Lumpabilities and Noninterference in PEPA

Alberto Casagrande

UNIVERSITÀ
DEGLI STUDI
DI UDINE

PEPA - Performance Evaluation Process Algebra

A **stocastic process algebra** for modelling systems [H90]

PEPA - Performance Evaluation Process Algebra

A **stocastic process algebra** for modelling systems [H90]

$$P ::= (a, \lambda).\, P \mid P + P \mid P \underset{\mathrm{A}}{\bowtie} P \mid P/\mathrm{A} \mid A$$

PEPA - Performance Evaluation Process Algebra

A **stocastic process algebra** for modelling systems [H90]

$$P ::= (a, \lambda).\, P \mid P + P \mid P \underset{\mathrm{A}}{\bowtie} P \mid P/\mathrm{A} \mid A$$

- *Actions:* $(a, \lambda).\, Q$
- *Choises:* $P + Q$
- *Cooperations:* $P \underset{\mathrm{A}}{\bowtie} Q$
- *Hidings:* P/A
- *Definitions:* $A = P$

From Operational Semantics to CTMC

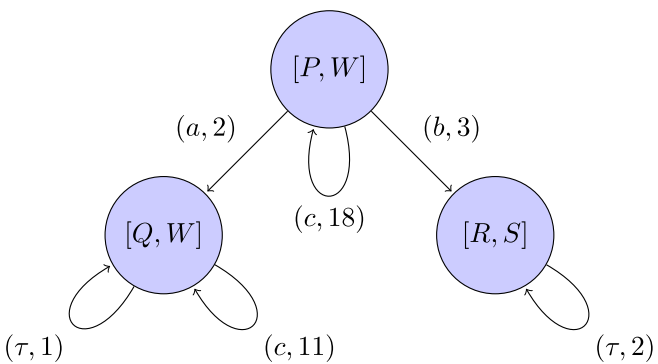
PEPA processes are **Continuous-Time Markov Chains**

```
P = (a, 2).Q + (b, 3).R + (c, 7).P;
W = (b, 5).S + (c, 11).W;

Q = (tau, 1).Q;
S = (tau, 1).S;
R = (tau, 1).R;

P<b>W
```

From Operational Semantics to CTMC

PEPA processes are **Continuous-Time Markov Chains**

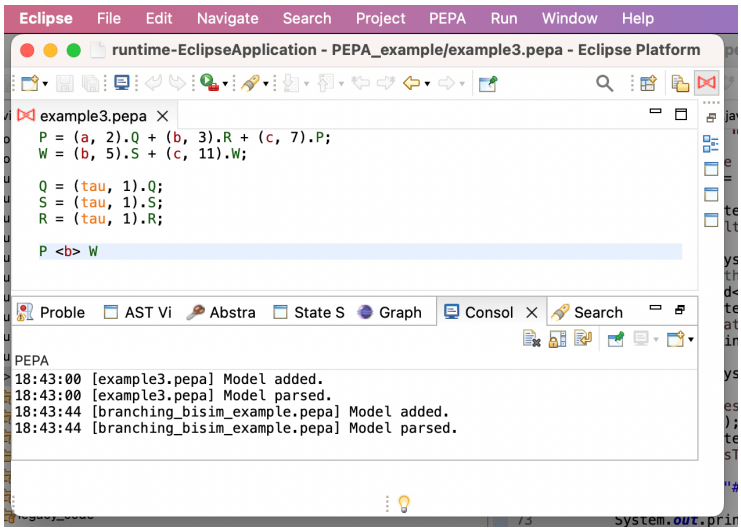
```
P = (a, 2).Q + (b, 3).R + (c, 7).P;
W = (b, 5).S + (c, 11).W;

Q = (tau, 1).Q;
S = (tau, 1).S;
R = (tau, 1).R;

P<b>W
```



- nodes are model states
- transitions are (synchonized) actions

PEPA Implementation

- model specification
- steady state analysis
- scalability analysis



Non-Interference

Is a property avoiding information leaks due to system behaviours

Non-Interference

Is a property avoiding information leaks due to system behaviours

- Non-τ actions are either **high level** (H) or **low level** (L)
- **Low level users** can only perform low level actions
- High actions should be "invisible" to low level users

Non-Interference

Is a property avoiding information leaks due to system behaviours

- Non-τ actions are either **high level** (H) or **low level** (L)
- **Low level users** can only perform low level actions
- High actions should be "invisible" to low level users

What does "invisible" mean?

Non-Interference

Is a property avoiding information leaks due to system behaviours

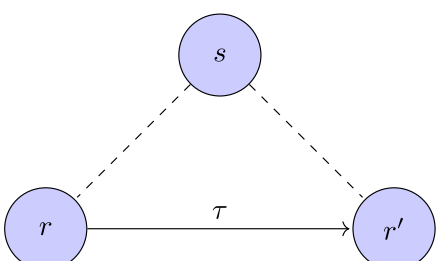
- Non-τ actions are either **high level** (H) or **low level** (L)
- **Low level users** can only perform low level actions
- High actions should be "invisible" to low level users

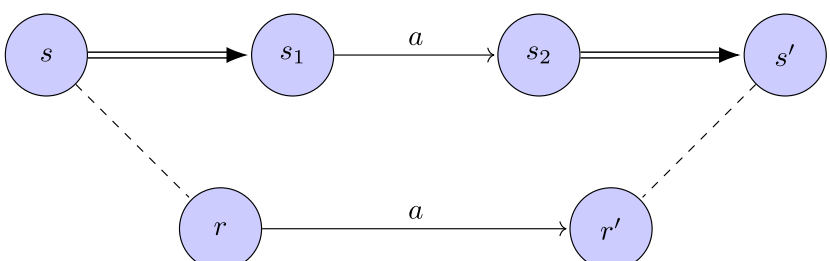
What does "invisible" mean? Equivalent w.r.t. some relation

Weak Bisimulations

R_w is symmetric and if $(r, s) \in R_w$ and $r \xrightarrow{a} r'$, either:

