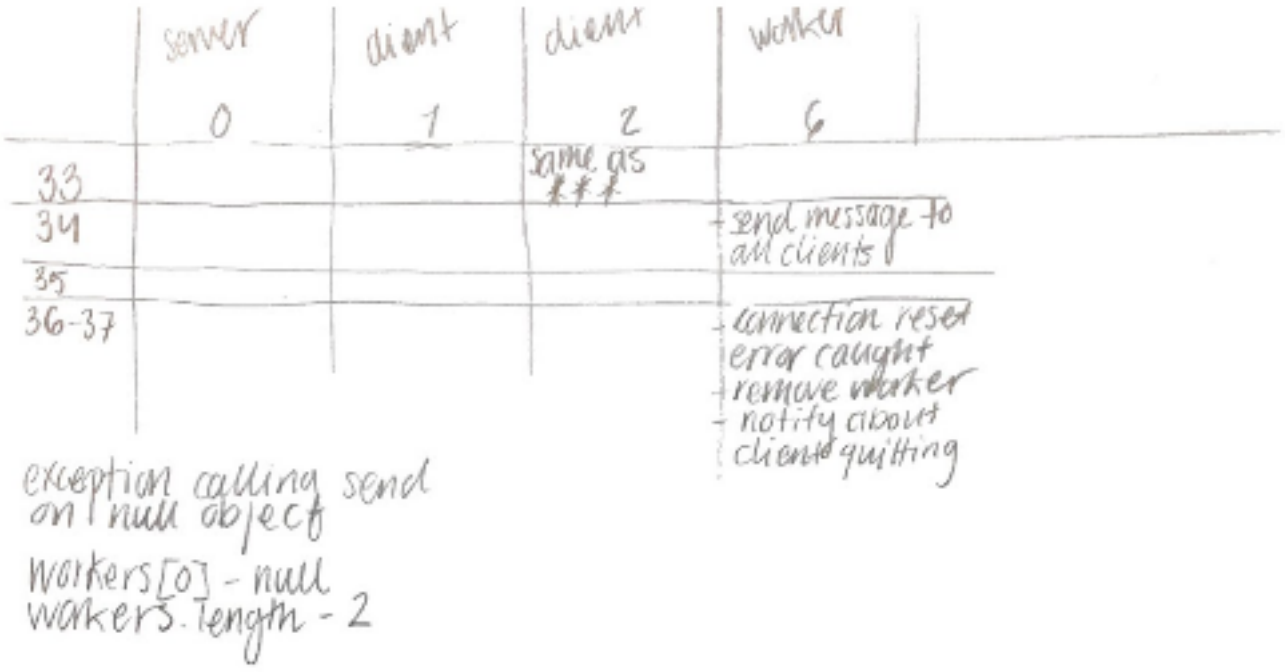
Chat Server

2.2 Diagnosis

Trace for the original Chat server:

	server 0	client 1	client 2	worker 6
0	- create chat server - server socket opened			
1		- create chat client w. id * - run exec which connects to the server		
2	- accept connection - create worker - start worker thread ** - open server socket (listen for next client)			
3				
4				
5		- report connection *** successful - create Input Stream Reader - -11- BufferedReader - -11- OutputStream Writer - send hello world to out - flush out - print recieved line from BufferedReader		
6			same as *	
7-8	same as **			
9-10	- close server socket - max connections - print about shutdown			
11-12				thread 3 - BreakGenerator nothing
13				- report about running - create printwriter
14-17				
18-24				- register worker - create BufferedReader - try to read from BufferedReader - send message to all clients
10, 22				
25-29				
30		- report message recieved - close OutputStream Writer		
31	nothing	nothing		



The explanation of how the failure happens is as follows:

- The ChatServer is created and accepts connections.
- A ChatClient is created and accepted by the ChatServer
- A designated worker to the ChatClient is created and started.
- The ChatServer goes back to accepting
- The worker does its work, eg:
    - Goes through the workers array, that now looks like [w1, null]
    - However, the field n of ChatServer is 1, and the last null is not touched
- Finally worker calls remove on its field chatServer to remove itself
- A new ChatClient is created and accepted by ChatServer
- A designated worker to that ChatClient is created and started.
- The server is now done and exits.
- When the new worker goes through the array when doing sendAll the array is [null, w2]
- This causes a NullPointerException - send is called on the first element, null.

The defect was fixed by exchanging the array of workers with a HashMap with worker-id (idx) as key and the worker as value. The fix can be seen here (no change to the ChatClient class):

```
public class ChatServer {
        static HashMap<Integer, Worker> workers = new HashMap<>();

        int n = 0;

        public ChatServer(int maxServ) {
        int port = 4444;
        boolean init = true;
        Socket sock;
        ServerSocket servsock = null;
        try {
                servsock = new ServerSocket(port);
```

```
                while (maxServ-- != 0) {
                sock = servsock.accept();
                Worker worker = null;
                try{
                        worker = new Worker(sock, this);
                        //if(Verify.getBoolean()) { throw new IOException("Simulated
exception"); }
                }catch(IOException e){
                        //assert(false);
                        init = false;
                }
                if(init){
                        assert(init);
                        new Thread(worker).start();
                }
                }
        servsock.close();
        } catch(IOException ioe) {
                System.err.println("Server: " + ioe);
        }
        System.out.println("Server shutting down.");
        }

        public static void main(String args[]) throws IOException {
    if (args.length == 0) {
                new ChatServer(-1);
    } else {
        new ChatServer(Integer.parseInt(args[0]));
        }
        }

        public synchronized void sendAll(String s) {
        for (Worker w: workers.values()) {
                w.send(s);
        }
        }

        public synchronized void remove(int i) {
        //workers[i] = null;
        workers.remove(i);
        sendAll("Client " + i + " quit.");
        }
        public synchronized void putWorker(int idx, Worker w){
        workers.put(idx, w);
        }
        public  synchronized Worker getWorker(int id){
        return workers.get(id);
        }

        public synchronized int getAndUpdateN(){
    n++;
    return n-1;
        }
}
```

3 Optional Tasks

3.1 Validation & Documentation

1. Our solution is likely correct because instead of removing workers from an array, (replacing them with null), we now remove the mapping in our HashMap. Workers are therefore removed and added in a thread safe way. When sendAll() iterates over workers.values() the function will never call send() on a null value.
2. We can never prove something to be correct, only to be false. There could still be some other issues in the overall system, like the server becoming overloaded if too many clients are started. Or the content of the messages received not matching exactly with the ones being sent.

3.2 Resource Initialization

We initialized the communications resources **in** and **out** of the worker threads in the constructor of Worker and threw a java.util.IOException if a resource could not be initialized. To simulate a case where some I/O resource cannot be initialized we injected an exception using Verify.getBoolean.

ChatServer.java line 21:

```
public Worker(Socket s, ChatServer cs) throws IOException{
        chatServer = cs;
        sock = s;

        if(Verify.getBoolean()) { throw new IOException("Simulated exception"); }
       out = new PrintWriter(sock.getOutputStream(), true);
       in = new BufferedReader(new InputStreamReader(sock.getInputStream()));

}
```

ChatServer.java line 70:

```
try{
        worker = new Worker(sock, this);
}catch(IOException e){
        assert(false);
       init = false;
}
if(init){
        assert(init);
       assert(false);
       new Thread(worker).start();
}
```

To show that our program handles the exception correctly we added assert(false) in the catch on row 79 which produced the trace below:

```
====================================================== system under test
ChatServer.main("2")+ChatClient.main()+ChatClient.main()

====================================================== search started: 5/3/19 3:31 PM

====================================================== error 1
gov.nasa.jpf.vm.NoUncaughtExceptionsProperty
```

```
java.lang.AssertionError
    at ChatServer.<init>(ChatServer.java:76)
    at ChatServer.main(ChatServer.java:96)


====================================================== trace #1
------------------------------------------------------ transition #0 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"ROOT" ,1/3,isCascaded:false}
        [3168 insn w/o sources]
  ChatServer.java:57            : public class ChatServer {
        [2 insn w/o sources]
  ChatServer.java:57            : public class ChatServer {
  ChatServer.java:58            : static HashMap<Integer, Worker> workers = new HashMap<>();
        [10 insn w/o sources]
  ChatServer.java:58            : static HashMap<Integer, Worker> workers = new HashMap<>();
  ChatServer.java:1             : import java.io.BufferedReader;
        [1 insn w/o sources]
  ChatServer.java:93            : if (args.length == 0) {
  ChatServer.java:96            : new ChatServer(Integer.parseInt(args[0]));
        [2 insn w/o sources]
  ChatServer.java:96            : new ChatServer(Integer.parseInt(args[0]));
  ChatServer.java:62            : public ChatServer(int maxServ) {
        [1 insn w/o sources]
  ChatServer.java:60            : int n = 0;
  ChatServer.java:63            : int port = 4444;
  ChatServer.java:64            : boolean init = true;
  ChatServer.java:66            : ServerSocket servsock = null;
  ChatServer.java:68            : servsock = new ServerSocket(port);
        [81 insn w/o sources]
  ChatServer.java:68            : servsock = new ServerSocket(port);
  ChatServer.java:69            : while (maxServ-- != 0) {
  ChatServer.java:70            : sock = servsock.accept();
        [141 insn w/o sources]
------------------------------------------------------ transition #1 thread: 1
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_ACCEPT" ,1/2,isCascaded:false}
        [3168 insn w/o sources]
  ChatClient.java:13            : static int currID = 0;
  ChatClient.java:1             : /* $Id: ChatClient.java 723 2009-09-24 07:48:58Z cartho $ */
        [1 insn w/o sources]
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:19            : public ChatClient() {
        [1 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
        [2 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
  ChatClient.java:21            : id = currID++;
  ChatClient.java:22            : }
  ChatClient.java:23            : }
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:27            : Socket socket = new Socket();
        [10 insn w/o sources]
  ChatClient.java:27            : Socket socket = new Socket();
        [116 insn w/o sources]
  ChatClient.java:27            : Socket socket = new Socket();
  ChatClient.java:28            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
        [20 insn w/o sources]
  ChatClient.java:28            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
  ChatClient.java:29            : socket.connect(addr);
        [23 insn w/o sources]
------------------------------------------------------ transition #2 thread: 0
nas.java.net.choice.NasThreadChoice {id:"CONNECT" ,1/3,isCascaded:false}
        [3 insn w/o sources]
```

```
  ChatServer.java:70            : sock = servsock.accept();
  ChatServer.java:71            : Worker worker = null;
  ChatServer.java:73            : worker = new Worker(sock, this);
        [1 insn w/o sources]
  ChatServer.java:10            : class Worker implements Runnable {
        [2 insn w/o sources]
  ChatServer.java:10            : class Worker implements Runnable {
  ChatServer.java:1             : import java.io.BufferedReader;
  ChatServer.java:73            : worker = new Worker(sock, this);
  ChatServer.java:17            : public Worker(Socket s, ChatServer cs) throws IOException{
        [1 insn w/o sources]
  ChatServer.java:18            : chatServer = cs;
  ChatServer.java:19            : sock = s;
  ChatServer.java:21            : if(Verify.getBoolean()) { throw new IOException("Simulated
exception"); }
        [1 insn w/o sources]
------------------------------------------------------ transition #3 thread: 0
gov.nasa.jpf.vm.BooleanChoiceGenerator[[id="verifyGetBoolean",isCascaded:false,{>true,false}]
        [2 insn w/o sources]
  ChatServer.java:21            : if(Verify.getBoolean()) { throw new IOException("Simulated
exception"); }
        [35 insn w/o sources]
  ChatServer.java:21            : if(Verify.getBoolean()) { throw new IOException("Simulated
exception"); }
  ChatServer.java:75            : }catch(IOException e){
  ChatServer.java:76            : assert(false);
        [24 insn w/o sources]

====================================================== snapshot #1
thread
java.lang.Thread:{id:0,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
  call stack:
    at ChatServer.<init>(ChatServer.java:76)
    at ChatServer.main(ChatServer.java:96)

thread
gov.nasa.jpf.FinalizerThread:{id:3,name:finalizer,status:WAITING,priority:5,isDaemon:false,lockCount:0
,suspendCount:0}
  waiting on: java.lang.Object@c7
  call stack:

thread
java.lang.Thread:{id:1,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
  call stack:
    at java.net.Socket.connect(Socket.java)
    at java.net.Socket.connect(Socket.java:77)
    at java.net.Socket.connect(Socket.java:70)
    at ChatClient.exec(ChatClient.java:29)
    at ChatClient.main(ChatClient.java:16)

thread
gov.nasa.jpf.FinalizerThread:{id:4,name:finalizer,status:WAITING,priority:5,isDaemon:false,lockCount:0
,suspendCount:0}
  waiting on: java.lang.Object@18c
  call stack:

thread
java.lang.Thread:{id:2,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
  call stack:
```

```
thread
gov.nasa.jpf.FinalizerThread:{id:5,name:finalizer,status:WAITING,priority:5,isDaemon:false,lockCount:0
,suspendCount:0}
  waiting on: java.lang.Object@251
  call stack:


====================================================== results
error #1: gov.nasa.jpf.vm.NoUncaughtExceptionsProperty "java.lang.AssertionError  at
ChatServer.<init>(Cha..."

====================================================== statistics
elapsed time:   00:00:00
states:         new=4,visited=0,backtracked=0,end=0
search:         maxDepth=4,constraints=0
choice generators:  thread=3 (signal=0,lock=1,sharedRef=0,threadApi=0,reschedule=0), data=1
heap:           new=1013,released=7,maxLive=973,gcCycles=3
instructions:   6935
max memory:     88MB
loaded code:    classes=77,methods=3617

====================================================== search finished: 5/3/19 3:31 PM
```

