Space Exploration

EECS 4315

www.cse.yorku.ca/course/4315/

Nondeterminism

Nondeterministic code is code that, even for the same input, can exhibit different behaviours on different runs, as opposed to deterministic code.

- Randomization and
- concurrency

both give rise to nondeterminism.

Testing Nondeterministic Code

```
public class RandomFraction
{
  public static void run()
  {
    Random random = new Random(System.currentTimeMi
    System.out.print(1 / random.nextInt(1000000));
  }
}
```

Question

If we run the above app 1,000,000 times, what is the probability that it does not throw an exception in any of those runs?

Testing Nondeterministic Code

Answer

- The probability of choosing zero is $\frac{1}{1{,}000{,}000}$.
- The probability of not choosing zero is $1 - \frac{1}{1{,}000{,}000} = \frac{999{,}999}{1{,}000{,}000}$.
- The probability of not choosing zero one million times in a row is $(\frac{999{,}999}{1{,}000{,}000})^{1{,}000{,}000} \approx 0.37$.

Limitations of Testing

Limitations of testing of nondeterministic code include

- no guarantee that all different behaviours have been checked, and
- errors may be difficult to reproduce.

Alternatives to Testing

To detect bugs in nondeterministic code, testing needs to be supplemented with other approaches.

Question

How to tackle the limitations of testing of nondeterministic code?

Alternatives to Testing

To detect bugs in nondeterministic code, testing needs to be supplemented with other approaches.

Question

How to tackle the limitations of testing of nondeterministic code?

Answer

Control the nondeterminism: this allows us to

- systematically check all different behaviours and
- reproduce errors.

Exercises

In groups of two or three, solve the following exercises. Fill in the body of the following `main` method.

```
public class Exercise
{
  public static void main(String[] args)
  {
    Random random = new Random();

  }
}
```

using only the `nextBoolean` method of the `Random` class.

Exercises

1. The app prints either 1 or 2, both with probability 0.5.
2. The app prints 1, 2, 3, or 4, each with probability 0.25.
3. The app prints any integer, each with positive but not necessarily equal probability.

Executions

- The first app has two different executions.
- The second app has four different executions.

Question

How many different executions has the third application?

Executions

- The first app has two different executions.
- The second app has four different executions.

Question

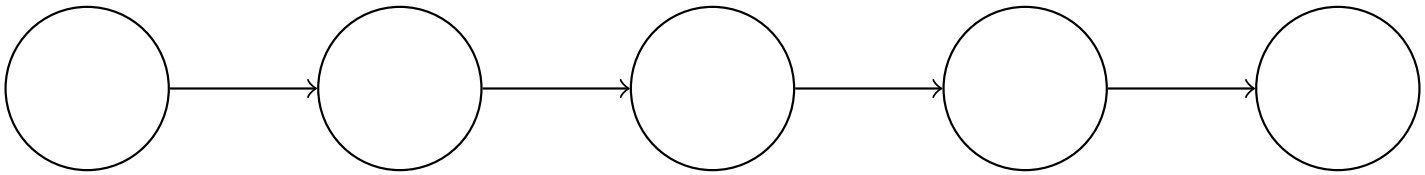
How many different executions has the third application?

Answer

$2^{32} = 4,294,967,296.$

Execution

An execution consists of states connected by transitions.



State

A state of a Java virtual machine (JVM) includes

- the heap,
- for each thread
  - its state (runnable, waiting, terminated, ...),
  - its stack,
  - etc,
- etc.

`https://docs.oracle.com/javase/8/docs/platform/jvmti/jvmti.html`

Transition

A transition of a JVM takes the JVM from one state to another by executing a bytecode instruction.

