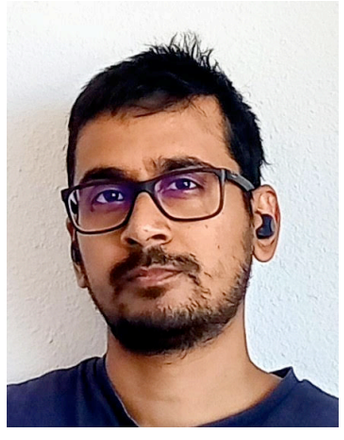
Reachability in extensions of Petri nets

FCT 2025, Wrocław, 2025-09-15

Wojciech Czerwiński

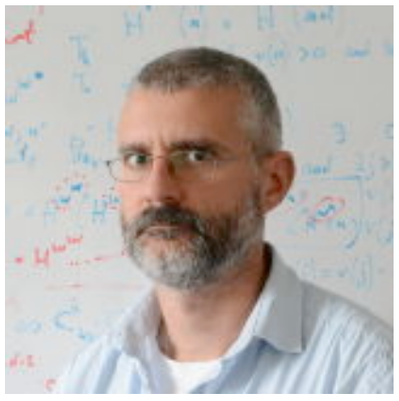


Arka Ghosh

Roland Guttenberg

Ismael Jecker

Łukasz Kamiński



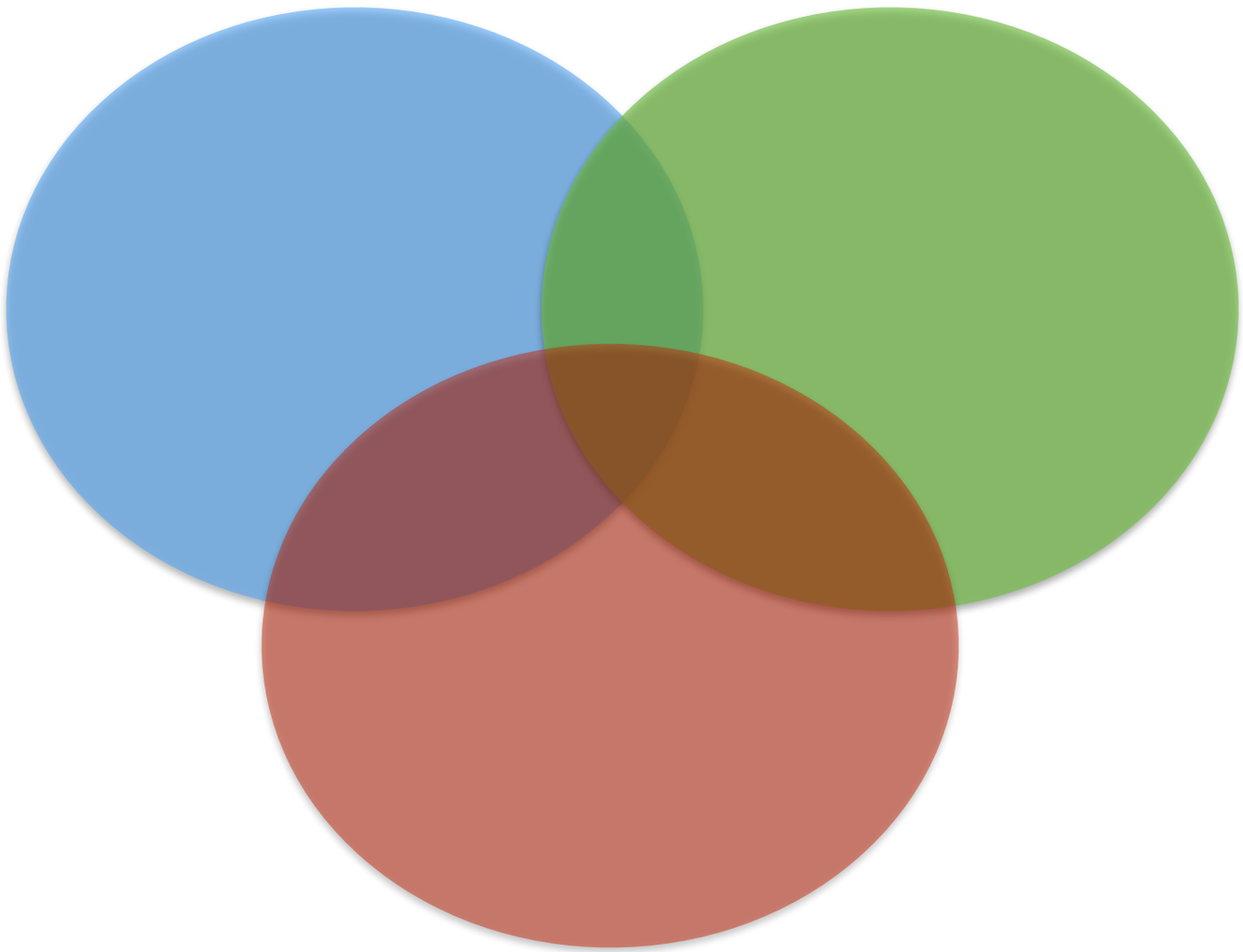
Ranko Lazic

Jerome Leroux

Filip Mazowiecki

Łukasz Orlikowski

Sylvain Schmitz