- $a = \tau$ and $(r', s) \in R_w$
- $s \Rightarrow s_1 \xrightarrow{a} s_2 \Rightarrow s'$ and $(r', s') \in R_w$

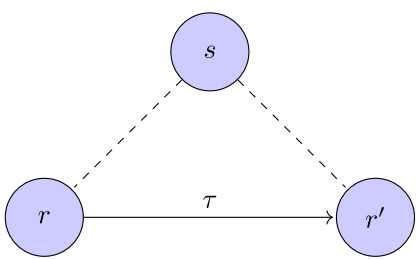


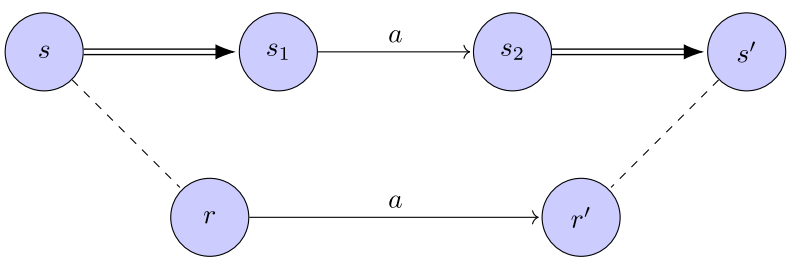


Brancing Bisimulations

R_b is symmetric and if $(r, s) \in R_b$ and $r \xrightarrow{a} r'$, either:

- $a = \tau$ and $(r', s) \in R_b$
- $s \Rightarrow s_1 \xrightarrow{a} s_2 \Rightarrow s'$ and $(r, s_1), (r', s_2), (r', s') \in R_b$

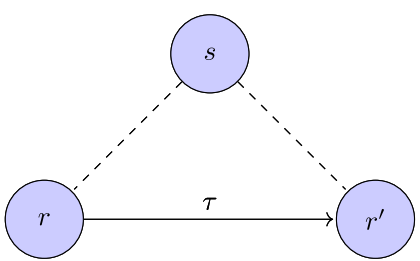


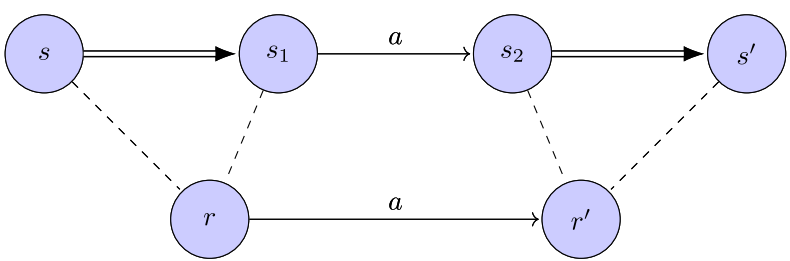


Brancing Bisimulations

R_b is symmetric and if $(r, s) \in R_b$ and $r \xrightarrow{a} r'$, either:

- $a = \tau$ and $(r', s) \in R_b$
- $s \Rightarrow s_1 \xrightarrow{a} s_2 \Rightarrow s'$ and $(r, s_1), (r', s_2), (r', s') \in R_b$





Lumpable Bisimulations

[HPR21] R_l is an equivalence relation on $\mathcal{P}$ and s.t. if $(r, s) \in R_l$, then, for all a and for all $C \in \mathcal{P}/R_l$ that either:

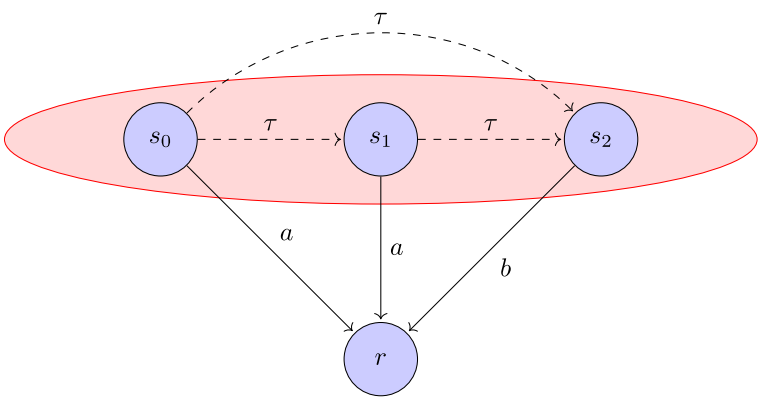
- $a \neq \tau$ or
- $a = \tau$ and $r, s \notin C$,

the following equation holds

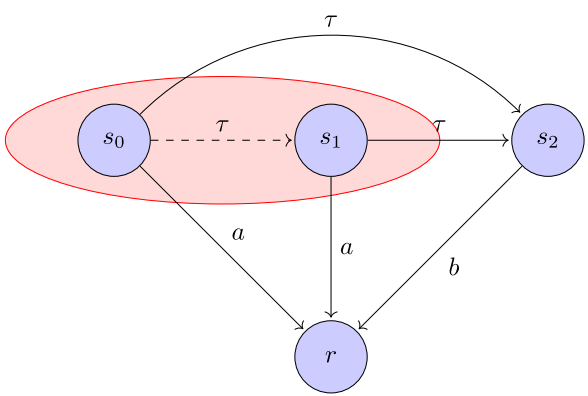
$$\sum_{c \in C} \text{rate}\left(r \xrightarrow{a} c\right) = \sum_{c' \in C} \text{rate}\left(s \xrightarrow{a} c'\right)$$

Branching vs Lumplable

Branching



Lumpable



Persistent Stochastic Non-Interference

Definition [HPR21]

$$P \in \mathrm{PSNI} \iff \forall P' \text{ s.t. } P \to^* P' \text{ and } \forall H \in \mathcal{C}_H \quad (P' \underset{\mathcal{H}}{\bowtie} 0)/\mathcal{H} \approx_l (P' \underset{\mathcal{H}}{\bowtie} H)/\mathcal{H}$$

