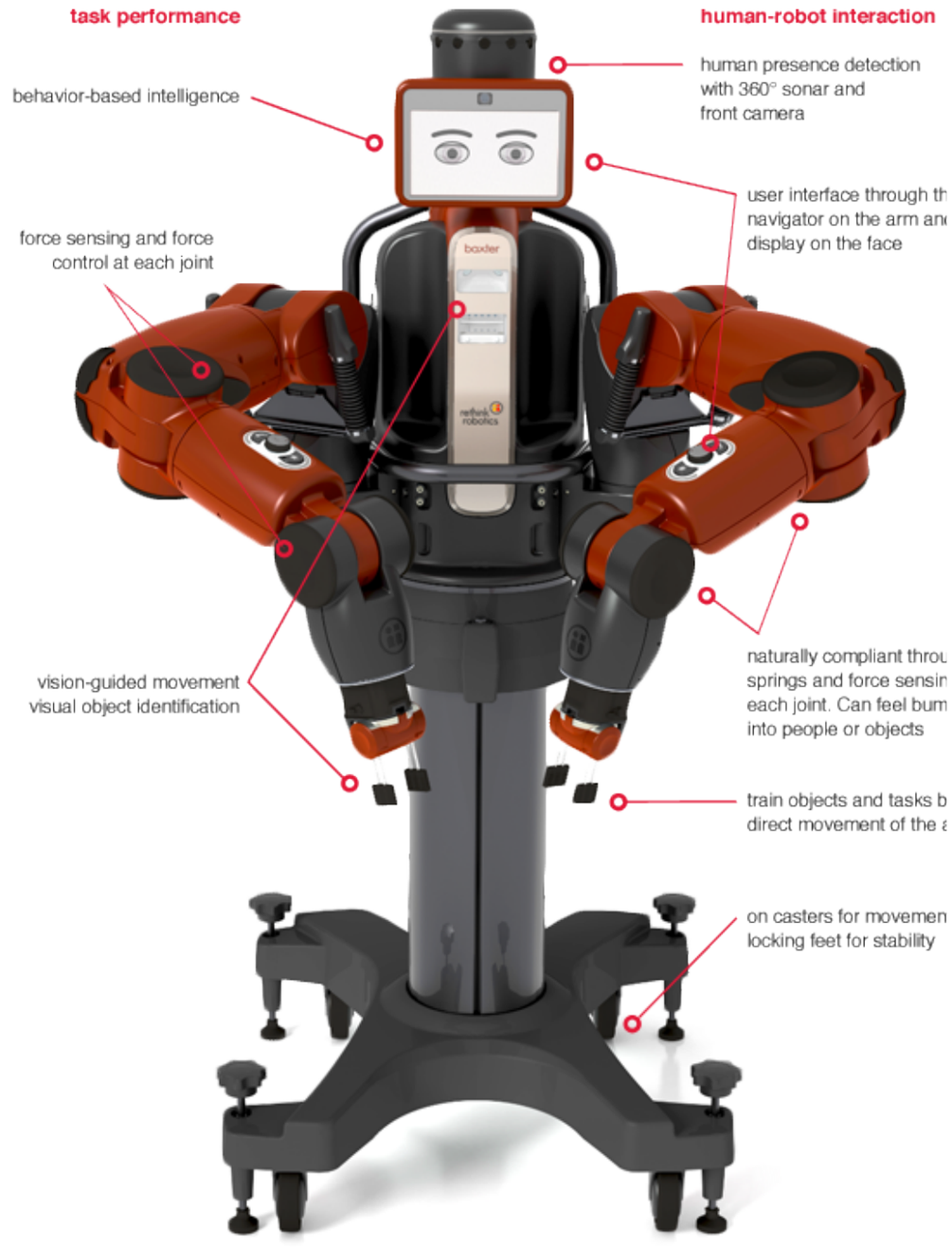
Inheritance

A problem with interfaces

TurtleBot

PR2



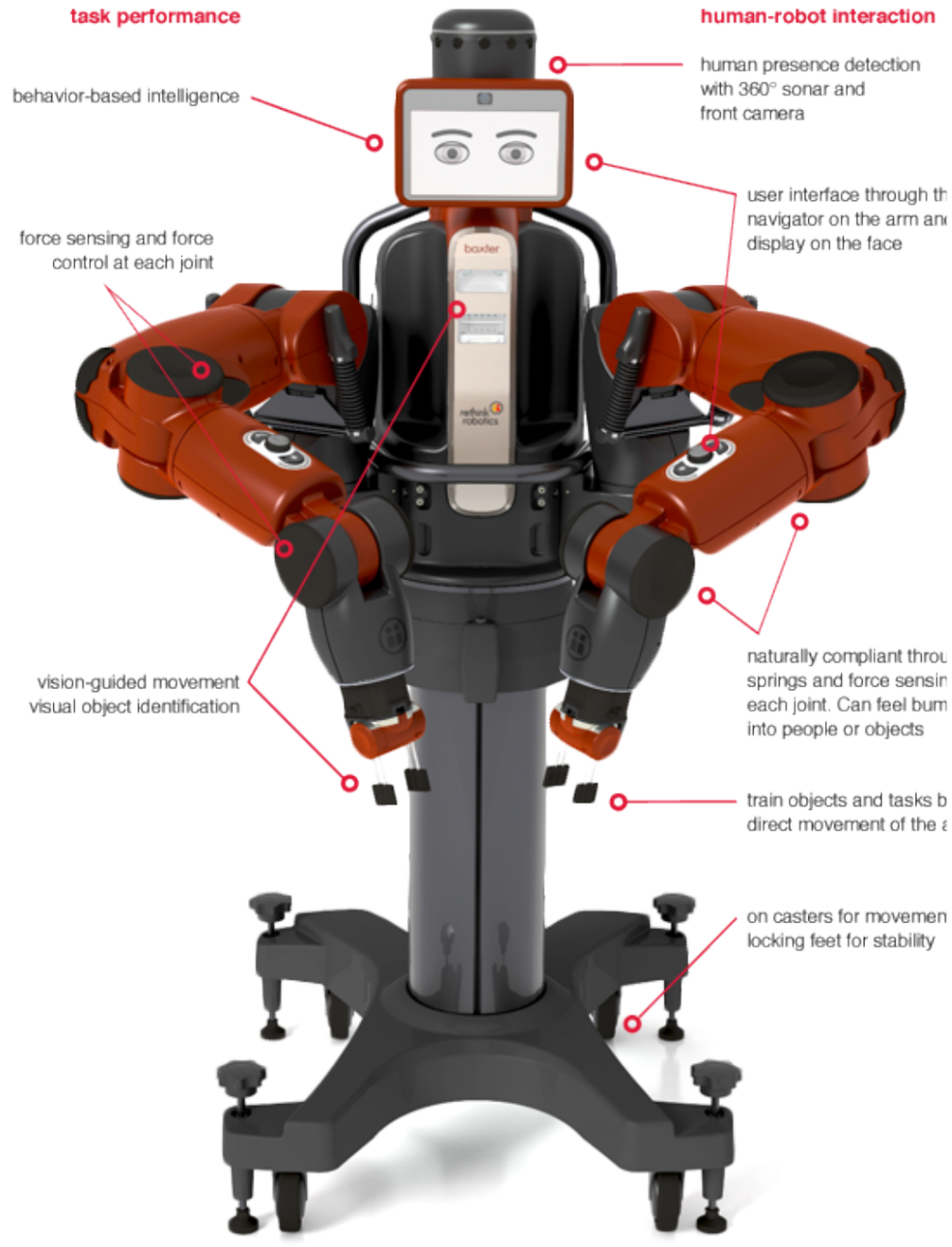
Baxter

The orange, it burns...it burns!

A problem with interfaces

TurtleBot

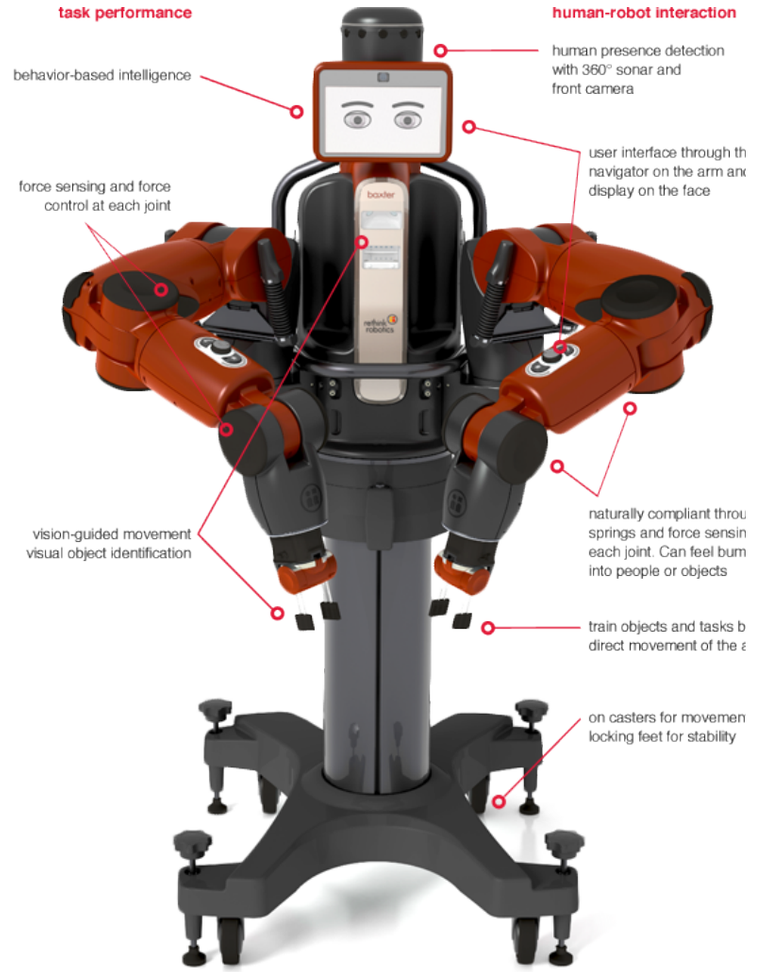
