# Reasoning about Actions

## Simple Transition System in ASP

Ira A. Fulton Schools of Engineering
Arizona State University

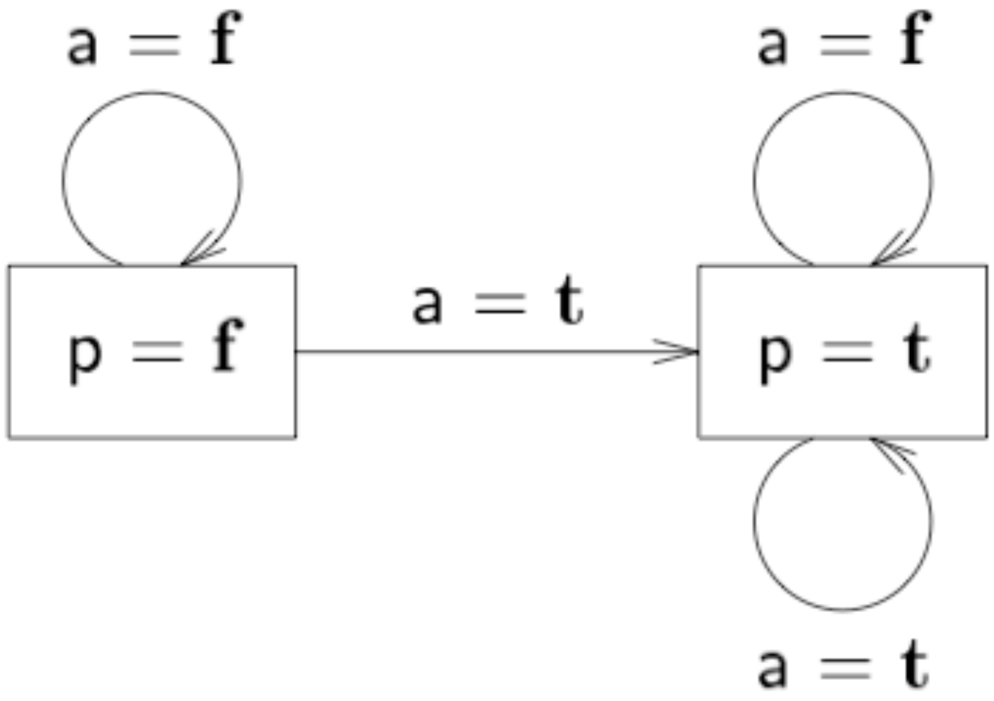
# Objectives

### Objective

Model simple transition systems in answer set programming

# Transition System

- A **transition system** is a directed graph
  - whose vertices correspond to the states of the world
  - whose edges are labelled by actions



# Representing Simple Transition System in ASP (I)

- **Let's describe transitions.**
- **We use the following atoms:**
  - `p(t,0), p(f,0), p(t,1), p(f,1), a(0).`
- **The effect of executing the action is described by**

```
p(t,1) :- a(0).
```

- **We still need to describe**

  (i) how to determine the value of p in the initial state

  (ii) how to determine whether action a is executed

  (iii) how to determine the value of p in the final state if a is not executed

  (iv) exactly one of the atoms `p(t,T), p(f,T)` is true at any time T = 0 or 1

# Representing Simple Transition System in ASP (II)

- **(i) how to determine the value of p in the initial state**
  - Answer: The initial state of the system is arbitrary:
  - `1{p(t,0); p(f,0)}1.`
- **Whichever causes determine the initial state of the system, they are outside the theory**
- **In other words, the value of p in the initial state is “exogenous.”**
- **The above rule also covers part of (iv) in the initial state**

# Representing Simple Transition System in ASP (III)

## (ii) how to determine whether action a is executed

- Answer: whichever causes determine whether or not the action is executed, they are outside the theory; the value of a is exogenous

```
{a(0)}.
```