Persistent Stochastic Non-Interference

Definition [HPR21]

$$P \in \mathrm{PSNI} \iff \forall P' \text{ s.t. } P \to^* P' \text{ and } \forall H \in \mathrm{C}_H \quad (P' \underset{\mathrm{H}}{\triangleright\!\triangleleft} 0)/\mathrm{H} \approx_l (P' \underset{\mathrm{H}}{\triangleright\!\triangleleft} H)/\mathrm{H}$$

Theorem [HPR21]

$$P \in \mathrm{PSNI} \iff (P \underset{\mathrm{H}}{\triangleright\!\triangleleft} 0)/\mathrm{H} \approx_l^{\mathrm{H}} P$$

Adding PSNI in PEPA Plugin

Two steps

- action level labelling
- PSNI testing command

Adding PSNI in PEPA Plugin

Two steps

- action level labelling (PEPA language)
- PSNI testing command (PEPA-plugin UI)

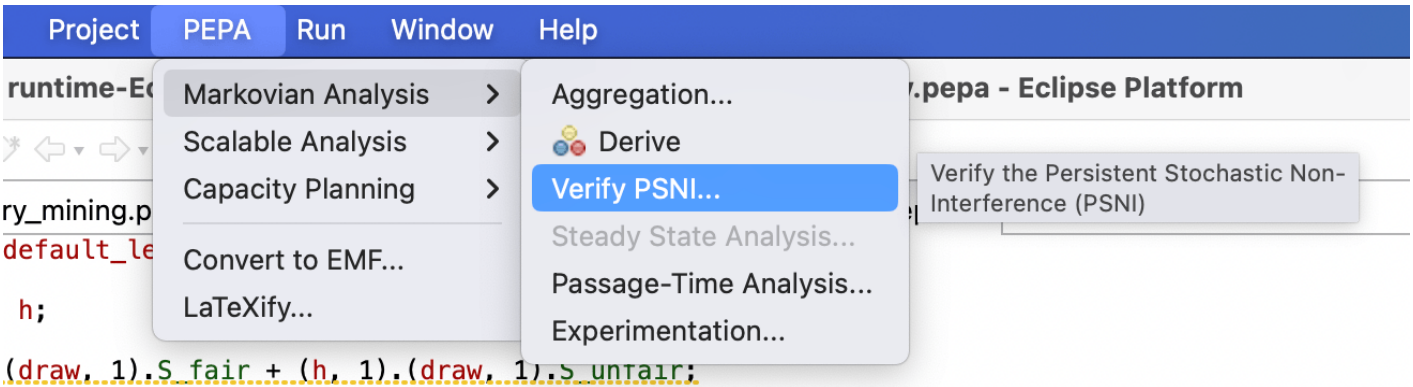
Action Level Labelling

```
 1 ar = 2;
 2 br = 3;
 3
 4 P = (a, ar).Q + (b, br).R + (c, 7).P;
 5 W = (b, 5).S + (c, 11).W;
 6
 7 Q = (tau, 1).Q;
 8 S = (tau, 1).S;
 9 R = (tau, 1).R;
10
11 P<b>W
```

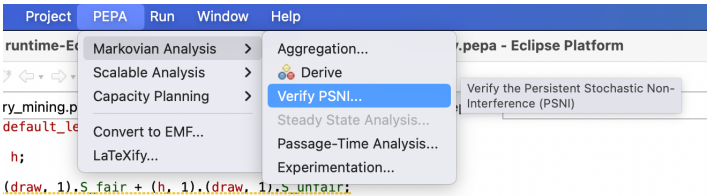
Action Level Labelling

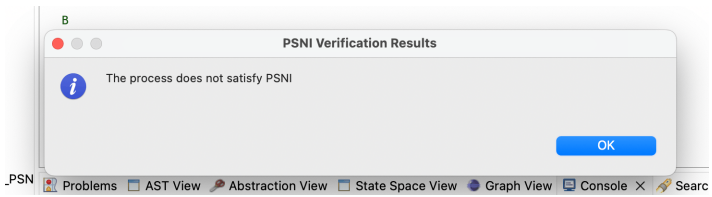
```
1  ar = 2;
2  br = 3;
3
4  set_default_level low;
5
6  high a, b;
7
8  P = (a, ar).Q + (b, br).R + (c, 7).P;
9  W = (b, 5).S + (c, 11).W;
10
11 Q = (tau, 1).Q;
12 S = (tau, 1).S;
13 R = (tau, 1).R;
14
15 P<b>W
```

