Hyperproperties

Systems, Properties, and Hyperproperties

A Hyperproperty is a set of Systems

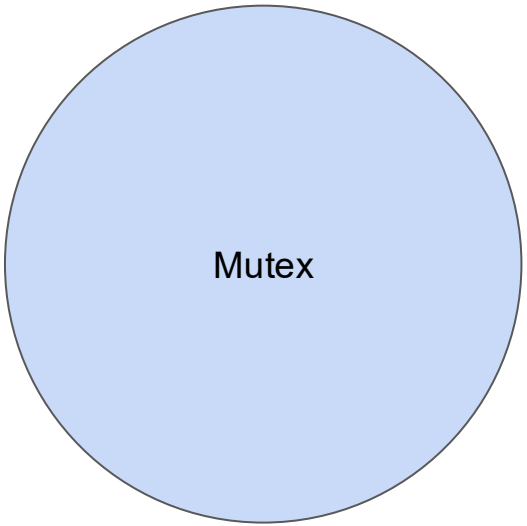
The End

We now know what hyperproperties are.

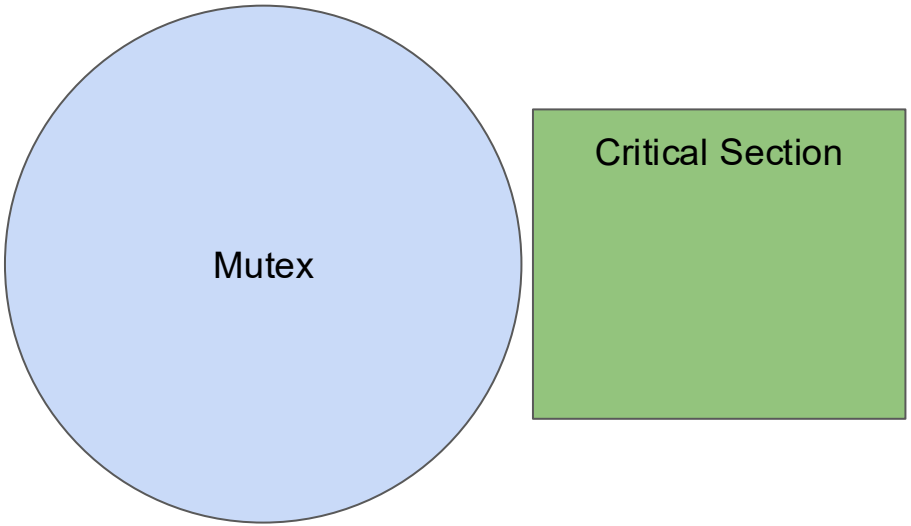
Any questions?

Let’s specify ‘correct’ mutual exclusion

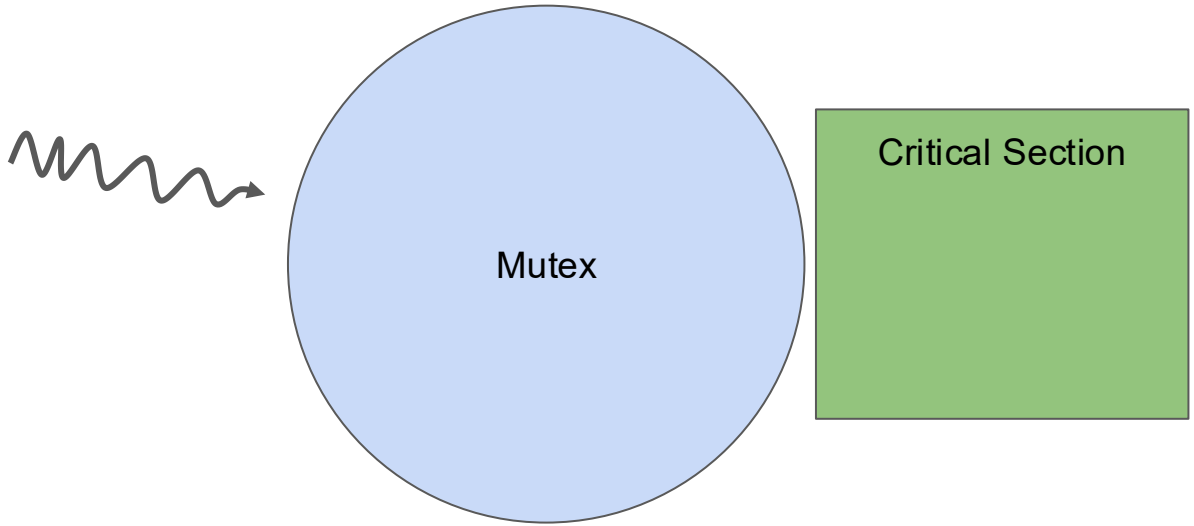
Let’s specify ‘correct’ mutual exclusion



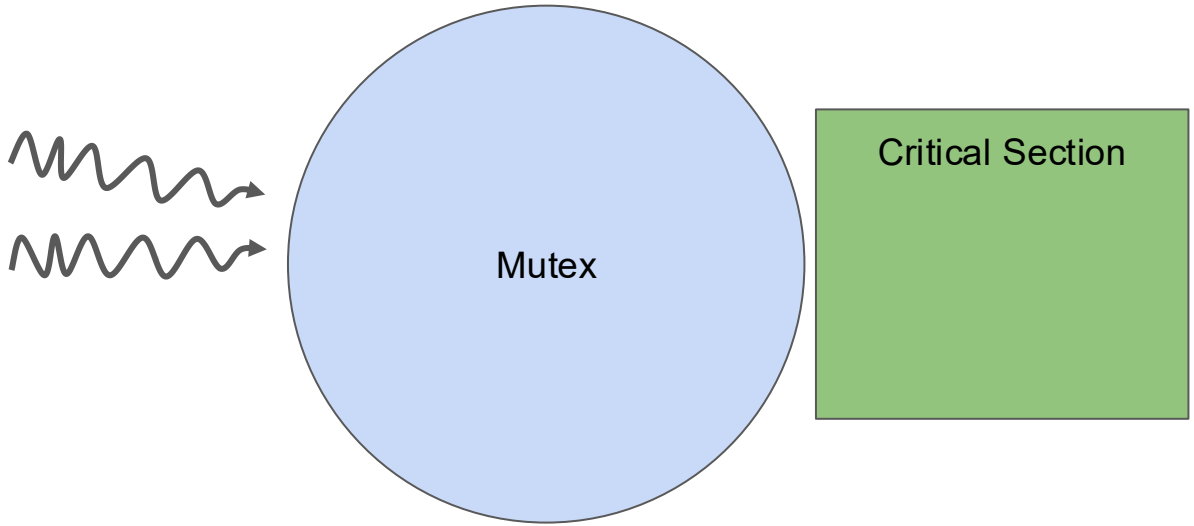
Let’s specify ‘correct’ mutual exclusion



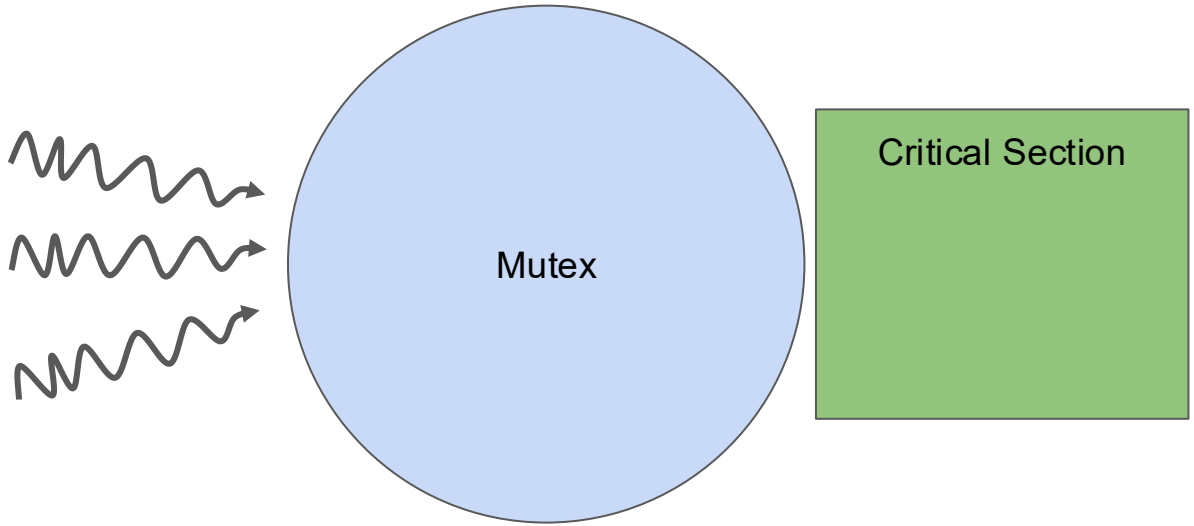
Let’s specify ‘correct’ mutual exclusion



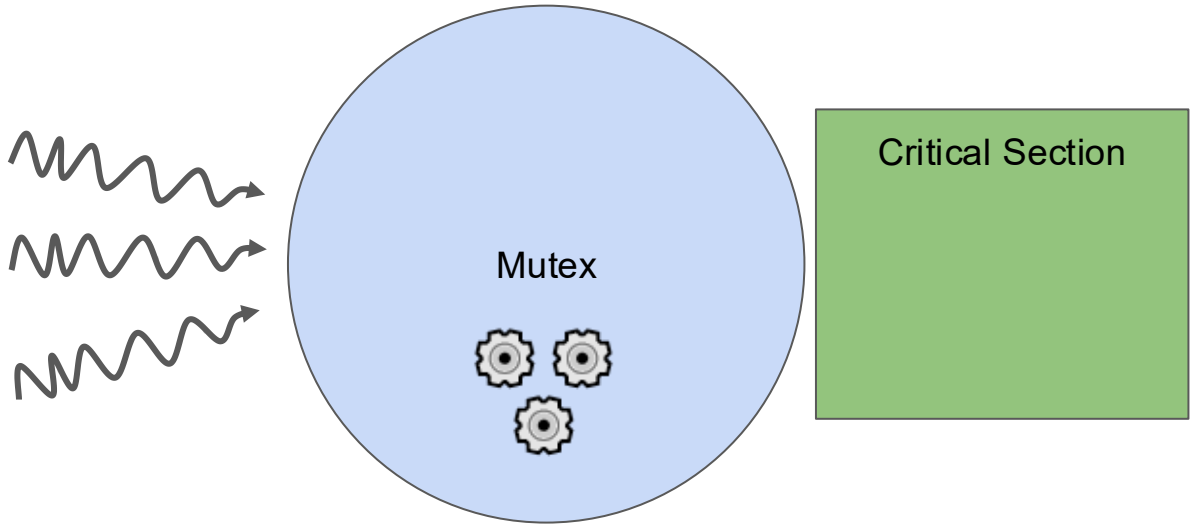
Let’s specify ‘correct’ mutual exclusion



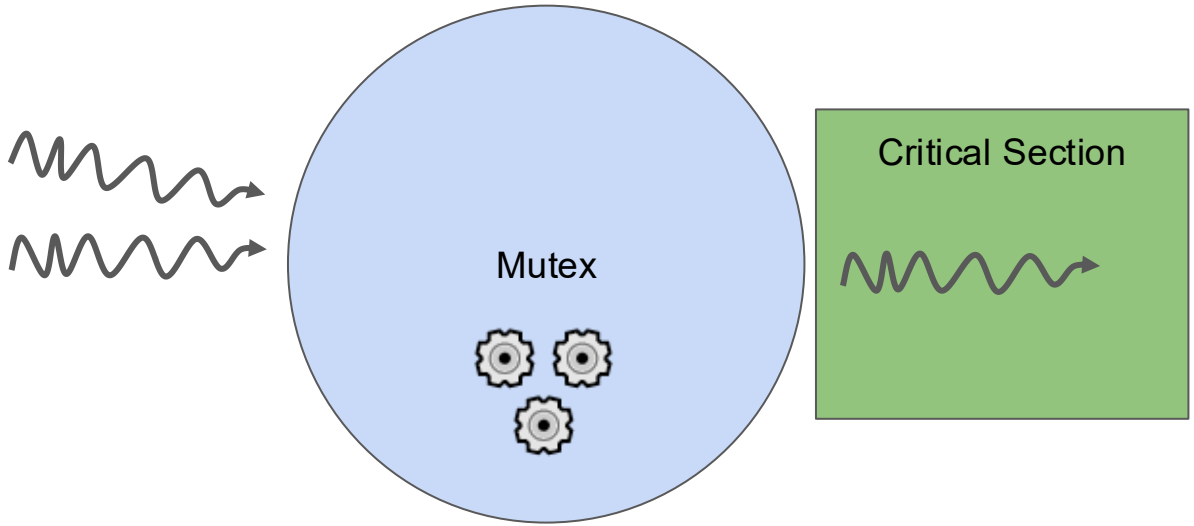
Let’s specify ‘correct’ mutual exclusion