3.3 Thread Handling

Our method for a rejected execution prints to sys-out: “No more threads”.

```
public ChatServer(int maxServ) {
        int port = 4444;
        boolean init = true;
        Socket sock;
        ServerSocket servsock = null;
        BlockingQueue<Runnable> wQueue = new LinkedBlockingQueue();
        RejectedExecutionHandler rH = new RejectedExecutionHandler(){
        @Override
        public synchronized void rejectedExecution(Runnable r, ThreadPoolExecutor executor){
                System.out.println("No more threads");
        }
        };
        ThreadPoolExecutor executor = new ThreadPoolExecutor(10, 2, 10, TimeUnit.SECONDS,
wQueue, rH);

        try {
                servsock = new ServerSocket(port);
                while (maxServ-- != 0) {
                sock = servsock.accept();
                Worker worker = null;
                try{
                        worker = new Worker(sock, this);
                }catch(IOException e){
                        init = false;
                }
                if(init){
                        assert(init);
                        executor.execute(new Thread(worker));
                }
                }
```

We can see the sys-out print signifying the execution of the RejectedExecutionHandler here, taken from the trace where only one worker is accepted.

```
====================================================== search started: 4/30/19 10:09 AM
Thread running: Thread[main,5,main]
Adding 0 to workers
Registered worker 0.
Client 0 connected.
0: Received [0]0: Hello, world!
No more threads ← HERE
Server shutting down.
```

Our model class for the ThreadPoolExecutor:

```
public class ThreadPoolExecutor {
        private volatile int launchedThreads;
        private volatile int maximumPoolSize;
        private volatile RejectedExecutionHandler handler;
        public ThreadPoolExecutor(int corePoolSize, int maximumPoolSize,
                        long keepAliveTime, TimeUnit unit,
                        BlockingQueue<Runnable> workQueue,
                        RejectedExecutionHandler handler) {

        this.maximumPoolSize = maximumPoolSize;
        this.handler = handler;
        }

        public void shutdown() { } // stub

        public void execute(Runnable r) {
        if(launchedThreads<maximumPoolSize){
                new Thread(r).start();
                launchedThreads++;
        } else handler.rejectedExecution(r, this);
        }
}
```

We configured the ThreadExecutionHandler to accept 0,1 or 2 workers, by passing different values to the “maximumPoolSize” parameter in the constructor (highlighted in greed in the code excerpts). This lead to the following traces.

