Incremental, Inductive Coverability

(to appear at CAV 2013)

Johannes Kloos Rupak Majumdar **Filip Niksic** Ruzica Piskac

Max Planck Institute for Software Systems

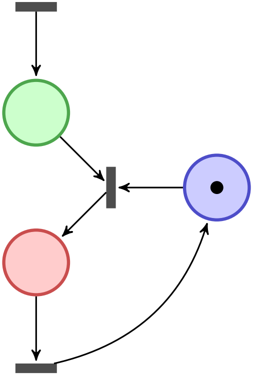
March 30, 2013, Belgrade

Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```

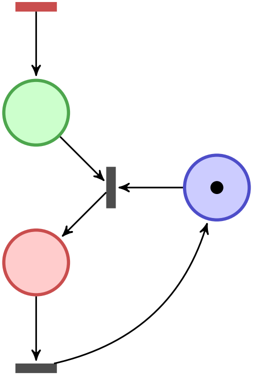
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



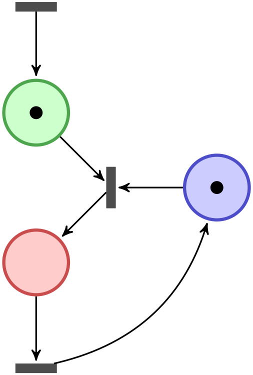
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



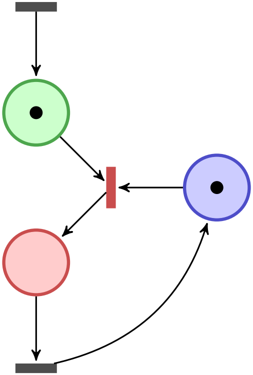
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



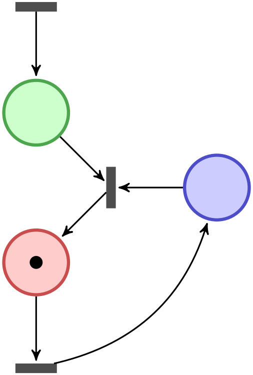
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



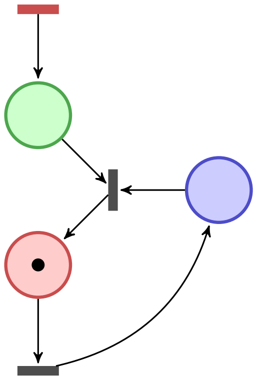
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



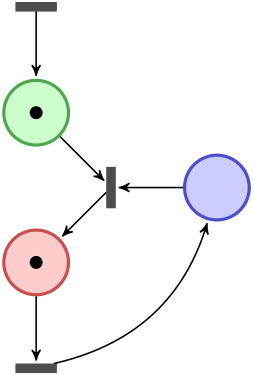
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



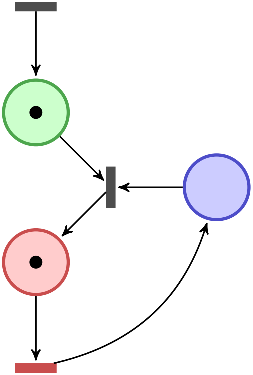
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



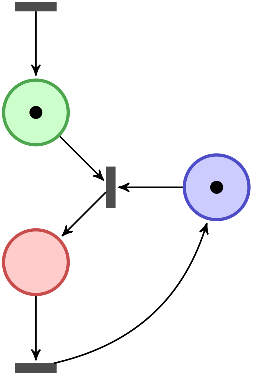
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



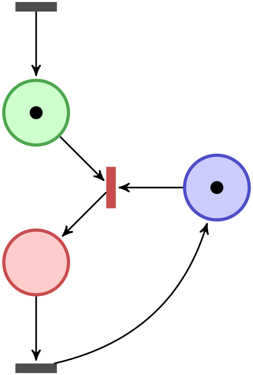
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



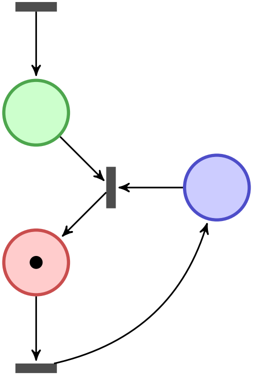
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



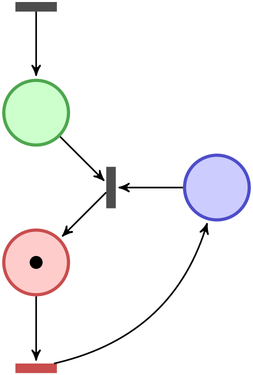
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



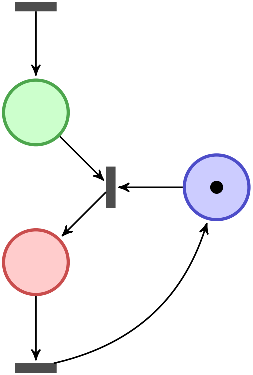
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