PEPA Plug-in Integration

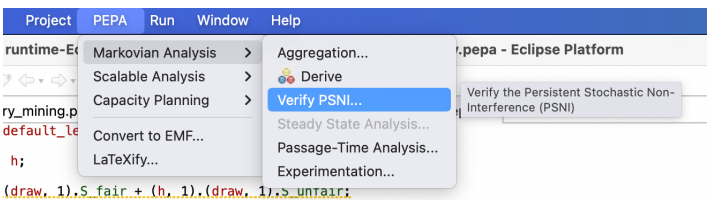


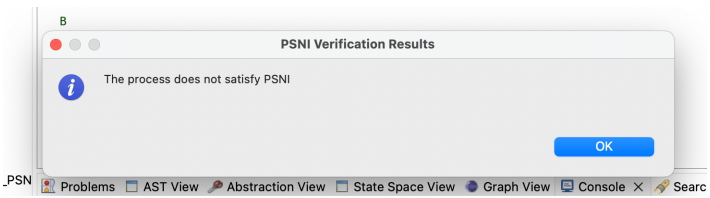
PEPA Plug-in Integration

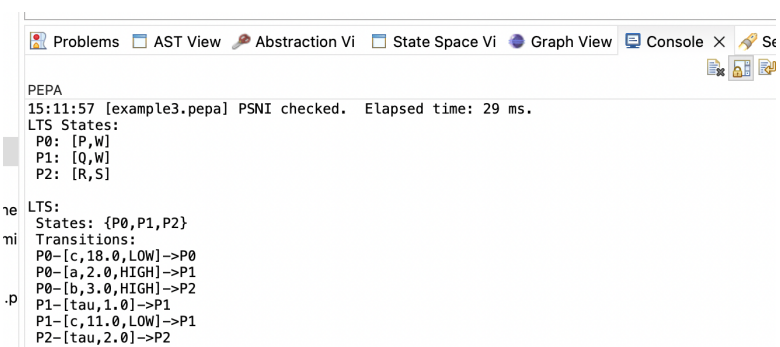




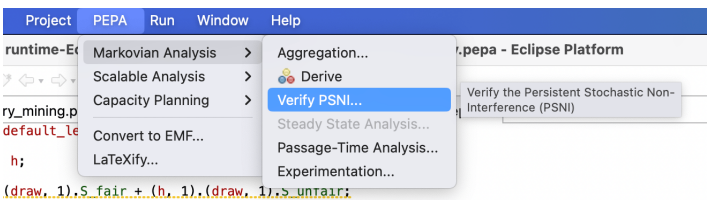
PEPA Plug-in Integration

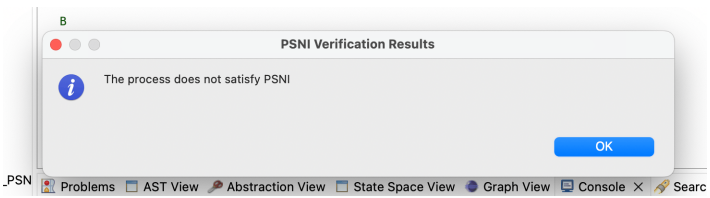


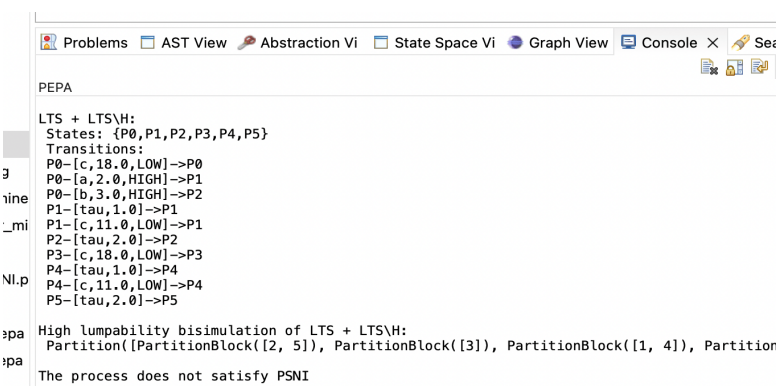




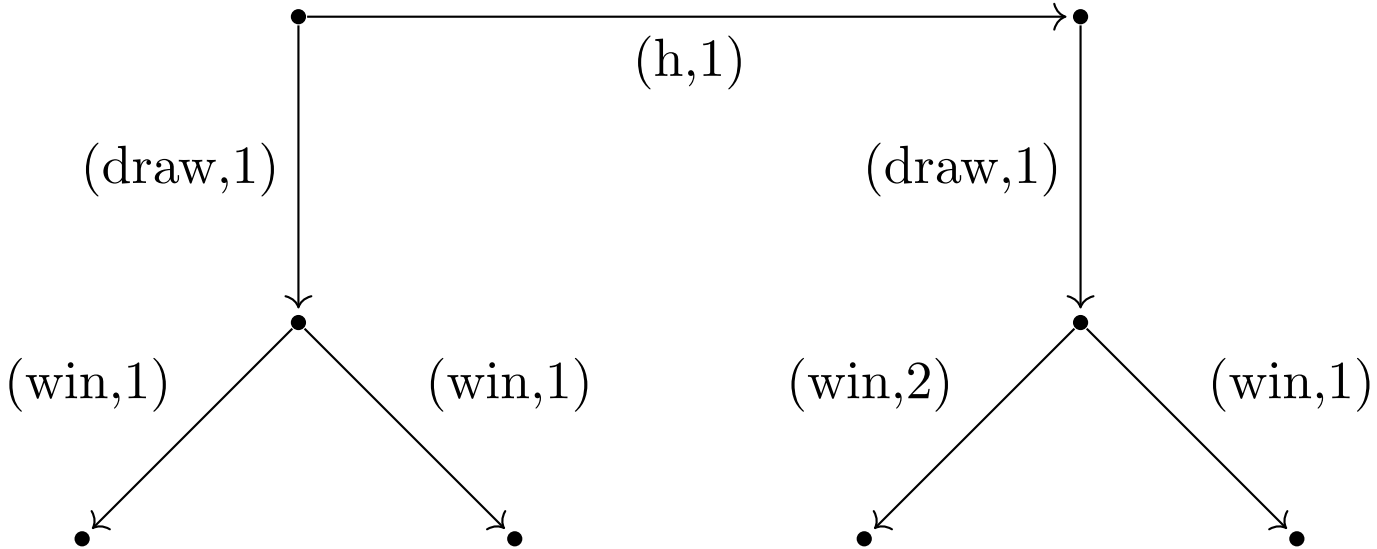
PEPA Plug-in Integration