Let’s specify ‘correct’ mutual exclusion

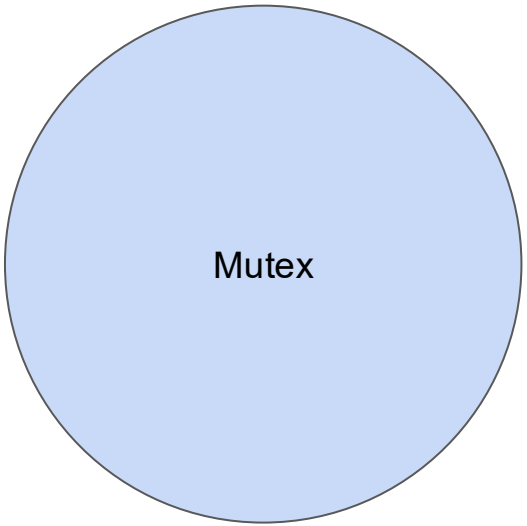


Let’s specify ‘correct’ mutual exclusion

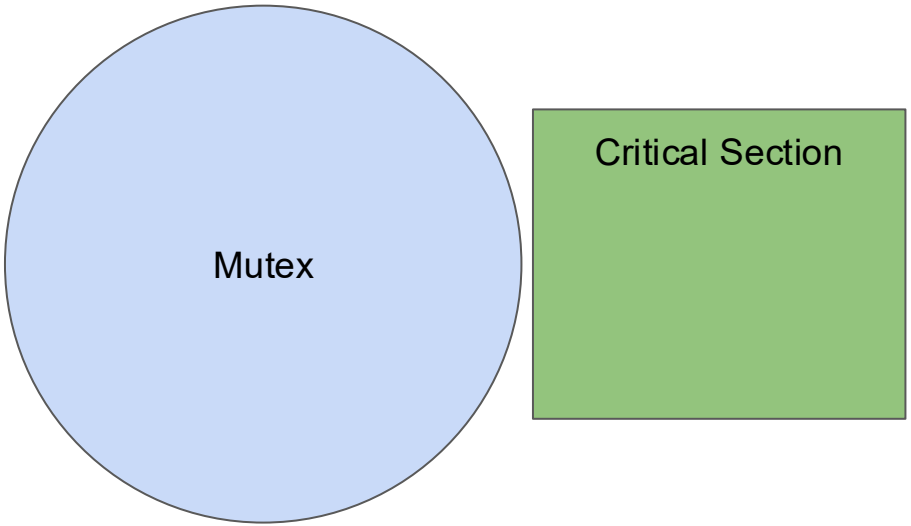


What we want - 1

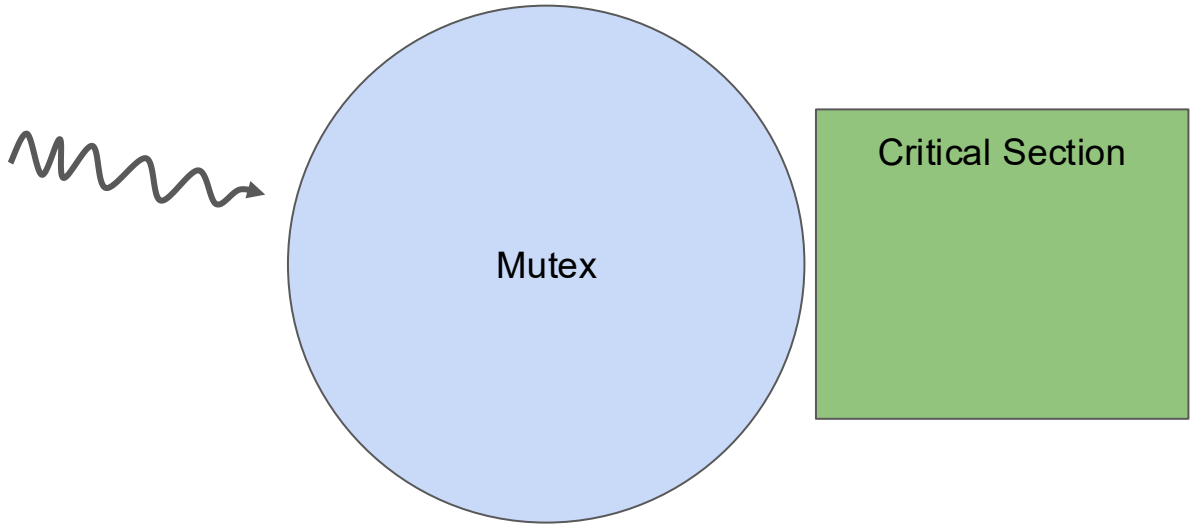
What we want - 1



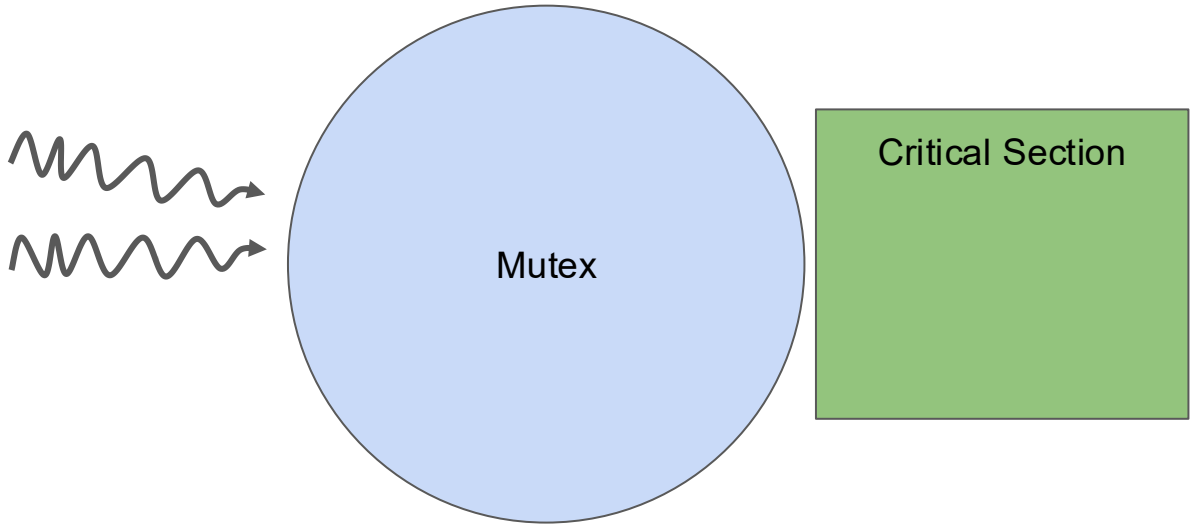
What we want - 1



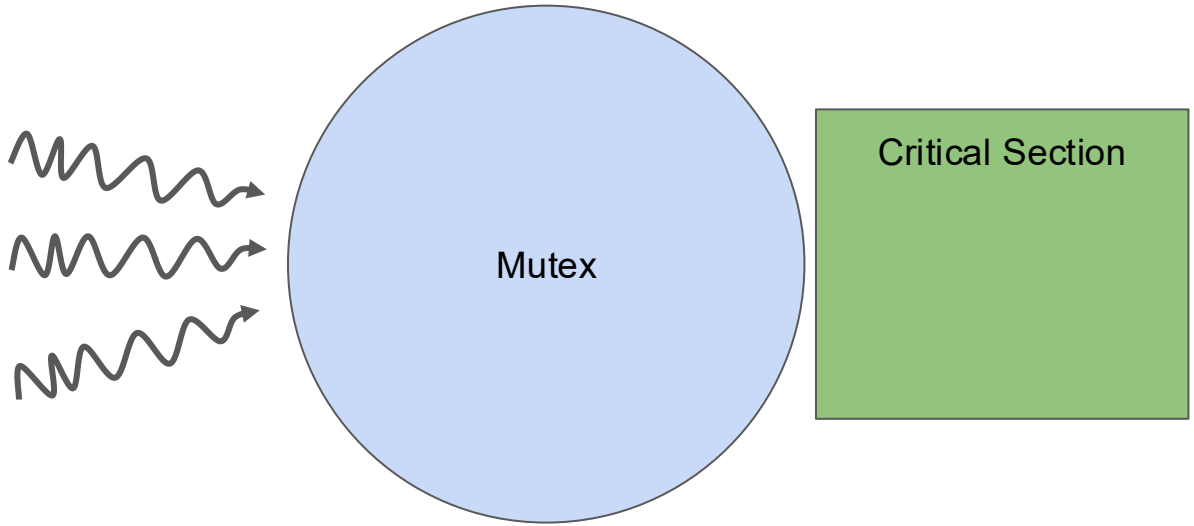
What we want - 1



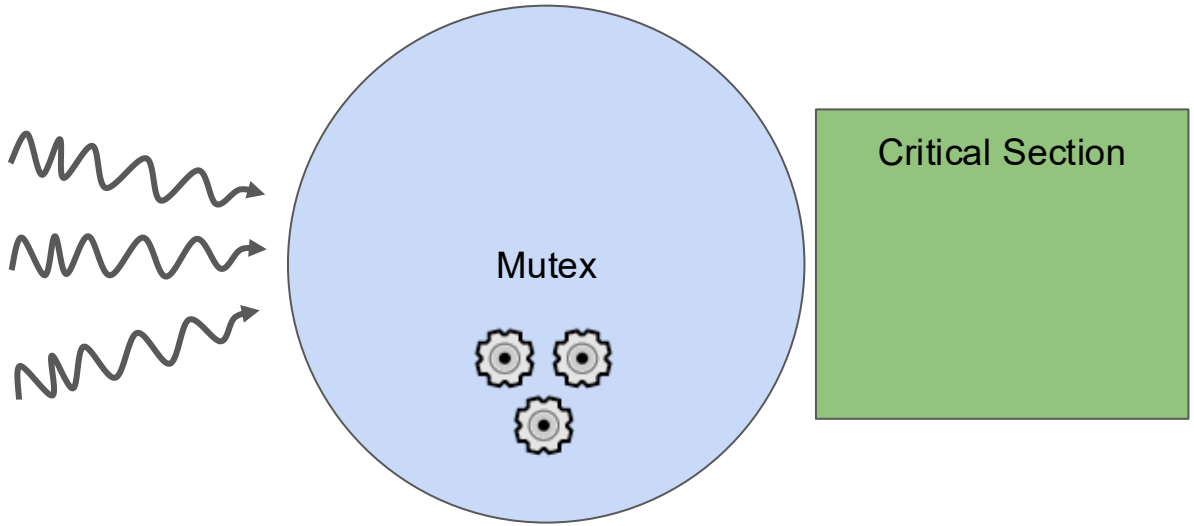
What we want - 1



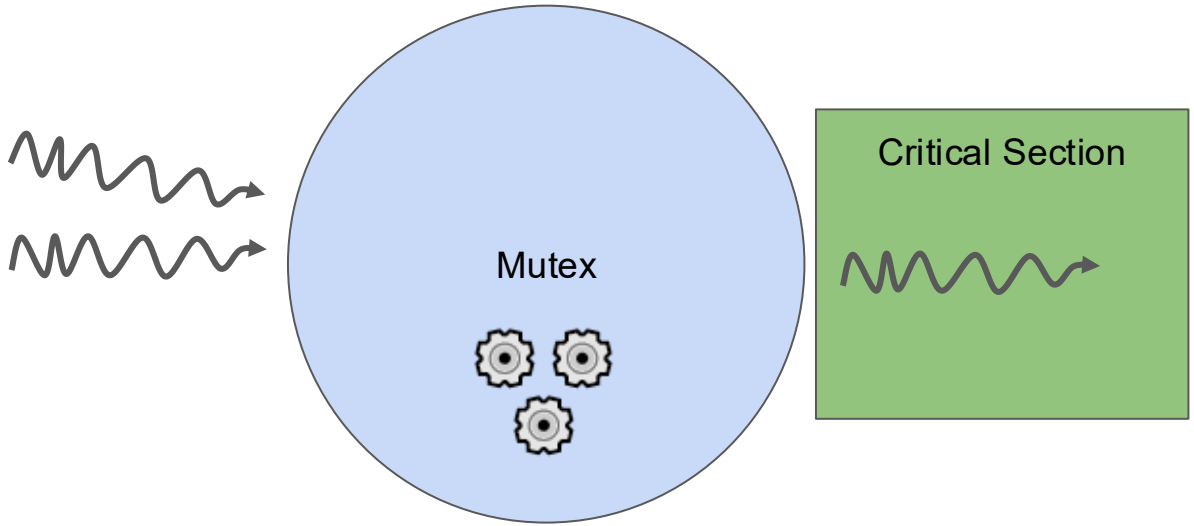
What we want - 1



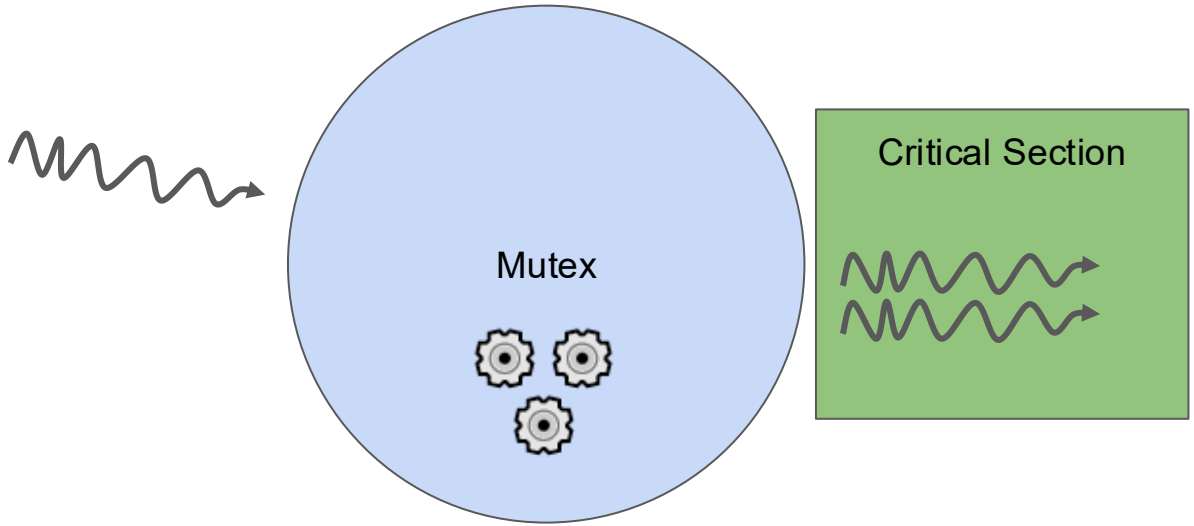
What we want - 1



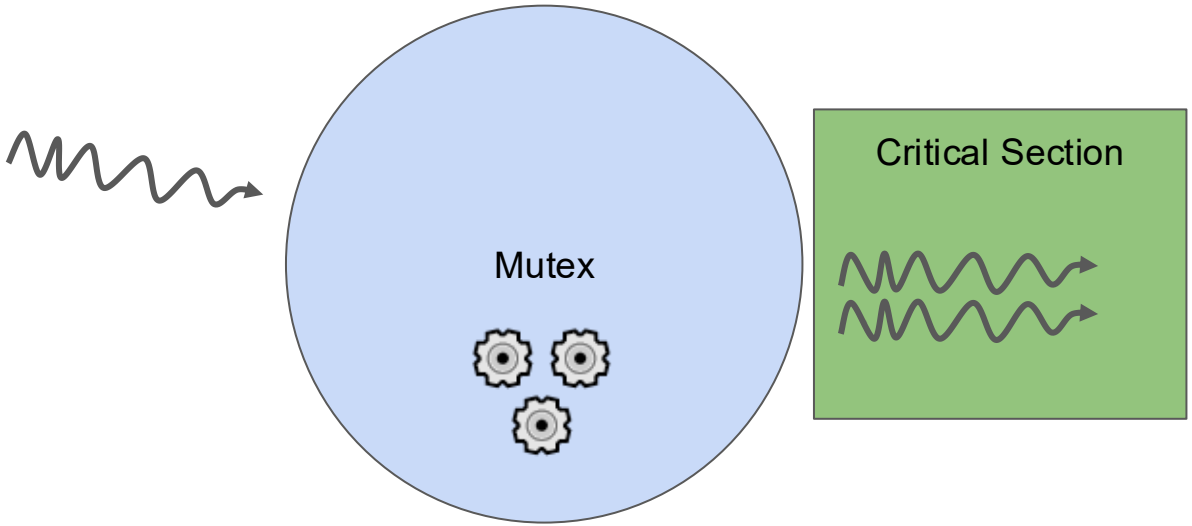
What we want - 1



What we want - 1

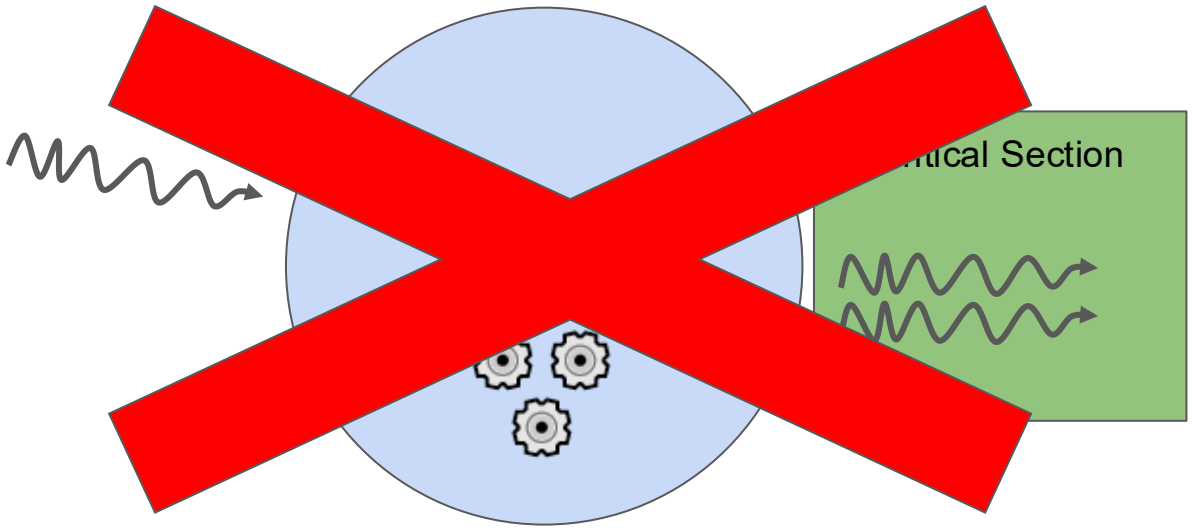


What we want - 1



Oops! Something bad happened!

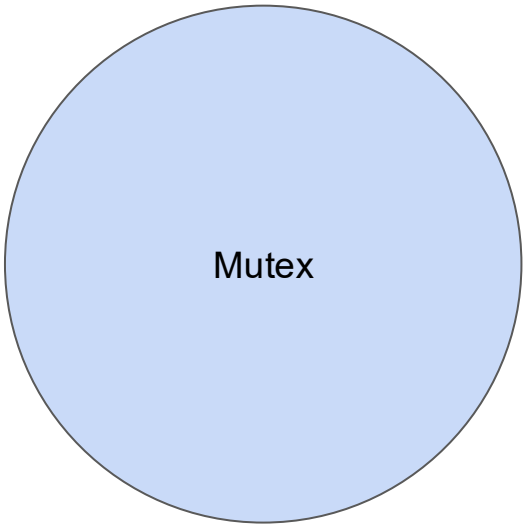
What we want - 1



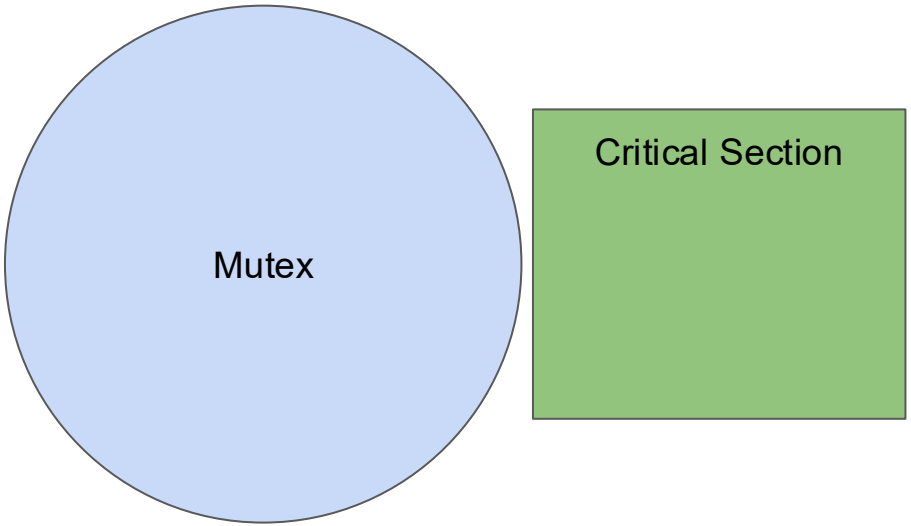
Oops! Something bad happened!