PR2



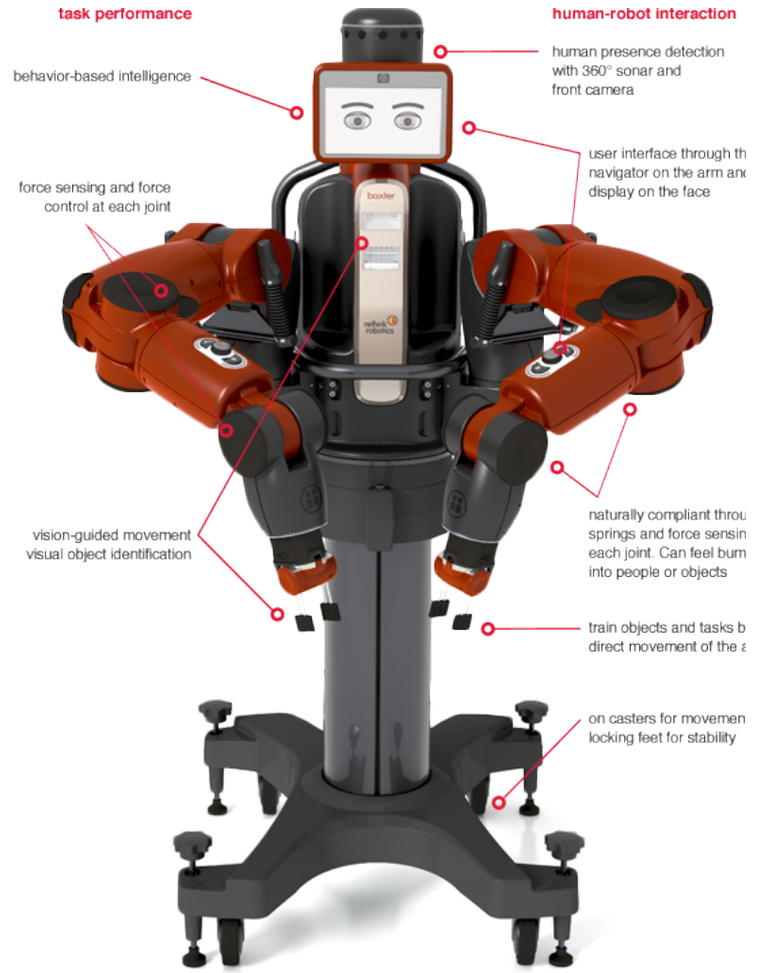
Baxter

http://goo.gl/pNaVn

```
public class Robot {
    private String myName;
    public Robot(String name) {
        myName = name;
    }

    public String getName() {
        return myName;
    }
}
```