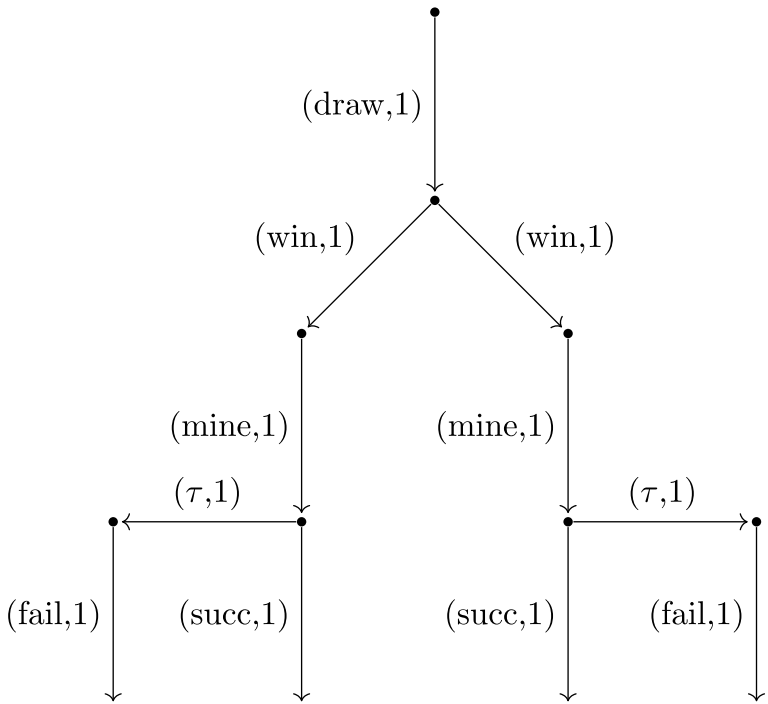
Example 1: Non-Uniform Lottery



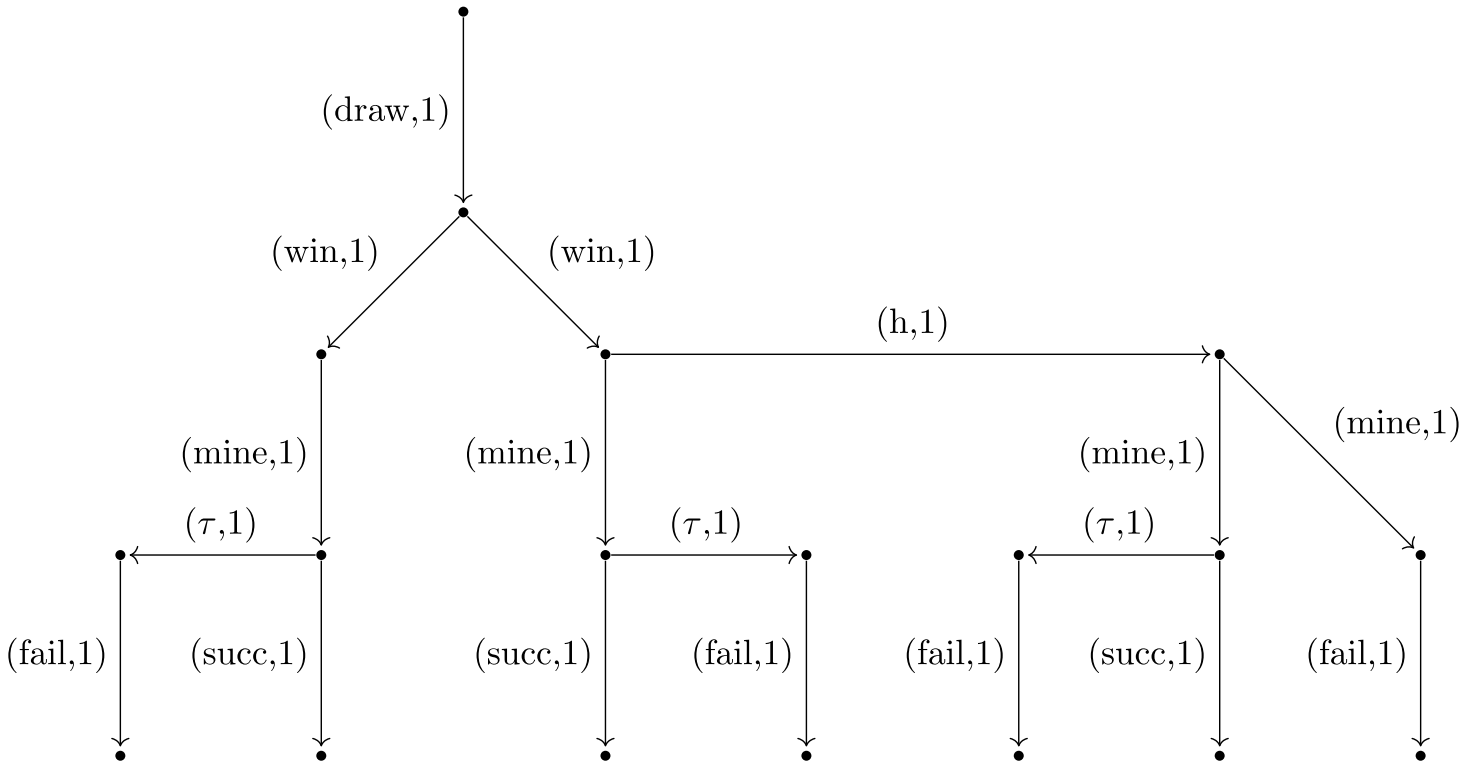
Example 1: Non-Uniform Lottery

```
1  set_default_level low;
2
3  high h;
4
5  B0 = (draw, 1).D0 + (h, 1).(draw, 1).D1;
6
7  D0 = (win, 1).W0 + (win, 1).W1;
8
9  W0 = (tau, 1).W0;
10 W1 = (tau, 1).W1;
11
12 D1 = (win, 2).W2 + (win, 1).W3;
13
14 W2 = (tau, 1).W2;
15 W3 = (tau, 1).W3;
16
17 B0
```