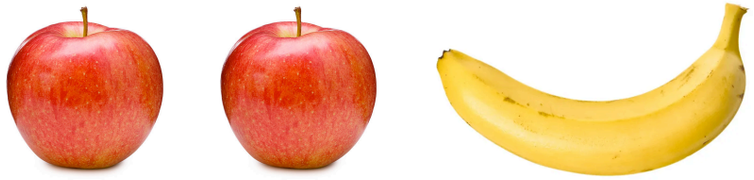


I. Reachability in Petri nets

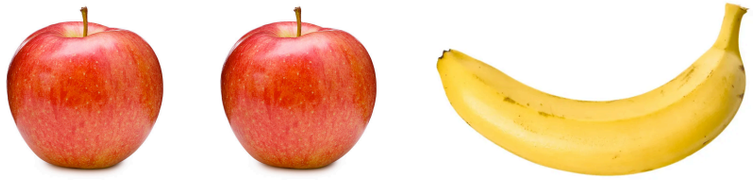
II. Extensions of Petri nets

Kindergarten fruit-trading

Kindergarten fruit-trading

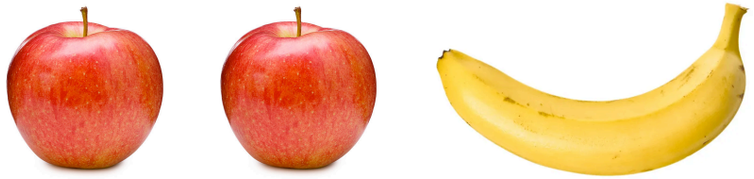
Alice has  but likes 

Kindergarten fruit-trading

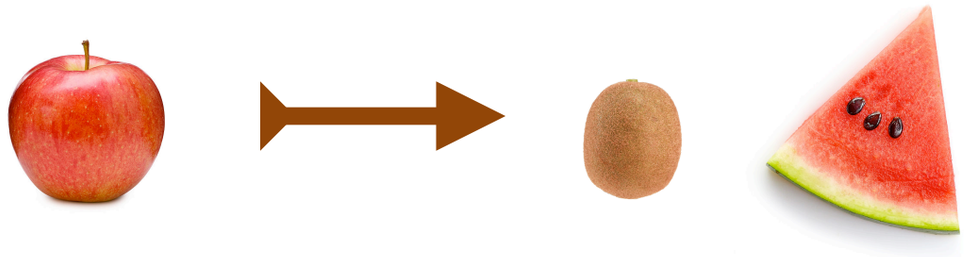
Alice has  but likes 

Kindergarten fruit-trading rules:

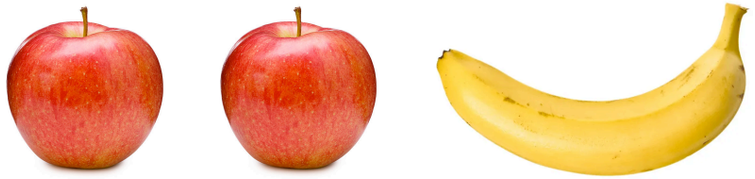
Kindergarten fruit-trading

Alice has  but likes 

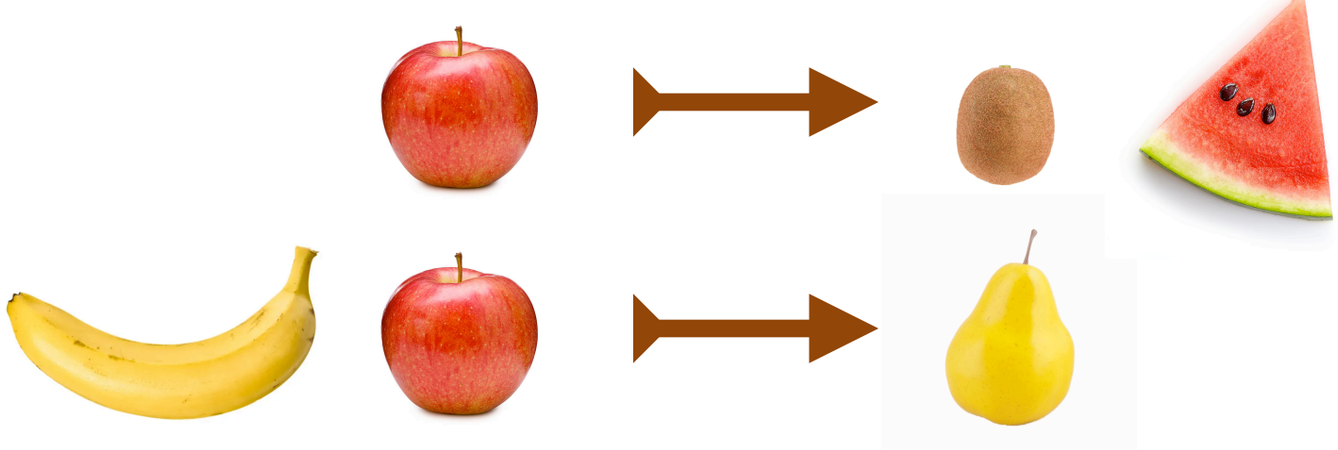
Kindergarten fruit-trading rules:



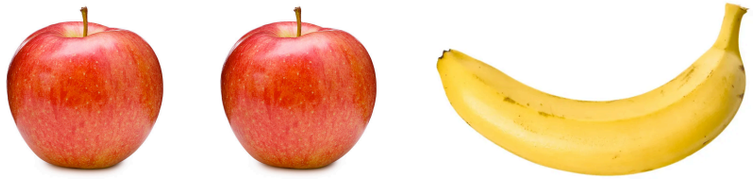
Kindergarten fruit-trading

Alice has  but likes 

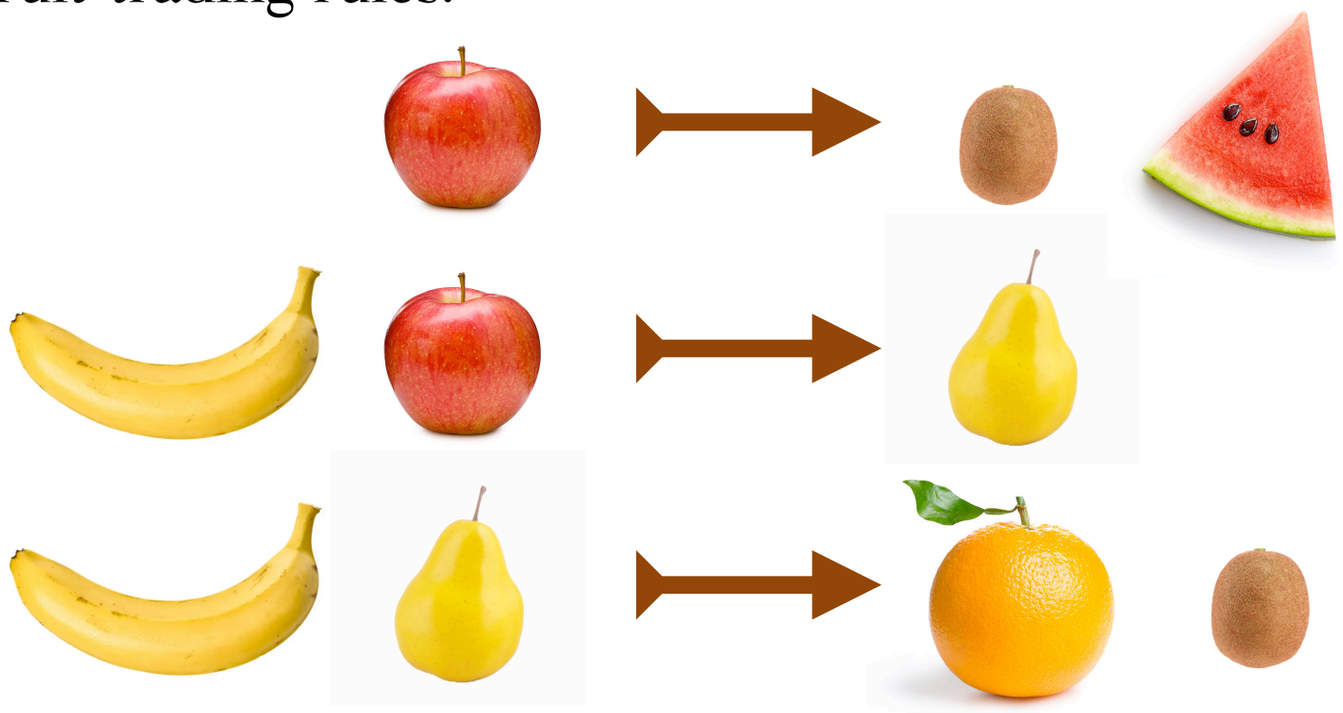
Kindergarten fruit-trading rules:



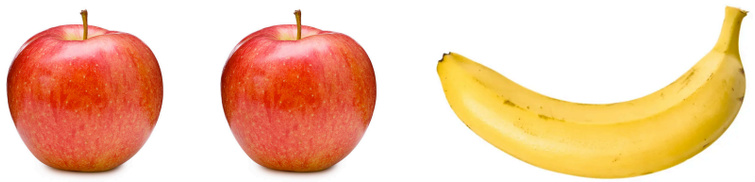
Kindergarten fruit-trading

Alice has  but likes 

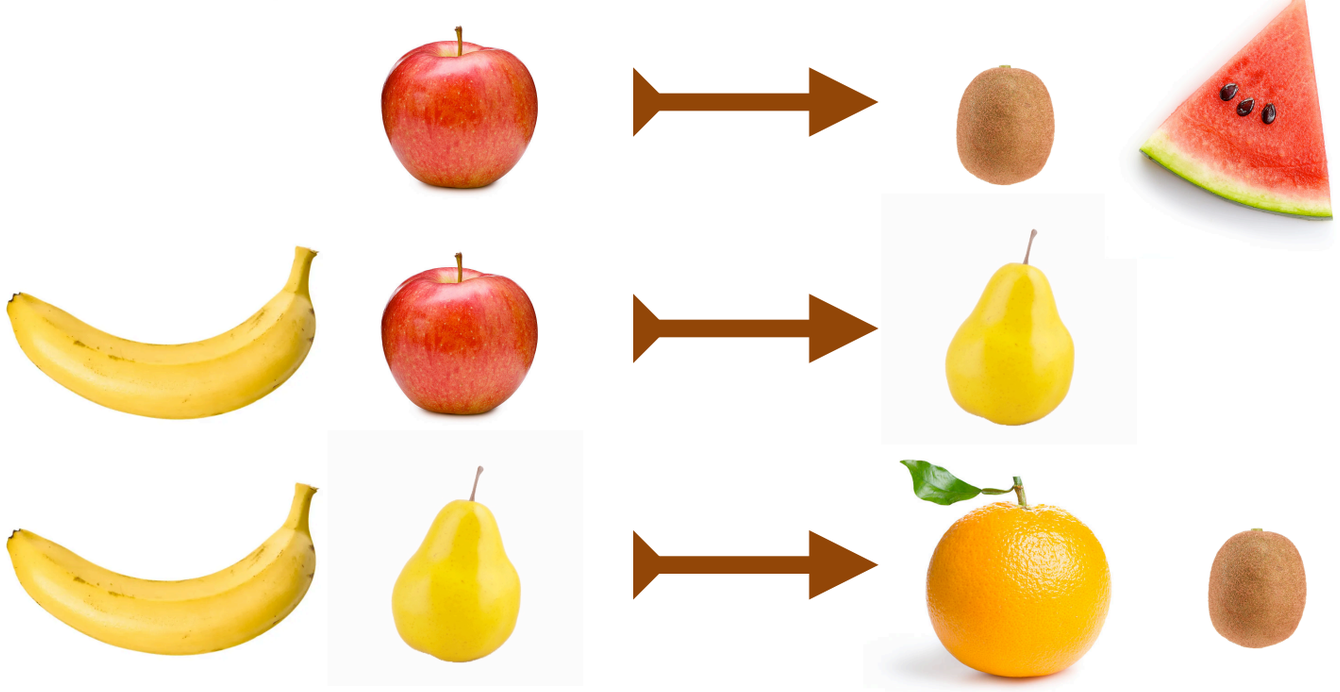
Kindergarten fruit-trading rules:



Kindergarten fruit-trading

Alice has  but likes 

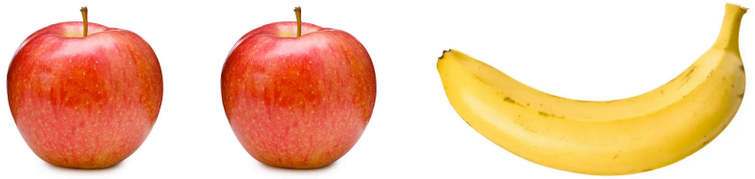
Kindergarten fruit-trading rules:



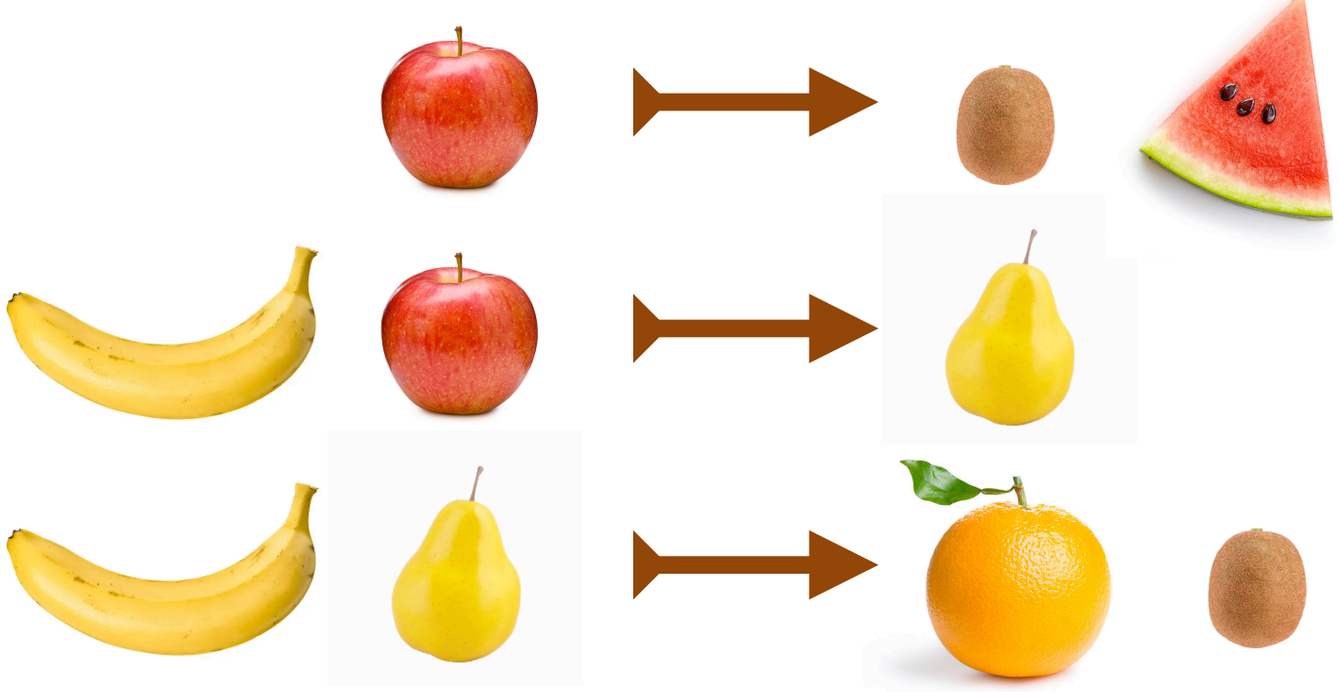
Decision problem:

 *  ?