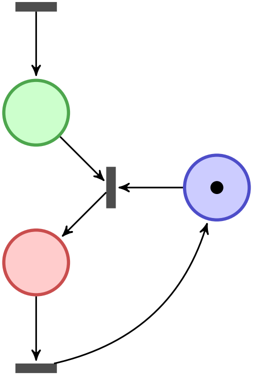
Abstraction Using Petri Nets

```
void thread()
{
    //Non-critical section

    synchronized(lock)
    {
        //Critical section
    }
}
```



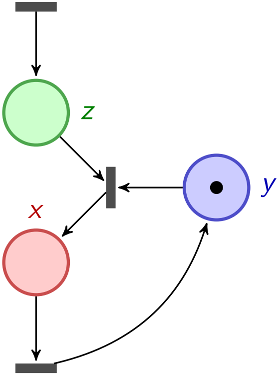
Coverability Problem

- Can two or more threads be in the critical section?
- Can we reach $(x, y, z) \geq (2, 0, 0)$?
- Can we cover $(2, 0, 0)$?



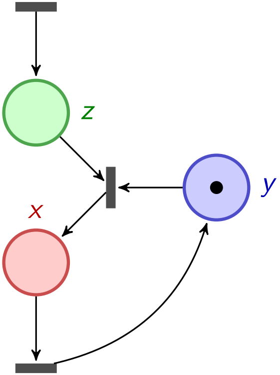
Coverability Problem

- Can two or more threads be in the critical section?
- Can we reach $(x, y, z) \geq (2, 0, 0)$?
- Can we cover $(2, 0, 0)$?

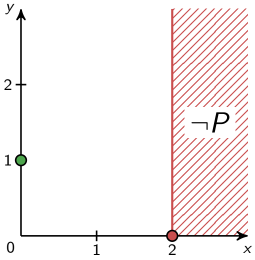


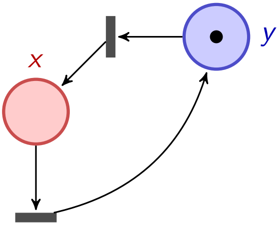
Coverability Problem

- Can two or more threads be in the critical section?
- Can we reach $(x, y, z) \geq (2, 0, 0)$?
- Can we cover $(2, 0, 0)$?