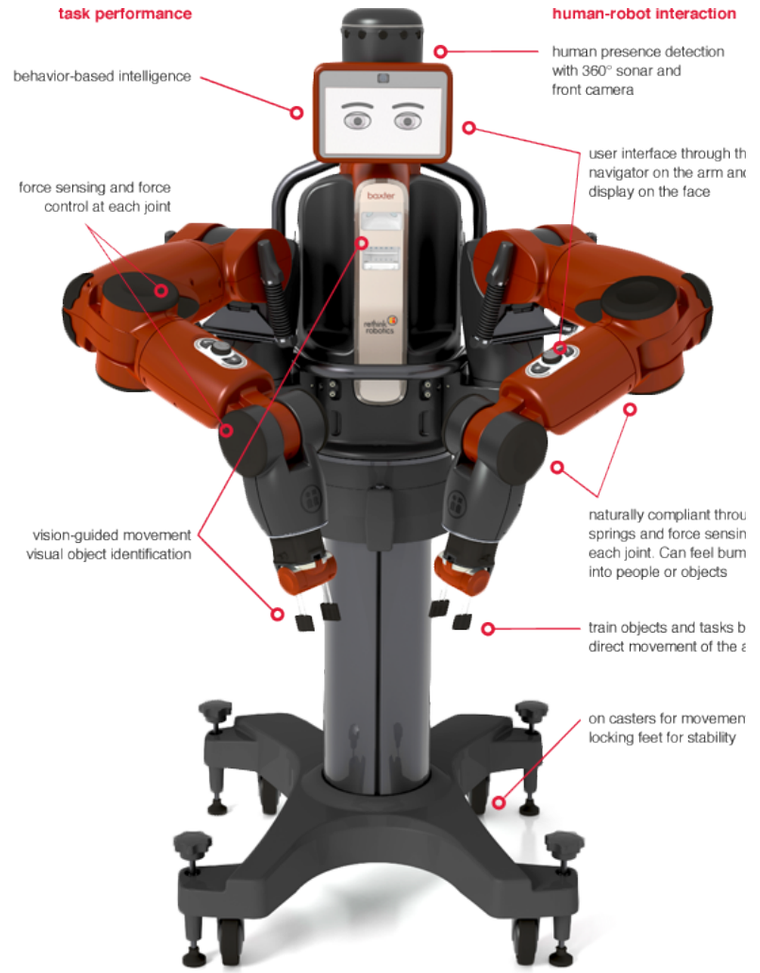
```java
public class Robot {
   private String myName;
   public Robot(String name) {
      myName = name;
   }

   public String getName() {
      return myName;
   }
}
```




```java
public class TwoArmedRobot extends Robot {
   public TwoArmedRobot(String name) {
      super(name);
   }

   public void raiseArms() {
      // ...arm control code.
   }
}
```

Subclass *Superclass*

```
public class TwoArmedRobot extends Robot {
   public TwoArmedRobot(String name) {
      super(name);
   }

   public void raiseArms() {
      // ...arm control code.
   }
}
```




```
Robot r = new Robot("R2D2");
System.out.println("r is named " + r.getName());
r.raiseArms(); // DOES NOT WORK
TwoArmedRobot pr2 = new TwoArmedRobot("pr2");
System.out.println("pr2 is named " + pr2.getName());
pr2.raiseArms();
```

Dispatch

```
public class A {
   // Code.
}
```

```
C c = new C();
c.doSomethingCool();
```

```
public class B extends A {
   // Code.
}
```

```
public class C extends B {
   // Code!
}
```

Check if class C has the method (or instance variable).
If not, check if class B has it.
If not, check if class A has it.
If not, check if Object has it.

Dispatch

```
public class A {
   public int x;
   public void doSomethingOk() {
     // Code!
   }
}

public class B extends A {
   public void doSomethingCool() {
     // Code!
   }
}

public class C extends B {
   public void doSomethingAwesome() {
     // Code!
   }
}
```

Capital letter "Oh"

http://goo.gl/fOVXq

Check if I have the method or variable.
Then (recursively) check my superclass.