# Representing Simple Transition System in ASP (IV)

- **(iii) how to determine the value of p in the final state $s_1$ if a is not executed**
- **(iv) exactly one of the atoms `p(t,T), p(f,T)` is true at any time T**
  - Answer: when action a is not executed, the value of p in the next state is determined by "the commonsense law of inertia":

```
{p(t,1)} :- p(t,0).
{p(f,1)} :- p(f,0).
:- not 1{p(t,1); p(f,1)}1.
```

# Questions

```
p(t,1) :- a(0).
1{p(t,0); p(f,0)}1.
{a(0)}.
{p(t,1)} :- p(t,0).
{p(f,1)} :- p(f,0).
:- not 1{p(t,1); p(f,1)}1.
```

**Q**: What will be the value of p at time 1 when p is false at time 0 and a(0) is false?

{ p(f, 0), p(f, 1) }

**Q**: What will be the value of p at time 1 when p is false at time 0 and a(0) is true?

{ p(f, 0), a(0), p(t, 1) }

# ASP Solution to the Frame Problem

```
{p(t,1)} :- p(t,0).
{p(f,1)} :- p(f,0).
:- not 1{p(t,1); p(f,1)}1.
```

- Second rule says if the value of p is f at time 0, then decide arbitrarily whether to assert that p is f at time 1.
- In the absence of additional information about p at time 1, asserting p(f,1) will be the only option (instead of not asserting it), as the last rule requires one of p(t,1), p(f,1) must be true.
- But if we are given conflicting information about the value of p at time 1, then not asserting p(t,1) is the only option.

# Problem

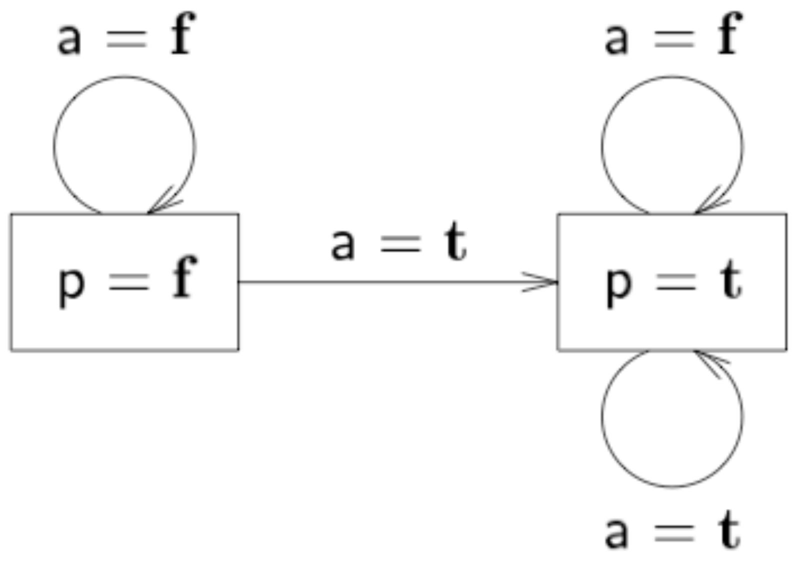
Use clingo to check that the program consisting of the six rules above has 4 stable models, each of which correspond to the transitions in the simple transition system.

# Representing Histories of Simple Transition System

- To get a theory whose models correspond to the histories of the simple domain whose length is m (m ≥ 0), we introduce
  - atoms `p(t,i), p(f,i)` for `i = 0,...,m`
  - `a(i)` for `i = 0,...,m-1.`
- The values of p(t,i), p(f,i) characterize state $s_i$ – it gives the value of the parameter p in that state.
- The value of `a(i)` characterizes the event occurring between states $s_i$ and $s_{i+1}$ – it tells us whether that event included the execution of action a.