Java Code

```
public class HelloWorld
{
  public static void main(String[] args)
  {
    System.out.println("Hello World");
  }
}
```

Java Bytecode

The command

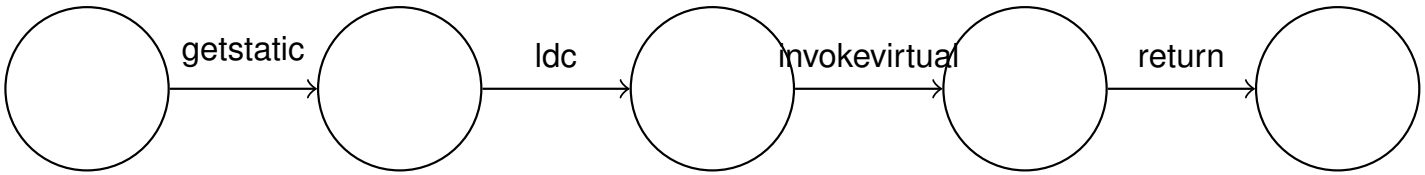
```
javap -c HelloWorld.class
```

produces

```
0: getstatic
// of attribute System.out
// of class PrintStream
3: ldc
// String "Hello World"
5: invokevirtual
// of method println
// with argument String
8: return
```

Java Code and Execution

```
public class HelloWorld
{
  public static void main(String[] args)
  {
    System.out.println("Hello World");
  }
}
```



Java Code

```
public class RedOrGreen
{
  public static void main(String[] args)
  {
    Random random = new Random();
    if (random.nextBoolean())
    {
      System.out.println("Red");
    }
    else
    {
      System.out.println("Green");
    }
  }
}
```

Java Bytecode

```
0: new
3: dup
4: invokespecial
7: astore_1
8: aload_1
9: invokevirtual
12: ifeq
15: getstatic
18: ldc
20: invokevirtual
23: goto
26: getstatic
29: ldc
31: invokevirtual
34: return
```

Executions

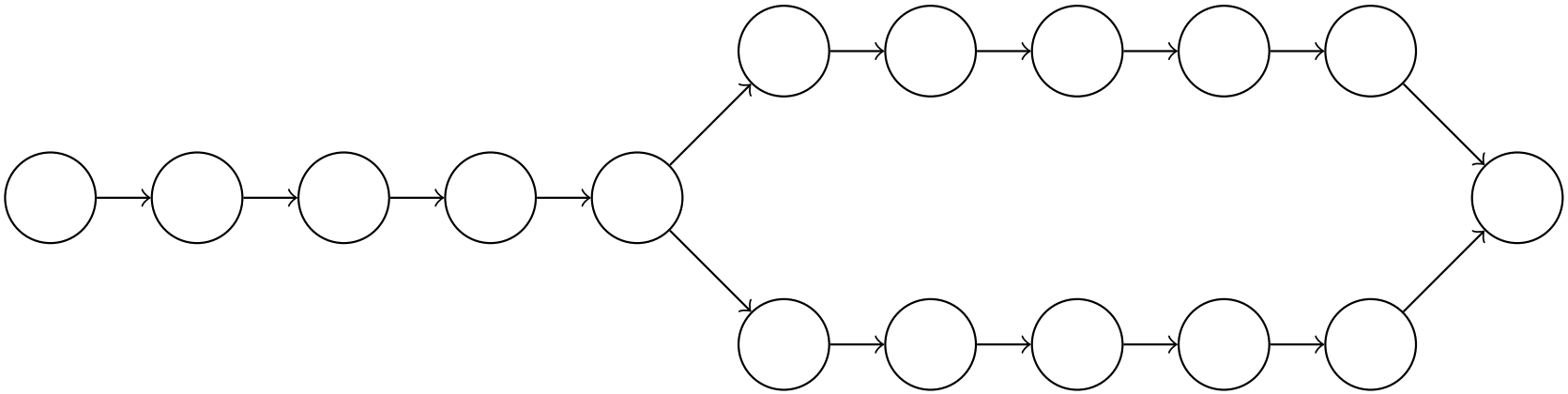
Question

Draw the state-transition diagram.

Executions

Question

Draw the state-transition diagram.



State Space Explosion Problem

Problem

The size of the state space, that is, the number of states, may become very large.

State Space Explosion Problem

Problem

The size of the state space, that is, the number of states, may become very large.