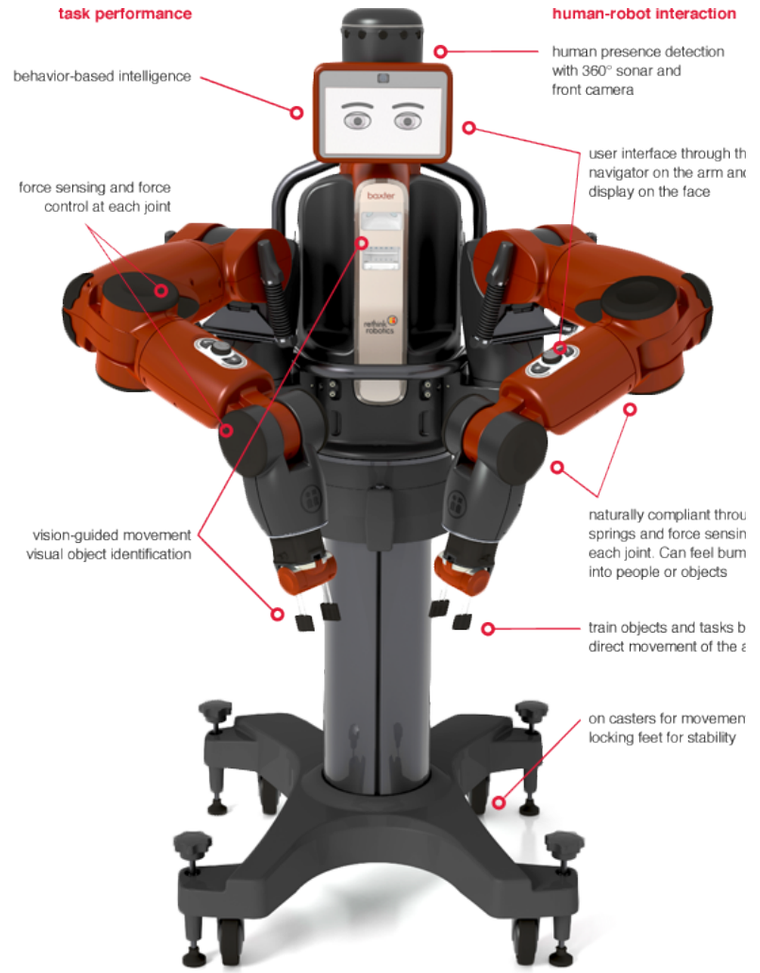
Overloading

```
public class D {
   public void doSomething() {
      System.out.println(“D”);
   }
}
```

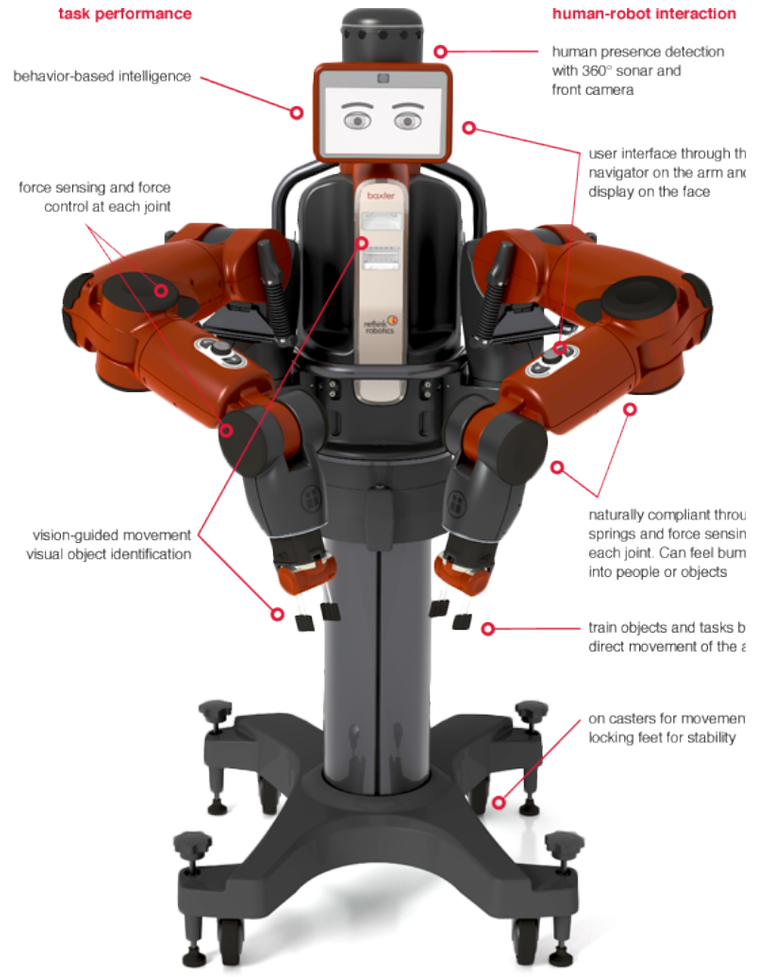
```
public class E extends D{
   public void doSomething() {
      System.out.println(“E”);
   }
}
```

Check if I have the method or variable.
Then (recursively) check my superclass.

```java
public class Robot {
    private String myName;
    public Robot(String name) {
        myName = name;
    }

    public String getName() {
        return myName;
    }
}

public class TwoArmedRobot extends Robot {
    public TwoArmedRobot(String name) {
        super(name);
    }

    public void raiseArms() {
        // ...arm control code.
    }
}
```



```
public abstract class Robot {
    private String myName;
    public Robot(String name) {
        myName = name;
    }

    public String getName() {
        return myName;
    }

    public abstract void eStop();
}

public class TwoArmedRobot extends Robot {
    public TwoArmedRobot(String name) {
        super(name);
    }

    public void raiseArms() {
        // ...arm control code.
    }

    public void eStop() {
        // STOP.
    }
}
```



Abstract methods work like interfaces.

When do I use...

Interfaces are very common:
"These classes can all do the same stuff."

Data inheritance is pretty common:
"These classes need some of the same instance variables."

Code inheritance is less common:
"These classes share part of *an implementation.*"

You can implement *multiple interfaces; you can only* extend *one class.*

To look into, if you're curious: the "protected" and "final" keywords.

Boggle!

Words words words

Words words words

Words words words

Words words words

...and others.

Words words words

...and others.

Automatic Boggle

You may assume a set dictionary of words.

We want an algorithm for finding every word on the board.

Tries!

Is ‘e’ a prefix?

Tries!

Is ‘e’ a prefix?

Is ‘ey’ a prefix?

Tries!

Is ‘e’ a prefix?

Is ‘ey’ a prefix?

Is ‘eyr’ a prefix?

Tries!

Is ‘e’ a prefix?

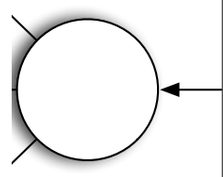
Is ‘ey’ a prefix?

Is ‘eyr’ a prefix?