Example 2: Fair Lottery Verifier



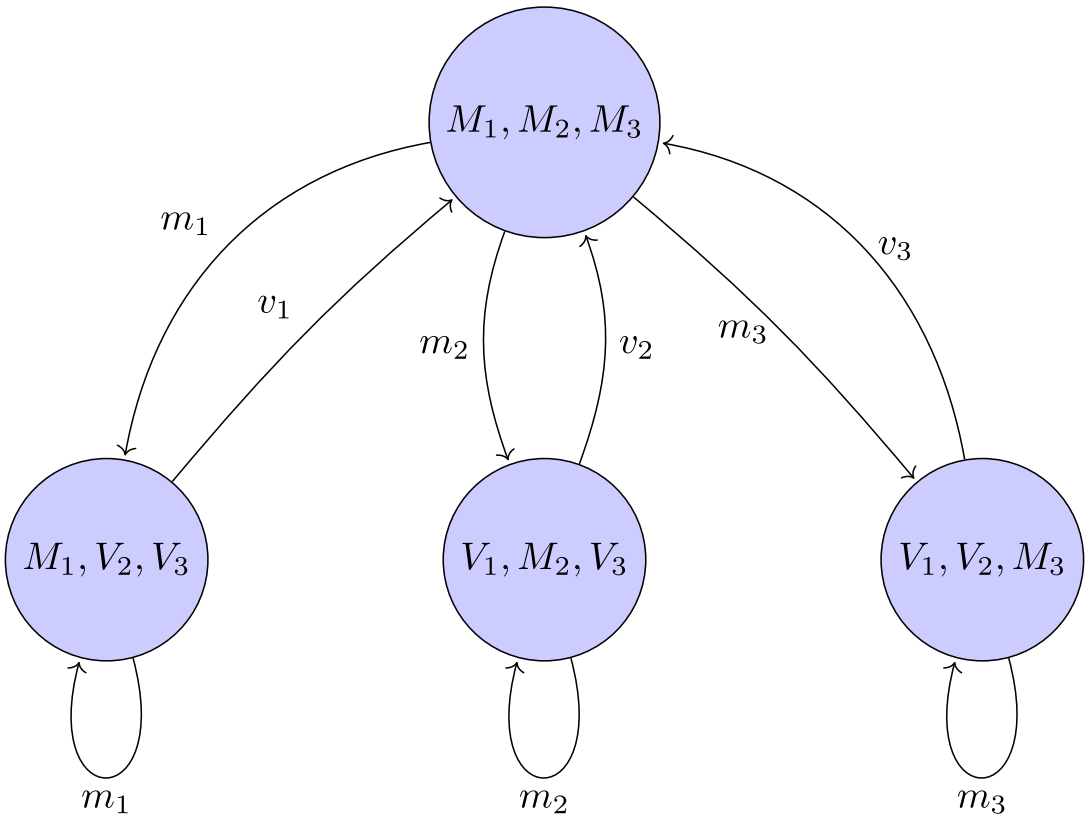
Example 2: Unfair Lottery Verifier



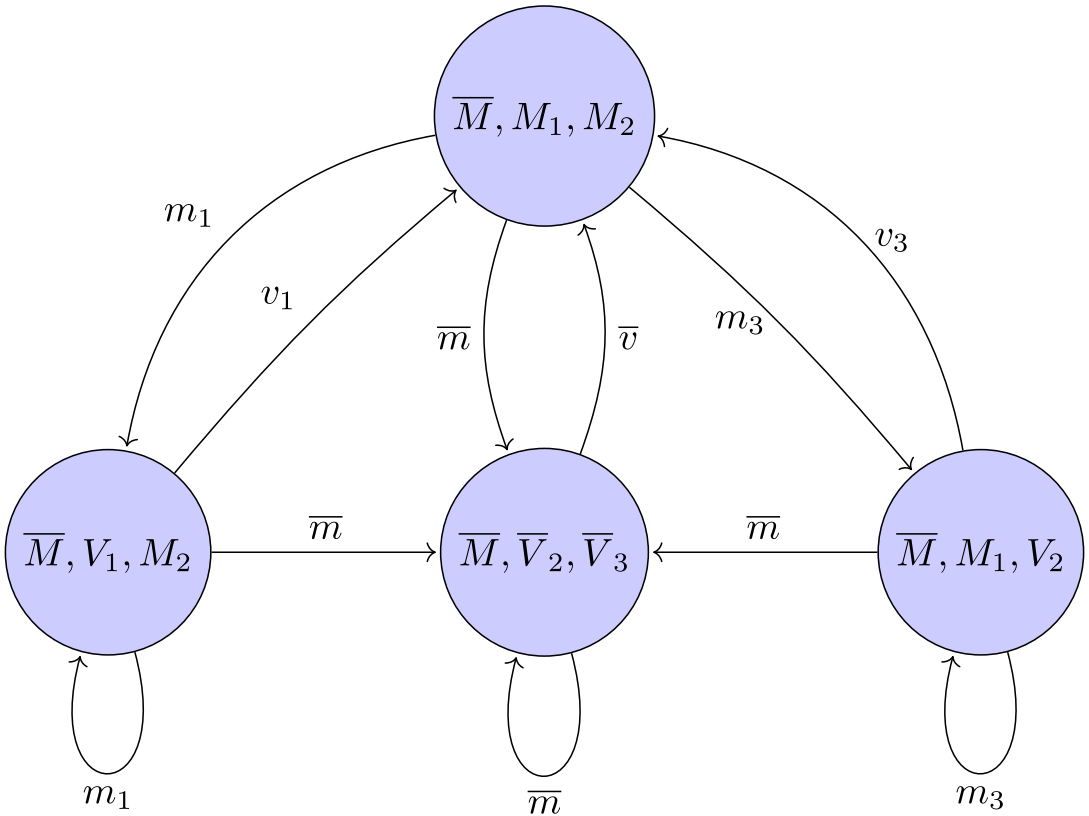
Example 2: Unfair Lottery Verifier

```
1  set_default_level low;
2  high h;
3
4  B = (win0, 1).W0 + (win1, 1).W1;
5
6  W0 = (mine, 1).((tau,1).(fail,1).F0 + S0);
7
8  W1 = (mine, 1).((tau,1).(fail,1).F1 + S1)
9       + (h, 1).((mine,1).(fail, 1).F2
10                  + (mine, 1).((tau,1).(fail,1).F3 + S3));
11
12 F0 = (tau, 1).F0;
13 S0 = (tau, 1).S0;
14 ...
15 F3 = (tau, 1).F3;
16 S3 = (tau, 1).S3;
17
18 B
```