This is one of the major challenges in model checking.

Model Checking

Develop a model (states connected by transitions) of the code and check properties of the model.

Model Checking

Model checking was developed independently by Clarke and Emerson and by Queille and Sifakis in early 1980s.

Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In, Dexter Kozen, editor, *Proceedings of Workshop on Logic of Programs*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Yorktown Heights, NY, USA, May 1981. Springer-Verlag.

Jean-Pierre Queille and Joseph Sifakis. Specification and verification of concurrent systems in CESAR. In, Mariangiola Dezani-Ciancaglini and Ugo Montanari, editors, *Proceedings of the 5th International Symposium on Programming*, volume 137 of *Lecture Notes in Computer Science*, pages 337–351. Torino, Italy, April 1982. Springer-Verlag.

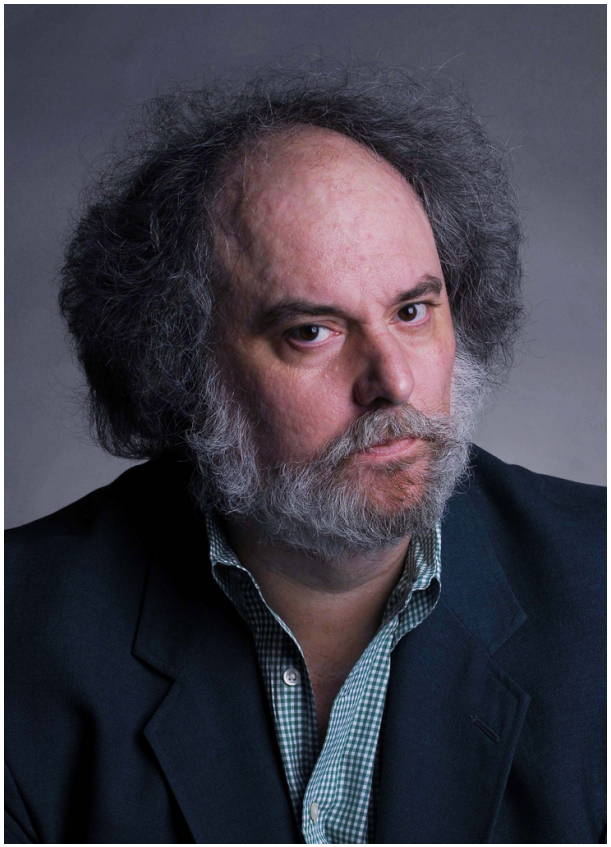
Edmund Clarke

- Recipient of the Turing Award (2007)
- Recipient of the ACM Paris Kanellakis Award (1999)
- Member of the National Academy of Engineering (2005)
- Member of the American Academy of Arts and Sciences (2011)

Source: Dennis Hamilton

Allen Emerson

- Recipient of the Turing Award (2007)
- Recipient of the ACM Paris Kanellakis Award (1999)
- Recipient of the CMU Newell Medal (1999)



Source: Marsha Miller

Joseph Sifakis

- Recipient of the Turing Award (2007)
- Grand officer of France's national order of merit (2008)
- Commander in France's legion of honour (2011)

Source: David Monniaux

Jean-Pierre Queille

Source: unknown

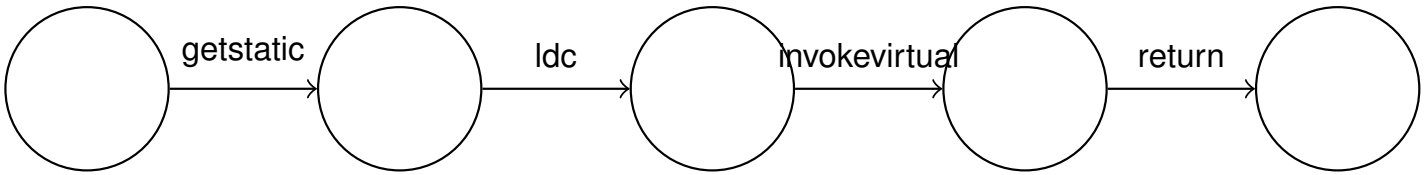
Model of a System

A model of a system is an abstraction of the system.