What we want - 2

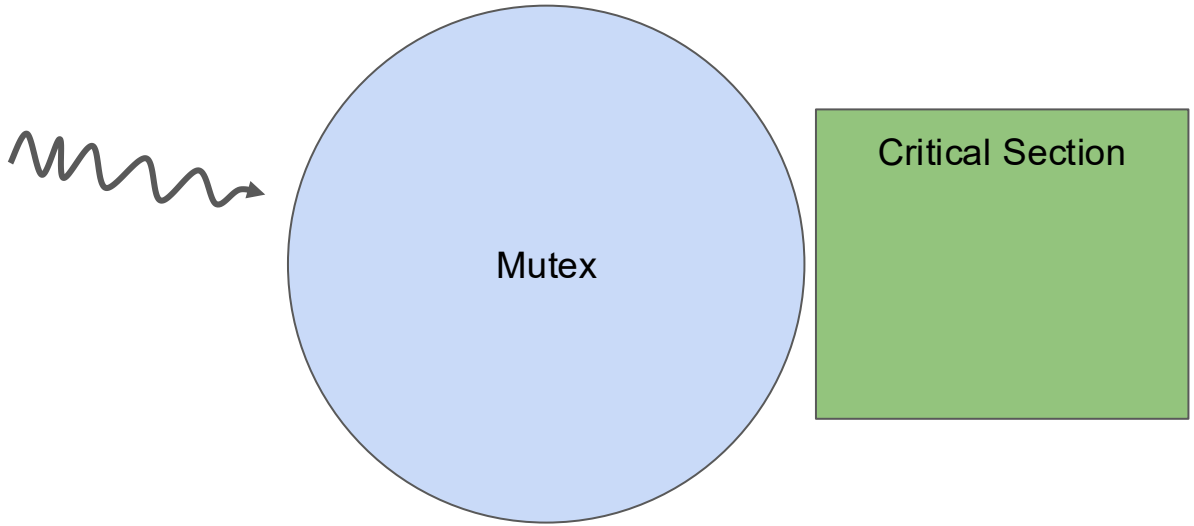
What we want - 2



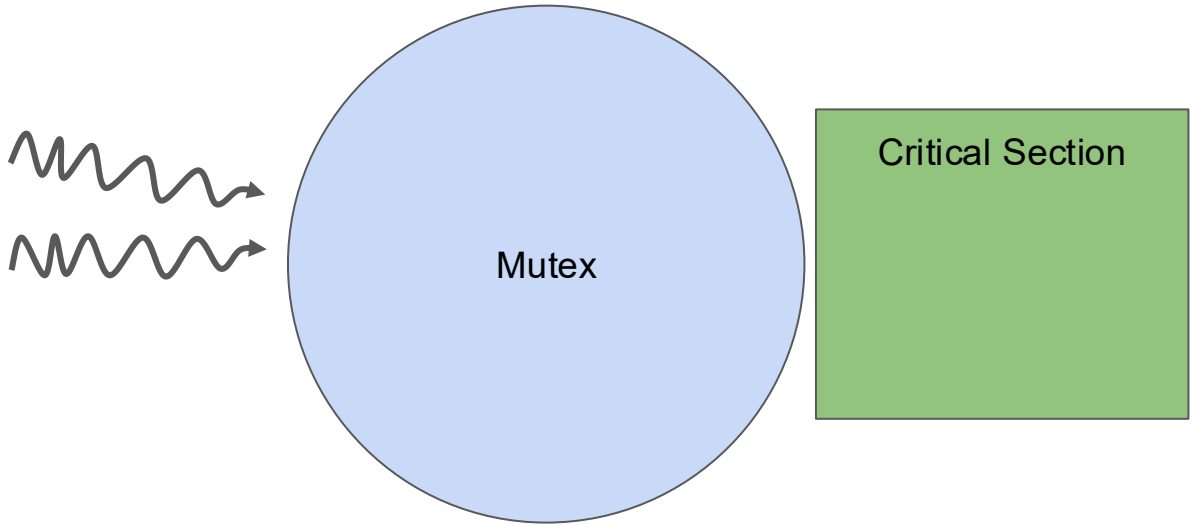
What we want - 2



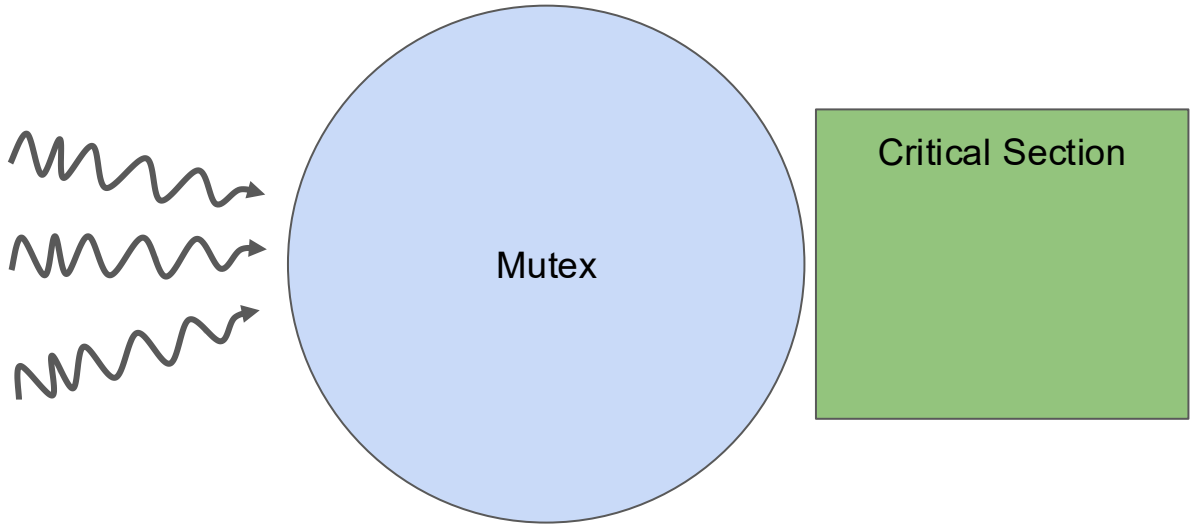
What we want - 2



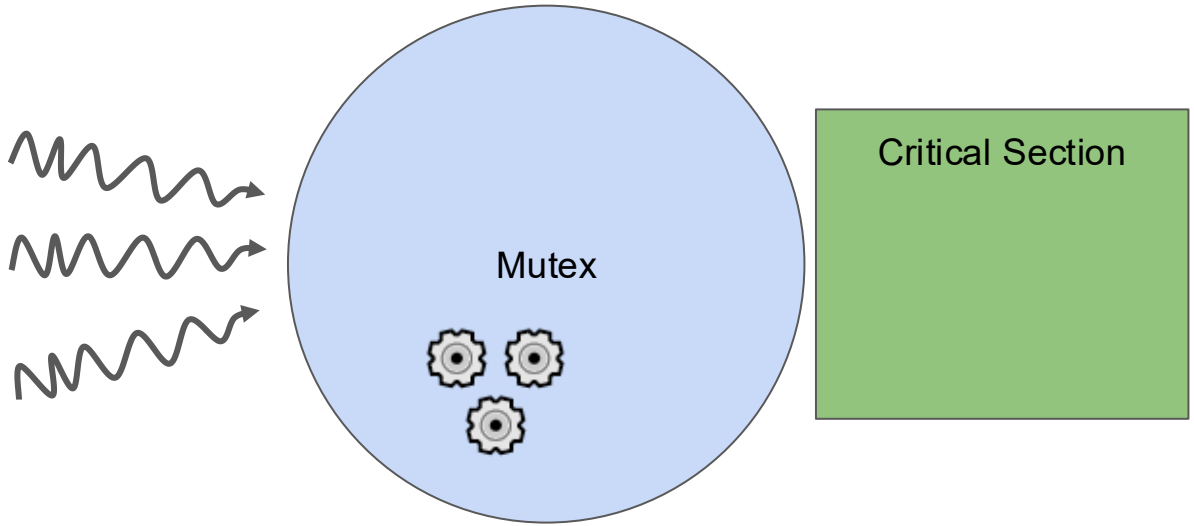
What we want - 2



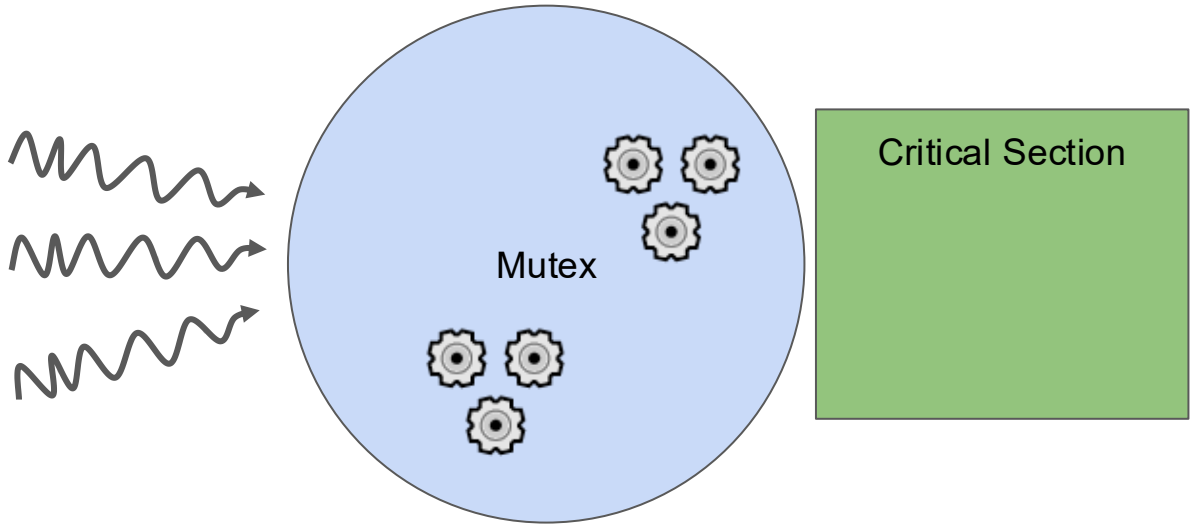
What we want - 2



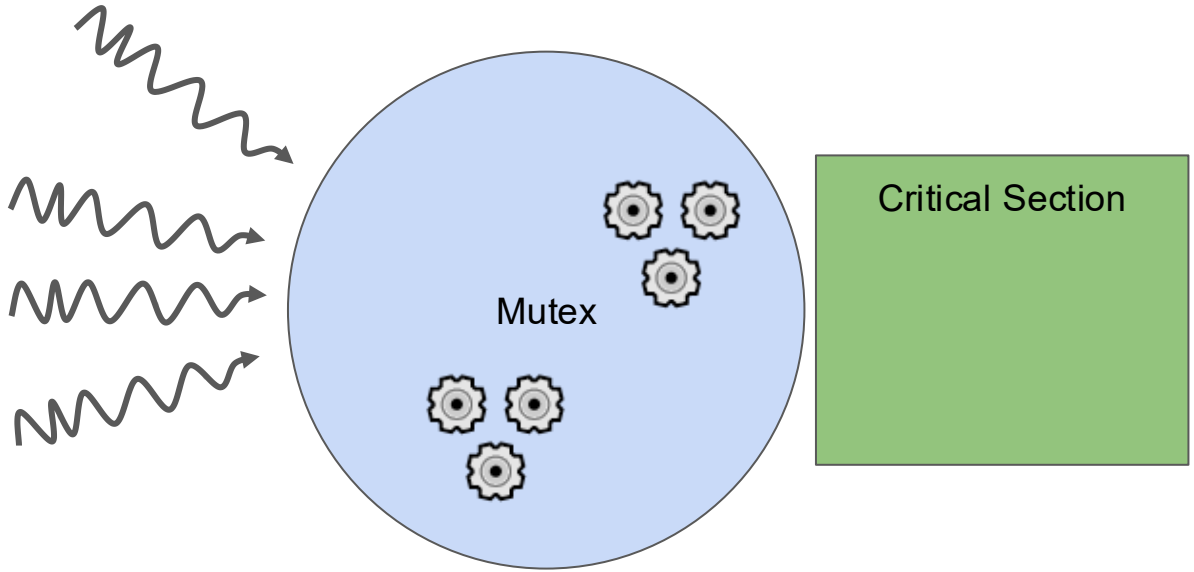
What we want - 2



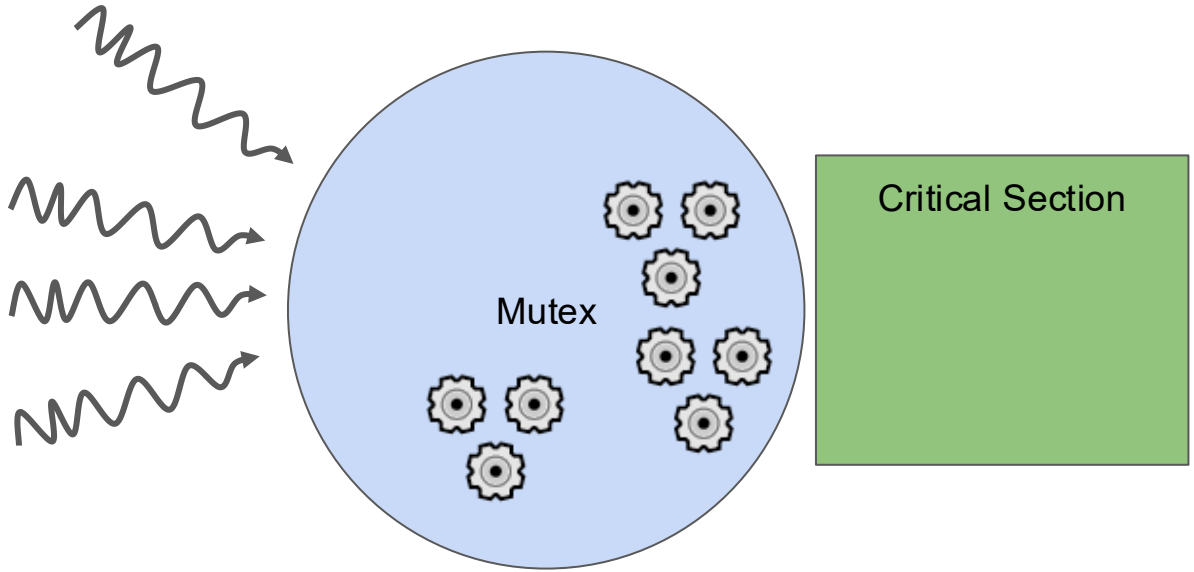
What we want - 2



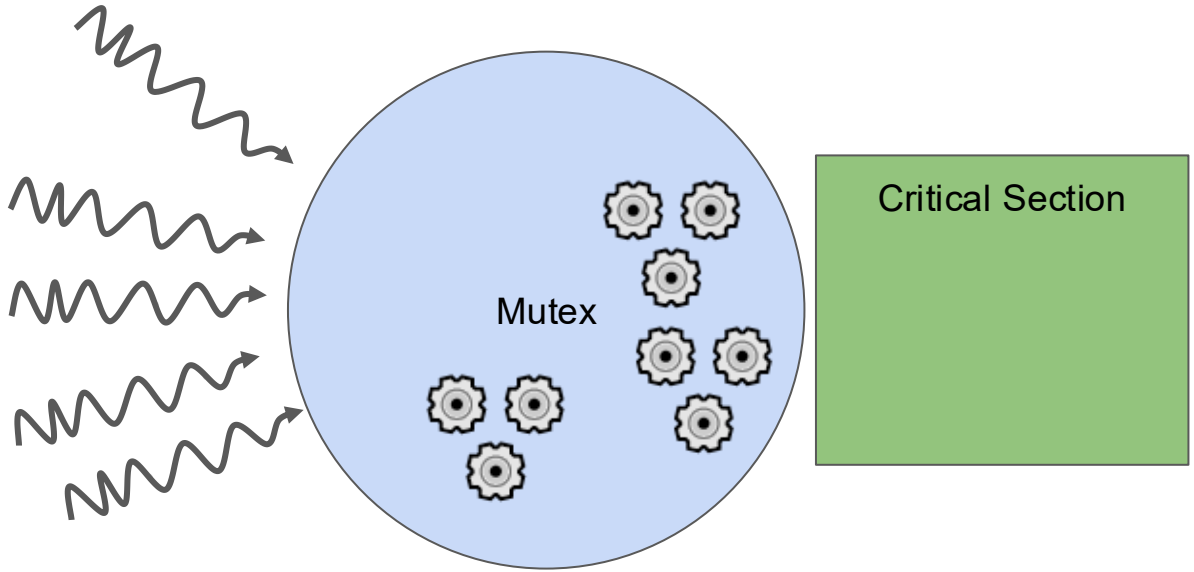
What we want - 2



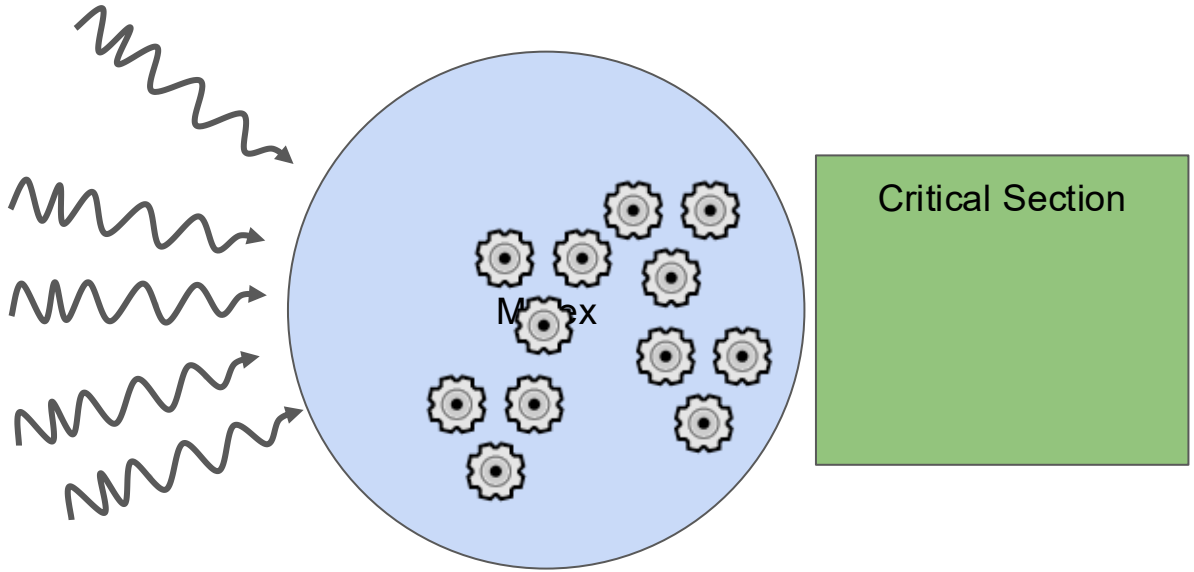
What we want - 2



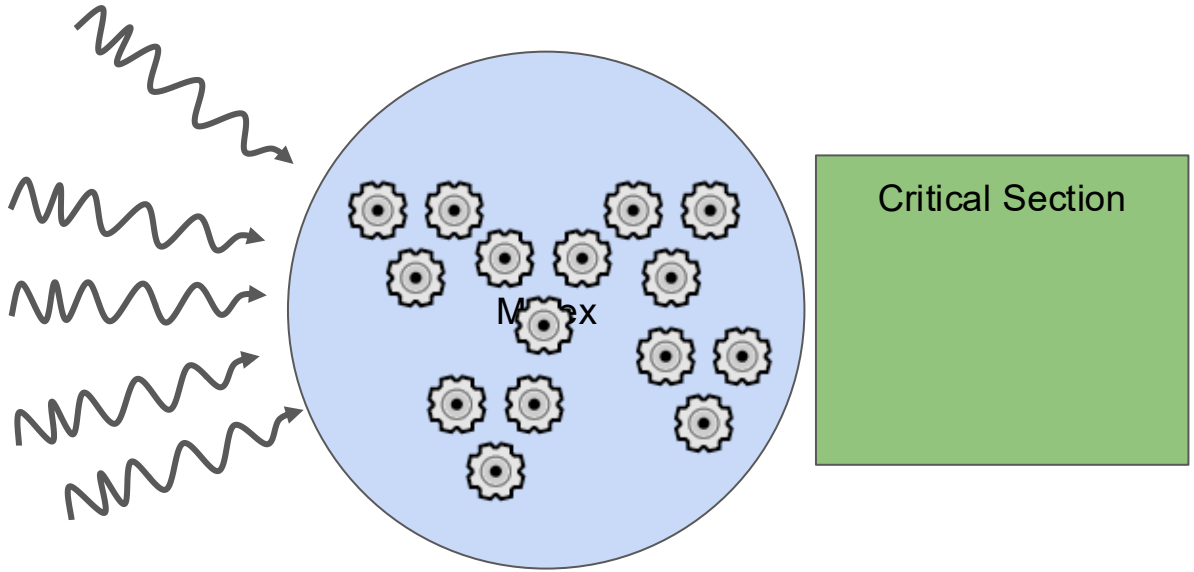
What we want - 2



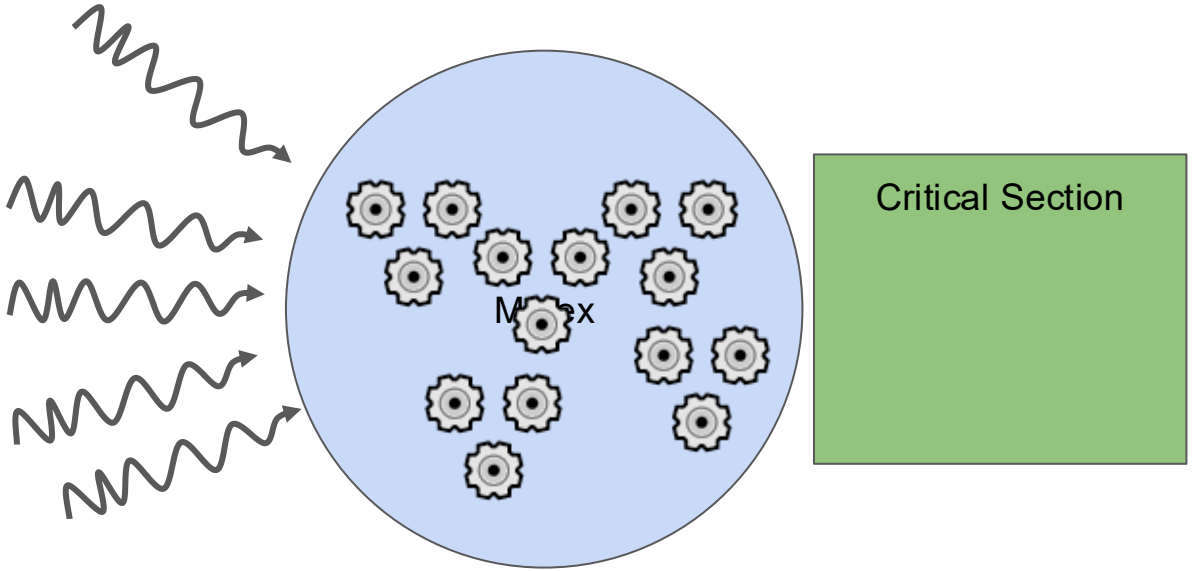
What we want - 2



What we want - 2

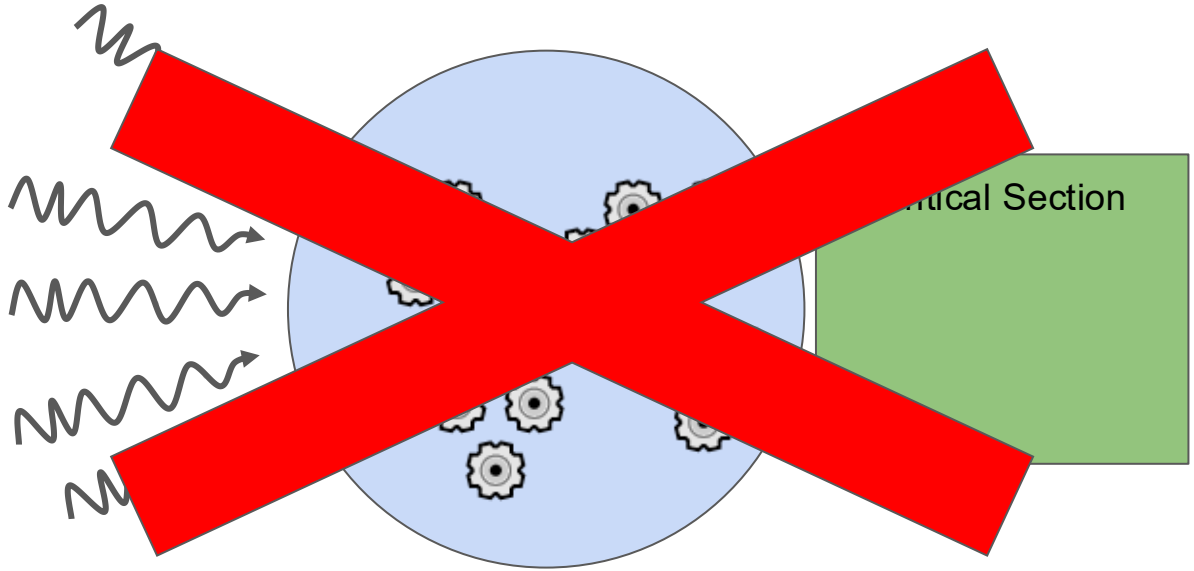


What we want - 2



Hmm! Nothing good is happening…

What we want - 2



Hmm! Nothing good is happening…

2 Different Flavours of “Properties”

2 Different Flavours of “Properties”

1. **Bad things shouldn’t happen**

2 Different Flavours of “Properties”

1. **Bad things shouldn’t happen**
    a. Things should not go ‘wrong’

2 Different Flavours of “Properties”

1. **Bad things shouldn’t happen**
    a. Things should not go ‘wrong’
    b. Your system should not make a mistake

2 Different Flavours of “Properties”

1. **Bad things shouldn’t happen**
    a. Things should not go ‘wrong’
    b. Your system should not make a mistake
    c. Your system should be **safe**

2 Different Flavours of “Properties”

1. **Bad things shouldn’t happen**
    a. Things should not go ‘wrong’
    b. Your system should not make a mistake
    c. Your system should be **safe**

2 Different Flavours of “Properties”

1. **Bad things shouldn’t happen**
    a. Things should not go ‘wrong’
    b. Your system should not make a mistake
    c. Your system should be **safe**

1. **Things should always ‘happen’**

2 Different Flavours of “Properties”

1. **Bad things shouldn’t happen**
    a. Things should not go ‘wrong’
    b. Your system should not make a mistake
    c. Your system should be **safe**

1. **Things should always ‘happen’**
    a. Good things should always happen

2 Different Flavours of “Properties”

1. **Bad things shouldn’t happen**
    a. Things should not go ‘wrong’
    b. Your system should not make a mistake
    c. Your system should be **safe**

1. **Things should always ‘happen’**
    a. Good things should always happen
    b. Things should always progress

2 Different Flavours of “Properties”

1. **Bad things shouldn’t happen**
    a. Things should not go ‘wrong’
    b. Your system should not make a mistake
    c. Your system should be **safe**

1. **Things should always ‘happen’**
    a. Good things should always happen
    b. Things should always progress
    c. Your system should be **live**

Some History (and some future)

Some History (and some future)

- Lamport recognized these properties

Some History (and some future)

- Lamport recognized these properties
- Could give formal definitions of safety

Some History (and some future)

- Lamport recognized these properties
- Could give formal definitions of safety
- Could not give formal definition of liveness, and posed it as a challenge

Some History (and some future)

- Lamport recognized these properties
- Could give formal definitions of safety
- Could not give formal definition of liveness, and posed it as a challenge
- Alpern and Schneider solved this, and gave a satisfactory definition of liveness

Some History (and some future)

- Lamport recognized these properties
- Could give formal definitions of safety
- Could not give formal definition of liveness, and posed it as a challenge
- Alpern and Schneider solved this, and gave a satisfactory definition of liveness

Some History (and some future)

- Lamport recognized these properties
- Could give formal definitions of safety
- Could not give formal definition of liveness, and posed it as a challenge
- Alpern and Schneider solved this, and gave a satisfactory definition of liveness

1. What are these formal definitions?

Some History (and some future)

- Lamport recognized these properties
- Could give formal definitions of safety
- Could not give formal definition of liveness, and posed it as a challenge
- Alpern and Schneider solved this, and gave a satisfactory definition of liveness

1. What are these formal definitions?
2. Anyone can make a definition. What makes these definitions ‘correct’?

Systems, Behaviours, States

Systems, Behaviours, States

What is a State?

Systems, Behaviours, States

What is a State?

- If you freeze time / take a snapshot, you have a state of the system

Systems, Behaviours, States

What is a State?