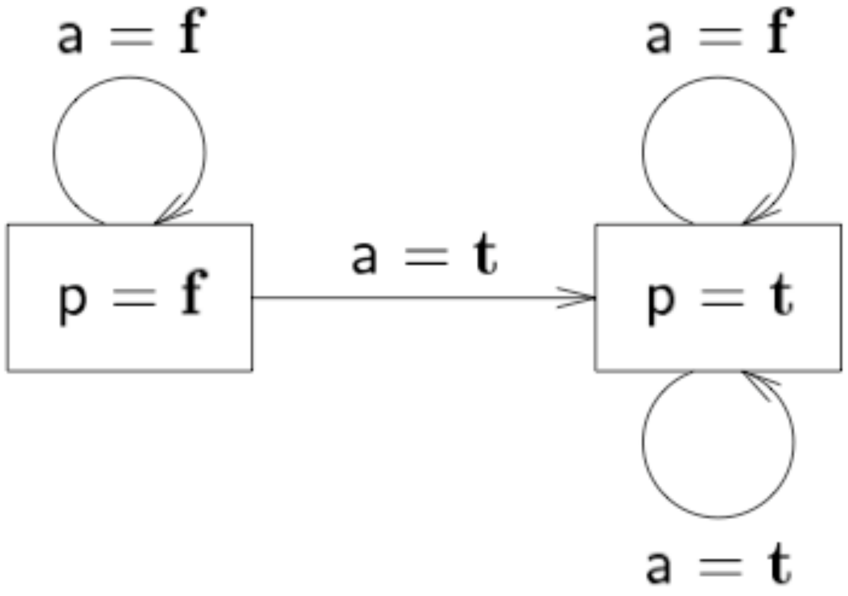
# Simple Transition System in the Language of Clingo

```
% File ‘simple.lp’
boolean(t;f).

% direct effect
p(t,T+1) :- a(T), T=0..m-1.

% initial status are exogenous
1{p(B,0):boolean(B)}1.

% uniqueness and existence of values
:- not 1{p(B,T):boolean(B)}1, T=1..m.

% actions are exogenous
{a(T)} :- T=0..m-1.

% commonsense law of inertia
{p(B,T+1)} :- p(B,T), T=0..m-1.
```



# Find All States

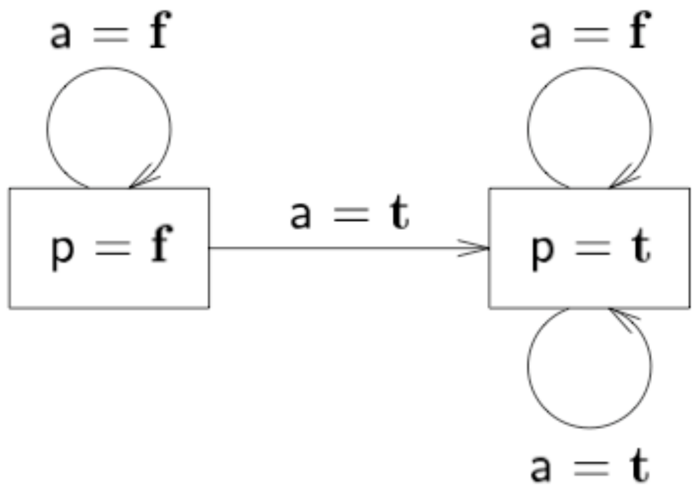
```
$ clingo simple.lp -c m=0 0
clingo version 5.3.0
Reading from simple
Solving...
Answer: 1
p(f,0)
Answer: 2
p(t,0)
SATISFIABLE

Models      : 2
Calls       : 1
Time        : 0.013s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time    : 0.005s
```



# Find All Transitions

```
$ clingo simple -c m=1 0
clingo version 5.2.1
Reading from simple
Solving...
Answer: 1
p(t,1) p(t,0)
Answer: 2
a(0) p(t,1) p(t,0)
Answer: 3
p(f,1) p(f,0)
Answer: 4
a(0) p(t,1) p(f,0)
SATISFIABLE
```

```
Models        : 4
Calls         : 1
Time          : 0.025s (Solving:
0.00s 1st Model: 0.00s Unsat:
0.00s)
CPU Time      : 0.007s
```



# Wrap-Up

# Problem

If we replace the rule

```
:- not 1{p(B,T) : boolean(B)}1, T=1..m.
```

of the program with

```
1{p(B,T) : boolean(B)}1 :- T=1..m.
```

How will the stable models change?