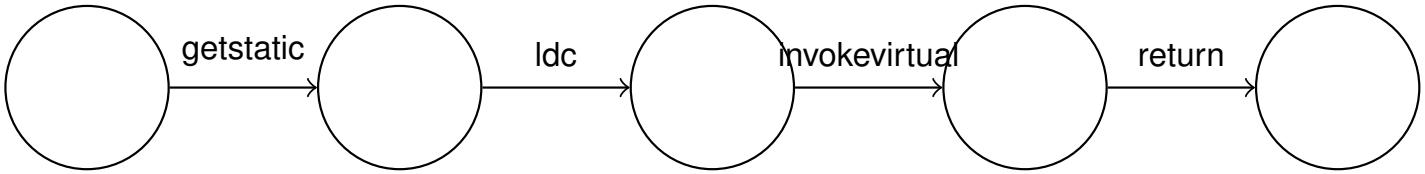
There are many levels of abstraction and, hence, a system can be modelled in many different ways.

A Model of a System

```
public class HelloWorld
{
  public static void main(String[] args)
  {
    System.out.println("Hello World");
  }
}
```



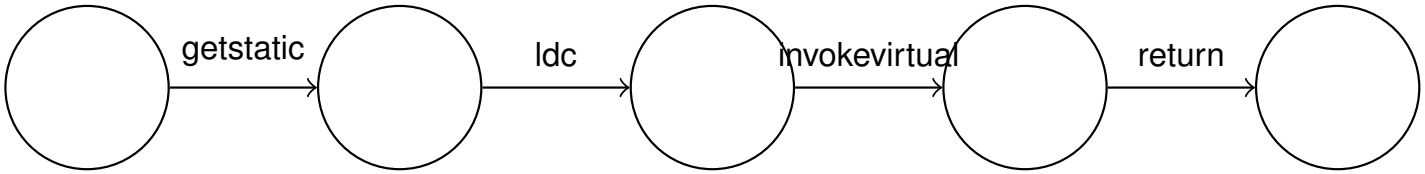
A Model of a System



Question

What are the three entities that make up the above model?

A Model of a System



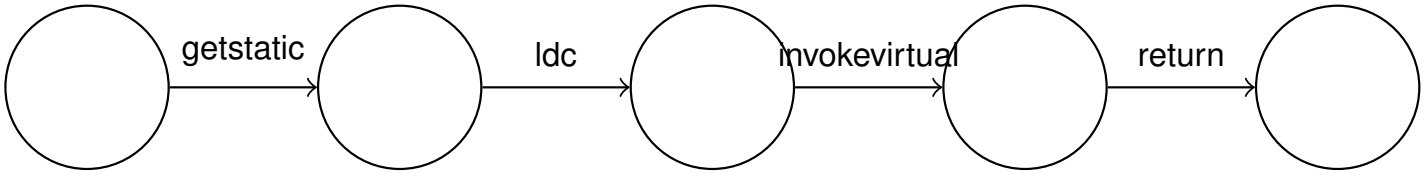
Question

What are the three entities that make up the above model?

Answer

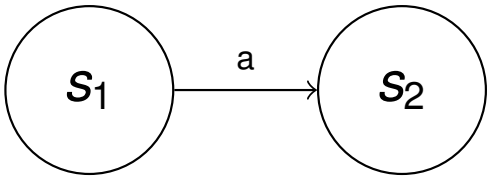
States, transitions and actions (such as getstatic, ldc, . . .).