Accepting 0 workers:

```
====================================================== system under test
ChatServer.main("2")+ChatClient.main()+ChatClient.main()

====================================================== search started
No more threads
Client 0 connected.
No more threads
Server shutting down.
Client 0 connected.

====================================================== error 1
gov.nasa.jpf.vm.NotDeadlockedProperty
deadlock encountered:
  thread
java.lang.Thread:{id:1,name:main,status:WAITING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
  thread
java.lang.Thread:{id:2,name:main,status:WAITING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}


====================================================== trace #1
------------------------------------------------------ transition #0 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"ROOT" ,1/1,isCascaded:false}
```

```
        [3168 insn w/o sources]
  ChatServer.java:64            : public class ChatServer {
        [2 insn w/o sources]
  ChatServer.java:64            : public class ChatServer {
  ChatServer.java:65            : static HashMap<Integer, Worker> workers = new HashMap<>();
        [10 insn w/o sources]
  ChatServer.java:65            : static HashMap<Integer, Worker> workers = new HashMap<>();
  ChatServer.java:1             : import java.io.BufferedReader;
        [1 insn w/o sources]
  ChatServer.java:114           : if (args.length == 0) {
  ChatServer.java:117           : new ChatServer(Integer.parseInt(args[0]));
        [2 insn w/o sources]
  ChatServer.java:117           : new ChatServer(Integer.parseInt(args[0]));
  ChatServer.java:69            : public ChatServer(int maxServ) {
        [1 insn w/o sources]
  ChatServer.java:67            : int n = 0;
  ChatServer.java:70            : int port = 4444;
  ChatServer.java:71            : boolean init = true;
  ChatServer.java:73            : ServerSocket servsock = null;
  ChatServer.java:75            : BlockingQueue<Runnable> wQueue = new LinkedBlockingQueue();
        [207 insn w/o sources]
  ChatServer.java:75            : BlockingQueue<Runnable> wQueue = new LinkedBlockingQueue();
  ChatServer.java:76            : RejectedExecutionHandler rH = new RejectedExecutionHandler(){
        [1 insn w/o sources]
  ChatServer.java:76            : RejectedExecutionHandler rH = new RejectedExecutionHandler(){
  ChatServer.java:83            : ThreadPoolExecutor executor = new ThreadPoolExecutor(10, 0, 10,
TimeUnit.SECONDS, wQueue, rH);
        [258 insn w/o sources]
  ChatServer.java:83            : ThreadPoolExecutor executor = new ThreadPoolExecutor(10, 0, 10,
TimeUnit.SECONDS, wQueue, rH);
  env/java/util/concurrent/ThreadPoolExecutor.java:20 : RejectedExecutionHandler handler) {
        [1 insn w/o sources]
  env/java/util/concurrent/ThreadPoolExecutor.java:22 : this.maximumPoolSize = maximumPoolSize;
  env/java/util/concurrent/ThreadPoolExecutor.java:23 : this.handler = handler;
  env/java/util/concurrent/ThreadPoolExecutor.java:24 : }
  ChatServer.java:83            : ThreadPoolExecutor executor = new ThreadPoolExecutor(10, 0, 10,
TimeUnit.SECONDS, wQueue, rH);
  ChatServer.java:86            : servsock = new ServerSocket(port);
        [81 insn w/o sources]
  ChatServer.java:86            : servsock = new ServerSocket(port);
  ChatServer.java:87            : while (maxServ-- != 0) {
  ChatServer.java:88            : sock = servsock.accept();
        [141 insn w/o sources]
-------------------------------------------------- transition #1 thread: 1
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_ACCEPT" ,1/2,isCascaded:false}
        [3168 insn w/o sources]
  ChatClient.java:13            : static int currID = 0;
  ChatClient.java:1             : /* $Id: ChatClient.java 723 2009-09-24 07:48:58Z cartho $ */
        [1 insn w/o sources]
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:19            : public ChatClient() {
        [1 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
        [2 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
  ChatClient.java:21            : id = currID++;
  ChatClient.java:22            : }
  ChatClient.java:23            : }
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:27            : Socket socket = new Socket();
        [10 insn w/o sources]
  ChatClient.java:27            : Socket socket = new Socket();
        [116 insn w/o sources]
  ChatClient.java:27            : Socket socket = new Socket();
  ChatClient.java:28            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
        [20 insn w/o sources]
  ChatClient.java:28            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
  ChatClient.java:29            : socket.connect(addr);
        [23 insn w/o sources]
-------------------------------------------------- transition #2 thread: 0
nas.java.net.choice.NasThreadChoice {id:"CONNECT" ,1/3,isCascaded:false}
```

```
      [3 insn w/o sources]
  ChatServer.java:88            : sock = servsock.accept();
  ChatServer.java:89            : Worker worker = null;
  ChatServer.java:91            : worker = new Worker(sock, this);
      [1 insn w/o sources]
  ChatServer.java:17            : class Worker implements Runnable {
      [2 insn w/o sources]
  ChatServer.java:17            : class Worker implements Runnable {
  ChatServer.java:1             : import java.io.BufferedReader;
  ChatServer.java:91            : worker = new Worker(sock, this);
  ChatServer.java:24            : public Worker(Socket s, ChatServer cs) throws IOException{
      [1 insn w/o sources]
  ChatServer.java:25            : chatServer = cs;
  ChatServer.java:26            : sock = s;
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
      [29 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
      [270 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
  ChatServer.java:29            : in = new BufferedReader(new
      [7 insn w/o sources]
  ChatServer.java:29            : in = new BufferedReader(new
      [8 insn w/o sources]
  ChatServer.java:29            : in = new BufferedReader(new
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
      [29 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
      [23 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
      [48 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
  ChatServer.java:33            : }
  ChatServer.java:34            : }
  ChatServer.java:91            : worker = new Worker(sock, this);
  ChatServer.java:98            : }
  ChatServer.java:99            : if(init){
  ChatServer.java:100           : assert(init);
  ChatServer.java:102           : executor.execute(new Thread(worker));
      [145 insn w/o sources]
  ChatServer.java:102           : executor.execute(new Thread(worker));
  env/java/util/concurrent/ThreadPoolExecutor.java:29 : if(launchedThreads<maximumPoolSize){
  env/java/util/concurrent/ThreadPoolExecutor.java:32 : } else handler.rejectedExecution(r, this);
  ChatServer.java:80            : System.out.println("No more threads");
      [2 insn w/o sources]
  ChatServer.java:81            : }
  env/java/util/concurrent/ThreadPoolExecutor.java:33 : }
  ChatServer.java:104           : }
  ChatServer.java:87            : while (maxServ-- != 0) {
  ChatServer.java:88            : sock = servsock.accept();
      [130 insn w/o sources]
------------------------------------------------------ transition #3 thread: 1
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_ACCEPT" ,1/2,isCascaded:false}
      [4 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
      [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
      [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
      [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
      [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
      [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
      [2 insn w/o sources]
  ChatClient.java:31            : InputStreamReader istr =
      [8 insn w/o sources]
  ChatClient.java:31            : InputStreamReader istr =
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
      [29 insn w/o sources]
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
```

```
        [23 insn w/o sources]
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
        [7 insn w/o sources]
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
        [48 insn w/o sources]
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
  ChatClient.java:34            : OutputStreamWriter out =
        [8 insn w/o sources]
  ChatClient.java:34            : OutputStreamWriter out =
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
        [29 insn w/o sources]
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
        [17 insn w/o sources]
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [55 insn w/o sources]
  ChatClient.java:37            : out.flush();
        [1 insn w/o sources]
  ChatClient.java:38            : for (int i = 0; i < 1; i++) {
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [77 insn w/o sources]
------------------------------------------------------ transition #4 thread: 2
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_READ" ,1/1,isCascaded:false}
        [3168 insn w/o sources]
  ChatClient.java:13            : static int currID = 0;
  ChatClient.java:1             : /* $Id: ChatClient.java 723 2009-09-24 07:48:58Z cartho $ */
        [1 insn w/o sources]
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:19            : public ChatClient() {
        [1 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
        [2 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
  ChatClient.java:21            : id = currID++;
  ChatClient.java:22            : }
  ChatClient.java:23            : }
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:27            : Socket socket = new Socket();
        [10 insn w/o sources]
  ChatClient.java:27            : Socket socket = new Socket();
        [116 insn w/o sources]
  ChatClient.java:27            : Socket socket = new Socket();
  ChatClient.java:28            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
        [20 insn w/o sources]
  ChatClient.java:28            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
  ChatClient.java:29            : socket.connect(addr);
        [23 insn w/o sources]
------------------------------------------------------ transition #5 thread: 0
nas.java.net.choice.NasThreadChoice {id:"CONNECT" ,1/2,isCascaded:false}
        [3 insn w/o sources]
  ChatServer.java:88            : sock = servsock.accept();
  ChatServer.java:89            : Worker worker = null;
  ChatServer.java:91            : worker = new Worker(sock, this);
  ChatServer.java:24            : public Worker(Socket s, ChatServer cs) throws IOException{
        [1 insn w/o sources]
  ChatServer.java:25            : chatServer = cs;
```

```
  ChatServer.java:26            : sock = s;
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
        [29 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
        [255 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
  ChatServer.java:29            : in = new BufferedReader(new
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [29 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [23 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [48 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
  ChatServer.java:33            : }
  ChatServer.java:34            : }
  ChatServer.java:91            : worker = new Worker(sock, this);
  ChatServer.java:98            : }
  ChatServer.java:99            : if(init){
  ChatServer.java:100           : assert(init);
  ChatServer.java:102           : executor.execute(new Thread(worker));
        [145 insn w/o sources]
  ChatServer.java:102           : executor.execute(new Thread(worker));
  env/java/util/concurrent/ThreadPoolExecutor.java:29 : if(launchedThreads<maximumPoolSize){
  env/java/util/concurrent/ThreadPoolExecutor.java:32 : } else handler.rejectedExecution(r, this);
  ChatServer.java:80            : System.out.println("No more threads");
        [2 insn w/o sources]
  ChatServer.java:81            : }
  env/java/util/concurrent/ThreadPoolExecutor.java:33 : }
  ChatServer.java:104           : }
  ChatServer.java:87            : while (maxServ-- != 0) {
  ChatServer.java:105           : servsock.close();
        [2 insn w/o sources]
  ChatServer.java:108           : }
  ChatServer.java:109           : System.out.println("Server shutting down.");
        [2 insn w/o sources]
  ChatServer.java:110           : executor.shutdown();
  env/java/util/concurrent/ThreadPoolExecutor.java:26 : public void shutdown() { } // stub
  ChatServer.java:111           : }
  ChatServer.java:117           : new ChatServer(Integer.parseInt(args[0]));
  ChatServer.java:119           : }
  ChatServer.java:3             : import java.io.InputStreamReader;
------------------------------------------------------ transition #6 thread: 2
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"TERMINATE"
,1/1,isCascaded:false,attrs:[gov.nasa.jpf.vm.GlobalSchedulingPoint@5e57643e]}
        [4 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:31            : InputStreamReader istr =
        [8 insn w/o sources]
  ChatClient.java:31            : InputStreamReader istr =
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
        [29 insn w/o sources]
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
        [23 insn w/o sources]
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
        [7 insn w/o sources]
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
        [48 insn w/o sources]
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
```

```
  ChatClient.java:34            : OutputStreamWriter out =
        [8 insn w/o sources]
  ChatClient.java:34            : OutputStreamWriter out =
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
        [29 insn w/o sources]
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
        [17 insn w/o sources]
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [55 insn w/o sources]
  ChatClient.java:37            : out.flush();
        [1 insn w/o sources]
  ChatClient.java:38            : for (int i = 0; i < 1; i++) {
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [77 insn w/o sources]

====================================================== snapshot #1
thread
java.lang.Thread:{id:1,name:main,status:WAITING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
  owned locks:java.io.InputStreamReader@4ae,java.lang.Object@4ad
  waiting on: java.lang.Object@434
  call stack:
    at java.net.SocketInputStream.read(SocketInputStream.java)
    at java.io.InputStreamReader.read(InputStreamReader.java:93)
    at java.io.BufferedReader.fill(BufferedReader.java:161)
    at java.io.BufferedReader.readLine(BufferedReader.java:324)
    at java.io.BufferedReader.readLine(BufferedReader.java:389)
    at ChatClient.exec(ChatClient.java:39)
    at ChatClient.main(ChatClient.java:16)

thread
java.lang.Thread:{id:2,name:main,status:WAITING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
  owned locks:java.io.InputStreamReader@5b6,java.lang.Object@5b5
  waiting on: java.lang.Object@572
  call stack:
    at java.net.SocketInputStream.read(SocketInputStream.java)
    at java.io.InputStreamReader.read(InputStreamReader.java:93)
    at java.io.BufferedReader.fill(BufferedReader.java:161)
    at java.io.BufferedReader.readLine(BufferedReader.java:324)
    at java.io.BufferedReader.readLine(BufferedReader.java:389)
    at ChatClient.exec(ChatClient.java:39)
    at ChatClient.main(ChatClient.java:16)


====================================================== results
error #1: gov.nasa.jpf.vm.NotDeadlockedProperty "deadlock encountered:   thread
java.lang.Thread:{i..."

====================================================== statistics
elapsed time:   00:00:00
states:         new=7,visited=0,backtracked=0,end=1
search:         maxDepth=7,constraints=0
choice generators:  thread=7 (signal=0,lock=1,sharedRef=0,threadApi=0,reschedule=1), data=0
heap:           new=1488,released=108,maxLive=1389,gcCycles=7
instructions:   12904
max memory:     241MB
loaded code:    classes=115,methods=5072
```

```
====================================================== search finished
```