- If you freeze time / take a snapshot, you have a state of the system
- Your state contains everything of interest to you

Systems, Behaviours, States

What is a State?

- If you freeze time / take a snapshot, you have a state of the system
- Your state contains everything of interest to you
  - Memory contents?

Systems, Behaviours, States

What is a State?

- If you freeze time / take a snapshot, you have a state of the system
- Your state contains everything of interest to you
  - Memory contents?
  - Program counter value?

Systems, Behaviours, States

What is a State?

- If you freeze time / take a snapshot, you have a state of the system
- Your state contains everything of interest to you
  - Memory contents?
  - Program counter value?
  - Variable values?

Systems, Behaviours, States

What is a State?

- If you freeze time / take a snapshot, you have a state of the system
- Your state contains everything of interest to you
  - Memory contents?
  - Program counter value?
  - Variable values?

```
i=0;
while(true)
   {
       i+=1;
   }
```

Systems, Behaviours, States

What is a State?

- If you freeze time / take a snapshot, you have a state of the system
- Your state contains everything of interest to you
  - Memory contents?
  - Program counter value?
  - Variable values?

```
i=0;
while(true)
   {
       i+=1;
   }
```



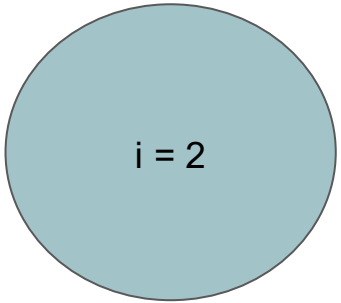
Systems, Behaviours, States

What is a State?

- If you freeze time / take a snapshot, you have a state of the system
- Your state contains everything of interest to you
  - Memory contents?
  - Program counter value?
  - Variable values?

```
i=0;
while(true)
   {
       i+=1;
   }
```





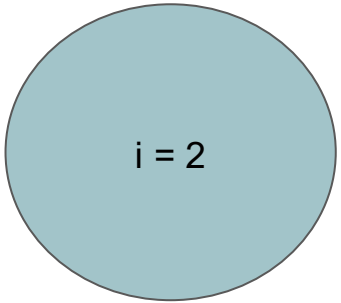
Systems, Behaviours, States

What is a State?

- If you freeze time / take a snapshot, you have a state of the system
- Your state contains everything of interest to you
  - Memory contents?
  - Program counter value?
  - Variable values?

```
i=0;
while(true)
   {
       i+=1;
   }
```





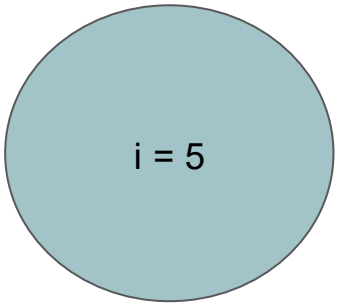
Systems, Behaviours, States

What is a State?

- If you freeze time / take a snapshot, you have a state of the system
- Your state contains everything of interest to you
  - Memory contents?
  - Program counter value?
  - Variable values?

```
i=0;
while(true)
   {
       i+=1;
   }
```





Systems, Behaviours, States

Systems, Behaviours, States

What is a Behaviour?

Systems, Behaviours, States

What is a Behaviour?

- Also known as **execution/trace**.

Systems, Behaviours, States

What is a Behaviour?

- Also known as **execution/trace**.
- Simply the sequence of states, as the program executes

Systems, Behaviours, States

What is a Behaviour?

- Also known as **execution/trace**.
- Simply the sequence of states, as the program executes

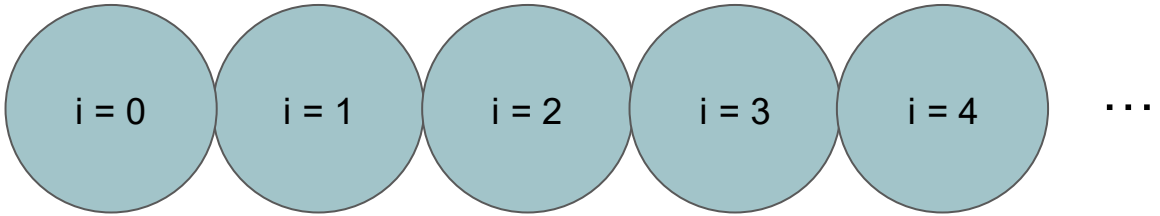
```
i=0;
while(true)
    {
        i+=1;
    }
```

Systems, Behaviours, States

What is a Behaviour?

- Also known as **execution/trace**.
- Simply the sequence of states, as the program executes

```
i=0;
while(true)
    {
        i+=1;
    }
```

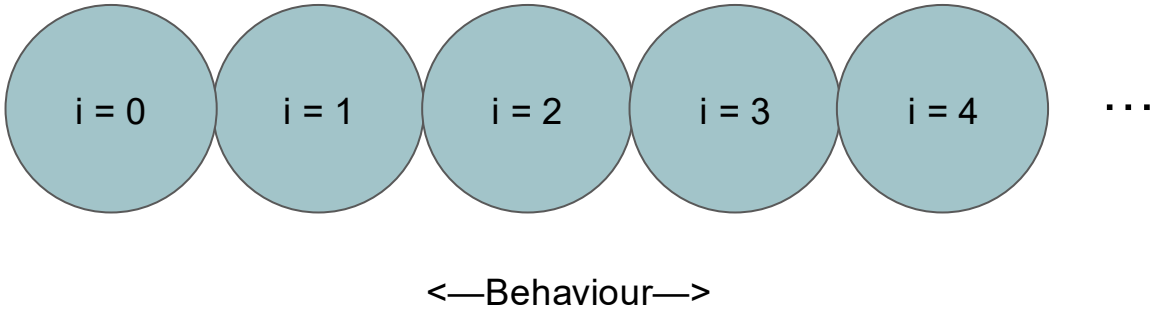


Systems, Behaviours, States

What is a Behaviour?

- Also known as **execution/trace**.
- Simply the sequence of states, as the program executes

```
i=0;
while(true)
    {
        i+=1;
    }
```



Systems, Behaviours, States

Systems, Behaviours, States

What is a System?

Systems, Behaviours, States

What is a System?

- A bunch of **executions**.

Systems, Behaviours, States

What is a System?

- A bunch of **executions**.
- Mathematically, a system is just a **set of executions**

Systems, Behaviours, States

What is a System?

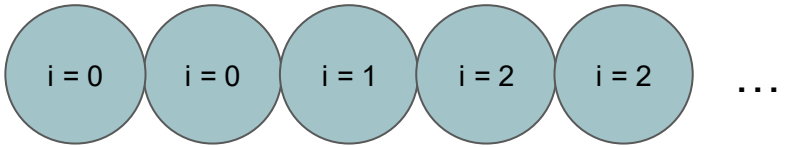
- A bunch of **executions**.
- Mathematically, a system is just a **set of executions**

```
i=0;
while(true)
  {
    if rand() % 2:
      i+=1
  }
```

Systems, Behaviours, States

What is a System?

- A bunch of **executions**.
- Mathematically, a system is just a **set of executions**



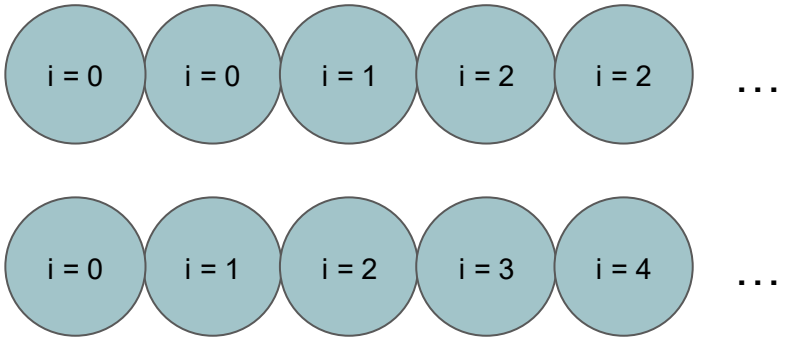

```
i=0;
while(true)
  {
    if rand() % 2:
      i+=1
  }
```

Systems, Behaviours, States

What is a System?

- A bunch of **executions**.
- Mathematically, a system is just a **set of executions**

```
i=0;
while(true)
  {
    if rand() % 2:
      i+=1
  }
```

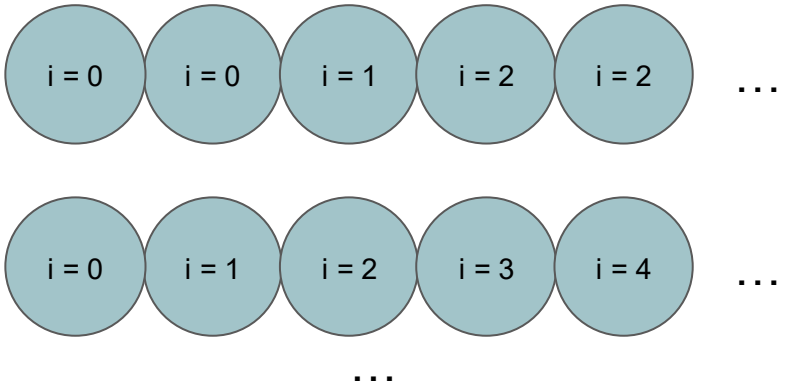


Systems, Behaviours, States

What is a System?

- A bunch of **executions**.
- Mathematically, a system is just a **set of executions**

```
i=0;
while(true)
  {
    if rand() % 2:
      i+=1
  }
```

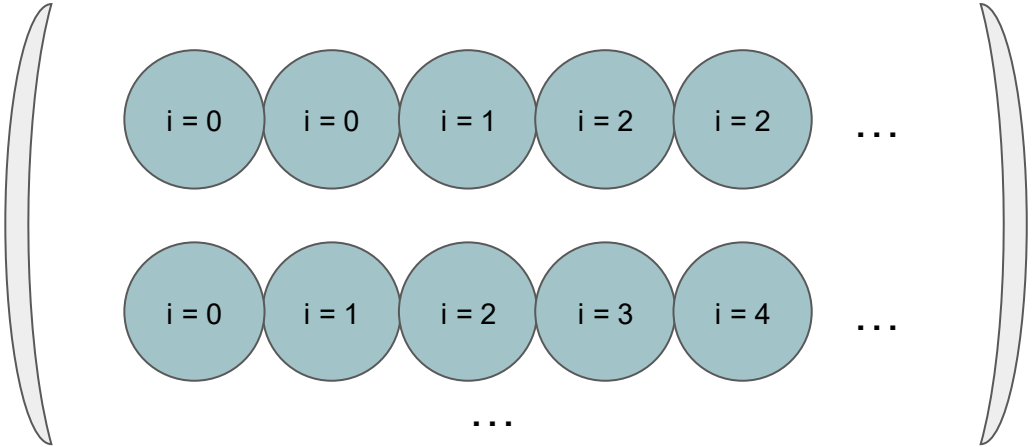


Systems, Behaviours, States

What is a System?