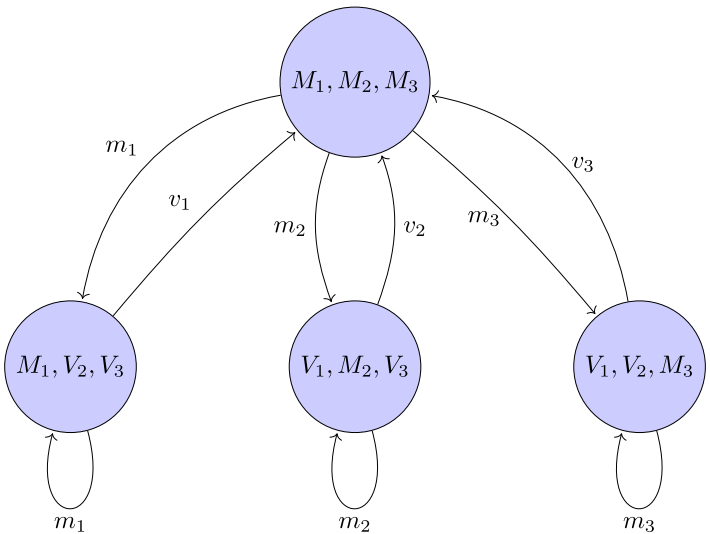
Example 3: Fair Miners [SMMvMR22]

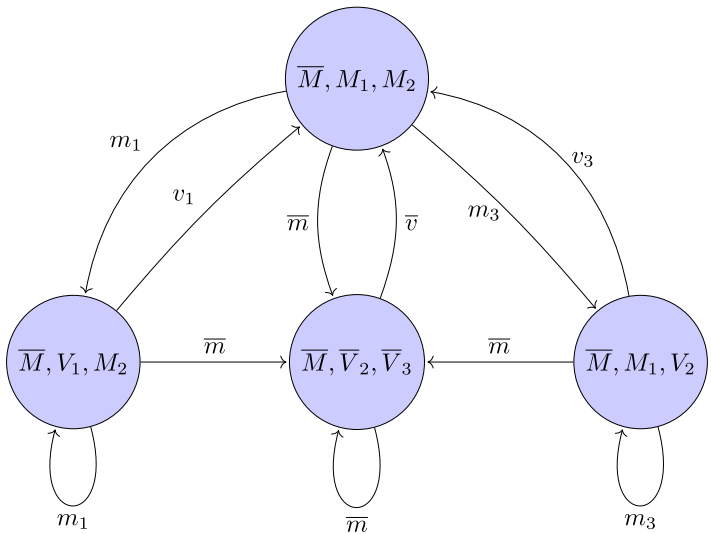


Example 3: Unfair Miner [SMMvMR22]

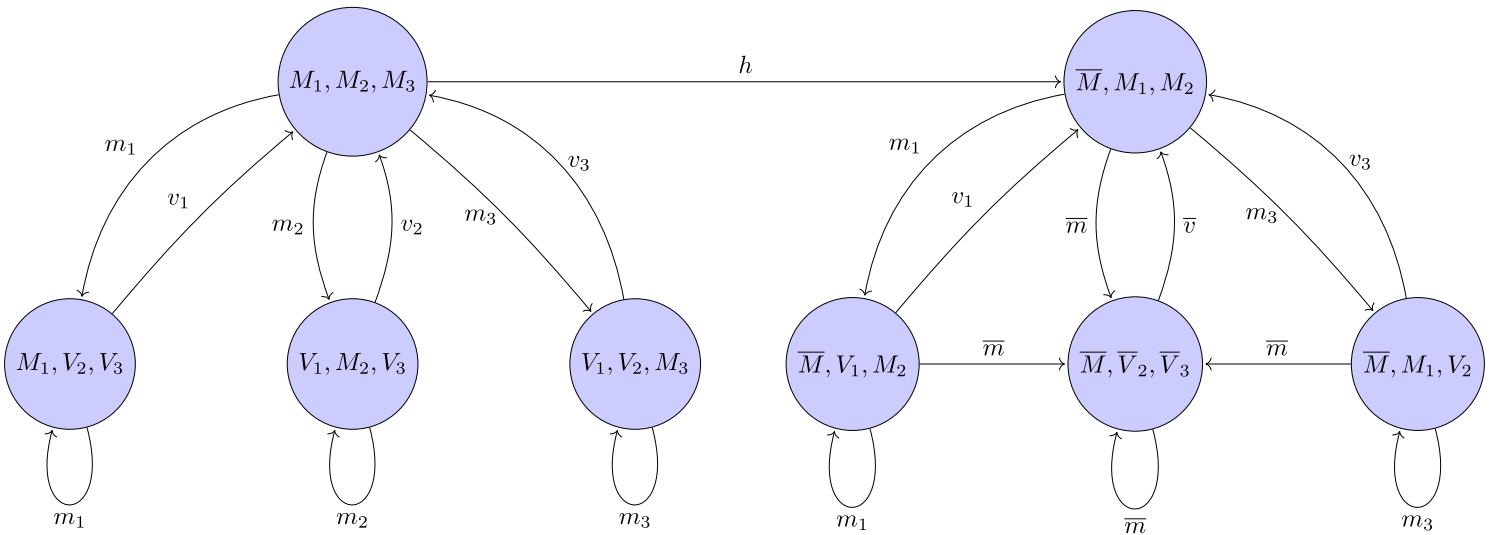


Example 3: Fair-Unfair Miner [SMMvMR22]