Accepting 1 worker

```
====================================================== system under test
ChatServer.main("2")+ChatClient.main()+ChatClient.main()

====================================================== search started
Thread running: Thread[main,5,main]
Adding 0 to workers
Registered worker 0.
Client 0 connected.
0: Received [0]0: Hello, world!
No more threads
Server shutting down.
Client 0 connected.

====================================================== error 1
gov.nasa.jpf.vm.NotDeadlockedProperty
deadlock encountered:
  thread
java.lang.Thread:{id:2,name:main,status:WAITING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}


====================================================== trace #1
------------------------------------------------------ transition #0 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"ROOT" ,1/1,isCascaded:false}
        [3168 insn w/o sources]
  ChatServer.java:64             : public class ChatServer {
        [2 insn w/o sources]
  ChatServer.java:64             : public class ChatServer {
  ChatServer.java:65             : static HashMap<Integer, Worker> workers = new HashMap<>();
        [10 insn w/o sources]
  ChatServer.java:65             : static HashMap<Integer, Worker> workers = new HashMap<>();
  ChatServer.java:1              : import java.io.BufferedReader;
        [1 insn w/o sources]
  ChatServer.java:114            : if (args.length == 0) {
  ChatServer.java:117            : new ChatServer(Integer.parseInt(args[0]));
        [2 insn w/o sources]
  ChatServer.java:117            : new ChatServer(Integer.parseInt(args[0]));
  ChatServer.java:69             : public ChatServer(int maxServ) {
        [1 insn w/o sources]
  ChatServer.java:67             : int n = 0;
  ChatServer.java:70             : int port = 4444;
  ChatServer.java:71             : boolean init = true;
  ChatServer.java:73             : ServerSocket servsock = null;
  ChatServer.java:75             : BlockingQueue<Runnable> wQueue = new LinkedBlockingQueue();
        [207 insn w/o sources]
  ChatServer.java:75             : BlockingQueue<Runnable> wQueue = new LinkedBlockingQueue();
  ChatServer.java:76             : RejectedExecutionHandler rH = new RejectedExecutionHandler(){
        [1 insn w/o sources]
  ChatServer.java:76             : RejectedExecutionHandler rH = new RejectedExecutionHandler(){
  ChatServer.java:83             : ThreadPoolExecutor executor = new ThreadPoolExecutor(10, 1, 10,
TimeUnit.SECONDS, wQueue, rH);
        [258 insn w/o sources]
  ChatServer.java:83             : ThreadPoolExecutor executor = new ThreadPoolExecutor(10, 1, 10,
TimeUnit.SECONDS, wQueue, rH);
  env/java/util/concurrent/ThreadPoolExecutor.java:20 : RejectedExecutionHandler handler) {
        [1 insn w/o sources]
  env/java/util/concurrent/ThreadPoolExecutor.java:22 : this.maximumPoolSize = maximumPoolSize;
  env/java/util/concurrent/ThreadPoolExecutor.java:23 : this.handler = handler;
  env/java/util/concurrent/ThreadPoolExecutor.java:24 : }
  ChatServer.java:83             : ThreadPoolExecutor executor = new ThreadPoolExecutor(10, 1, 10,
TimeUnit.SECONDS, wQueue, rH);
  ChatServer.java:86             : servsock = new ServerSocket(port);
        [81 insn w/o sources]
  ChatServer.java:86             : servsock = new ServerSocket(port);
  ChatServer.java:87             : while (maxServ-- != 0) {
  ChatServer.java:88             : sock = servsock.accept();
```

```
        [141 insn w/o sources]
------------------------------------------------------ transition #1 thread: 1
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_ACCEPT" ,1/2,isCascaded:false}
        [3168 insn w/o sources]
  ChatClient.java:13            : static int currID = 0;
  ChatClient.java:1             : /* $Id: ChatClient.java 723 2009-09-24 07:48:58Z cartho $ */
        [1 insn w/o sources]
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:19            : public ChatClient() {
        [1 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
        [2 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
  ChatClient.java:21            : id = currID++;
  ChatClient.java:22            : }
  ChatClient.java:23            : }
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:27            : Socket socket = new Socket();
        [10 insn w/o sources]
  ChatClient.java:27            : Socket socket = new Socket();
        [116 insn w/o sources]
  ChatClient.java:27            : Socket socket = new Socket();
  ChatClient.java:28            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
        [20 insn w/o sources]
  ChatClient.java:28            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
  ChatClient.java:29            : socket.connect(addr);
        [23 insn w/o sources]
------------------------------------------------------ transition #2 thread: 0
nas.java.net.choice.NasThreadChoice {id:"CONNECT" ,1/3,isCascaded:false}
        [3 insn w/o sources]
  ChatServer.java:88            : sock = servsock.accept();
  ChatServer.java:89            : Worker worker = null;
  ChatServer.java:91            : worker = new Worker(sock, this);
        [1 insn w/o sources]
  ChatServer.java:17            : class Worker implements Runnable {
        [2 insn w/o sources]
  ChatServer.java:17            : class Worker implements Runnable {
  ChatServer.java:1             : import java.io.BufferedReader;
  ChatServer.java:91            : worker = new Worker(sock, this);
  ChatServer.java:24            : public Worker(Socket s, ChatServer cs) throws IOException{
        [1 insn w/o sources]
  ChatServer.java:25            : chatServer = cs;
  ChatServer.java:26            : sock = s;
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
        [29 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
        [270 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
  ChatServer.java:29            : in = new BufferedReader(new
        [7 insn w/o sources]
  ChatServer.java:29            : in = new BufferedReader(new
        [8 insn w/o sources]
  ChatServer.java:29            : in = new BufferedReader(new
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [29 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [23 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [48 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
  ChatServer.java:33            : }
  ChatServer.java:34            : }
  ChatServer.java:91            : worker = new Worker(sock, this);
  ChatServer.java:98            : }
  ChatServer.java:99            : if(init){
  ChatServer.java:100           : assert(init);
  ChatServer.java:102           : executor.execute(new Thread(worker));
        [145 insn w/o sources]
  ChatServer.java:102           : executor.execute(new Thread(worker));
  env/java/util/concurrent/ThreadPoolExecutor.java:29 : if(launchedThreads<maximumPoolSize){
  env/java/util/concurrent/ThreadPoolExecutor.java:30 : r.run();
```

```
      [6 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
      [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
      [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
      [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
      [59 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
      [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
      [2 insn w/o sources]
  ChatServer.java:38            : idx = chatServer.getAndUpdateN();
  ChatServer.java:140           : n++;
  ChatServer.java:141           : return n-1;
  ChatServer.java:38            : idx = chatServer.getAndUpdateN();
  ChatServer.java:41            : assert(out != null);
  ChatServer.java:42            : assert(chatServer.getWorker(idx) == null);
  ChatServer.java:136           : return workers.get(id);
      [2 insn w/o sources]
  ChatServer.java:136           : return workers.get(id);
      [35 insn w/o sources]
  ChatServer.java:136           : return workers.get(id);
  ChatServer.java:42            : assert(chatServer.getWorker(idx) == null);
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
      [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
      [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
      [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
      [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
      [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
      [2 insn w/o sources]
  ChatServer.java:44            : chatServer.putWorker(idx, this);
  ChatServer.java:133           : workers.put(idx, w);
      [2 insn w/o sources]
  ChatServer.java:133           : workers.put(idx, w);
      [142 insn w/o sources]
  ChatServer.java:133           : workers.put(idx, w);
  ChatServer.java:134           : }
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
      [2 insn w/o sources]
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
      [2 insn w/o sources]
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
      [2 insn w/o sources]
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
      [2 insn w/o sources]
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
      [2 insn w/o sources]
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
      [2 insn w/o sources]
  ChatServer.java:47            : String s = null;
  ChatServer.java:48            : while ((s = in.readLine()) != null) {
      [77 insn w/o sources]
---------------------------------------------------- transition #3 thread: 1
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_READ" ,1/2,isCascaded:false}
      [4 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
      [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
      [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
      [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
      [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
```

```
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:31            : InputStreamReader istr =
        [8 insn w/o sources]
  ChatClient.java:31            : InputStreamReader istr =
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
        [29 insn w/o sources]
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
        [23 insn w/o sources]
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
        [7 insn w/o sources]
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
        [48 insn w/o sources]
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
  ChatClient.java:34            : OutputStreamWriter out =
        [8 insn w/o sources]
  ChatClient.java:34            : OutputStreamWriter out =
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
        [29 insn w/o sources]
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
        [17 insn w/o sources]
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [41 insn w/o sources]
------------------------------------------------------ transition #4 thread: 0
nas.java.net.choice.NasThreadChoice {id:"WRITE" ,1/3,isCascaded:false}
        [414 insn w/o sources]
  ChatServer.java:48            : while ((s = in.readLine()) != null) {
  ChatServer.java:49            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:49            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:49            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:49            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:49            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:49            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:49            : chatServer.sendAll("[" + idx + "]" + s);
  ChatServer.java:122           : for (Worker w: workers.values()) {
        [25 insn w/o sources]
  ChatServer.java:122           : for (Worker w: workers.values()) {
        [59 insn w/o sources]
  ChatServer.java:122           : for (Worker w: workers.values()) {
        [6 insn w/o sources]
  ChatServer.java:122           : for (Worker w: workers.values()) {
        [307 insn w/o sources]
  ChatServer.java:122           : for (Worker w: workers.values()) {
  ChatServer.java:123           : w.send(s);
  ChatServer.java:60            : out.println(s);
        [324 insn w/o sources]
  ChatServer.java:61            : }
  ChatServer.java:124           : }
  ChatServer.java:122           : for (Worker w: workers.values()) {
        [5 insn w/o sources]
  ChatServer.java:122           : for (Worker w: workers.values()) {
  ChatServer.java:125           : }
  ChatServer.java:49            : chatServer.sendAll("[" + idx + "]" + s);
  ChatServer.java:48            : while ((s = in.readLine()) != null) {
```

```
        [77 insn w/o sources]
------------------------------------------------------ transition #5 thread: 1
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_READ" ,1/2,isCascaded:false}
        [15 insn w/o sources]
  ChatClient.java:37            : out.flush();
        [1 insn w/o sources]
  ChatClient.java:38            : for (int i = 0; i < 1; i++) {
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [541 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:38            : for (int i = 0; i < 1; i++) {
  ChatClient.java:41            : out.close();
        [13 insn w/o sources]
------------------------------------------------------ transition #6 thread: 1
nas.java.net.choice.NasThreadChoice {id:"SOCKET_CLOSE" ,1/2,isCascaded:false}
        [4 insn w/o sources]
  ChatClient.java:44            : }
  ChatClient.java:45            : }
  ChatClient.java:3             : import java.io.BufferedReader;
------------------------------------------------------ transition #7 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"TERMINATE"
,1/2,isCascaded:false,attrs:[gov.nasa.jpf.vm.GlobalSchedulingPoint@140e5a13]}
        [30 insn w/o sources]
  ChatServer.java:48            : while ((s = in.readLine()) != null) {
  ChatServer.java:51            : sock.close();
        [1 insn w/o sources]
------------------------------------------------------ transition #8 thread: 0
nas.java.net.choice.NasThreadChoice {id:"SOCKET_CLOSE" ,1/2,isCascaded:false}
        [2 insn w/o sources]
  ChatServer.java:55            : chatServer.remove(idx);
  ChatServer.java:129           : workers.remove(i);
        [2 insn w/o sources]
  ChatServer.java:129           : workers.remove(i);
        [102 insn w/o sources]
  ChatServer.java:129           : workers.remove(i);
  ChatServer.java:130           : sendAll("Client " + i + " quit.");
        [2 insn w/o sources]
  ChatServer.java:130           : sendAll("Client " + i + " quit.");
        [2 insn w/o sources]
  ChatServer.java:130           : sendAll("Client " + i + " quit.");
        [2 insn w/o sources]
  ChatServer.java:130           : sendAll("Client " + i + " quit.");
        [2 insn w/o sources]
  ChatServer.java:130           : sendAll("Client " + i + " quit.");
        [2 insn w/o sources]
  ChatServer.java:130           : sendAll("Client " + i + " quit.");
  ChatServer.java:122           : for (Worker w: workers.values()) {
        [7 insn w/o sources]
  ChatServer.java:122           : for (Worker w: workers.values()) {
        [41 insn w/o sources]
  ChatServer.java:122           : for (Worker w: workers.values()) {
        [5 insn w/o sources]
  ChatServer.java:122           : for (Worker w: workers.values()) {
  ChatServer.java:125           : }
  ChatServer.java:56            : }
  ChatServer.java:57            : }
        [1 insn w/o sources]
  env/java/util/concurrent/ThreadPoolExecutor.java:31 : launchedThreads++;
  env/java/util/concurrent/ThreadPoolExecutor.java:33 : }
  ChatServer.java:104           : }
```

```
  ChatServer.java:87            : while (maxServ-- != 0) {
  ChatServer.java:88            : sock = servsock.accept();
        [130 insn w/o sources]
------------------------------------------------------ transition #9 thread: 2
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_ACCEPT" ,1/1,isCascaded:false}
        [3168 insn w/o sources]
  ChatClient.java:13            : static int currID = 0;
  ChatClient.java:1             : /* $Id: ChatClient.java 723 2009-09-24 07:48:58Z cartho $ */
        [1 insn w/o sources]
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:19            : public ChatClient() {
        [1 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
        [2 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
  ChatClient.java:21            : id = currID++;
  ChatClient.java:22            : }
  ChatClient.java:23            : }
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:27            : Socket socket = new Socket();
        [10 insn w/o sources]
  ChatClient.java:27            : Socket socket = new Socket();
        [116 insn w/o sources]
  ChatClient.java:27            : Socket socket = new Socket();
  ChatClient.java:28            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
        [20 insn w/o sources]
  ChatClient.java:28            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
  ChatClient.java:29            : socket.connect(addr);
        [23 insn w/o sources]
------------------------------------------------------ transition #10 thread: 0
nas.java.net.choice.NasThreadChoice {id:"CONNECT" ,1/2,isCascaded:false}
        [3 insn w/o sources]
  ChatServer.java:88            : sock = servsock.accept();
  ChatServer.java:89            : Worker worker = null;
  ChatServer.java:91            : worker = new Worker(sock, this);
  ChatServer.java:24            : public Worker(Socket s, ChatServer cs) throws IOException{
        [1 insn w/o sources]
  ChatServer.java:25            : chatServer = cs;
  ChatServer.java:26            : sock = s;
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
        [29 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
        [255 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
  ChatServer.java:29            : in = new BufferedReader(new
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [29 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [23 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [48 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
  ChatServer.java:33            : }
  ChatServer.java:34            : }
  ChatServer.java:91            : worker = new Worker(sock, this);
  ChatServer.java:98            : }
  ChatServer.java:99            : if(init){
  ChatServer.java:100           : assert(init);
  ChatServer.java:102           : executor.execute(new Thread(worker));
        [145 insn w/o sources]
  ChatServer.java:102           : executor.execute(new Thread(worker));
  env/java/util/concurrent/ThreadPoolExecutor.java:29 : if(launchedThreads<maximumPoolSize){
  env/java/util/concurrent/ThreadPoolExecutor.java:32 : } else handler.rejectedExecution(r, this);
  ChatServer.java:80            : System.out.println("No more threads");
        [2 insn w/o sources]
  ChatServer.java:81            : }
  env/java/util/concurrent/ThreadPoolExecutor.java:33 : }
  ChatServer.java:104           : }
  ChatServer.java:87            : while (maxServ-- != 0) {
  ChatServer.java:105           : servsock.close();
        [2 insn w/o sources]
```

```
  ChatServer.java:108           : }
  ChatServer.java:109           : System.out.println("Server shutting down.");
        [2 insn w/o sources]
  ChatServer.java:110           : executor.shutdown();
  env/java/util/concurrent/ThreadPoolExecutor.java:26 : public void shutdown() { } // stub
  ChatServer.java:111           : }
  ChatServer.java:117           : new ChatServer(Integer.parseInt(args[0]));
  ChatServer.java:119           : }
  ChatServer.java:3             : import java.io.InputStreamReader;
------------------------------------------------------ transition #11 thread: 2
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"TERMINATE"
,1/1,isCascaded:false,attrs:[gov.nasa.jpf.vm.GlobalSchedulingPoint@140e5a13]}
        [4 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:30            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:31            : InputStreamReader istr =
        [8 insn w/o sources]
  ChatClient.java:31            : InputStreamReader istr =
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
        [29 insn w/o sources]
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
        [23 insn w/o sources]
  ChatClient.java:32            : new InputStreamReader(socket.getInputStream());
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
        [7 insn w/o sources]
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
        [48 insn w/o sources]
  ChatClient.java:33            : BufferedReader in = new BufferedReader(istr);
  ChatClient.java:34            : OutputStreamWriter out =
        [8 insn w/o sources]
  ChatClient.java:34            : OutputStreamWriter out =
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
        [29 insn w/o sources]
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
        [17 insn w/o sources]
  ChatClient.java:35            : new OutputStreamWriter(socket.getOutputStream());
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:36            : out.write(id + ": Hello, world!\n");
        [55 insn w/o sources]
  ChatClient.java:37            : out.flush();
        [1 insn w/o sources]
  ChatClient.java:38            : for (int i = 0; i < 1; i++) {
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [2 insn w/o sources]
  ChatClient.java:39            : System.out.println(id + ": Received " + in.readLine());
        [77 insn w/o sources]

====================================================== snapshot #1
thread
java.lang.Thread:{id:2,name:main,status:WAITING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
```

```
  owned locks:java.io.InputStreamReader@70c,java.lang.Object@70b
  waiting on: java.lang.Object@6c6
  call stack:
    at java.net.SocketInputStream.read(SocketInputStream.java)
    at java.io.InputStreamReader.read(InputStreamReader.java:93)
    at java.io.BufferedReader.fill(BufferedReader.java:161)
    at java.io.BufferedReader.readLine(BufferedReader.java:324)
    at java.io.BufferedReader.readLine(BufferedReader.java:389)
    at ChatClient.exec(ChatClient.java:39)
    at ChatClient.main(ChatClient.java:16)


====================================================== results
error #1: gov.nasa.jpf.vm.NotDeadlockedProperty "deadlock encountered:   thread
java.lang.Thread:{i..."

====================================================== statistics
elapsed time:   00:00:00
states:         new=12,visited=0,backtracked=0,end=1
search:         maxDepth=12,constraints=0
choice generators:  thread=12 (signal=0,lock=1,sharedRef=0,threadApi=0,reschedule=2), data=0
heap:           new=1830,released=175,maxLive=1662,gcCycles=12
instructions:   15366
max memory:     241MB
loaded code: classes=122,methods=5114
```