- A bunch of **executions**.
- Mathematically, a system is just a **set of executions**

```
i=0;
while(true)
  {
    if rand() % 2:
      i+=1
  }
```

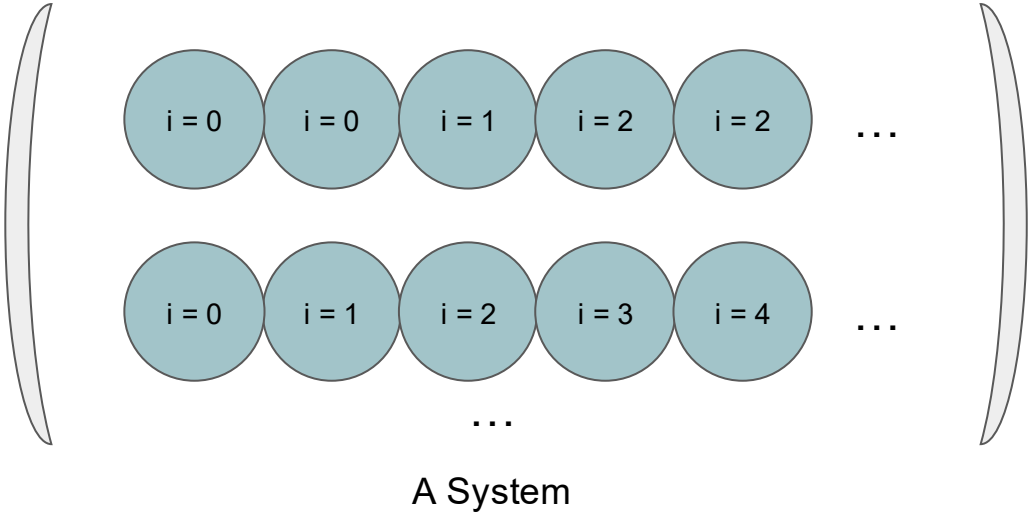


Systems, Behaviours, States

What is a System?

- A bunch of **executions**.
- Mathematically, a system is just a **set of executions**

```
i=0;
while(true)
  {
    if rand() % 2:
      i+=1
  }
```



A System

Properties

Properties

- We can think (at least Leslie Lamport did) of properties as a bunch of **desirable executions**

Properties

- We can think (at least Leslie Lamport did) of properties as a bunch of **desirable executions**
- Mathematically, a ‘property’ is a set of executions too!

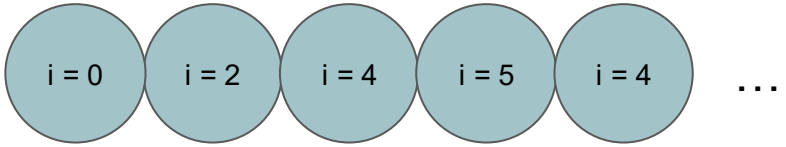
Properties

- We can think (at least Leslie Lamport did) of properties as a bunch of **desirable executions**
- Mathematically, a ‘property’ is a set of executions too!

“Eventually always i>3”

Properties

- We can think (at least Leslie Lamport did) of properties as a bunch of **desirable executions**
- Mathematically, a ‘property’ is a set of executions too!

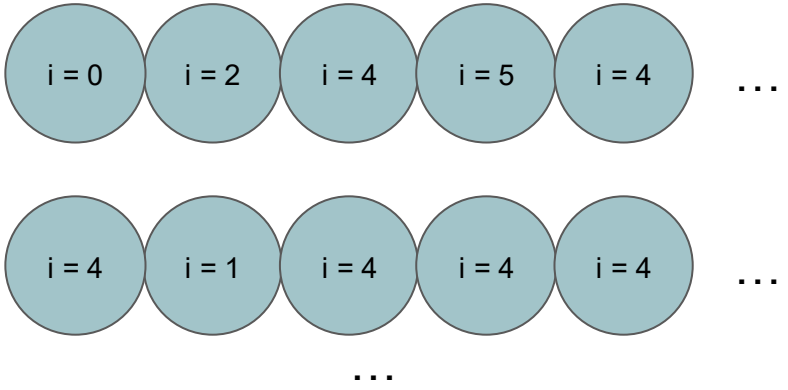


“Eventually always i>3”

Properties

- We can think (at least Leslie Lamport did) of properties as a bunch of **desirable executions**
- Mathematically, a ‘property’ is a set of executions too!

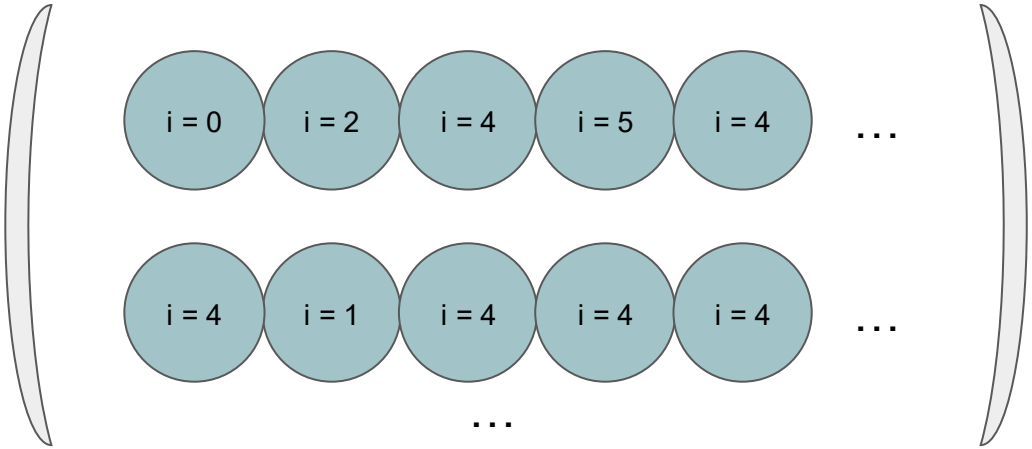
“Eventually always i>3”



Properties

- We can think (at least Leslie Lamport did) of properties as a bunch of **desirable executions**
- Mathematically, a ‘property’ is a set of executions too!

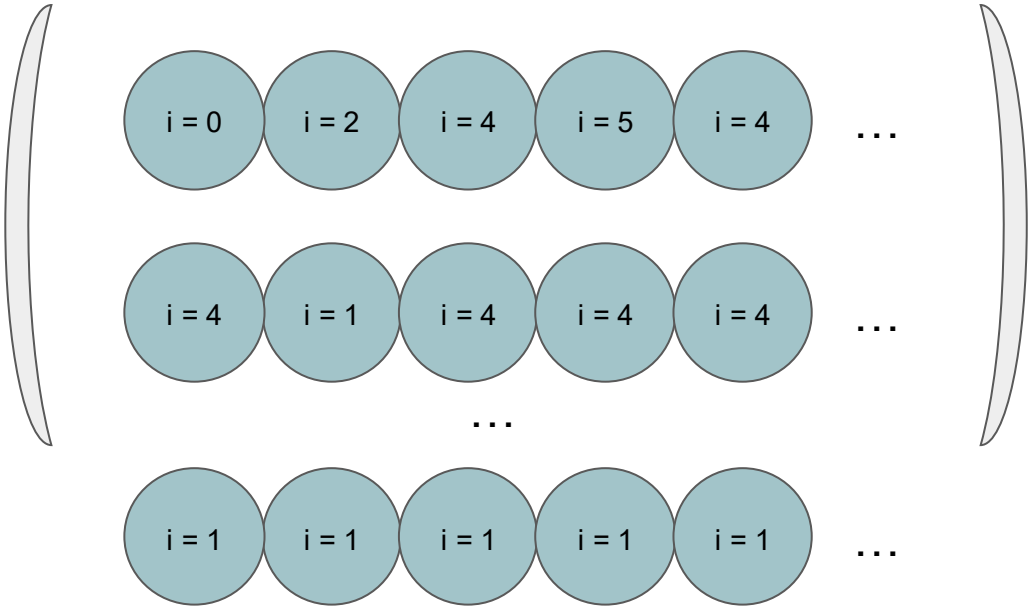
“Eventually always i>3”



Properties

- We can think (at least Leslie Lamport did) of properties as a bunch of **desirable executions**
- Mathematically, a ‘property’ is a set of executions too!

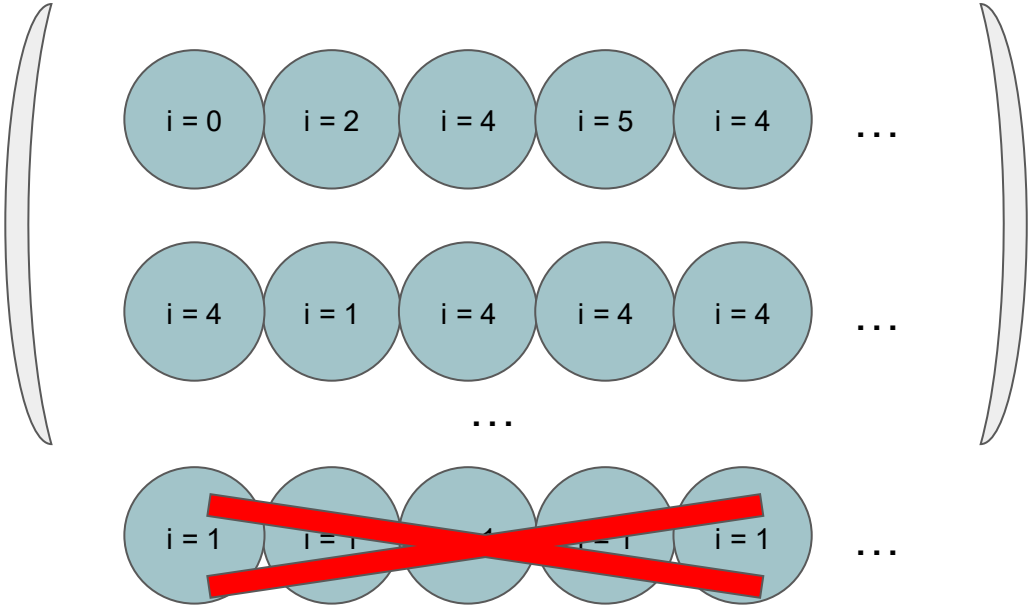
“Eventually always i>3”



Properties

- We can think (at least Leslie Lamport did) of properties as a bunch of **desirable executions**
- Mathematically, a ‘property’ is a set of executions too!

“Eventually always i>3”



System satisfies a Property

System satisfies a Property

- A system is a set of executions

System satisfies a Property

- A system is a set of executions
- A property is a set of desirable executions

System satisfies a Property

- A system is a set of executions
- A property is a set of desirable executions
- When does a system satisfy a property?

System satisfies a Property

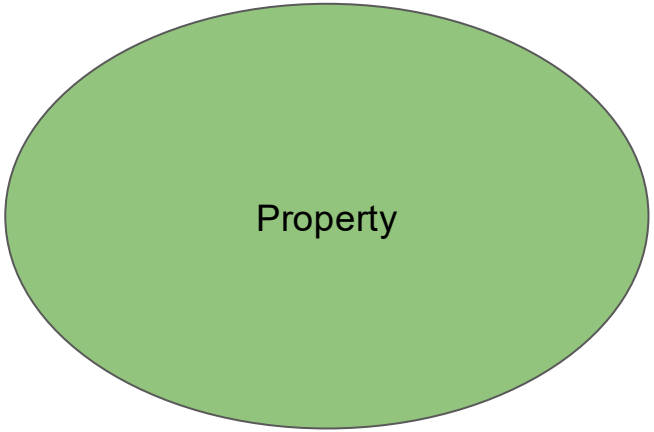
- A system is a set of executions
- A property is a set of desirable executions
- When does a system satisfy a property?
  - Every system execution is desirable

System satisfies a Property

- A system is a set of executions
- A property is a set of desirable executions
- When does a system satisfy a property?
  - Every system execution is desirable
  - system $\subseteq$ property

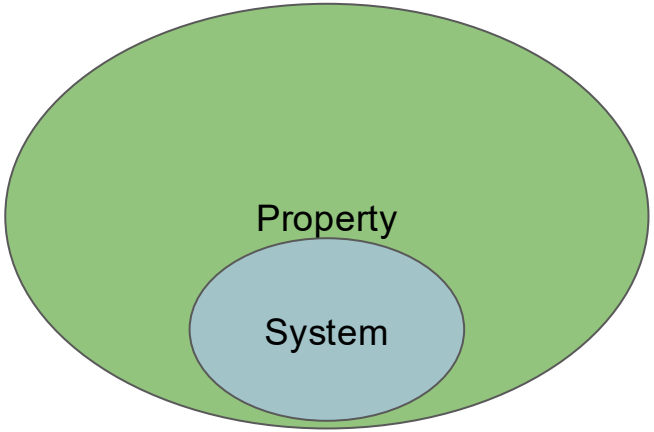
System satisfies a Property

- A system is a set of executions
- A property is a set of desirable executions
- When does a system satisfy a property?
  - Every system execution is desirable
  - system ⊆ property



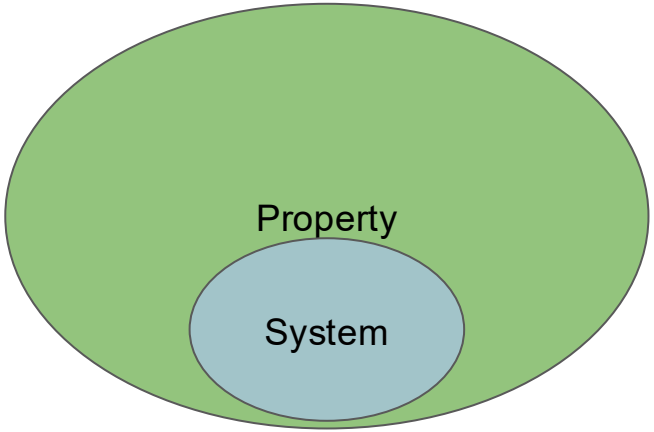
System satisfies a Property

- A system is a set of executions
- A property is a set of desirable executions
- When does a system satisfy a property?
  - Every system execution is desirable
  - system ⊆ property



System satisfies a Property

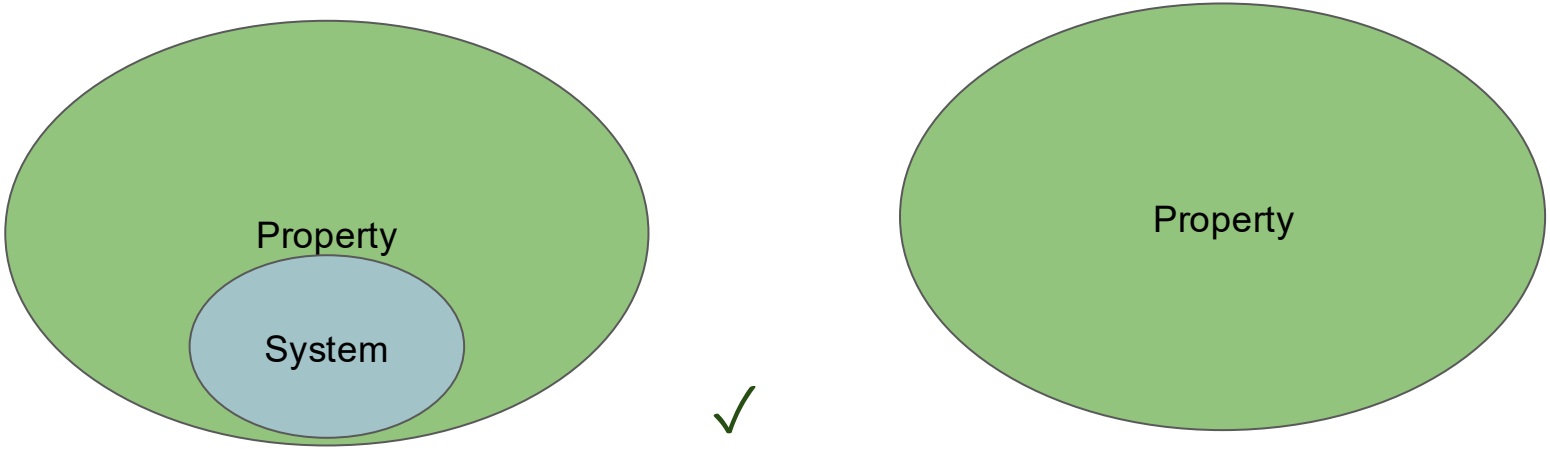
- A system is a set of executions
- A property is a set of desirable executions
- When does a system satisfy a property?
  - Every system execution is desirable
  - system ⊆ property



✓

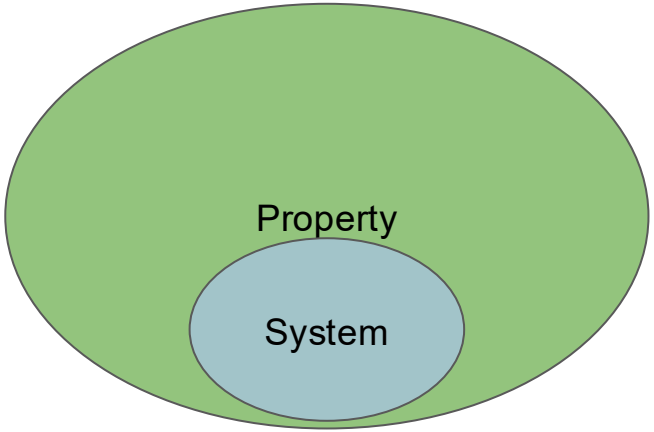
System satisfies a Property

- A system is a set of executions
- A property is a set of desirable executions
- When does a system satisfy a property?
  - Every system execution is desirable
  - system ⊆ property

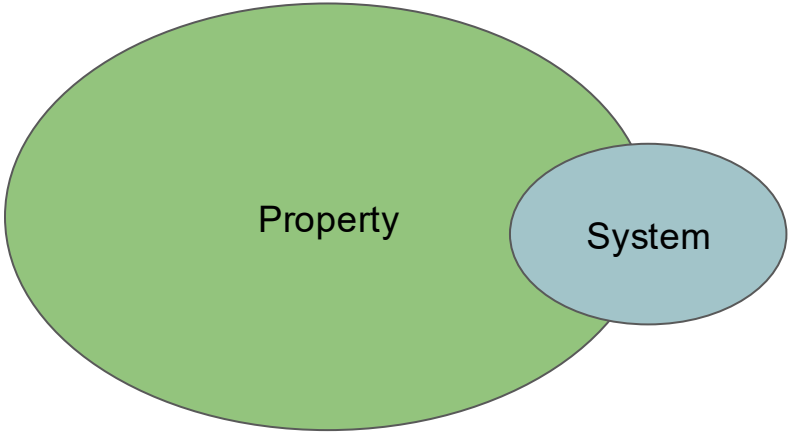


System satisfies a Property

- A system is a set of executions
- A property is a set of desirable executions
- When does a system satisfy a property?
  - Every system execution is desirable
  - system ⊆ property