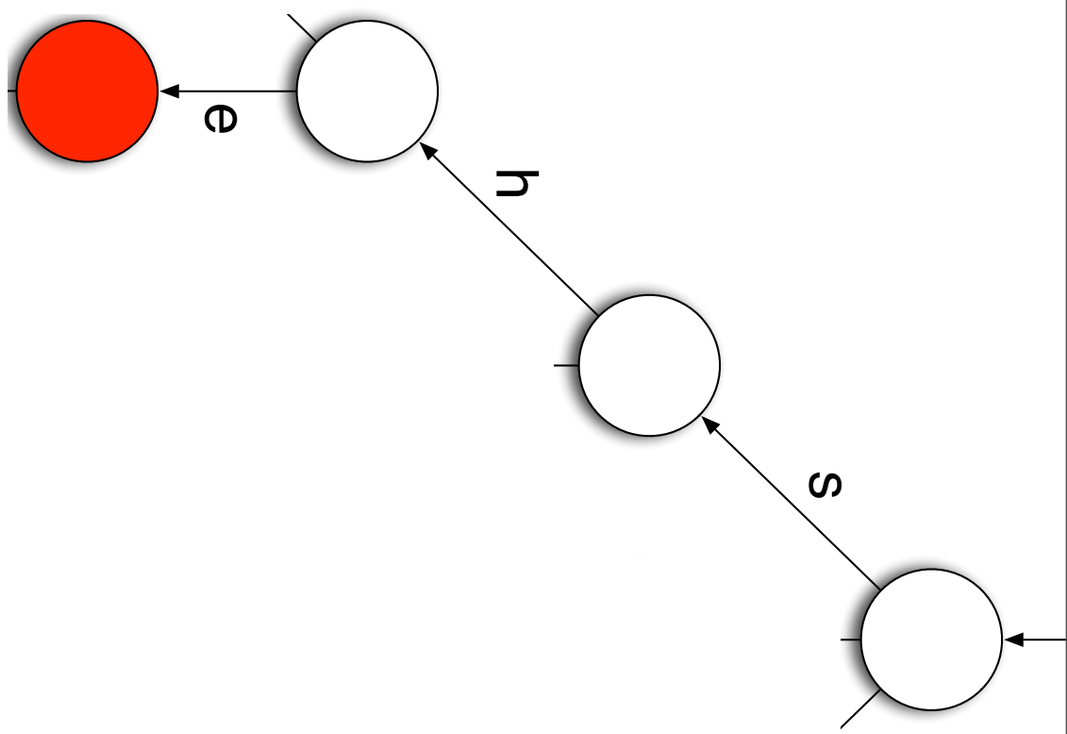
Is ‘eyri’ a prefix?

(and then we’re stuck)

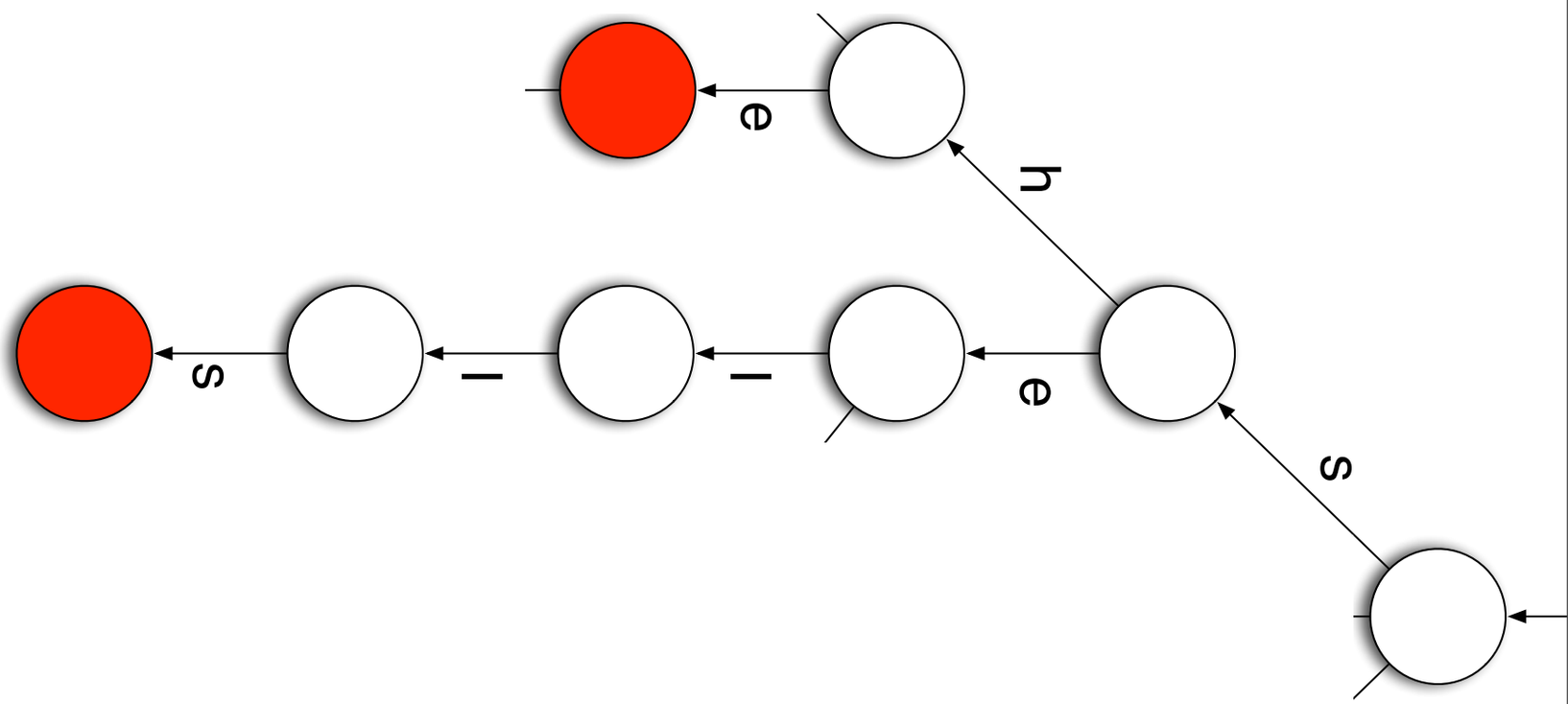
“She sells sea shells by the sea shore.”