Accepting 2 workers

```
====================================================== system under test
ChatServer.main("2")+ChatClient.main()+ChatClient.main()

====================================================== search started
Thread running: Thread[main,5,main]
Adding 0 to workers
Registered worker 0.
Client 0 connected.
0: Received [0]0: Hello, world!
Thread running: Thread[main,5,main]
Adding 1 to workers
Registered worker 1.
Client 0 connected.
0: Received [1]0: Hello, world!
Server shutting down.
0: Received [1]0: Hello, world!
0: Received [1]0: Hello, world!
Worker thread 1: java.net.SocketException: Connection reset
Server shutting down.
Client 0 connected.
Thread running: Thread[main,5,main]
Adding 1 to workers
Registered worker 1.
0: Received [0]0: Hello, world!
0: Received [0]0: Hello, world!
0: Received [0]0: Hello, world!
0: Received [0]0: Hello, world!
Worker thread 0: java.net.SocketException: Connection reset
Thread running: Thread[main,5,main]
Adding 1 to workers
Registered worker 1.
Client 0 connected.
0: Received [1]0: Hello, world!
Server shutting down.
0: Received [1]0: Hello, world!
0: Received [1]0: Hello, world!
Worker thread 1: java.net.SocketException: Connection reset
Server shutting down.
Client 0 connected.
Thread running: Thread[main,5,main]
Adding 1 to workers
Registered worker 1.
```

```
Worker thread 0: java.net.SocketException: Connection reset
0: Received [0]0: Hello, world!
Client 0 connected.
Client 0 connected.
Thread running: Thread[main,5,main]
Adding 0 to workers
Registered worker 0.
Thread running: Thread[main,5,main]
Adding 0 to workers
Registered worker 0.
Thread running: Thread[main,5,main]
Adding 0 to workers
Registered worker 0.
Client 0 connected.
Client 0 connected.
Thread running: Thread[main,5,main]
Adding 0 to workers
Registered worker 0.
Client 0 connected.
0: Received [0]0: Hello, world!
Thread running: Thread[main,5,main]
Adding 1 to workers
Registered worker 1.
Client 0 connected.
0: Received [1]0: Hello, world!
0: Received [1]0: Hello, world!
0: Received [1]0: Hello, world!
Client 0 connected.
Thread running: Thread[main,5,main]
Adding 1 to workers
Registered worker 1.
0: Received [0]0: Hello, world!
0: Received [0]0: Hello, world!
Worker thread 0: java.net.SocketException: Connection reset
Thread running: Thread[main,5,main]
Adding 1 to workers
Registered worker 1.
Client 0 connected.
0: Received [1]0: Hello, world!
0: Received [1]0: Hello, world!
0: Received [1]0: Hello, world!
Client 0 connected.
Thread running: Thread[main,5,main]
Adding 1 to workers
Registered worker 1.
Client 0 connected.
0: Received [0]0: Hello, world!
0: Received [0]0: Hello, world!
0: Received [0]0: Hello, world!
Worker thread 0: java.net.SocketException: Connection reset
Thread running: Thread[main,5,main]
Adding 0 to workers
Registered worker 0.
Client 0 connected.
Thread running: Thread[main,5,main]
Adding 0 to workers
Registered worker 0.
Client 0 connected.
Thread running: Thread[main,5,main]
Adding 0 to workers
Registered worker 0.

====================================================== results
no errors detected

====================================================== statistics
elapsed time:   00:00:01
states:         new=108,visited=60,backtracked=168,end=4
search:         maxDepth=17,constraints=0
choice generators:  thread=107 (signal=0,lock=1,sharedRef=0,threadApi=0,reschedule=10), data=0
heap:           new=12126,released=1871,maxLive=1740,gcCycles=168
```

```
instructions:    186626
max memory:      241MB
loaded code:     classes=126,methods=5143
```

3.4 Client-side verification

To check that the messages received in ChatClient conform to expectations we added and amended the following assert:

```
assert recieved.contains("Hello,") || recieved.contains("world") : "recieved was: " +
recieved;
```

The original property violation follows below. A client can receive more messages than this, we need to add more checks in the assertion. To amend the assertion we also added that “Client quit” can be a message recieved.

```
assert recieved.contains("Hello,") || recieved.contains("world") ||
recieved.contains("Client")||recieved.contains("quit") : "recieved was: " + recieved;
```