Kindergarten fruit-trading

Alice has  but likes 

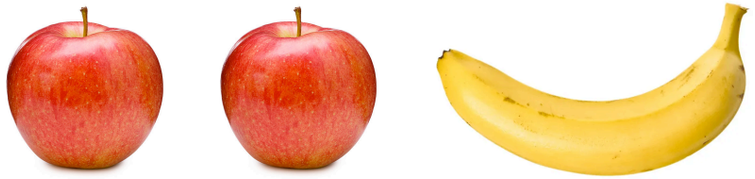
Kindergarten fruit-trading rules:



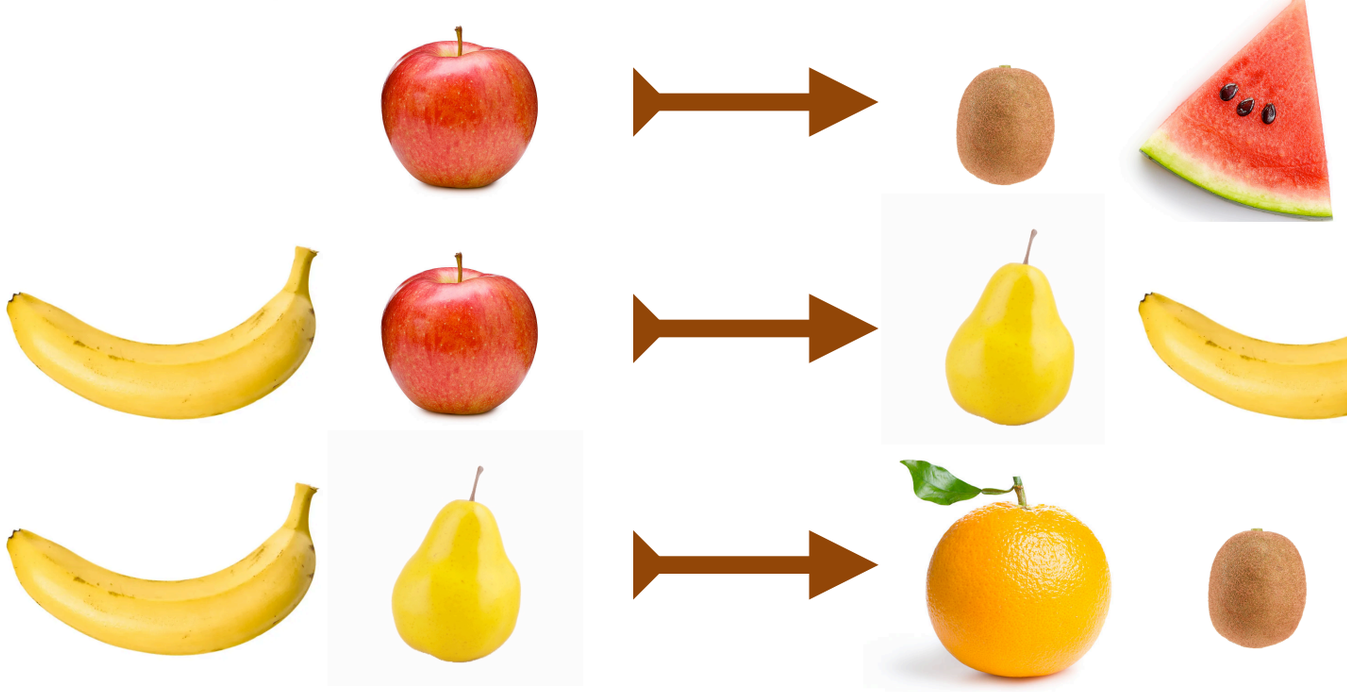
Decision problem:

 *  ?

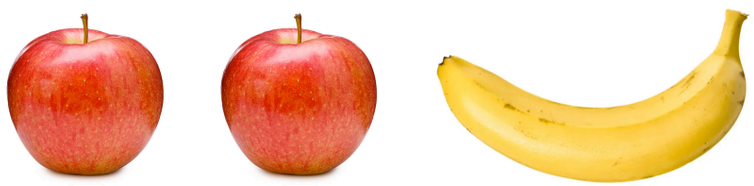
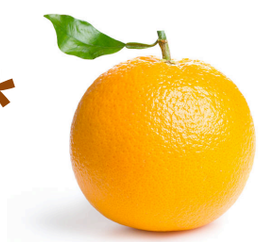
Kindergarten fruit-trading