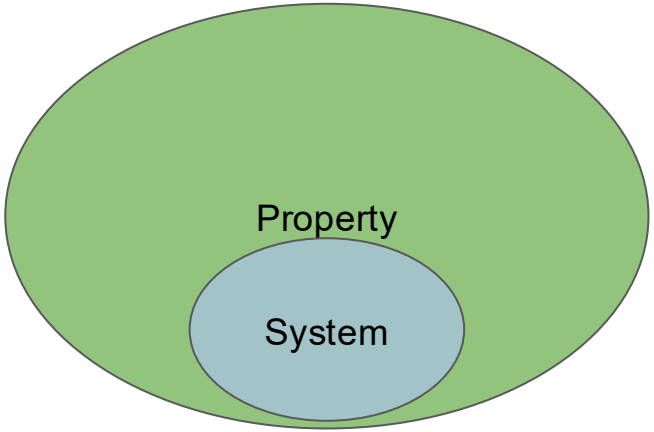


✓

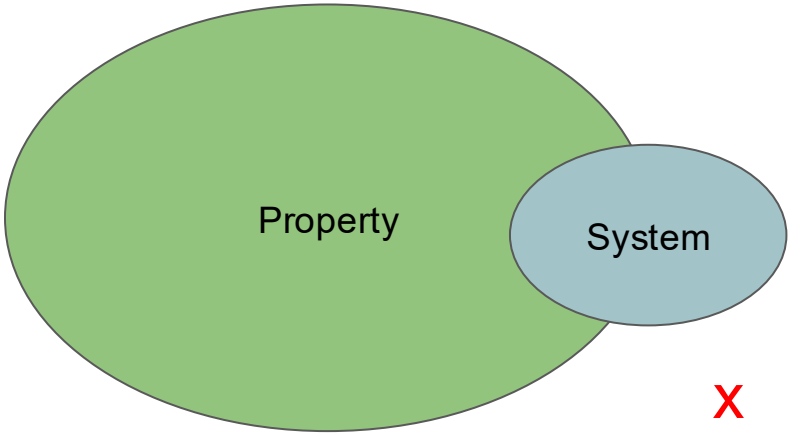


System satisfies a Property

- A system is a set of executions
- A property is a set of desirable executions
- When does a system satisfy a property?
  - Every system execution is desirable
  - system ⊆ property



✓



X

Safety and Liveness Properties

Safety and Liveness Properties

- We see that we can view a property as a set of desirable executions

Safety and Liveness Properties

- We see that we can view a property as a set of desirable executions
- We also saw 2 ‘flavours’ of properties:

Safety and Liveness Properties

- We see that we can view a property as a set of desirable executions
- We also saw 2 ‘flavours’ of properties:
  - Safety → Mistakes should not happen

Safety and Liveness Properties

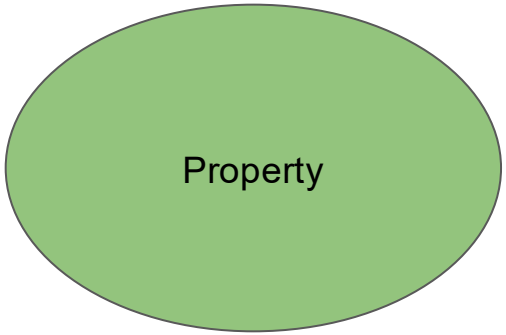
- We see that we can view a property as a set of desirable executions
- We also saw 2 ‘flavours’ of properties:
  - Safety → Mistakes should not happen
  - Liveness → ‘Something’ should happen

Safety and Liveness Properties

- We see that we can view a property as a set of desirable executions
- We also saw 2 ‘flavours’ of properties:
  - Safety → Mistakes should not happen
  - Liveness → ‘Something’ should happen
- Now, we want to know this:

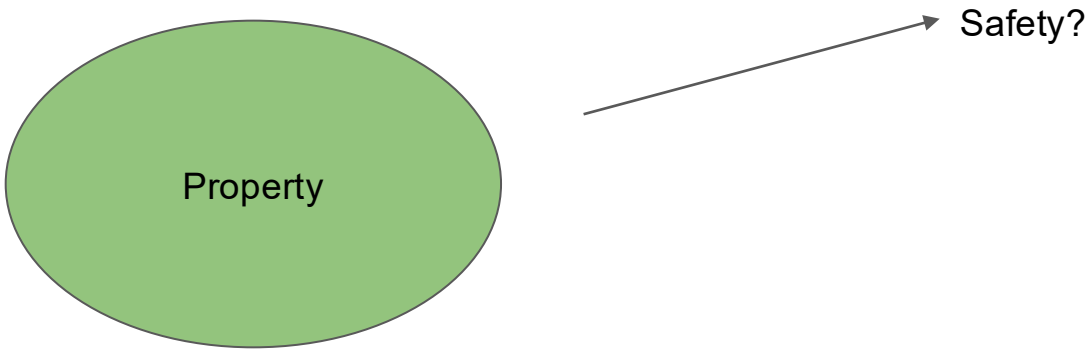
Safety and Liveness Properties

- We see that we can view a property as a set of desirable executions
- We also saw 2 ‘flavours’ of properties:
  - Safety → Mistakes should not happen
  - Liveness → ‘Something’ should happen
- Now, we want to know this:



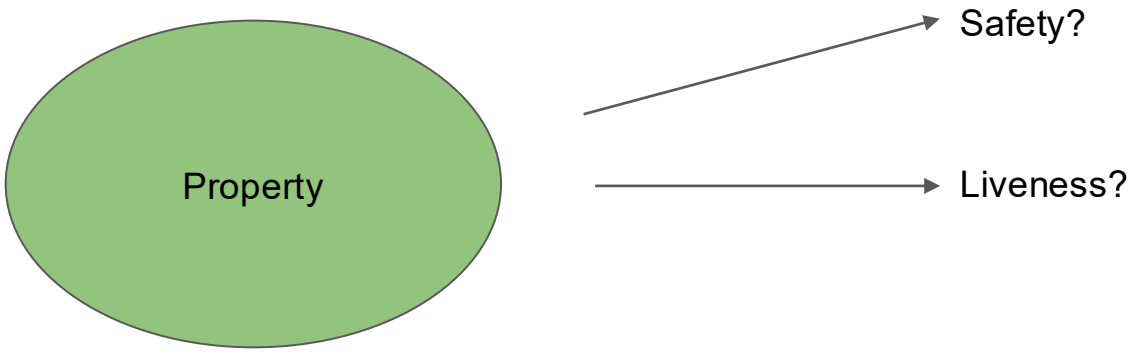
Safety and Liveness Properties

- We see that we can view a property as a set of desirable executions
- We also saw 2 ‘flavours’ of properties:
  - Safety → Mistakes should not happen
  - Liveness → ‘Something’ should happen
- Now, we want to know this:



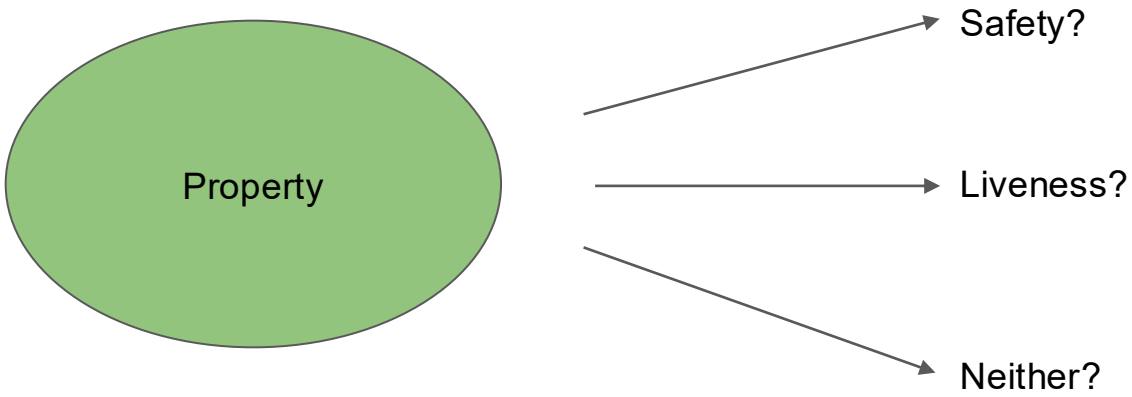
Safety and Liveness Properties

- We see that we can view a property as a set of desirable executions
- We also saw 2 ‘flavours’ of properties:
  - Safety → Mistakes should not happen
  - Liveness → ‘Something’ should happen
- Now, we want to know this:



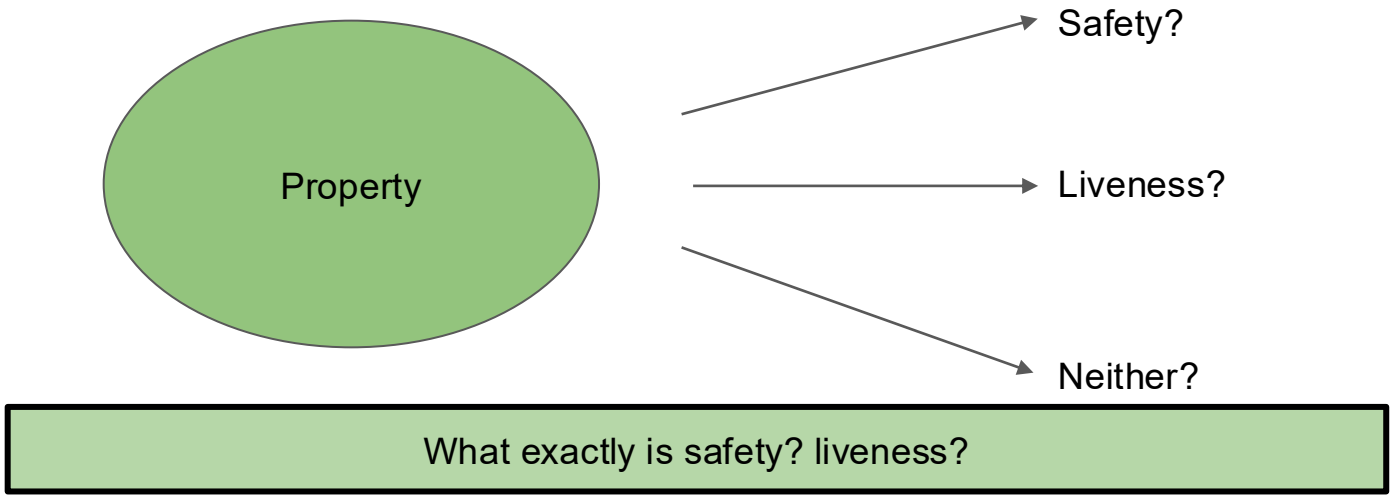
Safety and Liveness Properties

- We see that we can view a property as a set of desirable executions
- We also saw 2 ‘flavours’ of properties:
  - Safety → Mistakes should not happen
  - Liveness → ‘Something’ should happen
- Now, we want to know this:



Safety and Liveness Properties

- We see that we can view a property as a set of desirable executions
- We also saw 2 ‘flavours’ of properties:
  - Safety → Mistakes should not happen
  - Liveness → ‘Something’ should happen
- Now, we want to know this:



Safety Property

Safety Property

- Recall that in safety, mistakes/bad things should not happen.

Safety Property

- Recall that in safety, mistakes/bad things should not happen.
- You can view a **finite** history of an execution and say if something bad happened.

Safety Property

- Recall that in safety, mistakes/bad things should not happen.
- You can view a **finite** history of an execution and say if something bad happened.
- A safety property can be thought of as:

Safety Property

- Recall that in safety, mistakes/bad things should not happen.
- You can view a **finite** history of an execution and say if something bad happened.
- A safety property can be thought of as:
  - A collection of **all executions**

Safety Property

- Recall that in safety, mistakes/bad things should not happen.
- You can view a **finite** history of an execution and say if something bad happened.
- A safety property can be thought of as:
  - A collection of **all executions**
  - With bad executions removed

Safety Property

- Recall that in safety, mistakes/bad things should not happen.
- You can view a **finite** history of an execution and say if something bad happened.
- A safety property can be thought of as:
  - A collection of **all executions**
  - With bad executions removed
- Mathematically, a safety property is one that is constructed as follows:

Safety Property

- Recall that in safety, mistakes/bad things should not happen.
- You can view a **finite** history of an execution and say if something bad happened.
- A safety property can be thought of as:
  - A collection of **all executions**
  - With bad executions removed
- Mathematically, a safety property is one that is constructed as follows:
  - Let *E* be the set of all executions

Safety Property

- Recall that in safety, mistakes/bad things should not happen.
- You can view a **finite** history of an execution and say if something bad happened.
- A safety property can be thought of as:
  - A collection of **all executions**
  - With bad executions removed
- Mathematically, a safety property is one that is constructed as follows:
  - Let *E* be the set of all executions
  - Let *B* be a set of finite prefixes – These are the bad things that should not happen

Safety Property

- Recall that in safety, mistakes/bad things should not happen.
- You can view a **finite** history of an execution and say if something bad happened.
- A safety property can be thought of as:
  - A collection of **all executions**
  - With bad executions removed
- Mathematically, a safety property is one that is constructed as follows:
  - Let *E* be the set of all executions
  - Let *B* be a set of finite prefixes – These are the bad things that should not happen
  - Remove from *E* all executions that start with something in *B*

Safety Property - Examples

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**
- Is the following property a safety property?

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**
- Is the following property a safety property?
  - **Our system never starts as r**

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**
- Is the following property a safety property?
  - **Our system never starts as r**
  - Yes! Our set of bad things is {<r>}

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**
- Is the following property a safety property?
  - **Our system never starts as r**
  - Yes! Our set of bad things is {<r>}
- Is the following property a safety property?

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**
- Is the following property a safety property?
  - **Our system never starts as r**
  - Yes! Our set of bad things is {<r>}
- Is the following property a safety property?
  - **Our system is never r**

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**
- Is the following property a safety property?
  - **Our system never starts as r**
  - Yes! Our set of bad things is {<r>}
- Is the following property a safety property?
  - **Our system is never r**
  - Yes! Our set of bad things is {<r>, <g,r>, <g,g,r>, <g,g,g,r>, …}

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**
- Is the following property a safety property?
  - **Our system never starts as r**
  - Yes! Our set of bad things is {<r>}
- Is the following property a safety property?
  - **Our system is never r**
  - Yes! Our set of bad things is {<r>, <g,r>, <g,g,r>, <g,g,g,r>, …}
- Is the following a safety property?

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**
- Is the following property a safety property?
  - **Our system never starts as r**
  - Yes! Our set of bad things is {<r>}
- Is the following property a safety property?
  - **Our system is never r**
  - Yes! Our set of bad things is {<r>, <g,r>, <g,g,r>, <g,g,g,r>, …}
- Is the following a safety property?
  - **Our system becomes g sometime**

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**
- Is the following property a safety property?
  - **Our system never starts as r**
  - Yes! Our set of bad things is {<r>}
- Is the following property a safety property?
  - **Our system is never r**
  - Yes! Our set of bad things is {<r>, <g,r>, <g,g,r>, <g,g,g,r>, …}
- Is the following a safety property?
  - **Our system becomes g sometime**
  - What are the set of executions that make up this property?

Safety Property - Examples

- Our system can only be in 2 states here: **r** and **g**
- Is the following property a safety property?
  - **Our system never starts as r**
  - Yes! Our set of bad things is {<r>}
- Is the following property a safety property?
  - **Our system is never r**
  - Yes! Our set of bad things is {<r>, <g,r>, <g,g,r>, <g,g,g,r>, …}
- Is the following a safety property?
  - **Our system becomes g sometime**
  - What are the set of executions that make up this property?
  - What can be our bad things set?

Liveness

Liveness

- Intuitively, properties where you cannot make any ‘bad prefix’

Liveness

- Intuitively, properties where you cannot make any ‘bad prefix’
- Mathematically, a liveness property is one where:

Liveness

- Intuitively, properties where you cannot make any ‘bad prefix’
- Mathematically, a liveness property is one where:
  - **Any finite system** execution can be extended to an execution in the property

Liveness

- Intuitively, properties where you cannot make any ‘bad prefix’
- Mathematically, a liveness property is one where:
  - **Any finite system** execution can be extended to an execution in the property
- Why does this make sense?

Liveness

- Intuitively, properties where you cannot make any ‘bad prefix’
- Mathematically, a liveness property is one where:
  - **Any finite system** execution can be extended to an execution in the property
- Why does this make sense?
  - If you have a finite system execution which has no extension in the property,

Liveness

- Intuitively, properties where you cannot make any ‘bad prefix’
- Mathematically, a liveness property is one where:
  - **Any finite system** execution can be extended to an execution in the property
- Why does this make sense?
  - If you have a finite system execution which has no extension in the property,
  - You have a **bad prefix!**

Liveness

- Intuitively, properties where you cannot make any ‘bad prefix’
- Mathematically, a liveness property is one where:
  - **Any finite system** execution can be extended to an execution in the property
- Why does this make sense?
  - If you have a finite system execution which has no extension in the property,
  - You have a **bad prefix!**
  - Something bad happened!

Liveness

- Intuitively, properties where you cannot make any ‘bad prefix’
- Mathematically, a liveness property is one where:
  - **Any finite system** execution can be extended to an execution in the property
- Why does this make sense?
  - If you have a finite system execution which has no extension in the property,
  - You have a **bad prefix!**
  - Something bad happened!
- **“Our system becomes g eventually”** is liveness

Liveness

- Intuitively, properties where you cannot make any ‘bad prefix’
- Mathematically, a liveness property is one where:
  - **Any finite system** execution can be extended to an execution in the property
- Why does this make sense?
  - If you have a finite system execution which has no extension in the property,
  - You have a **bad prefix!**
  - Something bad happened!
- **“Our system becomes g eventually”** is liveness
  - Why?

Hmm. Cool? I’m not convinced

Hmm. Cool? I’m not convinced

- Neat definitions!

Hmm. Cool? I’m not convinced

- Neat definitions!
- But how do we know it’s ‘right’?