Trace:

```
====================================================== system under test
ChatServer.main("2")+ChatClient.main()+ChatClient.main()

====================================================== search started
id before0
currID 1
id after1
Client 1 connected.
id before0
currID 1
id after1
Server shutting down.
Thread running: Thread[Thread-2,5,main]
Adding 0 to workers
Registered worker 0.
1: Received [0]1: Hello, world!
Client 1 connected.
Worker thread 0: java.net.SocketException: Connection reset
Thread running: Thread[Thread-4,5,main]
Adding 1 to workers
Registered worker 1.
1: Received [1]1: Hello, world!
Worker thread 1: java.net.SocketException: Connection reset
1: Received [1]1: Hello, world!
Thread running: Thread[Thread-4,5,main]
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
```

```
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
Registered worker 1.
Registered worker 1.
Registered worker 1.
Registered worker 1.
Registered worker 1.
Registered worker 1.
Registered worker 1.
Registered worker 1.
Registered worker 1.
Registered worker 1.
1: Received [1]1: Hello, world!
Worker thread 1: java.net.SocketException: Connection reset
Worker thread 1: java.net.SocketException: Connection reset
Worker thread 1: java.net.SocketException: Connection reset
Worker thread 1: java.net.SocketException: Connection reset
1: Received [1]1: Hello, world!
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
Adding 1 to workers
Registered worker 1.
1: Received Client 0 quit.

====================================================== error 1
gov.nasa.jpf.vm.NoUncaughtExceptionsProperty
java.lang.AssertionError: recieved was: Client 0 quit.
    at ChatClient.exec(ChatClient.java:45)
    at ChatClient.main(ChatClient.java:16)


====================================================== trace #1
------------------------------------------------------ transition #0 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"ROOT" ,1/1,isCascaded:false}
        [3168 insn w/o sources]
  ChatServer.java:65            : public class ChatServer {
        [2 insn w/o sources]
  ChatServer.java:65            : public class ChatServer {
  ChatServer.java:66            : static HashMap<Integer, Worker> workers = new HashMap<>();
        [10 insn w/o sources]
  ChatServer.java:66            : static HashMap<Integer, Worker> workers = new HashMap<>();
  ChatServer.java:1             : import java.io.BufferedReader;
        [1 insn w/o sources]
  ChatServer.java:108           : if (args.length == 0) {
  ChatServer.java:111           : new ChatServer(Integer.parseInt(args[0]));
        [2 insn w/o sources]
  ChatServer.java:111           : new ChatServer(Integer.parseInt(args[0]));
  ChatServer.java:70            : public ChatServer(int maxServ) {
        [1 insn w/o sources]
  ChatServer.java:68            : int n = 0;
  ChatServer.java:71            : int port = 4444;
  ChatServer.java:72            : boolean init = true;
  ChatServer.java:74            : ServerSocket servsock = null;
  ChatServer.java:75            : BlockingQueue<Runnable> wQueue = new LinkedBlockingQueue();
        [207 insn w/o sources]
  ChatServer.java:75            : BlockingQueue<Runnable> wQueue = new LinkedBlockingQueue();
```

```
  ChatServer.java:76            : RejectedExecutionHandler rH = new RejectedExecutionHandler(){
        [1 insn w/o sources]
  ChatServer.java:76            : RejectedExecutionHandler rH = new RejectedExecutionHandler(){
  ChatServer.java:82            : ThreadPoolExecutor executor = new ThreadPoolExecutor(10, 2, 10,
TimeUnit.SECONDS, wQueue, rH);
        [258 insn w/o sources]
  ChatServer.java:82            : ThreadPoolExecutor executor = new ThreadPoolExecutor(10, 2, 10,
TimeUnit.SECONDS, wQueue, rH);
  env/java/util/concurrent/ThreadPoolExecutor.java:20 : RejectedExecutionHandler handler) {
        [1 insn w/o sources]
  env/java/util/concurrent/ThreadPoolExecutor.java:22 : this.maximumPoolSize = maximumPoolSize;
  env/java/util/concurrent/ThreadPoolExecutor.java:23 : this.handler = handler;
  env/java/util/concurrent/ThreadPoolExecutor.java:24 : }
  ChatServer.java:82            : ThreadPoolExecutor executor = new ThreadPoolExecutor(10, 2, 10,
TimeUnit.SECONDS, wQueue, rH);
  ChatServer.java:85            : servsock = new ServerSocket(port);
        [81 insn w/o sources]
  ChatServer.java:85            : servsock = new ServerSocket(port);
  ChatServer.java:86            : while (maxServ-- != 0) {
  ChatServer.java:87            : sock = servsock.accept();
        [141 insn w/o sources]
------------------------------------------------------ transition #1 thread: 1
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_ACCEPT" ,1/2,isCascaded:false}
        [3168 insn w/o sources]
  ChatClient.java:11            : public class ChatClient {
        [2 insn w/o sources]
  ChatClient.java:11            : public class ChatClient {
  ChatClient.java:13            : static int currID = 0;
  ChatClient.java:1             : /* $Id: ChatClient.java 723 2009-09-24 07:48:58Z cartho $ */
        [1 insn w/o sources]
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:19            : public ChatClient() {
        [1 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
        [2 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
  ChatClient.java:21            : System.out.println("id before" + id);
        [2 insn w/o sources]
  ChatClient.java:21            : System.out.println("id before" + id);
        [2 insn w/o sources]
  ChatClient.java:21            : System.out.println("id before" + id);
        [2 insn w/o sources]
  ChatClient.java:21            : System.out.println("id before" + id);
        [2 insn w/o sources]
  ChatClient.java:21            : System.out.println("id before" + id);
        [2 insn w/o sources]
  ChatClient.java:22            : int newID = (currID+1);
  ChatClient.java:23            : System.out.println("currID " + newID);
        [2 insn w/o sources]
  ChatClient.java:23            : System.out.println("currID " + newID);
        [2 insn w/o sources]
  ChatClient.java:23            : System.out.println("currID " + newID);
        [2 insn w/o sources]
  ChatClient.java:23            : System.out.println("currID " + newID);
        [2 insn w/o sources]
  ChatClient.java:23            : System.out.println("currID " + newID);
        [2 insn w/o sources]
  ChatClient.java:24            : id = newID;
  ChatClient.java:25            : System.out.println("id after" + id);
        [2 insn w/o sources]
  ChatClient.java:25            : System.out.println("id after" + id);
        [2 insn w/o sources]
```

```
  ChatClient.java:25            : System.out.println("id after" + id);
        [2 insn w/o sources]
  ChatClient.java:25            : System.out.println("id after" + id);
        [2 insn w/o sources]
  ChatClient.java:25            : System.out.println("id after" + id);
        [2 insn w/o sources]
  ChatClient.java:26            : }
  ChatClient.java:27            : }
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:31            : Socket socket = new Socket();
        [10 insn w/o sources]
  ChatClient.java:31            : Socket socket = new Socket();
        [116 insn w/o sources]
  ChatClient.java:31            : Socket socket = new Socket();
  ChatClient.java:32            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
        [20 insn w/o sources]
  ChatClient.java:32            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
  ChatClient.java:33            : socket.connect(addr);
        [23 insn w/o sources]
------------------------------------------------------ transition #2 thread: 0
nas.java.net.choice.NasThreadChoice {id:"CONNECT" ,1/3,isCascaded:false}
        [3 insn w/o sources]
  ChatServer.java:87            : sock = servsock.accept();
  ChatServer.java:88            : Worker worker = null;
  ChatServer.java:90            : worker = new Worker(sock, this);
        [1 insn w/o sources]
  ChatServer.java:17            : class Worker implements Runnable {
        [2 insn w/o sources]
  ChatServer.java:17            : class Worker implements Runnable {
  ChatServer.java:1             : import java.io.BufferedReader;
  ChatServer.java:90            : worker = new Worker(sock, this);
  ChatServer.java:24            : public Worker(Socket s, ChatServer cs) throws IOException{
        [1 insn w/o sources]
  ChatServer.java:25            : chatServer = cs;
  ChatServer.java:26            : sock = s;
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
        [29 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
        [270 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
  ChatServer.java:29            : in = new BufferedReader(new
        [7 insn w/o sources]
  ChatServer.java:29            : in = new BufferedReader(new
        [8 insn w/o sources]
  ChatServer.java:29            : in = new BufferedReader(new
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [29 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [23 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [48 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
  ChatServer.java:33            : }
  ChatServer.java:34            : }
  ChatServer.java:90            : worker = new Worker(sock, this);
  ChatServer.java:93            : }
  ChatServer.java:94            : if(init){
  ChatServer.java:95            : assert(init);
  ChatServer.java:96            : executor.execute(new Thread(worker));
        [145 insn w/o sources]
  ChatServer.java:96            : executor.execute(new Thread(worker));
  env/java/util/concurrent/ThreadPoolExecutor.java:29 : if(launchedThreads<maximumPoolSize){
```

```
  env/java/util/concurrent/ThreadPoolExecutor.java:30 : new Thread(r).start();
        [145 insn w/o sources]
  env/java/util/concurrent/ThreadPoolExecutor.java:30 : new Thread(r).start();
        [1 insn w/o sources]
------------------------------------------------------ transition #3 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"START" ,1/2,isCascaded:false}
        [2 insn w/o sources]
  env/java/util/concurrent/ThreadPoolExecutor.java:31 : launchedThreads++;
  env/java/util/concurrent/ThreadPoolExecutor.java:33 : }
  ChatServer.java:98             : }
  ChatServer.java:86             : while (maxServ-- != 0) {
  ChatServer.java:87             : sock = servsock.accept();
        [67 insn w/o sources]
------------------------------------------------------ transition #4 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [64 insn w/o sources]
------------------------------------------------------ transition #5 thread: 1
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_ACCEPT" ,1/3,isCascaded:false}
        [4 insn w/o sources]
  ChatClient.java:34             : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:34             : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:34             : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:34             : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:34             : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:34             : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:35             : InputStreamReader istr =
        [8 insn w/o sources]
  ChatClient.java:35             : InputStreamReader istr =
  ChatClient.java:36             : new InputStreamReader(socket.getInputStream());
        [29 insn w/o sources]
  ChatClient.java:36             : new InputStreamReader(socket.getInputStream());
        [23 insn w/o sources]
  ChatClient.java:36             : new InputStreamReader(socket.getInputStream());
  ChatClient.java:37             : BufferedReader in = new BufferedReader(istr);
        [7 insn w/o sources]
  ChatClient.java:37             : BufferedReader in = new BufferedReader(istr);
        [48 insn w/o sources]
  ChatClient.java:37             : BufferedReader in = new BufferedReader(istr);
  ChatClient.java:38             : OutputStreamWriter out =
        [8 insn w/o sources]
  ChatClient.java:38             : OutputStreamWriter out =
  ChatClient.java:39             : new OutputStreamWriter(socket.getOutputStream());
        [29 insn w/o sources]
  ChatClient.java:39             : new OutputStreamWriter(socket.getOutputStream());
        [17 insn w/o sources]
  ChatClient.java:39             : new OutputStreamWriter(socket.getOutputStream());
  ChatClient.java:40             : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:40             : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:40             : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:40             : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:40             : out.write(id + ": Hello, world!\n");
        [55 insn w/o sources]
```

```
  ChatClient.java:41            : out.flush();
        [1 insn w/o sources]
  ChatClient.java:42            : for (int i = 0; i < 1; i++) {
  ChatClient.java:43            : String recieved = in.readLine();
        [77 insn w/o sources]
------------------------------------------------------ transition #6 thread: 2
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_READ" ,1/2,isCascaded:false}
        [3168 insn w/o sources]
  ChatClient.java:11            : public class ChatClient {
        [2 insn w/o sources]
  ChatClient.java:11            : public class ChatClient {
  ChatClient.java:13            : static int currID = 0;
  ChatClient.java:1             : /* $Id: ChatClient.java 723 2009-09-24 07:48:58Z cartho $ */
        [1 insn w/o sources]
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:19            : public ChatClient() {
        [1 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
        [2 insn w/o sources]
  ChatClient.java:20            : synchronized(getClass()) {
  ChatClient.java:21            : System.out.println("id before" + id);
        [2 insn w/o sources]
  ChatClient.java:21            : System.out.println("id before" + id);
        [2 insn w/o sources]
  ChatClient.java:21            : System.out.println("id before" + id);
        [2 insn w/o sources]
  ChatClient.java:21            : System.out.println("id before" + id);
        [2 insn w/o sources]
  ChatClient.java:21            : System.out.println("id before" + id);
        [2 insn w/o sources]
  ChatClient.java:22            : int newID = (currID+1);
  ChatClient.java:23            : System.out.println("currID " + newID);
        [2 insn w/o sources]
  ChatClient.java:23            : System.out.println("currID " + newID);
        [2 insn w/o sources]
  ChatClient.java:23            : System.out.println("currID " + newID);
        [2 insn w/o sources]
  ChatClient.java:23            : System.out.println("currID " + newID);
        [2 insn w/o sources]
  ChatClient.java:23            : System.out.println("currID " + newID);
        [2 insn w/o sources]
  ChatClient.java:24            : id = newID;
  ChatClient.java:25            : System.out.println("id after" + id);
        [2 insn w/o sources]
  ChatClient.java:25            : System.out.println("id after" + id);
        [2 insn w/o sources]
  ChatClient.java:25            : System.out.println("id after" + id);
        [2 insn w/o sources]
  ChatClient.java:25            : System.out.println("id after" + id);
        [2 insn w/o sources]
  ChatClient.java:25            : System.out.println("id after" + id);
        [2 insn w/o sources]
  ChatClient.java:26            : }
  ChatClient.java:27            : }
  ChatClient.java:16            : new ChatClient().exec();
  ChatClient.java:31            : Socket socket = new Socket();
        [10 insn w/o sources]
  ChatClient.java:31            : Socket socket = new Socket();
        [116 insn w/o sources]
  ChatClient.java:31            : Socket socket = new Socket();
  ChatClient.java:32            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
        [20 insn w/o sources]
```

```
  ChatClient.java:32            : InetSocketAddress addr = new InetSocketAddress("localhost", 4444);
  ChatClient.java:33            : socket.connect(addr);
        [23 insn w/o sources]
------------------------------------------------------ transition #7 thread: 0
nas.java.net.choice.NasThreadChoice {id:"CONNECT" ,1/3,isCascaded:false}
        [3 insn w/o sources]
  ChatServer.java:87            : sock = servsock.accept();
  ChatServer.java:88            : Worker worker = null;
  ChatServer.java:90            : worker = new Worker(sock, this);
  ChatServer.java:24            : public Worker(Socket s, ChatServer cs) throws IOException{
        [1 insn w/o sources]
  ChatServer.java:25            : chatServer = cs;
  ChatServer.java:26            : sock = s;
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
        [29 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
        [82 insn w/o sources]
------------------------------------------------------ transition #8 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [112 insn w/o sources]
------------------------------------------------------ transition #9 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [63 insn w/o sources]
  ChatServer.java:28            : out = new PrintWriter(sock.getOutputStream(), true);
  ChatServer.java:29            : in = new BufferedReader(new
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [29 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [23 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
        [48 insn w/o sources]
  ChatServer.java:30            : InputStreamReader(sock.getInputStream()));
  ChatServer.java:33            : }
  ChatServer.java:34            : }
  ChatServer.java:90            : worker = new Worker(sock, this);
  ChatServer.java:93            : }
  ChatServer.java:94            : if(init){
  ChatServer.java:95            : assert(init);
  ChatServer.java:96            : executor.execute(new Thread(worker));
        [27 insn w/o sources]
------------------------------------------------------ transition #10 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [119 insn w/o sources]
  ChatServer.java:96            : executor.execute(new Thread(worker));
  env/java/util/concurrent/ThreadPoolExecutor.java:29 : if(launchedThreads<maximumPoolSize){
  env/java/util/concurrent/ThreadPoolExecutor.java:30 : new Thread(r).start();
        [27 insn w/o sources]
------------------------------------------------------ transition #11 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [119 insn w/o sources]
  env/java/util/concurrent/ThreadPoolExecutor.java:30 : new Thread(r).start();
        [1 insn w/o sources]
------------------------------------------------------ transition #12 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"START" ,1/3,isCascaded:false}
        [2 insn w/o sources]
  env/java/util/concurrent/ThreadPoolExecutor.java:31 : launchedThreads++;
  env/java/util/concurrent/ThreadPoolExecutor.java:33 : }
  ChatServer.java:98            : }
  ChatServer.java:86            : while (maxServ-- != 0) {
  ChatServer.java:99            : servsock.close();
------------------------------------------------------ transition #13 thread: 0
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/3,isCascaded:false}
```

```
  ChatServer.java:99            : servsock.close();
        [2 insn w/o sources]
  ChatServer.java:102           : }
  ChatServer.java:103           : System.out.println("Server shutting down.");
        [2 insn w/o sources]
  ChatServer.java:104           : executor.shutdown();
  env/java/util/concurrent/ThreadPoolExecutor.java:26 : public void shutdown() { } // stub
  ChatServer.java:105           : }
  ChatServer.java:111           : new ChatServer(Integer.parseInt(args[0]));
  ChatServer.java:113           : }
  ChatServer.java:3             : import java.io.InputStreamReader;
------------------------------------------------------ transition #14 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"TERMINATE" ,1/2,isCascaded:false}