A Model of a System

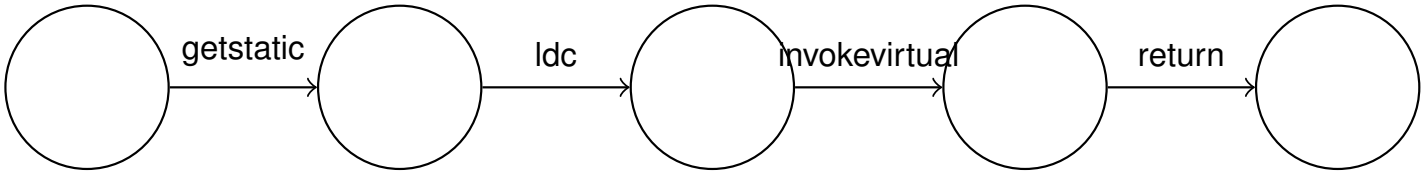


Question

Given a set of states S and a set of actions A, how can we mathematically model a transition from state s_1 to state s_2 labelled with action a?

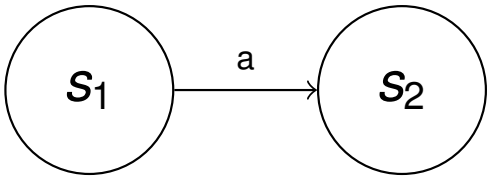


A Model of a System



Question

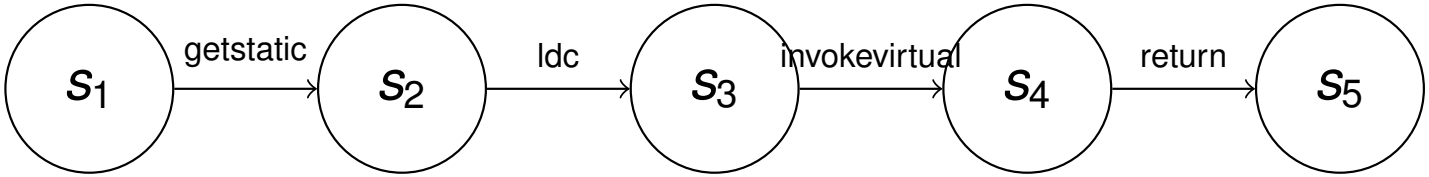
Given a set of states S and a set of actions A, how can we mathematically model a transition from state s_1 to state s_2 labelled with action a?



Answer

(s_1, a, s_2)

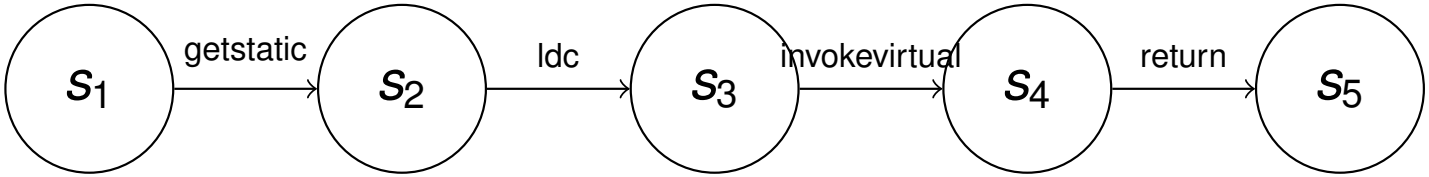
A Model of a System



Question

How can we model all the labelled transitions?

A Model of a System



Question

How can we model all the labelled transitions?

Answer

$\{(s_1, \text{getstatic}, s_2), (s_2, \text{ldc}, s_3), (s_3, \text{invokevirtual}, s_4), (s_4, \text{return}, s_5)\}$

A Model of a System

$\{(s_1, \text{getstatic}, s_2), (s_2, \text{ldc}, s_3), (s_3, \text{invokevirtual}, s_4), (s_4, \text{return}, s_5)\}$ is a subset of $S \times A \times S$

Question

$\{(s_1, \text{getstatic}, s_2), (s_2, \text{ldc}, s_3), (s_3, \text{invokevirtual}, s_4), (s_4, \text{return}, s_5)\}$ is a . . . over the sets S, A and S.

A Model of a System

$\{(s_1, \text{getstatic}, s_2), (s_2, \text{ldc}, s_3), (s_3, \text{invokevirtual}, s_4), (s_4, \text{return}, s_5)\}$ is a subset of $S \times A \times S$

Question

$\{(s_1, \text{getstatic}, s_2), (s_2, \text{ldc}, s_3), (s_3, \text{invokevirtual}, s_4), (s_4, \text{return}, s_5)\}$ is a . . . over the sets S, A and S.