Hmm. Cool? I’m not convinced

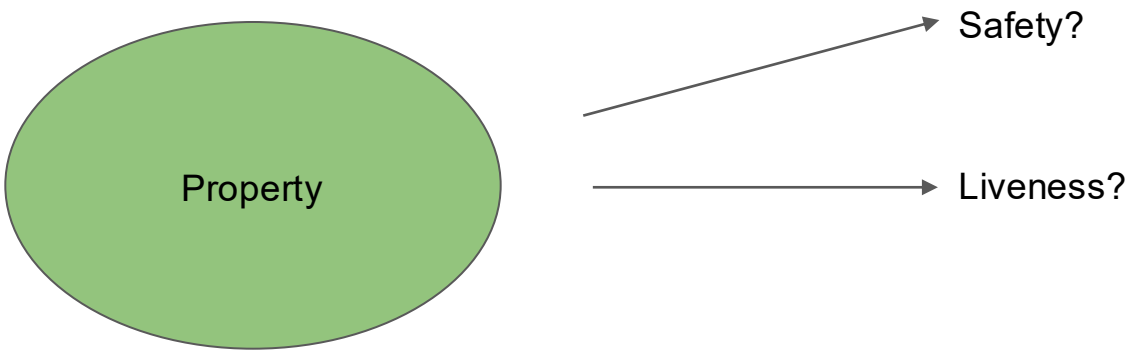
- Neat definitions!
- But how do we know it’s ‘right’?
- **Famous Result:**

Hmm. Cool? I’m not convinced

- Neat definitions!
- But how do we know it’s ‘right’?
- **Famous Result:**
  - Every property is an intersection of a safety and a liveness property (Alpern, Schneider)

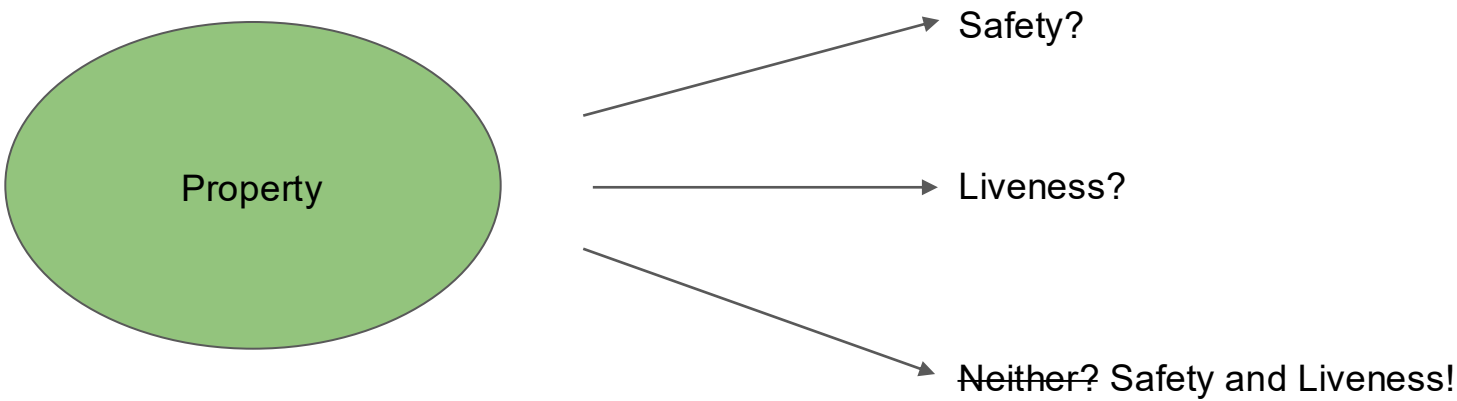
Hmm. Cool? I’m not convinced

- Neat definitions!
- But how do we know it’s ‘right’?
- **Famous Result:**
  - Every property is an intersection of a safety and a liveness property (Alpern, Schneider)



Hmm. Cool? I’m not convinced

- Neat definitions!
- But how do we know it’s ‘right’?
- **Famous Result:**
  - Every property is an intersection of a safety and a liveness property (Alpern, Schneider)



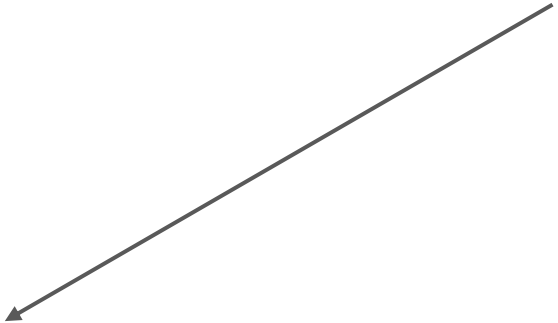
Both safety and liveness?

Both safety and liveness?

“Our system terminates with x=3”

Both safety and liveness?

“Our system terminates with x=3”

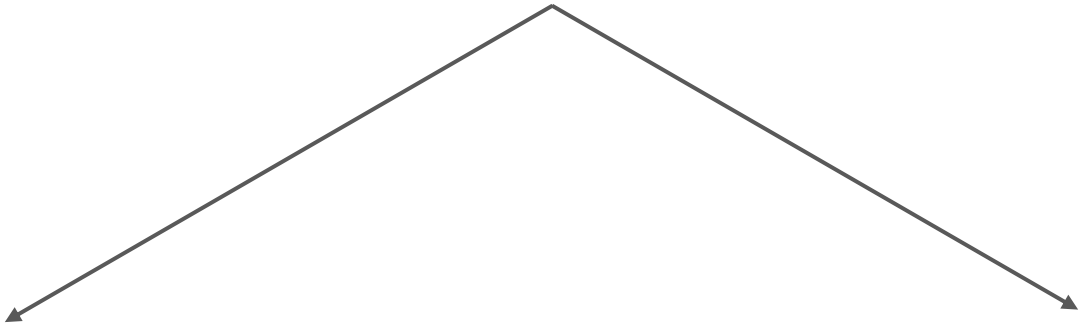


“If system terminates, x=3”

- Safety

Both safety and liveness?

“Our system terminates with x=3”



“If system terminates, x=3”

- Safety

“System eventually terminates”

- Liveness

Taking a step back

Taking a step back

What we did:

Taking a step back

What we did:

- Given a trace/behaviour, we check if it is desirable

Taking a step back

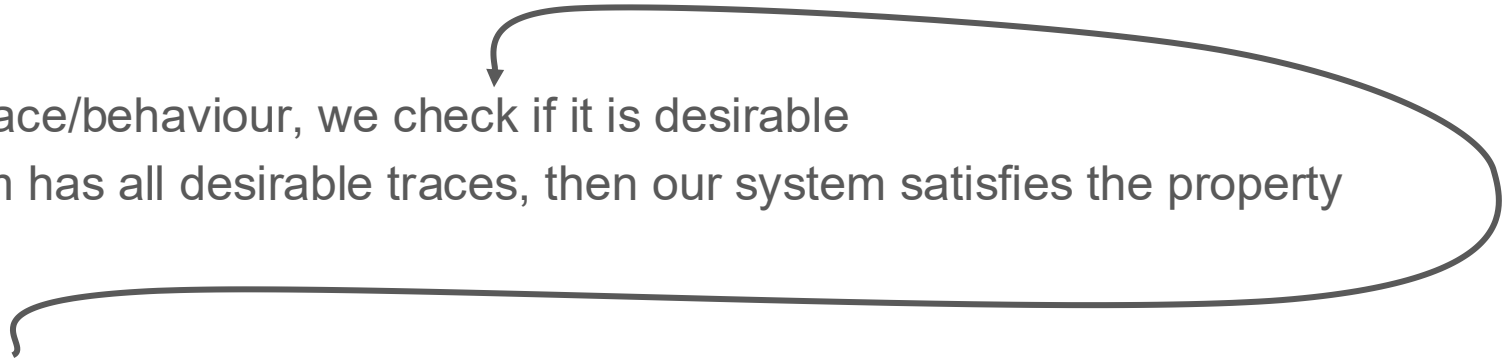
What we did:

- Given a trace/behaviour, we check if it is desirable
- If a system has all desirable traces, then our system satisfies the property

Taking a step back

What we did:

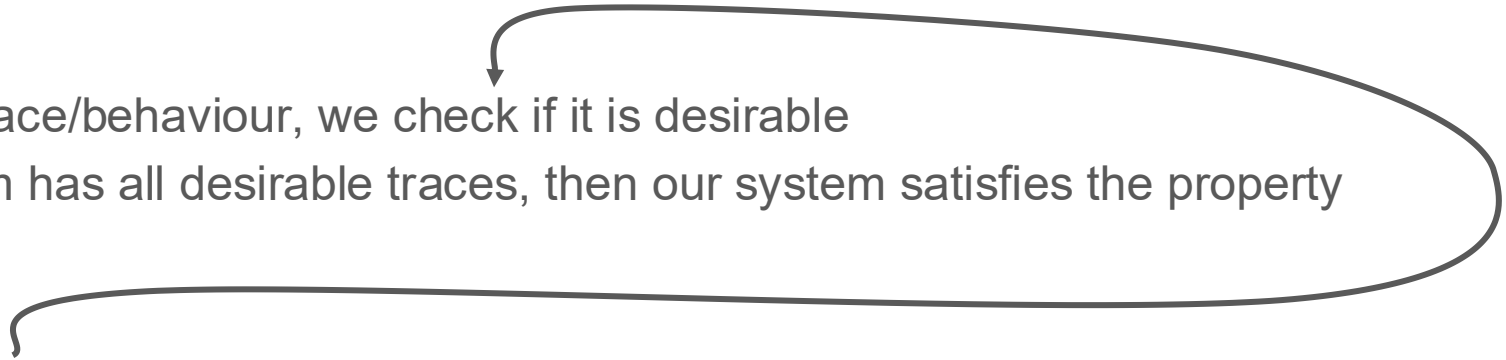
- Given a trace/behaviour, we check if it is desirable
- If a system has all desirable traces, then our system satisfies the property



Taking a step back

What we did:

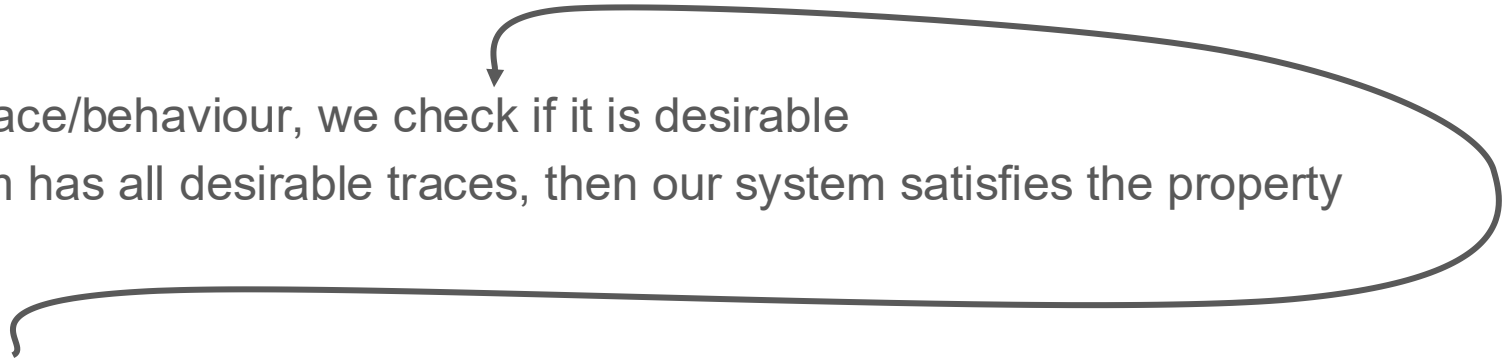
- Given a trace/behaviour, we check if it is desirable
- If a system has all desirable traces, then our system satisfies the property



Taking a step back

What we did:

- Given a trace/behaviour, we check if it is desirable
- If a system has all desirable traces, then our system satisfies the property

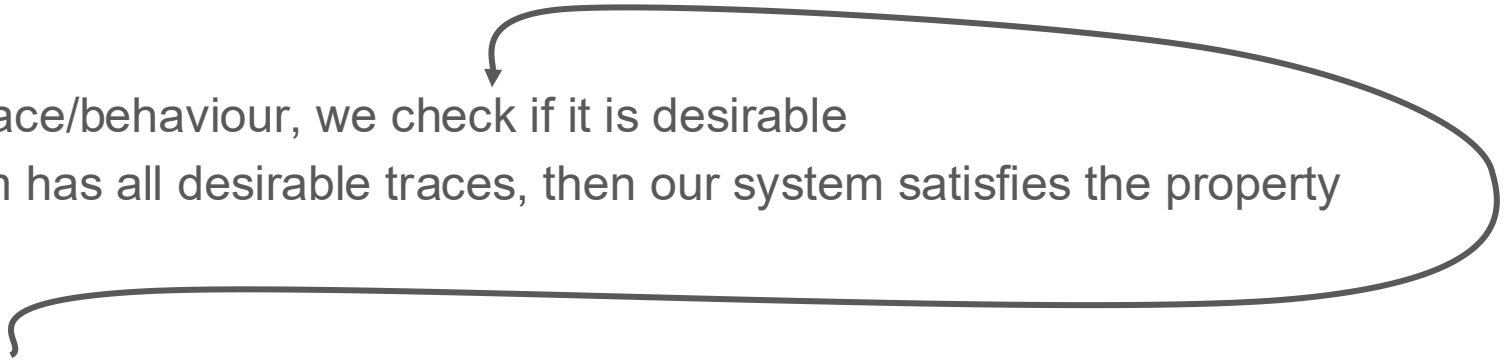


However, is this really possible?

Taking a step back

What we did:

- Given a trace/behaviour, we check if it is desirable
- If a system has all desirable traces, then our system satisfies the property



However, is this really possible?

- Perhaps, we cannot say, just by looking at a single trace, whether it is desirable

Beyond Simple Properties

Beyond Simple Properties

- Can you formulate the following:

Beyond Simple Properties

- Can you formulate the following:

“My system may have a terminating execution”

Beyond Simple Properties

- Can you formulate the following:

  “My system may have a terminating execution”

- Here, we cannot really look at a trace/execution and tell whether it is desirable.

Beyond Simple Properties

- Can you formulate the following:

“My system may have a terminating execution”

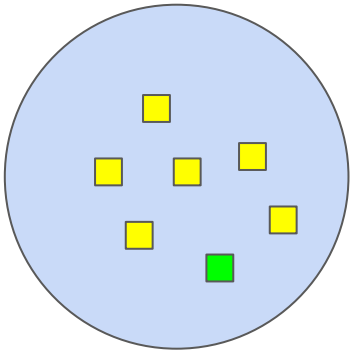
- Here, we cannot really look at a trace/execution and tell whether it is desirable.
- We actually have to look at the system (the set of executions) as a whole!

Beyond Simple Properties

- Can you formulate the following:

“My system may have a terminating execution”

- Here, we cannot really look at a trace/execution and tell whether it is desirable.
- We actually have to look at the system (the set of executions) as a whole!



Squares are executions.
Green square is a terminating execution

Hyperproperties

Hyperproperties

- It may be necessary to actually view the system as a whole to say whether it satisfies a ‘questions we have about it’.

Hyperproperties

- It may be necessary to actually view the system as a whole to say whether it satisfies a ‘questions we have about it’.
- Hyperproperties come into the picture here:

Hyperproperties

- It may be necessary to actually view the system as a whole to say whether it satisfies a ‘questions we have about it’.
- Hyperproperties come into the picture here:
  - While a trace property is a set of traces

Hyperproperties

- It may be necessary to actually view the system as a whole to say whether it satisfies a ‘questions we have about it’.
- Hyperproperties come into the picture here:
  - While a trace property is a set of traces
  - A Hyperproperty is a set of set of traces

Hyperproperties

- It may be necessary to actually view the system as a whole to say whether it satisfies a ‘questions we have about it’.
- Hyperproperties come into the picture here:
  - While a trace property is a set of traces
  - A Hyperproperty is a set of set of traces

Something that I keep in mind:

Hyperproperties

- It may be necessary to actually view the system as a whole to say whether it satisfies a ‘questions we have about it’.
- Hyperproperties come into the picture here:
  - While a trace property is a set of traces
  - A Hyperproperty is a set of set of traces

Something that I keep in mind:

- A trace property is viewed as a set of executions

Hyperproperties

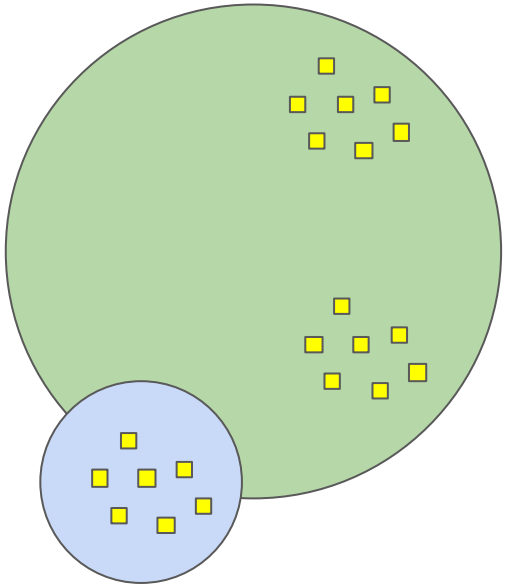
- It may be necessary to actually view the system as a whole to say whether it satisfies a ‘questions we have about it’.
- Hyperproperties come into the picture here:
  - While a trace property is a set of traces
  - A Hyperproperty is a set of set of traces

Something that I keep in mind:

- A trace property is viewed as a set of executions
- A Hyperproperty is viewed as a set of systems

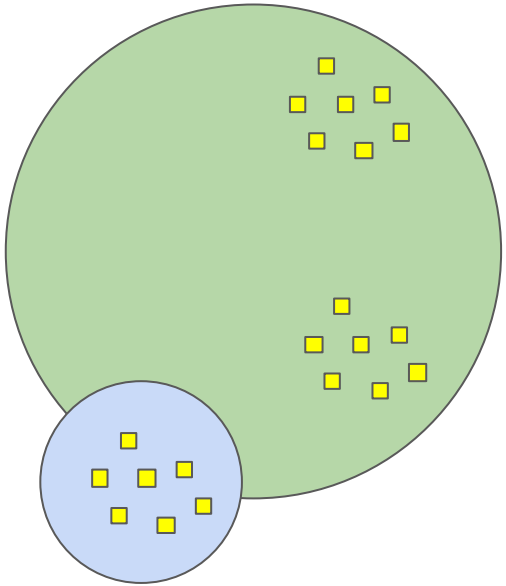
Hyperproperties vs Trace Properties

Hyperproperties vs Trace Properties

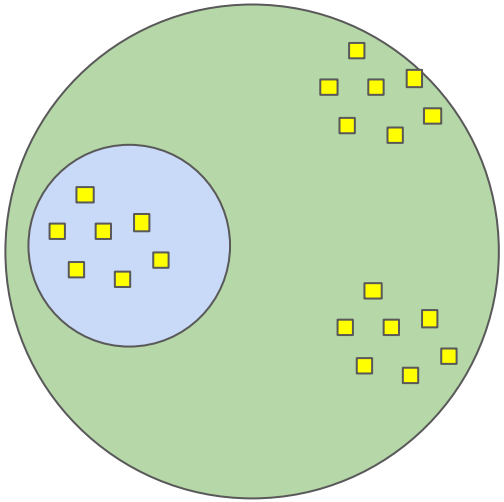


Blue system does not satisfy green property

Hyperproperties vs Trace Properties



Blue system does not satisfy green property



Blue system does satisfy green property

Hyperproperties in Action

Hyperproperties in Action

Everything you can ask about your system!

Hyperproperties in Action

Everything you can ask about your system!

- Information Flow Security Policies

Hyperproperties in Action

Everything you can ask about your system!

- Information Flow Security Policies
  - Noninterference

Hyperproperties in Action

Everything you can ask about your system!

- Information Flow Security Policies
  - Noninterference
  - Observational Determinism

Hyperproperties in Action

Everything you can ask about your system!