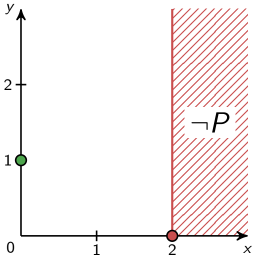


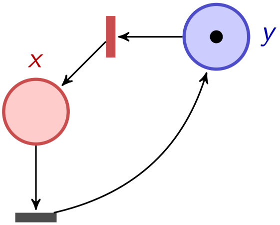
Visualization in the Coordinate System



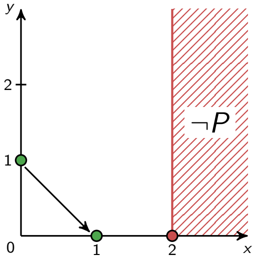


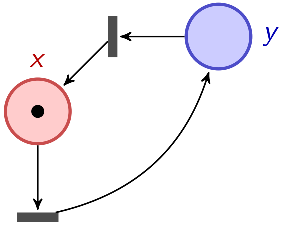
Visualization in the Coordinate System



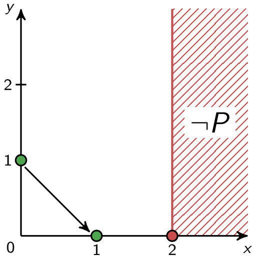


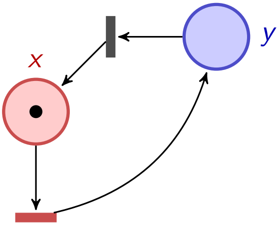
Visualization in the Coordinate System



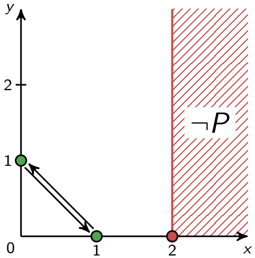


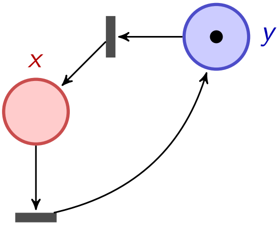
Visualization in the Coordinate System



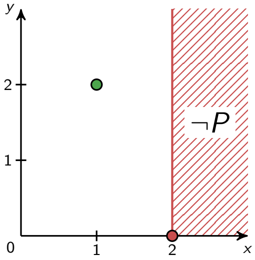


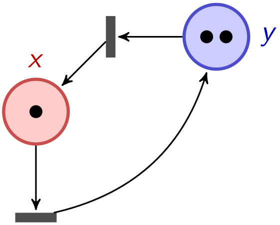
Visualization in the Coordinate System



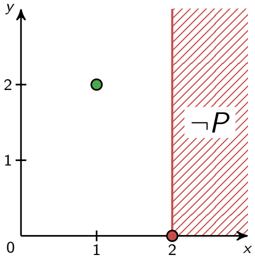


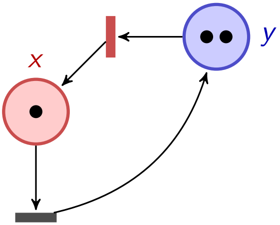
Visualization in the Coordinate System



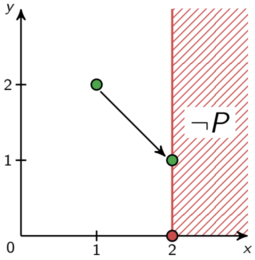


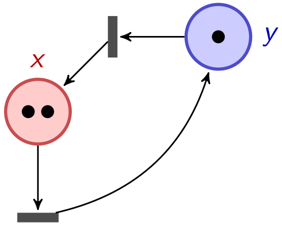
Visualization in the Coordinate System



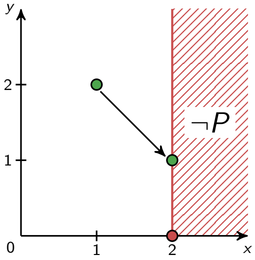


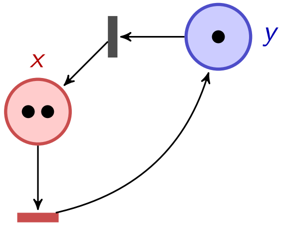
Visualization in the Coordinate System



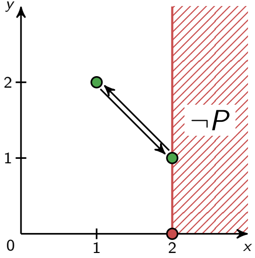


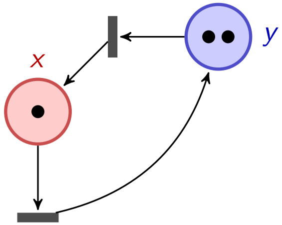
Visualization in the Coordinate System



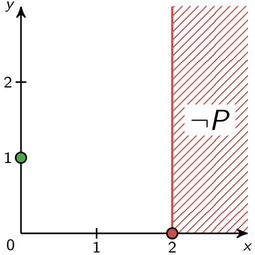


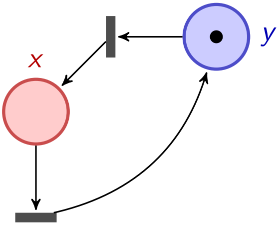
Visualization in the Coordinate System



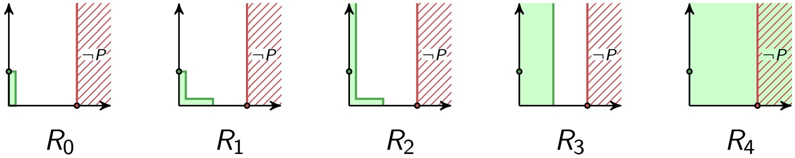


Visualization in the Coordinate System



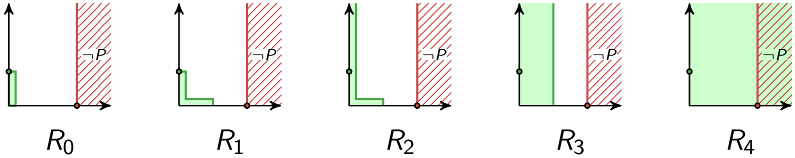


Overview of the Algorithm



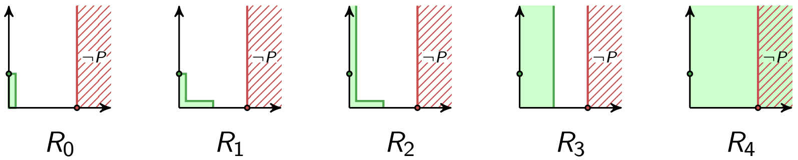
- R_k over-approximate states reachable in k steps.
- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\text{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

Overview of the Algorithm



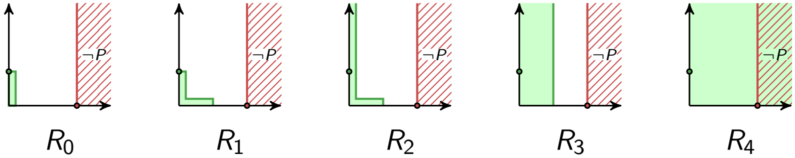
- R_k over-approximate states reachable in k steps.
- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\text{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

Overview of the Algorithm



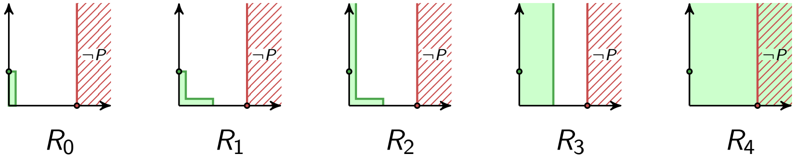
- R_k over-approximate states reachable in k steps.
- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\text{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

Overview of the Algorithm



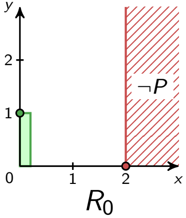
- R_k over-approximate states reachable in k steps.
- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\text{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

Overview of the Algorithm

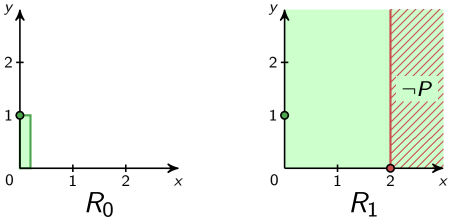


- R_k over-approximate states reachable in k steps.
- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\text{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