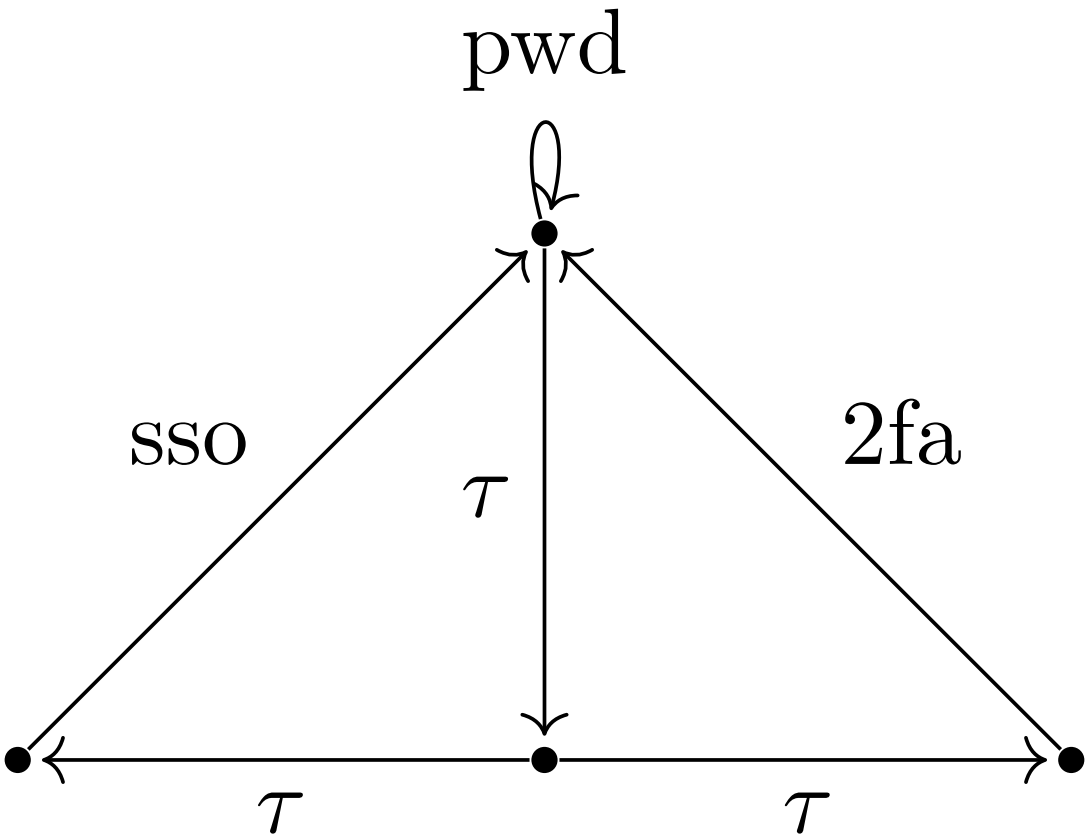
Example 3: Fair-Unfair Miner [SMMvMR22]



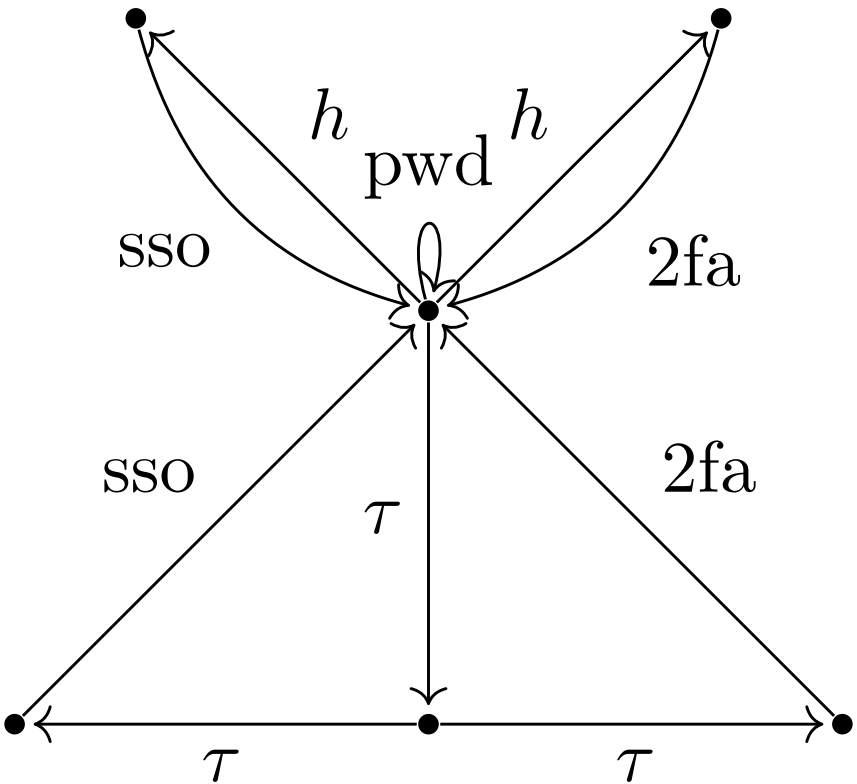
Example 3: Fair-Unfair Miner [SMMvMR22]

```
1  set_default_level low;
2
3  high h;
4
5  // 3 fair miners + h transition
6  P0 = (m1,0.5).P1 + (m2,0.5).P2 + (m3,0.5).P3 + (h, 0.1).P4;
7  P1 = (v1,0.3).P0 + (m1,0.5).P1;
8  P2 = (v2,0.3).P0 + (m2,0.5).P2;
9  P3 = (v3,0.3).P0 + (m3,0.5).P3;
10
11 // 2 fair miners, 1 unfair
12 P4 = (mu,0.5).P5 + (m1,0.5).P6 + (m2,0.5).P7;
13 P5 = (vu,0.3).P4 + (mu,0.5).P5;
14 P6 = (v1,0.3).P4 + (mu,0.5).P5 + (m1,0.5).P6;
15 P7 = (v2,0.3).P4 + (mu,0.5).P5 + (m2,0.5).P7;
16
17 P0
```

Example 4: DBMS Auth [EABR23]



Example 4: DBMS Auth Attack [EABR23]



Example 4: DBMS Auth Attack [EABR23]

```
1  set_default_level low;
2
3  high h;
4
5  WT0 = (l_pwd, 1).WT0
6          + (tau, 1).((tau, 1).(l_sso, 1).WT0
7          + (tau, 1).(l_2fa, 1).WT0)
8          + (h, 1).(l_sso, 1).WT0
9          + (h, 1).(l_2fa, 1).WT0;
10
11 WT0
```

Thank you!