = 0; i < n - noProbes; i++)
			{
				dest = getNextNode(source);
				if (dest == -1)
					break;
				System.out.println("\tProbe (P" + (origin) + ", P" + source + ", P" +
dest + ")");
				source = dest;
				if (origin == dest)
					break;
			}
			if (i != n - noProbes && dest != -1)
			{
				System.out.println("\nDeadlock Detected");
				break;
			}
			else if (noProbes == n - 2)
			{
				System.out.println("\nNo Deadlock Detected");
				  break;
			}
			origin = nextOrigin;
			source = origin;
		}
```

```
                while (true);
                System.out.println("-----------------------------------------------");
        }
        private static int getNextNode(int rowNum)
        {
                for (int j = 0; j < n; j++)
                {
                        if (nodes[rowNum][j])
                                return j;
                }
                return -1;
        }
}

/*
Output :
1.
C:\Documents and Settings\Administrator\My Documents>javac DeadlockDetection.java
C:\Documents and Settings\Administrator\My Documents>java  DeadlockDetection
Enter The Number Of Nodes : 4
Enter The Links (1 If Link Exists Else Any Number) :
        Link 1-->2: 1
        Link 1-->3: 0
        Link 1-->4: 0
        Link 2-->1: 0
        Link 2-->3: 1
        Link 2-->4: 0
        Link 3-->1: 0
        Link 3-->2: 0
        Link 3-->4: 1
        Link 4-->1: 0
        Link 4-->2: 1
        Link 4-->3: 0

Generated Probes (Origin, Source, Destination) :
-----------------------------------------------
Probe With Process P0 As Origin :
        Probe (P0, P0, P1)
        Probe (P0, P1, P2)
        Probe (P0, P2, P3)
        Probe (P0, P3, P1)
-----------------------------------------------
Probe With Process P1 As Origin :
        Probe (P1, P1, P2)
        Probe (P1, P2, P3)
        Probe (P1, P3, P1)

Deadlock Detected
-----------------------------------------------

2.
C:\Documents and Settings\Administrator\My Documents>java DeadlockDetection
```

```
Enter The Number Of Nodes : 4
Enter The Links (1 If Link Exists Else Any Number) :
        Link 1-->2: 1
        Link 1-->3: 0
        Link 1-->4: 0
        Link 2-->1: 0
        Link 2-->3: 1
        Link 2-->4: 0
        Link 3-->1: 0
        Link 3-->2: 0
        Link 3-->4: 1
        Link 4-->1: 1
        Link 4-->2: 0
        Link 4-->3: 0

Generated Probes (Origin, Source, Destination) :
-------------------------------------------------
Probe With Process P0 As Origin :
        Probe (P0, P0, P1)
        Probe (P0, P1, P2)
        Probe (P0, P2, P3)
        Probe (P0, P3, P0)

Deadlock Detected
-------------------------------------------------
```

3.

```
C:\Documents and Settings\Administrator\My Documents>java DeadlockDetection
Enter The Number Of Nodes : 4
Enter The Links (1 If Link Exists Else Any Number) :
        Link 1-->2: 1
        Link 1-->3: 0
        Link 1-->4: 0
        Link 2-->1: 0
        Link 2-->3: 1
        Link 2-->4: 0
        Link 3-->1: 0
        Link 3-->2: 0
        Link 3-->4: 1
        Link 4-->1: 0
        Link 4-->2: 0
        Link 4-->3: 0

Generated Probes (Origin, Source, Destination) :
-------------------------------------------------
Probe With Process P0 As Origin :
        Probe (P0, P0, P1)
        Probe (P0, P1, P2)
        Probe (P0, P2, P3)
-------------------------------------------------
Probe With Process P1 As Origin :
        Probe (P1, P1, P2)
```

```
      Probe (P1, P2, P3)
-------------------------------------------------
Probe With Process P2 As Origin :
      Probe (P2, P2, P3)

No Deadlock Detected
-------------------------------------------------

*/
```