        [8 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [59 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [2 insn w/o sources]
  ChatServer.java:38            : idx = chatServer.getAndUpdateN();
------------------------------------------------------ transition #15 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
  ChatServer.java:38            : idx = chatServer.getAndUpdateN();
  ChatServer.java:133           : n++;
  ChatServer.java:134           : return n-1;
  ChatServer.java:38            : idx = chatServer.getAndUpdateN();
  ChatServer.java:41            : assert(out != null);
  ChatServer.java:42            : assert(chatServer.getWorker(idx) == null);
------------------------------------------------------ transition #16 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
  ChatServer.java:42            : assert(chatServer.getWorker(idx) == null);
  ChatServer.java:129           : return workers.get(id);
        [2 insn w/o sources]
  ChatServer.java:129           : return workers.get(id);
        [35 insn w/o sources]
  ChatServer.java:129           : return workers.get(id);
  ChatServer.java:42            : assert(chatServer.getWorker(idx) == null);
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:44            : chatServer.putWorker(idx, this);
------------------------------------------------------ transition #17 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
  ChatServer.java:44            : chatServer.putWorker(idx, this);
  ChatServer.java:126           : workers.put(idx, w);
        [2 insn w/o sources]
```

```
  ChatServer.java:126           : workers.put(idx, w);
        [65 insn w/o sources]
------------------------------------------------------ transition #18 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"EXPOSE" ,1/2,isCascaded:false}
        [78 insn w/o sources]
  ChatServer.java:126           : workers.put(idx, w);
  ChatServer.java:127           : }
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
        [2 insn w/o sources]
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
        [2 insn w/o sources]
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
        [2 insn w/o sources]
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
        [2 insn w/o sources]
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
        [2 insn w/o sources]
  ChatServer.java:45            : System.out.println("Registered worker " + idx + ".");
        [2 insn w/o sources]
  ChatServer.java:47            : String s = null;
  ChatServer.java:48            : while ((s = in.readLine()) != null) {
        [10 insn w/o sources]
------------------------------------------------------ transition #19 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [47 insn w/o sources]
------------------------------------------------------ transition #20 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [435 insn w/o sources]
  ChatServer.java:48            : while ((s = in.readLine()) != null) {
  ChatServer.java:50            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:50            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:50            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:50            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:50            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:50            : chatServer.sendAll("[" + idx + "]" + s);
        [2 insn w/o sources]
  ChatServer.java:50            : chatServer.sendAll("[" + idx + "]" + s);
------------------------------------------------------ transition #21 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
  ChatServer.java:50            : chatServer.sendAll("[" + idx + "]" + s);
  ChatServer.java:116           : for (Worker w: workers.values()) {
        [23 insn w/o sources]
------------------------------------------------------ transition #22 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"EXPOSE" ,1/2,isCascaded:false}
        [3 insn w/o sources]
  ChatServer.java:116           : for (Worker w: workers.values()) {
        [59 insn w/o sources]
  ChatServer.java:116           : for (Worker w: workers.values()) {
        [6 insn w/o sources]
  ChatServer.java:116           : for (Worker w: workers.values()) {
        [307 insn w/o sources]
  ChatServer.java:116           : for (Worker w: workers.values()) {
  ChatServer.java:117           : w.send(s);
  ChatServer.java:61            : out.println(s);
        [5 insn w/o sources]
------------------------------------------------------ transition #23 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
```

```
        [41 insn w/o sources]
------------------------------------------------------ transition #24 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [109 insn w/o sources]
------------------------------------------------------ transition #25 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [79 insn w/o sources]
------------------------------------------------------ transition #26 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [31 insn w/o sources]
------------------------------------------------------ transition #27 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [26 insn w/o sources]
------------------------------------------------------ transition #28 thread: 1
nas.java.net.choice.NasThreadChoice {id:"WRITE" ,1/4,isCascaded:false}
        [465 insn w/o sources]
  ChatClient.java:43            : String recieved = in.readLine();
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:45            : assert recieved.contains("Hello,") || recieved.contains("world") :
"recieved was: " + recieved;
        [13 insn w/o sources]
  ChatClient.java:45            : assert recieved.contains("Hello,") || recieved.contains("world") :
"recieved was: " + recieved;
  ChatClient.java:42            : for (int i = 0; i < 1; i++) {
  ChatClient.java:49            : out.close();
        [13 insn w/o sources]
------------------------------------------------------ transition #29 thread: 1
nas.java.net.choice.NasThreadChoice {id:"SOCKET_CLOSE" ,1/4,isCascaded:false}
        [4 insn w/o sources]
  ChatClient.java:52            : }
  ChatClient.java:53            : }
  ChatClient.java:3             : import java.io.BufferedReader;
------------------------------------------------------ transition #30 thread: 2
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"TERMINATE"
,1/3,isCascaded:false,attrs:[gov.nasa.jpf.vm.GlobalSchedulingPoint@45afc369]}
        [4 insn w/o sources]
  ChatClient.java:34            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:34            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:34            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:34            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:34            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:34            : System.out.println("Client " + id + " connected.");
        [2 insn w/o sources]
  ChatClient.java:35            : InputStreamReader istr =
        [8 insn w/o sources]
  ChatClient.java:35            : InputStreamReader istr =
```

```
  ChatClient.java:36            : new InputStreamReader(socket.getInputStream());
        [29 insn w/o sources]
  ChatClient.java:36            : new InputStreamReader(socket.getInputStream());
        [23 insn w/o sources]
  ChatClient.java:36            : new InputStreamReader(socket.getInputStream());
  ChatClient.java:37            : BufferedReader in = new BufferedReader(istr);
        [7 insn w/o sources]
  ChatClient.java:37            : BufferedReader in = new BufferedReader(istr);
        [48 insn w/o sources]
  ChatClient.java:37            : BufferedReader in = new BufferedReader(istr);
  ChatClient.java:38            : OutputStreamWriter out =
        [8 insn w/o sources]
  ChatClient.java:38            : OutputStreamWriter out =
  ChatClient.java:39            : new OutputStreamWriter(socket.getOutputStream());
        [29 insn w/o sources]
  ChatClient.java:39            : new OutputStreamWriter(socket.getOutputStream());
        [17 insn w/o sources]
  ChatClient.java:39            : new OutputStreamWriter(socket.getOutputStream());
  ChatClient.java:40            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:40            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:40            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:40            : out.write(id + ": Hello, world!\n");
        [2 insn w/o sources]
  ChatClient.java:40            : out.write(id + ": Hello, world!\n");
        [55 insn w/o sources]
  ChatClient.java:41            : out.flush();
        [1 insn w/o sources]
  ChatClient.java:42            : for (int i = 0; i < 1; i++) {
  ChatClient.java:43            : String recieved = in.readLine();
        [77 insn w/o sources]
------------------------------------------------------ transition #31 thread: 4
nas.java.net.choice.NasThreadChoice {id:"BLOCKING_READ" ,1/2,isCascaded:false}
        [39 insn w/o sources]
  ChatServer.java:62            : }
  ChatServer.java:118           : }
  ChatServer.java:116           : for (Worker w: workers.values()) {
        [5 insn w/o sources]
  ChatServer.java:116           : for (Worker w: workers.values()) {
  ChatServer.java:119           : }
  ChatServer.java:50            : chatServer.sendAll("[" + idx + "]" + s);
  ChatServer.java:48            : while ((s = in.readLine()) != null) {
        [10 insn w/o sources]
------------------------------------------------------ transition #32 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [47 insn w/o sources]
------------------------------------------------------ transition #33 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [32 insn w/o sources]
  ChatServer.java:53            : } catch(IOException ioe) {
  ChatServer.java:54            : System.out.println("Worker thread " + idx + ": " + ioe);
        [2 insn w/o sources]
  ChatServer.java:54            : System.out.println("Worker thread " + idx + ": " + ioe);
        [2 insn w/o sources]
  ChatServer.java:54            : System.out.println("Worker thread " + idx + ": " + ioe);
        [2 insn w/o sources]
  ChatServer.java:54            : System.out.println("Worker thread " + idx + ": " + ioe);
        [2 insn w/o sources]
  ChatServer.java:54            : System.out.println("Worker thread " + idx + ": " + ioe);
        [14 insn w/o sources]
```

```
  ChatServer.java:54            : System.out.println("Worker thread " + idx + ": " + ioe);
        [2 insn w/o sources]
  ChatServer.java:54            : System.out.println("Worker thread " + idx + ": " + ioe);
        [2 insn w/o sources]
  ChatServer.java:56            : chatServer.remove(idx);
------------------------------------------------------ transition #34 thread: 6
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,2/2,isCascaded:false}
        [8 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [59 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [2 insn w/o sources]
  ChatServer.java:37            : System.out.println("Thread running: " + Thread.currentThread());
        [2 insn w/o sources]
  ChatServer.java:38            : idx = chatServer.getAndUpdateN();
------------------------------------------------------ transition #35 thread: 6
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,2/2,isCascaded:false}
  ChatServer.java:38            : idx = chatServer.getAndUpdateN();
  ChatServer.java:133           : n++;
  ChatServer.java:134           : return n-1;
------------------------------------------------------ transition #36 thread: 6
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"RELEASE" ,2/2,isCascaded:false}
  ChatServer.java:134           : return n-1;
  ChatServer.java:38            : idx = chatServer.getAndUpdateN();
  ChatServer.java:41            : assert(out != null);
  ChatServer.java:42            : assert(chatServer.getWorker(idx) == null);
------------------------------------------------------ transition #37 thread: 6
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,2/2,isCascaded:false}
  ChatServer.java:42            : assert(chatServer.getWorker(idx) == null);
  ChatServer.java:129           : return workers.get(id);
        [2 insn w/o sources]
  ChatServer.java:129           : return workers.get(id);
        [50 insn w/o sources]
  ChatServer.java:129           : return workers.get(id);
------------------------------------------------------ transition #38 thread: 6
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"RELEASE" ,2/2,isCascaded:false}
  ChatServer.java:129           : return workers.get(id);
  ChatServer.java:42            : assert(chatServer.getWorker(idx) == null);
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:43            : System.err.println("Adding "+idx+" to workers");
        [2 insn w/o sources]
  ChatServer.java:44            : chatServer.putWorker(idx, this);
------------------------------------------------------ transition #39 thread: 6
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,2/2,isCascaded:false}
  ChatServer.java:44            : chatServer.putWorker(idx, this);
  ChatServer.java:126           : workers.put(idx, w);
        [2 insn w/o sources]
```

```
  ChatServer.java:126           : workers.put(idx, w);
        [104 insn w/o sources]
  ChatServer.java:126           : workers.put(idx, w);
  ChatServer.java:127           : }
------------------------------------------------------ transition #40 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"RELEASE" ,1/2,isCascaded:false}
  ChatServer.java:56            : chatServer.remove(idx);
  ChatServer.java:122           : workers.remove(i);
        [2 insn w/o sources]
  ChatServer.java:122           : workers.remove(i);
        [102 insn w/o sources]
  ChatServer.java:122           : workers.remove(i);
  ChatServer.java:123           : sendAll("Client " + i + " quit.");
        [2 insn w/o sources]
  ChatServer.java:123           : sendAll("Client " + i + " quit.");
        [2 insn w/o sources]
  ChatServer.java:123           : sendAll("Client " + i + " quit.");
        [2 insn w/o sources]
  ChatServer.java:123           : sendAll("Client " + i + " quit.");
        [2 insn w/o sources]
  ChatServer.java:123           : sendAll("Client " + i + " quit.");
        [2 insn w/o sources]
  ChatServer.java:123           : sendAll("Client " + i + " quit.");
  ChatServer.java:116           : for (Worker w: workers.values()) {
        [7 insn w/o sources]
  ChatServer.java:116           : for (Worker w: workers.values()) {
        [77 insn w/o sources]
  ChatServer.java:116           : for (Worker w: workers.values()) {
        [6 insn w/o sources]
  ChatServer.java:116           : for (Worker w: workers.values()) {
        [19 insn w/o sources]
------------------------------------------------------ transition #41 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_OBJECT" ,1/2,isCascaded:false}
        [270 insn w/o sources]
------------------------------------------------------ transition #42 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_OBJECT" ,1/2,isCascaded:false}
        [2 insn w/o sources]
  ChatServer.java:116           : for (Worker w: workers.values()) {
  ChatServer.java:117           : w.send(s);
  ChatServer.java:61            : out.println(s);
------------------------------------------------------ transition #43 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"SHARED_OBJECT" ,1/2,isCascaded:false}
  ChatServer.java:61            : out.println(s);
        [5 insn w/o sources]
------------------------------------------------------ transition #44 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [41 insn w/o sources]
------------------------------------------------------ transition #45 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [109 insn w/o sources]
------------------------------------------------------ transition #46 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [79 insn w/o sources]
------------------------------------------------------ transition #47 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [31 insn w/o sources]
------------------------------------------------------ transition #48 thread: 4
gov.nasa.jpf.vm.choice.ThreadChoiceFromSet {id:"LOCK" ,1/2,isCascaded:false}
        [26 insn w/o sources]
------------------------------------------------------ transition #49 thread: 2
nas.java.net.choice.NasThreadChoice {id:"WRITE" ,1/3,isCascaded:false}
        [380 insn w/o sources]
```

```
  ChatClient.java:43            : String recieved = in.readLine();
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:44            : System.out.println(id + ": Received " + recieved);
        [2 insn w/o sources]
  ChatClient.java:45            : assert recieved.contains("Hello,") || recieved.contains("world") :
"recieved was: " + recieved;
        [12 insn w/o sources]
  ChatClient.java:45            : assert recieved.contains("Hello,") || recieved.contains("world") :
"recieved was: " + recieved;
        [12 insn w/o sources]
  ChatClient.java:45            : assert recieved.contains("Hello,") || recieved.contains("world") :
"recieved was: " + recieved;
        [2 insn w/o sources]
  ChatClient.java:45            : assert recieved.contains("Hello,") || recieved.contains("world") :
"recieved was: " + recieved;
        [2 insn w/o sources]
  ChatClient.java:45            : assert recieved.contains("Hello,") || recieved.contains("world") :
"recieved was: " + recieved;
        [2 insn w/o sources]
  ChatClient.java:45            : assert recieved.contains("Hello,") || recieved.contains("world") :
"recieved was: " + recieved;
        [2 insn w/o sources]
  ChatClient.java:45            : assert recieved.contains("Hello,") || recieved.contains("world") :
"recieved was: " + recieved;
        [50 insn w/o sources]

================================================== snapshot #1
thread
java.lang.Thread:{id:2,name:main,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount:0}
  call stack:
    at ChatClient.exec(ChatClient.java:45)
    at ChatClient.main(ChatClient.java:16)

thread
java.lang.Thread:{id:4,name:Thread-2,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount
:0}
  owned
locks:ChatServer@2de,java.io.BufferedWriter@5b5,java.io.OutputStreamWriter@5b6,java.lang.Object@474
  call stack:
    at java.net.SocketOutputStream.write(SocketOutputStream.java)
    at java.io.OutputStreamWriter.write(OutputStreamWriter.java:87)
    at java.io.BufferedWriter.flushBuffer(BufferedWriter.java:129)
    at java.io.BufferedWriter.flush(BufferedWriter.java:253)
    at java.io.PrintWriter.newLine(PrintWriter.java:482)
    at java.io.PrintWriter.println(PrintWriter.java:629)
    at java.io.PrintWriter.println(PrintWriter.java:740)
    at Worker.send(ChatServer.java:61)
    at ChatServer.sendAll(ChatServer.java:117)
    at ChatServer.remove(ChatServer.java:123)
    at Worker.run(ChatServer.java:56)
    at java.lang.Thread.run(Thread.java:348)
```

```
thread
java.lang.Thread:{id:6,name:Thread-4,status:RUNNING,priority:5,isDaemon:false,lockCount:0,suspendCount
:0}
  call stack:
    at ChatServer.putWorker(ChatServer.java:127)
    at Worker.run(ChatServer.java:44)
    at java.lang.Thread.run(Thread.java:348)


====================================================== results
error #1: gov.nasa.jpf.vm.NoUncaughtExceptionsProperty "java.lang.AssertionError: recieved was: Client
0 q..."

====================================================== statistics
elapsed time:   00:00:01
states:         new=231,visited=97,backtracked=278,end=28
search:         maxDepth=103,constraints=0
choice generators:  thread=230 (signal=0,lock=47,sharedRef=42,threadApi=2,reschedule=120), data=0
heap:           new=2709,released=3055,maxLive=1748,gcCycles=319
instructions:   74381
max memory:     112MB
loaded code:    classes=129,methods=5193
```