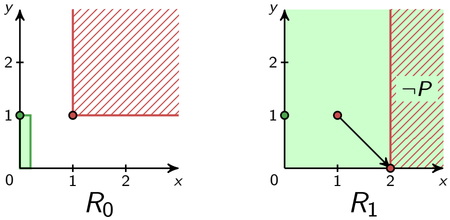
Proving the Uncoverability



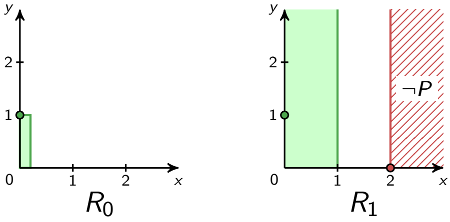
Proving the Uncoverability



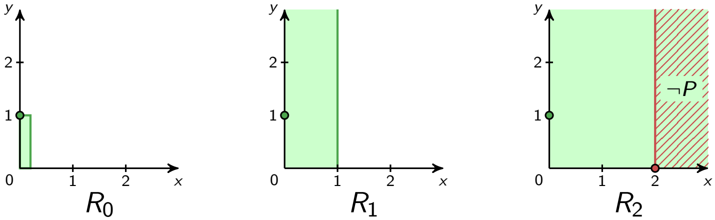
Proving the Uncoverability



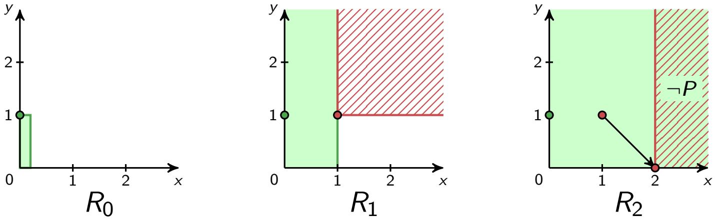
Proving the Uncoverability



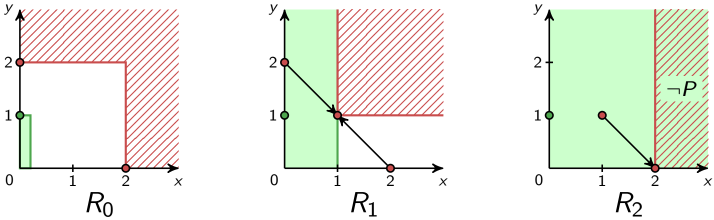
Proving the Uncoverability



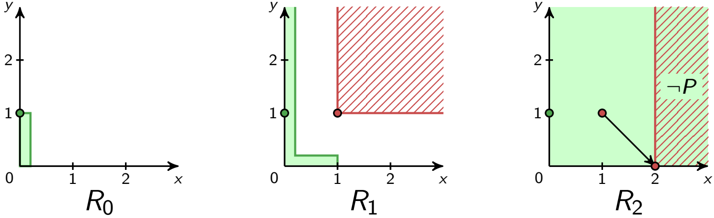
Proving the Uncoverability



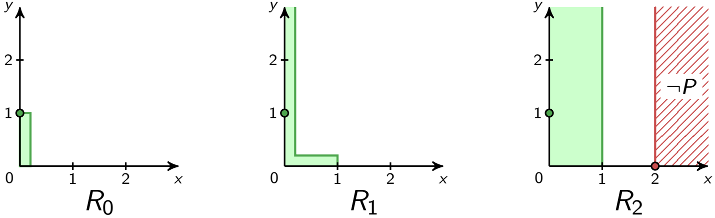
Proving the Uncoverability



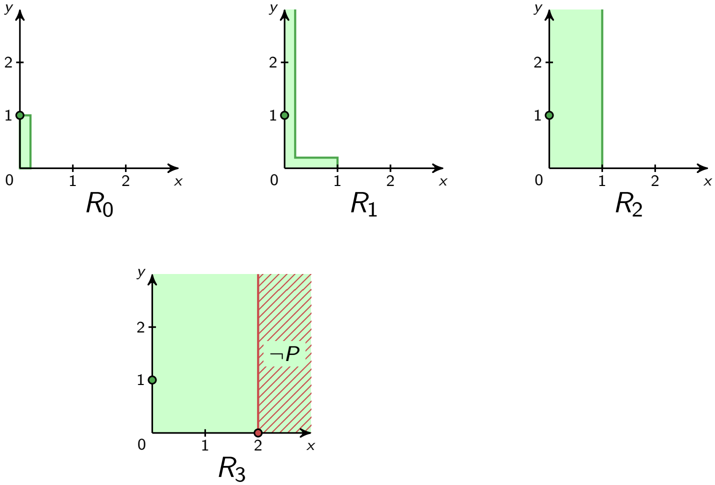
Proving the Uncoverability



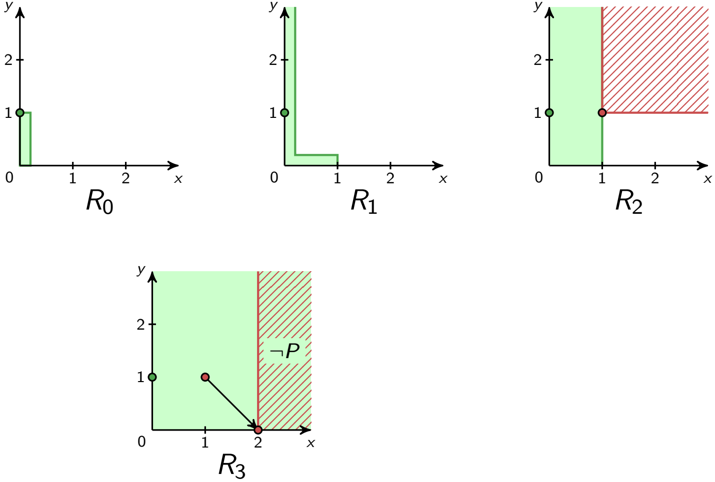
Proving the Uncoverability



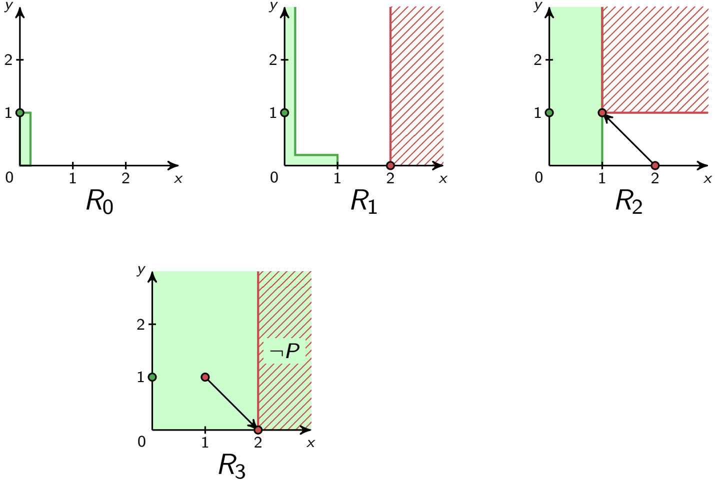
Proving the Uncoverability



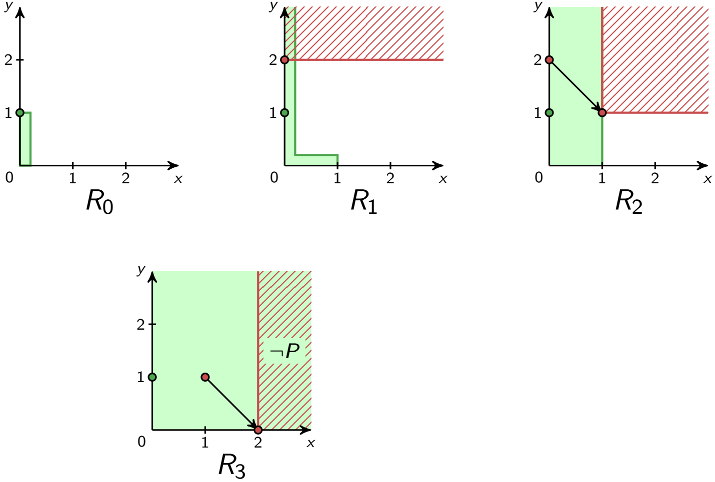
Proving the Uncoverability



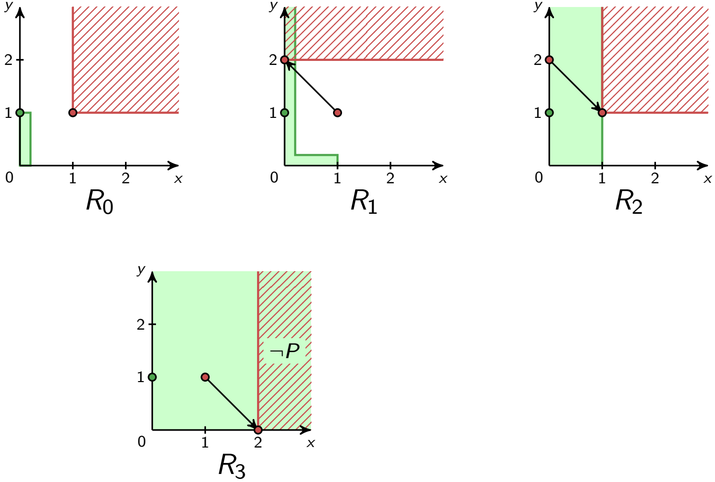
Proving the Uncoverability



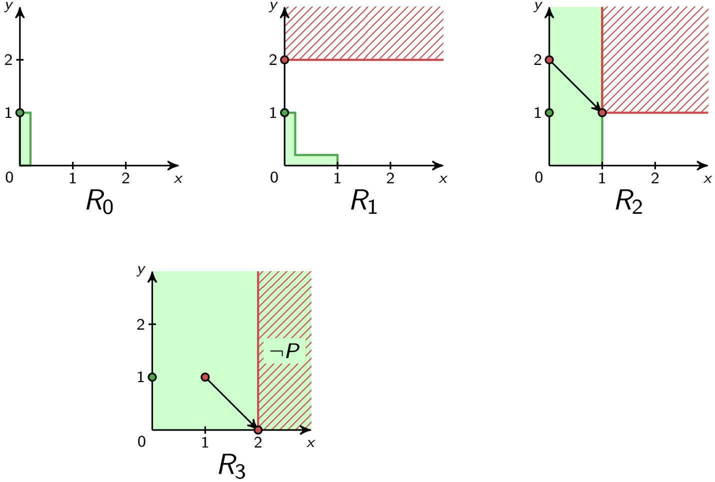
Proving the Uncoverability