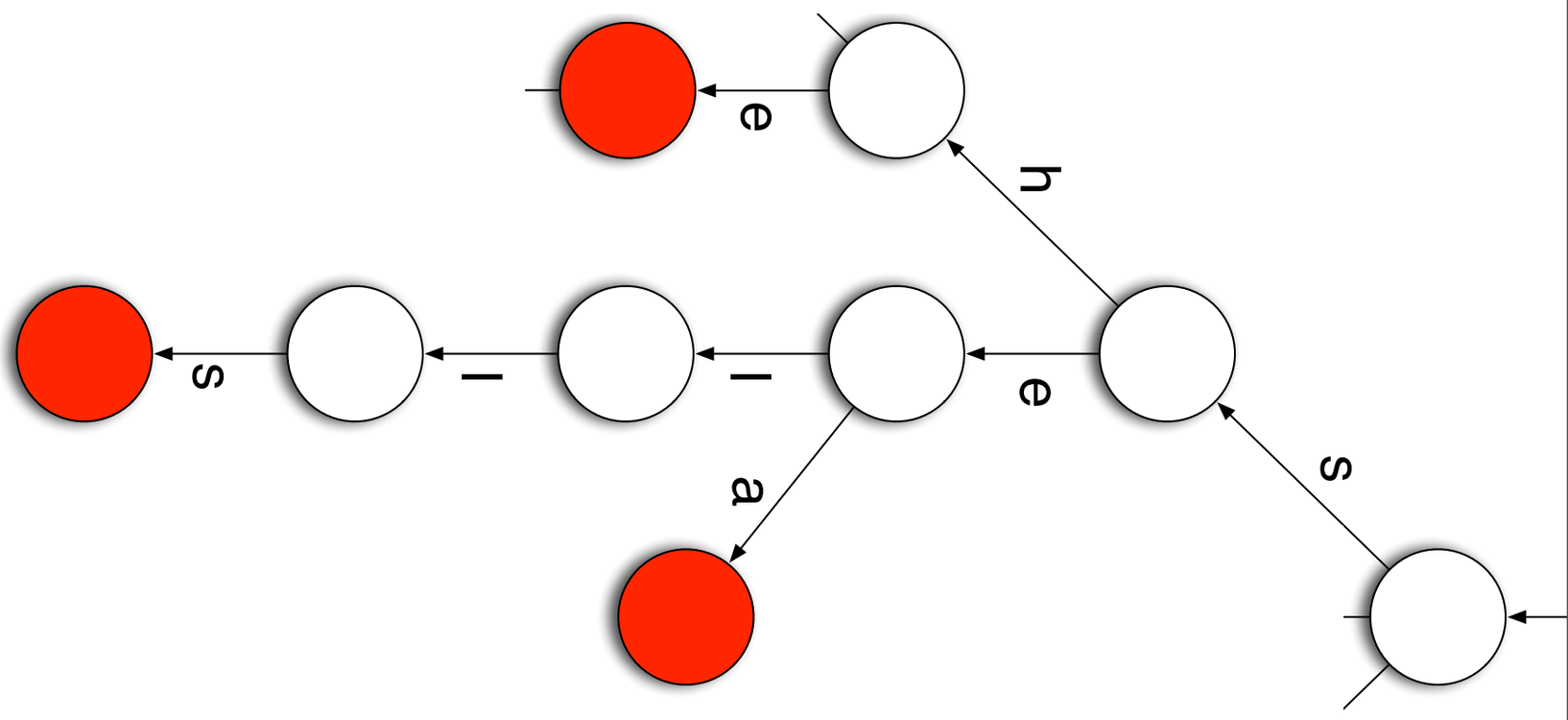
“She sells sea shells by the sea shore.”



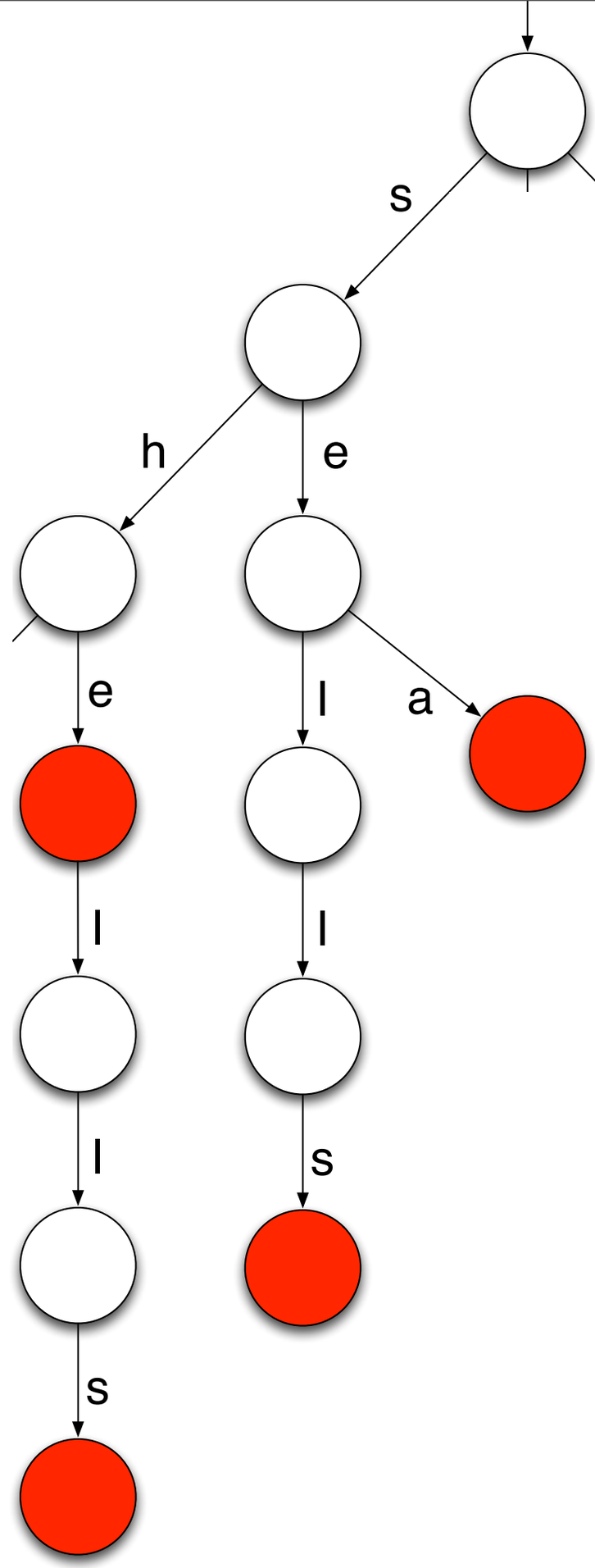
“She sells sea shells by the sea shore.”