- Information Flow Security Policies
  - Noninterference
  - Observational Determinism
- Service Level Agreements

Hyperproperties in Action

Everything you can ask about your system!

- Information Flow Security Policies
  - Noninterference
  - Observational Determinism
- Service Level Agreements
  - Average Response Time of System

Hyperproperties in Action

Everything you can ask about your system!

- Information Flow Security Policies
  - Noninterference
  - Observational Determinism
- Service Level Agreements
  - Average Response Time of System
  - Percentage Uptime

Hyperproperties in Action

Everything you can ask about your system!

- Information Flow Security Policies
  - Noninterference
  - Observational Determinism
- Service Level Agreements
  - Average Response Time of System
  - Percentage Uptime
- “Maybe” Questions

Hyperproperties in Action

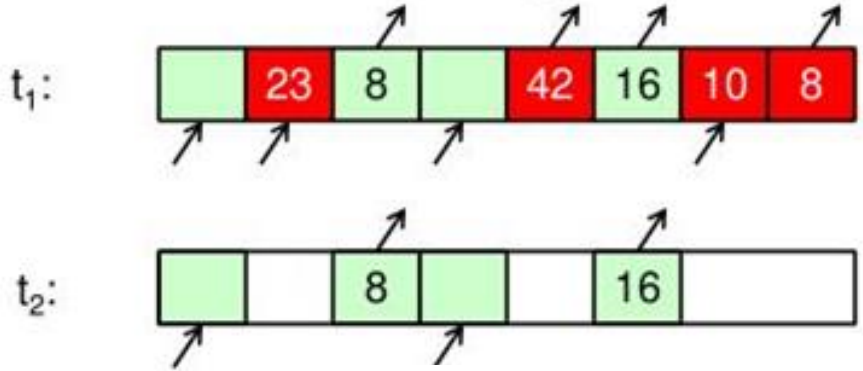
Everything you can ask about your system!

- Information Flow Security Policies
  - Noninterference
  - Observational Determinism
- Service Level Agreements
  - Average Response Time of System
  - Percentage Uptime
- “Maybe” Questions
  - System might do something

A Concrete Example

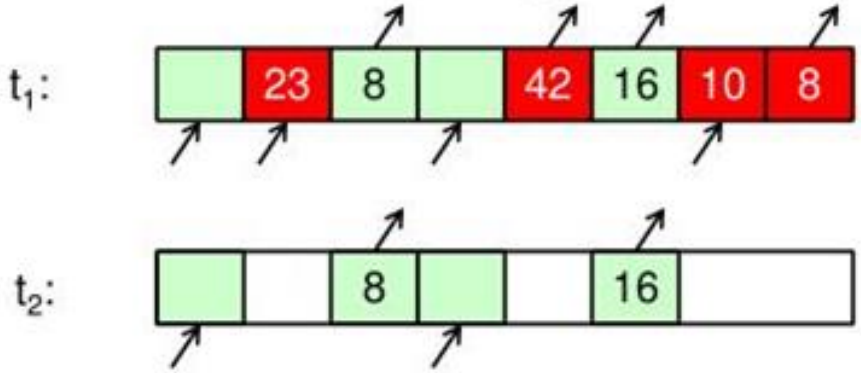
A Concrete Example

Noninterference [Goguen and Meseguer 1982]:
Commands of high security users have no effect on observations of low security users



A Concrete Example

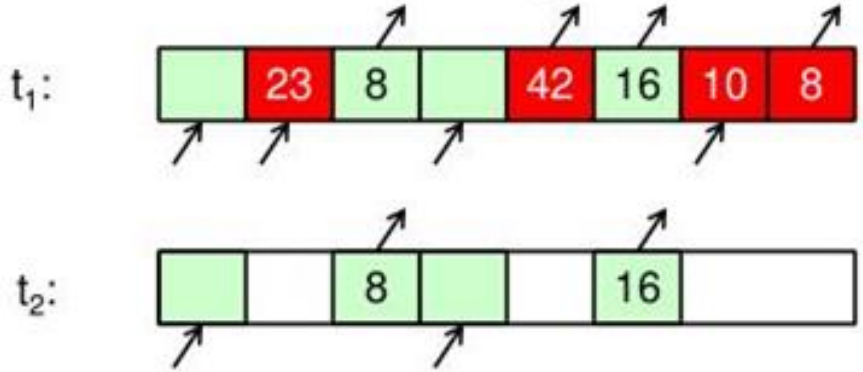
Noninterference [Goguen and Meseguer 1982]:
Commands of high security users have no effect on observations of low security users



Not safety!

A Concrete Example

Noninterference [Goguen and Meseguer 1982]:
Commands of high security users have no effect on observations of low security users



Not safety!

Taken from https://slideplayer.com/slide/9290816/

Hyperproperties - Safety and Liveness

Hyperproperties - Safety and Liveness

- Properties (sets of traces) can be reasoned about using safety and liveness

Hyperproperties - Safety and Liveness

- Properties (sets of traces) can be reasoned about using safety and liveness
- More interestingly, each property is an intersection of a safety and a liveness property

Hyperproperties - Safety and Liveness

- Properties (sets of traces) can be reasoned about using safety and liveness
- More interestingly, each property is an intersection of a safety and a liveness property
- Are there similar analogs for hyperproperties?

Hyperproperties - Safety and Liveness

- Properties (sets of traces) can be reasoned about using safety and liveness
- More interestingly, each property is an intersection of a safety and a liveness property
- Are there similar analogs for hyperproperties?

Yes!

Hypersafety

Hypersafety

- Recall how safety properties were constructed:

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions
  - We had *B*, a set of (finitely observable) **bad things**

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions
  - We had *B*, a set of (finitely observable) **bad things**
  - Concretely, it was a set of finite prefixes

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions
  - We had *B*, a set of (finitely observable) **bad things**
  - Concretely, it was a set of finite prefixes
  - From E, we removed executions starting with a bad thing.

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions
  - We had *B*, a set of (finitely observable) **bad things**
  - Concretely, it was a set of finite prefixes
  - From E, we removed executions starting with a bad thing.
- Can we do something similar for hyperproperties?

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions
  - We had *B*, a set of (finitely observable) **bad things**
  - Concretely, it was a set of finite prefixes
  - From E, we removed executions starting with a bad thing.
- Can we do something similar for hyperproperties?
  - We have *S*, the set of all **systems**

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions
  - We had *B*, a set of (finitely observable) **bad things**
  - Concretely, it was a set of finite prefixes
  - From E, we removed executions starting with a bad thing.
- Can we do something similar for hyperproperties?
  - We have *S*, the set of all **systems**
  - We make *B*, a set of bad things

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions
  - We had *B*, a set of (finitely observable) **bad things**
  - Concretely, it was a set of finite prefixes
  - From E, we removed executions starting with a bad thing.
- Can we do something similar for hyperproperties?
  - We have *S*, the set of all **systems**
  - We make *B*, a set of bad things
  - From *S,* we remove all systems ‘starting with’ a bad thing.

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions
  - We had *B*, a set of (finitely observable) **bad things**
  - Concretely, it was a set of finite prefixes
  - From E, we removed executions starting with a bad thing.
- Can we do something similar for hyperproperties?
  - We have *S*, the set of all **systems**
  - We make *B*, a set of bad things
  - From *S,* we remove all systems ‘starting with’ a bad thing.
- That’s cool. But

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions
  - We had *B*, a set of (finitely observable) **bad things**
  - Concretely, it was a set of finite prefixes
  - From E, we removed executions starting with a bad thing.
- Can we do something similar for hyperproperties?
  - We have *S*, the set of all **systems**
  - We make *B*, a set of bad things
  - From *S,* we remove all systems ‘starting with’ a bad thing.
- That’s cool. But
  - What’s a ‘bad thing’ for a system

Hypersafety

- Recall how safety properties were constructed:
  - We had *E*, the set of all executions
  - We had *B*, a set of (finitely observable) **bad things**
  - Concretely, it was a set of finite prefixes
  - From E, we removed executions starting with a bad thing.
- Can we do something similar for hyperproperties?
  - We have *S*, the set of all **systems**
  - We make *B*, a set of bad things
  - From *S,* we remove all systems ‘starting with’ a bad thing.
- That’s cool. But
  - What’s a ‘bad thing’ for a system
  - More importantly, how we know a system ‘starts with’ a bad thing?

Lifting ‘Bad things’ to systems/trace sets

Lifting ‘Bad things’ to systems/trace sets

- For trace properties, a bad thing was a finite trace.

Lifting ‘Bad things’ to systems/trace sets

- For trace properties, a bad thing was a finite trace.

Lifting ‘Bad things’ to systems/trace sets

- For trace properties, a bad thing was a finite trace.

Lifting ‘Bad things’ to systems/trace sets

- For trace properties, a bad thing was a finite trace.

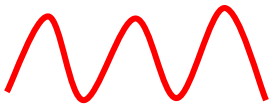
Lifting ‘Bad things’ to systems/trace sets

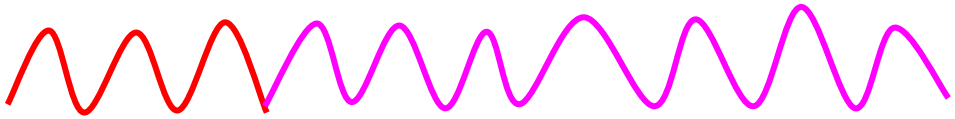
- For trace properties, a bad thing was a finite trace.

- For hyperproperties, a bad thing is a finite set of finite traces!

Lifting ‘Bad things’ to systems/trace sets

- For trace properties, a bad thing was a finite trace.

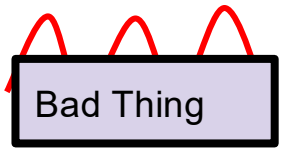




- For hyperproperties, a bad thing is a finite set of finite traces!

Lifting ‘Bad things’ to systems/trace sets

- For trace properties, a bad thing was a finite trace.

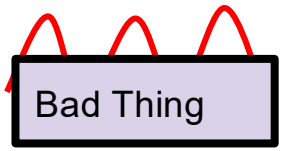




- For hyperproperties, a bad thing is a finite set of finite traces!

Lifting ‘Bad things’ to systems/trace sets

- For trace properties, a bad thing was a finite trace.

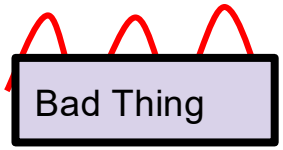




- For hyperproperties, a bad thing is a finite set of finite traces!

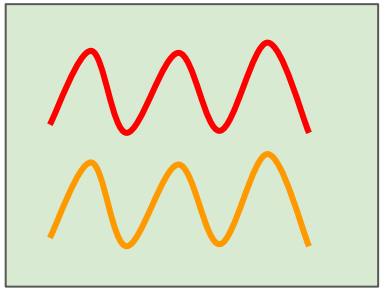
Lifting ‘Bad things’ to systems/trace sets

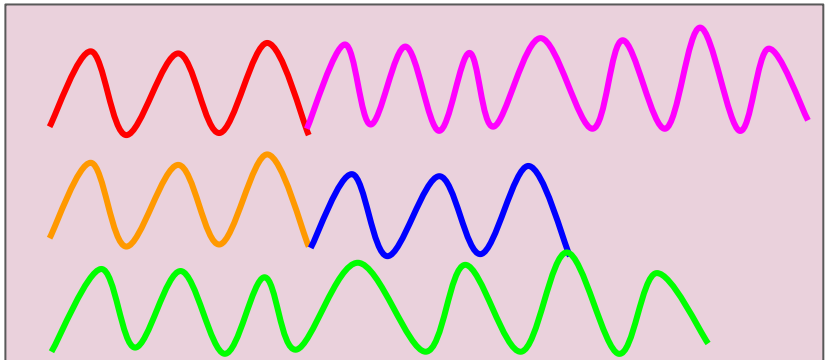
- For trace properties, a bad thing was a finite trace.





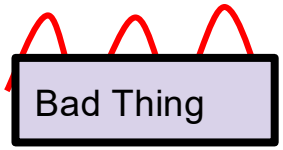
- For hyperproperties, a bad thing is a finite set of finite traces!





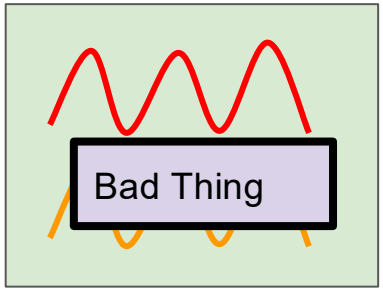
Lifting ‘Bad things’ to systems/trace sets

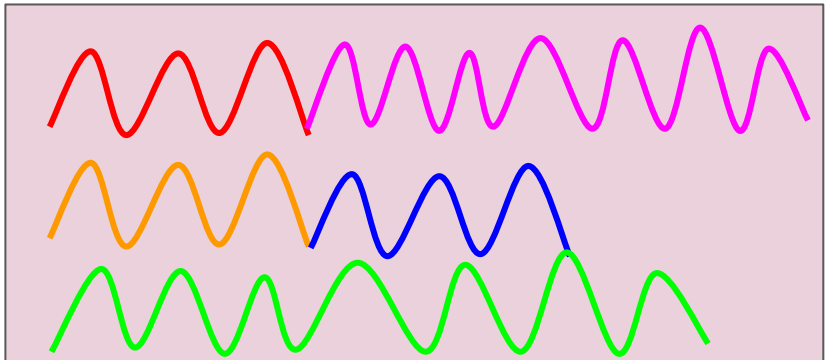
- For trace properties, a bad thing was a finite trace.





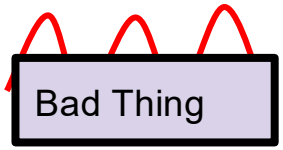
- For hyperproperties, a bad thing is a finite set of finite traces!





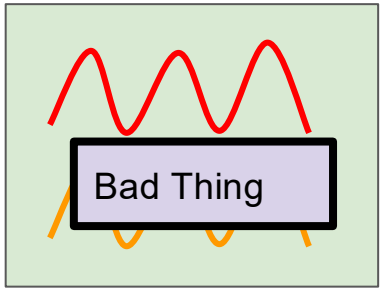
Lifting ‘Bad things’ to systems/trace sets

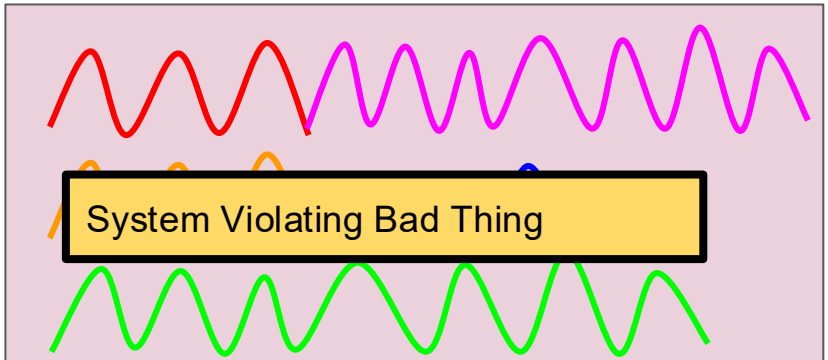
- For trace properties, a bad thing was a finite trace.





- For hyperproperties, a bad thing is a finite set of finite traces!





Lifting ‘Bad things’ to systems/trace sets

Lifting ‘Bad things’ to systems/trace sets

More formally,

Lifting ‘Bad things’ to systems/trace sets

More formally,

- An **Observation** is a finite set of finite traces

Lifting ‘Bad things’ to systems/trace sets

More formally,

- An **Observation** is a finite set of finite traces

Let T be a system (set of executions/infinite traces)

Lifting ‘Bad things’ to systems/trace sets

More formally,

- An **Observation** is a finite set of finite traces

Let T be a system (set of executions/infinite traces)

Let M be an observation.

Lifting ‘Bad things’ to systems/trace sets

More formally,

- An **Observation** is a finite set of finite traces

Let T be a system (set of executions/infinite traces)

Let M be an observation.

M ≤ T iff:

Lifting ‘Bad things’ to systems/trace sets

More formally,

- An **Observation** is a finite set of finite traces

Let T be a system (set of executions/infinite traces)

Let M be an observation.

M ≤ T iff:

- $\forall m \in M: (\exists t \in T : m \leq t)$

Lifting ‘Bad things’ to systems/trace sets

More formally,

- An **Observation** is a finite set of finite traces

Let T be a system (set of executions/infinite traces)

Let M be an observation.

M ≤ T iff:

- $\forall m \in M: (\exists t \in T : m \leq t)$
- It is now very easy to define hypersafety, and hyperliveness

Lifting ‘Bad things’ to systems/trace sets

More formally,

- An **Observation** is a finite set of finite traces

Let T be a system (set of executions/infinite traces)

Let M be an observation.

M ≤ T iff:

- $\forall m \in M: (\exists t \in T : m \leq t)$
- It is now very easy to define hypersafety, and hyperliveness
    - Left as an exercise ;)

More? Lot’s more!

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.
- K-Safety

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.
- K-Safety
  - A Hypersafety property where bad things are at most K-traces

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.
- K-Safety
  - A Hypersafety property where bad things are at most K-traces
- Examples of Hyperliveness?

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.
- K-Safety
  - A Hypersafety property where bad things are at most K-traces
- Examples of Hyperliveness?
  - Average Response Time

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.
- K-Safety
  - A Hypersafety property where bad things are at most K-traces
- Examples of Hyperliveness?
  - Average Response Time
  - ‘Maybe’ something happens

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.
- K-Safety
  - A Hypersafety property where bad things are at most K-traces
- Examples of Hyperliveness?
  - Average Response Time
  - ‘Maybe’ something happens
- Refinement

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.
- K-Safety
  - A Hypersafety property where bad things are at most K-traces
- Examples of Hyperliveness?
  - Average Response Time
  - ‘Maybe’ something happens
- Refinement
- Using this in practice?

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.
- K-Safety
  - A Hypersafety property where bad things are at most K-traces
- Examples of Hyperliveness?
  - Average Response Time
  - ‘Maybe’ something happens
- Refinement
- Using this in practice?
  - Logics?

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.
- K-Safety
  - A Hypersafety property where bad things are at most K-traces
- Examples of Hyperliveness?
  - Average Response Time
  - ‘Maybe’ something happens
- Refinement
- Using this in practice?
  - Logics?
  - Decision procedures?

More? Lot’s more!

- Every Hyper Property can be expressed as intersection of Hypersafety and Hyperliveness.
- K-Safety
  - A Hypersafety property where bad things are at most K-traces
- Examples of Hyperliveness?
  - Average Response Time
  - ‘Maybe’ something happens
- Refinement
- Using this in practice?
  - Logics?
  - Decision procedures?
  - RV for Hyperproperties?

Questions?