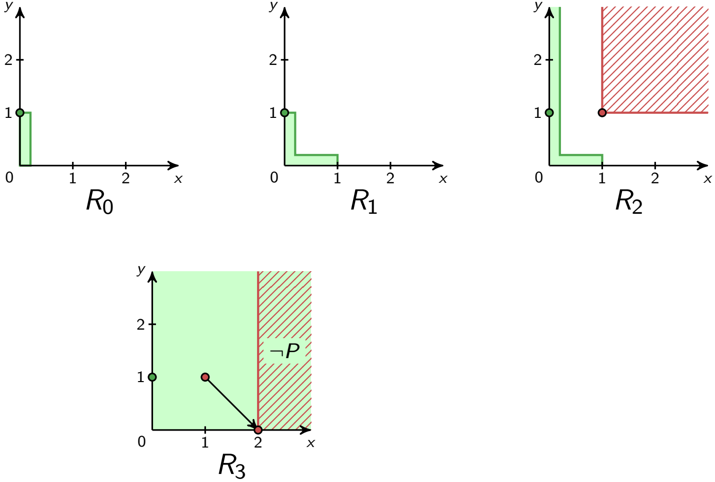
Proving the Uncoverability



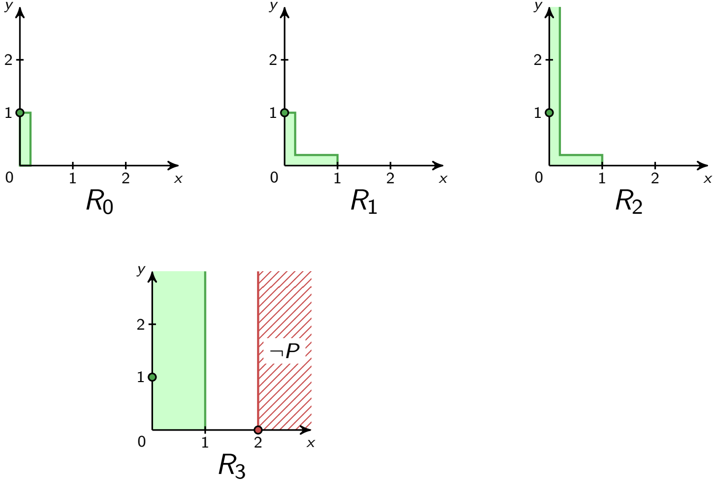
Proving the Uncoverability



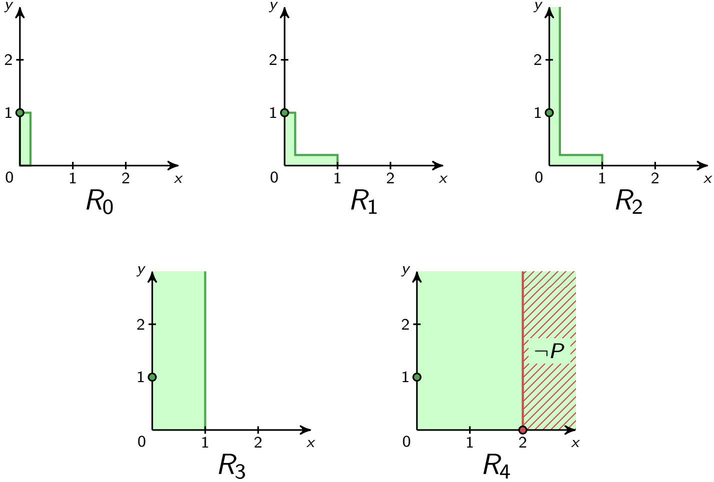
Proving the Uncoverability



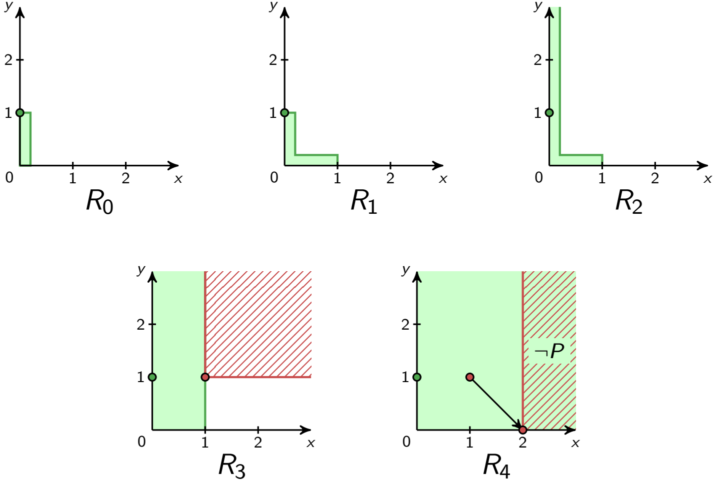
Proving the Uncoverability



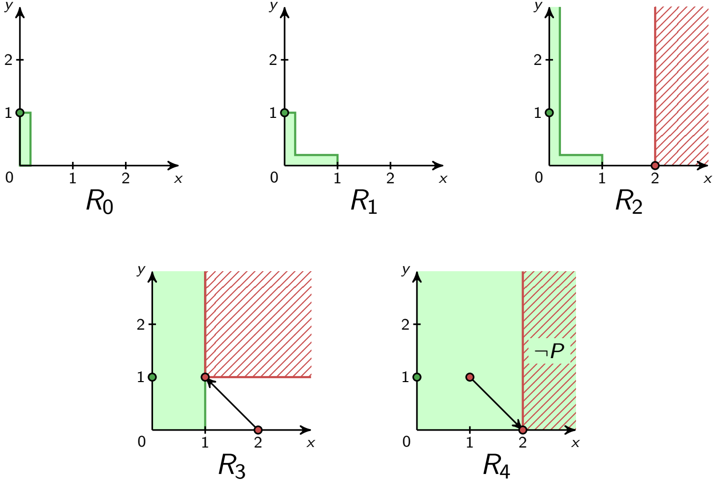
Proving the Uncoverability



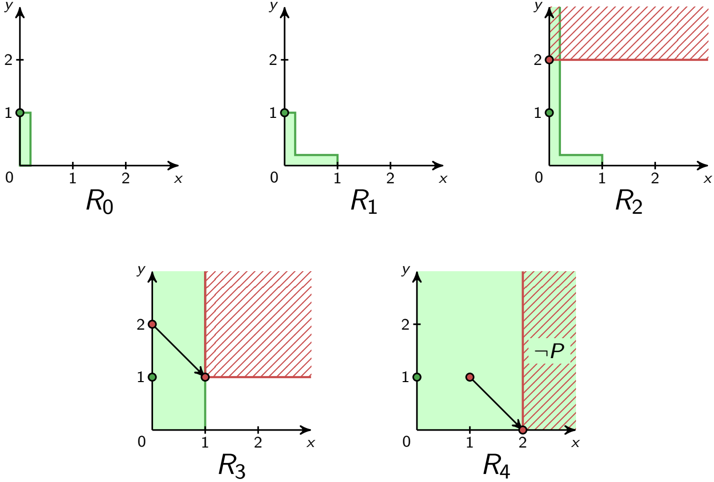
Proving the Uncoverability



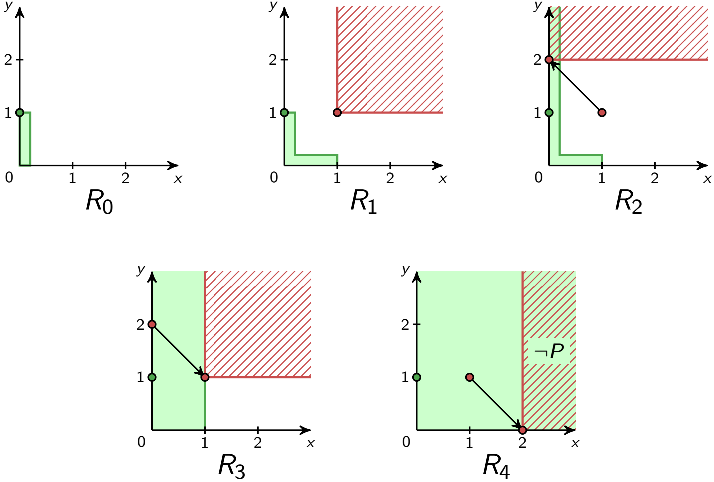
Proving the Uncoverability



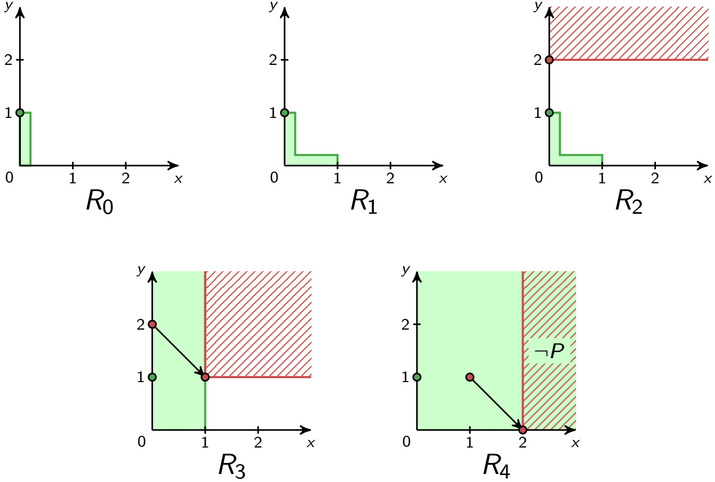
Proving the Uncoverability



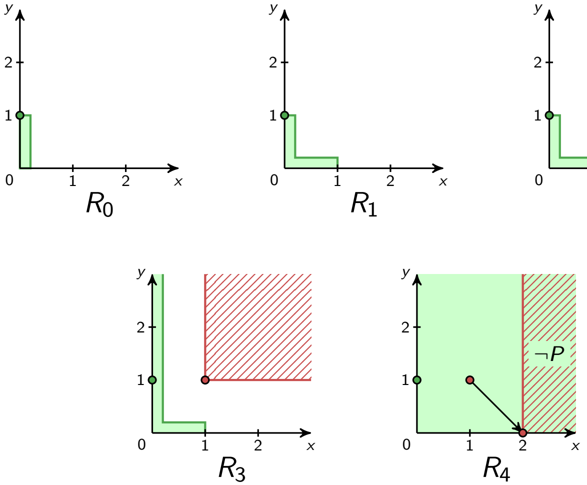
Proving the Uncoverability



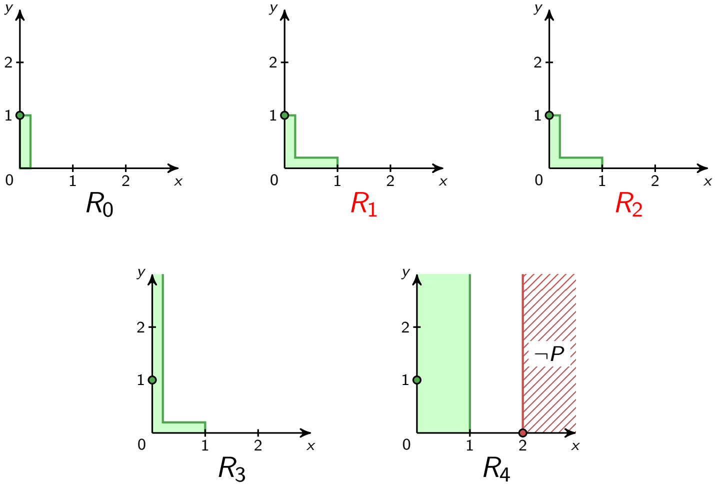
Proving the Uncoverability



Proving the Uncoverability



Proving the Uncoverability



Inductive Invariant

During the execution:

- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\text{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

If $R_i = R_{i+1}$:

- $I \subseteq R_i$
- $\text{post}(R_i) \subseteq R_i$
  $\Rightarrow R_i$ is inductive invariant
- $R_i \subseteq P$
  $\Rightarrow P$ is invariant.

Inductive Invariant

During the execution:

- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\mathrm{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

If $R_i = R_{i+1}$:

- $I \subseteq R_i$
- $\mathrm{post}(R_i) \subseteq R_i$
  $\Rightarrow$ R_i is inductive invariant
- $R_i \subseteq P$
  $\Rightarrow$ P is invariant.

Inductive Invariant

During the execution:

- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\mathrm{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

If $R_i = R_{i+1}$:

- $I \subseteq R_i$
- $\mathrm{post}(R_i) \subseteq R_i$
  $\Rightarrow R_i$ is inductive invariant
- $R_i \subseteq P$
  $\Rightarrow P$ is invariant.

Inductive Invariant

During the execution:

- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\text{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

If $R_i = R_{i+1}$:

- $I \subseteq R_i$
- $\text{post}(R_i) \subseteq R_i$
  $\Rightarrow R_i$ is inductive invariant
- $R_i \subseteq P$
  $\Rightarrow P$ is invariant.

Inductive Invariant

During the execution:

- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\mathrm{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

If $R_i = R_{i+1}$:

- $I \subseteq R_i$
- $\mathrm{post}(R_i) \subseteq R_i$

  $\Rightarrow$ R_i is inductive invariant
- $R_i \subseteq P$

  $\Rightarrow$ P is invariant.

Inductive Invariant

During the execution:

- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\mathrm{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

If $R_i = R_{i+1}$:

- $I \subseteq R_i$
- $\mathrm{post}(R_i) \subseteq R_i$
  $\Rightarrow R_i$ is inductive invariant
- $R_i \subseteq P$
  $\Rightarrow P$ is invariant.

Inductive Invariant

During the execution:

- $I \subseteq R_k$
- $R_k \subseteq R_{k+1}$
- $\text{post}(R_k) \subseteq R_{k+1}$
- $R_k \subseteq P$ for $k < N$

If $R_i = R_{i+1}$:

- $I \subseteq R_i$
- $\text{post}(R_i) \subseteq R_i$
  $\Rightarrow R_i$ is inductive invariant
- $R_i \subseteq P$
  $\Rightarrow P$ is invariant.

Incremental, Inductive Coverability

- Generalizes Aaron Bradley's IC3 to well-structured transition systems (which include Petri nets).
- Terminates for downward-finite WSTS.
- Efficient implementation for Petri nets.

Incremental, Inductive Coverability

- Generalizes Aaron Bradley's IC3 to well-structured transition systems (which include Petri nets).
- Terminates for downward-finite WSTS.
- Efficient implementation for Petri nets.

Incremental, Inductive Coverability

- Generalizes Aaron Bradley's IC3 to well-structured transition systems (which include Petri nets).
- Terminates for downward-finite WSTS.
- Efficient implementation for Petri nets.