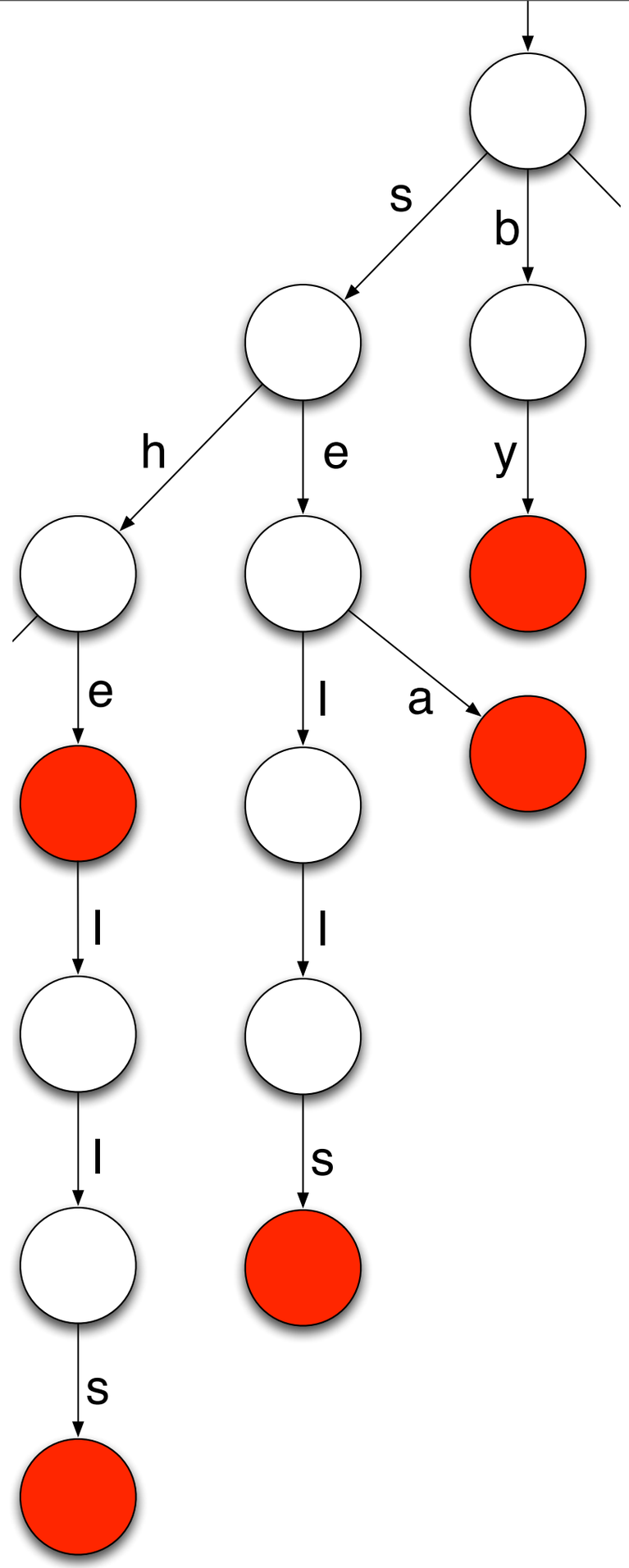
“She sells sea shells by the sea shore.”



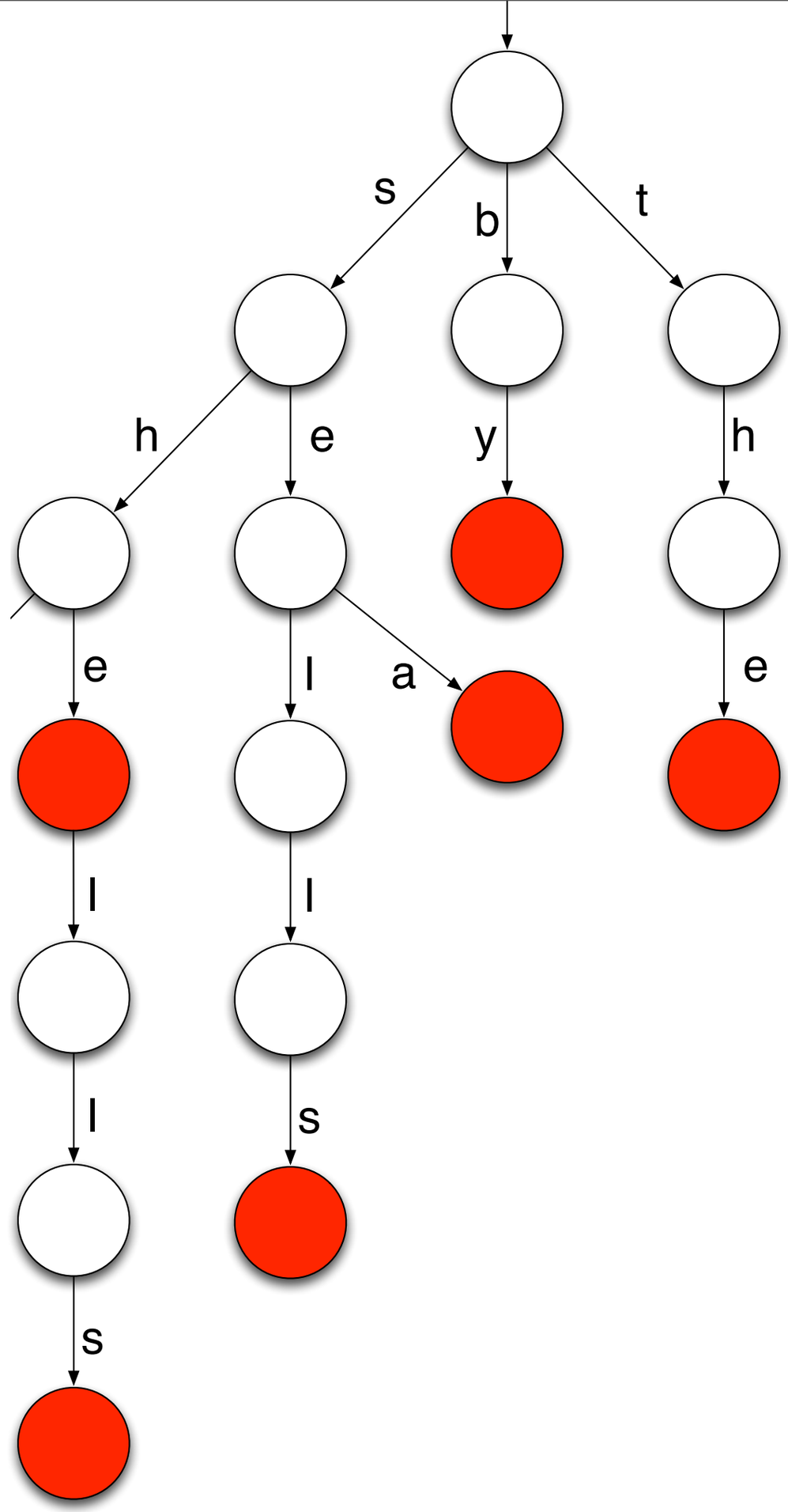
“She sells sea shells by the sea shore.”