3.5 Additional properties

The following additional properties were added:

Line 35 ChatServer.java

An assertion that the BufferedReader in and the Socket sock is not null when starting to run a worker.

```
assert(in !=null);
assert(sock !=null);
```

Line 47 ChatSerevr.java

An assertion that the worker only receives messages from the clients. This was done by asserting that all received messages contained the string “Hello”, “world”.

```
assert s.contains("Hello,")||s.contains("world") : "message was " + s;
```

Line 77 ChatServer.java

An assertion that the ServerSocket servsock is not null after creation.

```
servsock = new ServerSocket(port);
//3.5 added property
assert servsock !=null;
```

Line 83 ChatServer.java

An assertion that the Worker worker is not null after creation.

```
worker = new Worker(sock, this);
//3.5 added property
assert worker != null;
```

Line 121/126 ChatServer.java

An assertion that the Hashmap workers increase/decrease in size by one every put()/remove().

```
public synchronized void putWorker(int idx, Worker w){
        int sizebefore = workers.size();
```

```
        workers.put(idx, w);
        //3.5 added property
        assert workers.size() == sizebefore+1 : "sizebefore was "+sizebefore +" sizenow is
"+workers.size();
}
```

Can we check if the different combinations of clients, worker threads, and messages between clients actually occur as an assertion?

When running the program with one single client (by modifying the ChatServer.jpf file) we get a trace with around 600 lines after a couple of seconds. When adding another Client we left JPF running for several minutes, rendering a trace of around 45 000 lines.

This indicates that the state space, permutations or combinations, grows rapidly as the number of clients grows.

We believe that it can *not* be checked in an assertion if all combinations are being checked - at least not an assertion in the actual program we are checking. Checking combinations indicates that the program needs to be run several times, which makes an assertion non feasible. If the program is only being run one time it would still not work with an assertion, since only the assertion when checking the very last combination would pass, when all others are already made and documented.

We could potentially have another program that goes through the traces and counts combinations, after our program is done.

3.6 Bonus

We tried to use methods of java.net.Socket and java.net.ServerSocket in our assertions. However, it seems as if JPF lacks implementations of other methods than the most used, eg. accept(). Calling sock.getPort(), where sock is a java.net.Socket, results in a NoSuchMethodError when running JPF. However, the code compiles.

More bugs can be found when trying out other methods of java.net.Socket and java.net.ServerSocket, many seem to not be implemented.

Instead of showing a NoSuchMethodError, which is confusing since there is a method with said name in Java, maybe the error could be a bit more descriptive and state that the error is due to JPF not having it implemented.