Answer

relation

A Model of a System

$\{(s_1, \text{getstatic}, s_2), (s_2, \text{ldc}, s_3), (s_3, \text{invokevirtual}, s_4), (s_4, \text{return}, s_5)\}$ is a subset of $S \times A \times S$

Question

$\{(s_1, \text{getstatic}, s_2), (s_2, \text{ldc}, s_3), (s_3, \text{invokevirtual}, s_4), (s_4, \text{return}, s_5)\}$ is a . . . over the sets S, A and S.

Answer

relation

The relation is usually denoted by $\rightarrow$ and called the transition relation.

Labelled Transition System

Systems can be modelled by means of labelled transition systems.

Definition

A labelled transition system is a tuple $\langle S, A, \rightarrow, s \rangle$ consisting of

- a set S of states,
- a set A of actions,
- a transition relation $\rightarrow \subseteq S \times A \times S$, and
- a start state $s \in S$.

Labelled Transition System

Systems can be modelled by means of labelled transition systems.

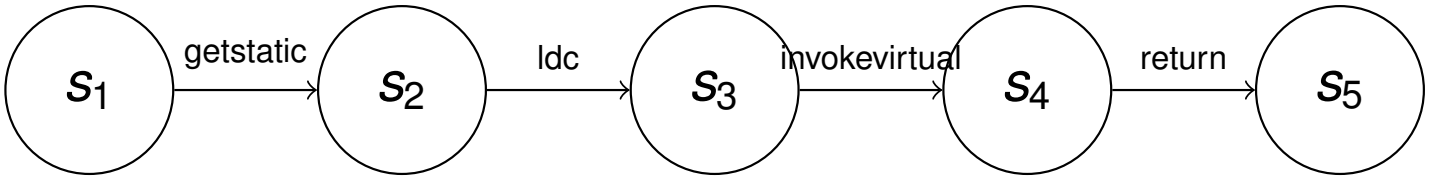
Definition

A labelled transition system is a tuple $\langle S, A, \rightarrow, s \rangle$ consisting of

- a set S of states,
- a set A of actions,
- a transition relation $\rightarrow \subseteq S \times A \times S$, and
- a start state $s \in S$.

Instead of $(s_1, a, s_2) \in \rightarrow$, we usually write $s_1 \xrightarrow{a} s_2$.

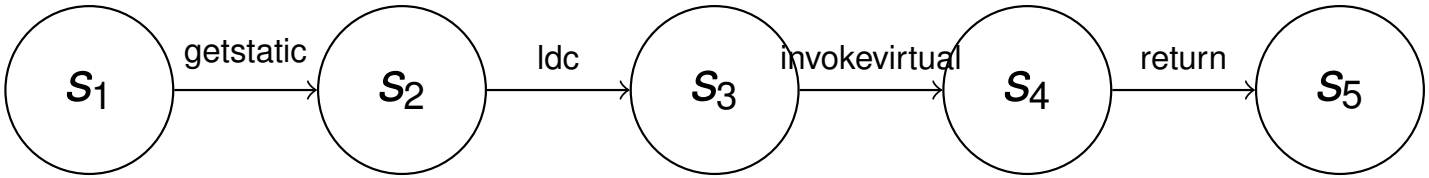
Labelled Transition System



Question

Give the corresponding labelled transition system.

Labelled Transition System



Question

Give the corresponding labelled transition system.

Answer

$$\langle \{s_1, s_2, s_3, s_4, s_5\},$$
$$\{\text{getstatic}, \text{ldc}, \text{invokevirtual}, \text{return}\},$$
$$\{(s_1, \text{getstatic}, s_2), (s_2, \text{ldc}, s_3), (s_3, \text{invokevirtual}, s_4), (s_4, \text{return}, s_5)\},$$
$$s_1 \rangle$$