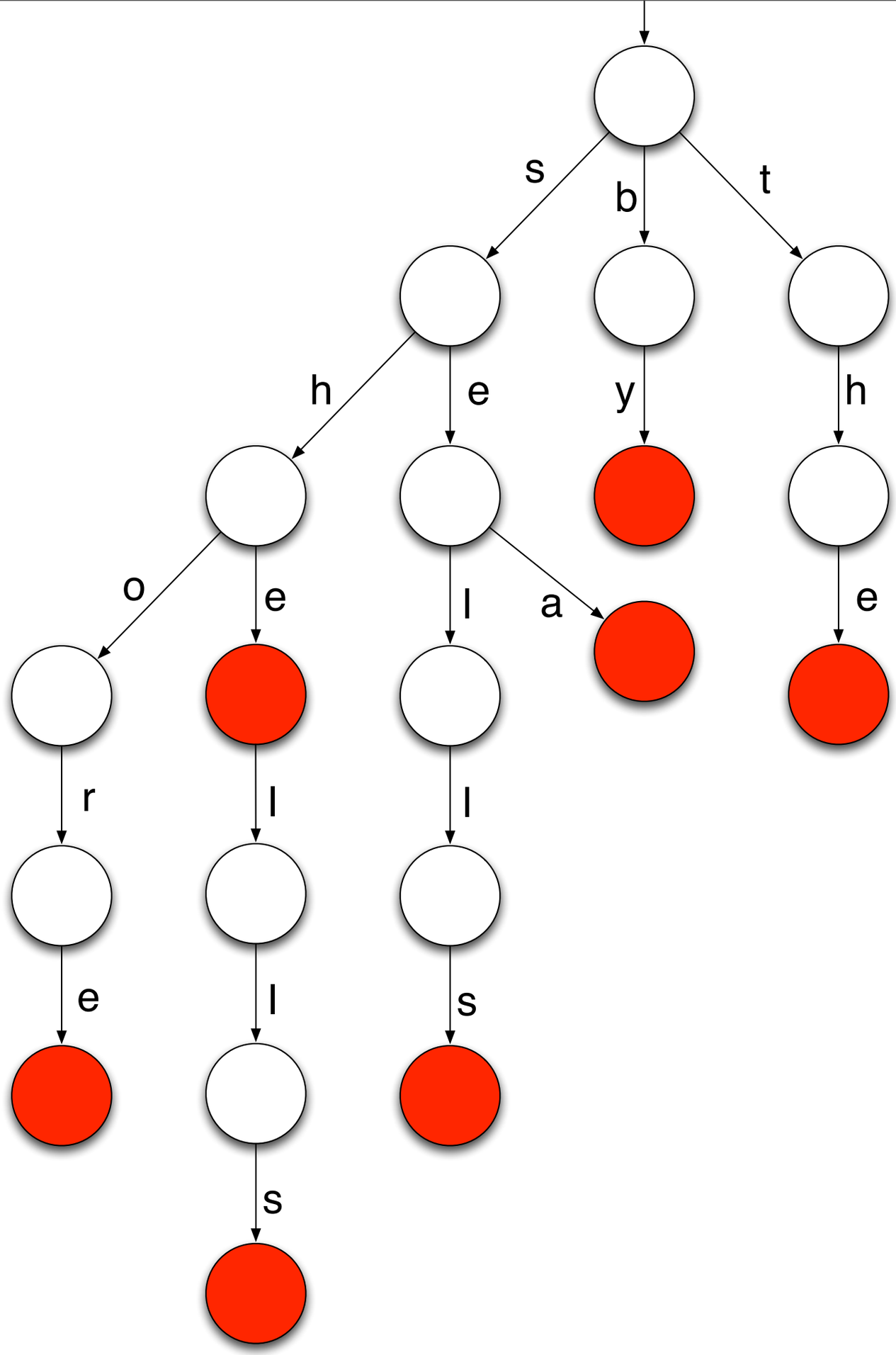
“She sells sea shells by the sea shore.”



“She sells sea shells by the sea shore.”



“She sells sea shells by the sea shore.”



“Peter piper picked a peck of pickled peppers”

“Xavier has x-rayed his xylophone”

“Four score and seven years ago”

“Ulysses usually uses union u-boats”