Alice has  but likes 

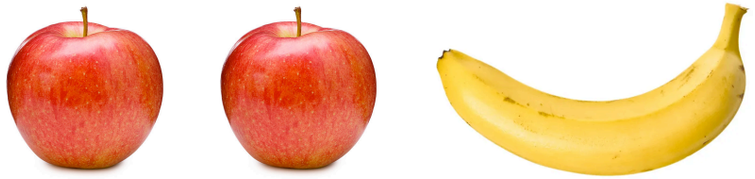
Kindergarten fruit-trading rules:



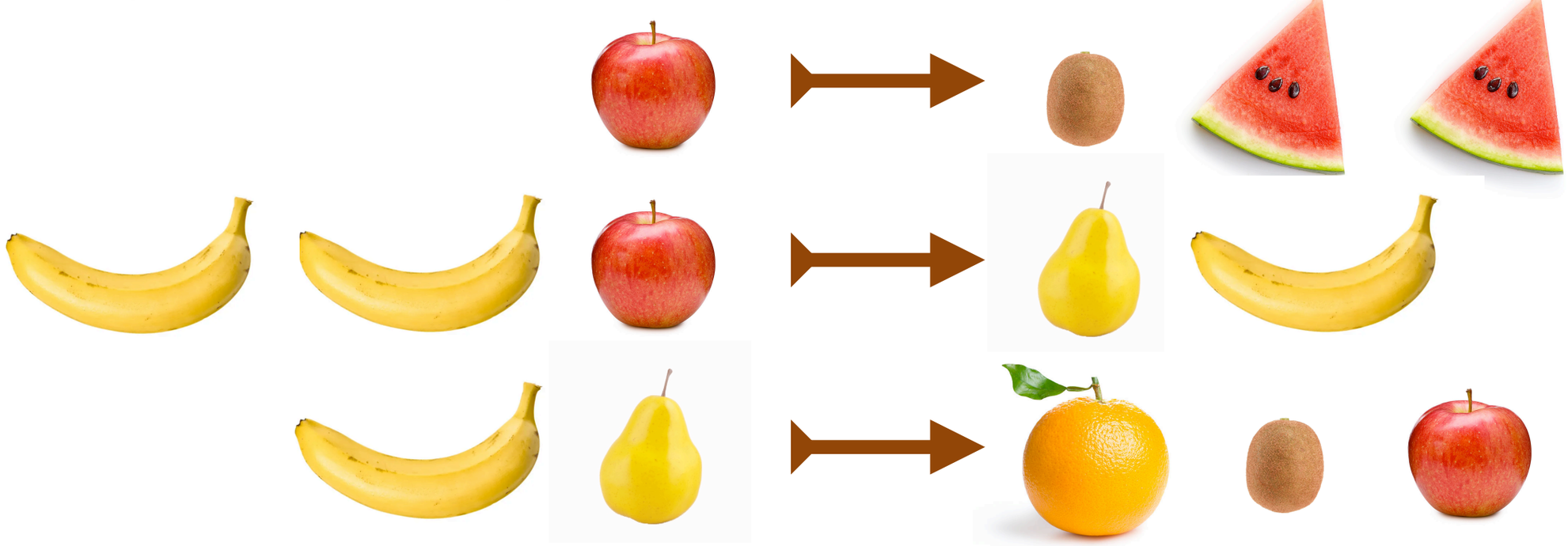
Decision problem:

 *   ?

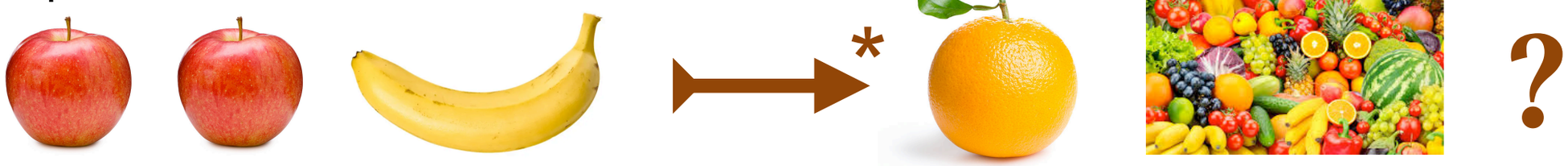
Kindergarten fruit-trading

Alice has  but likes 

Kindergarten fruit-trading rules:

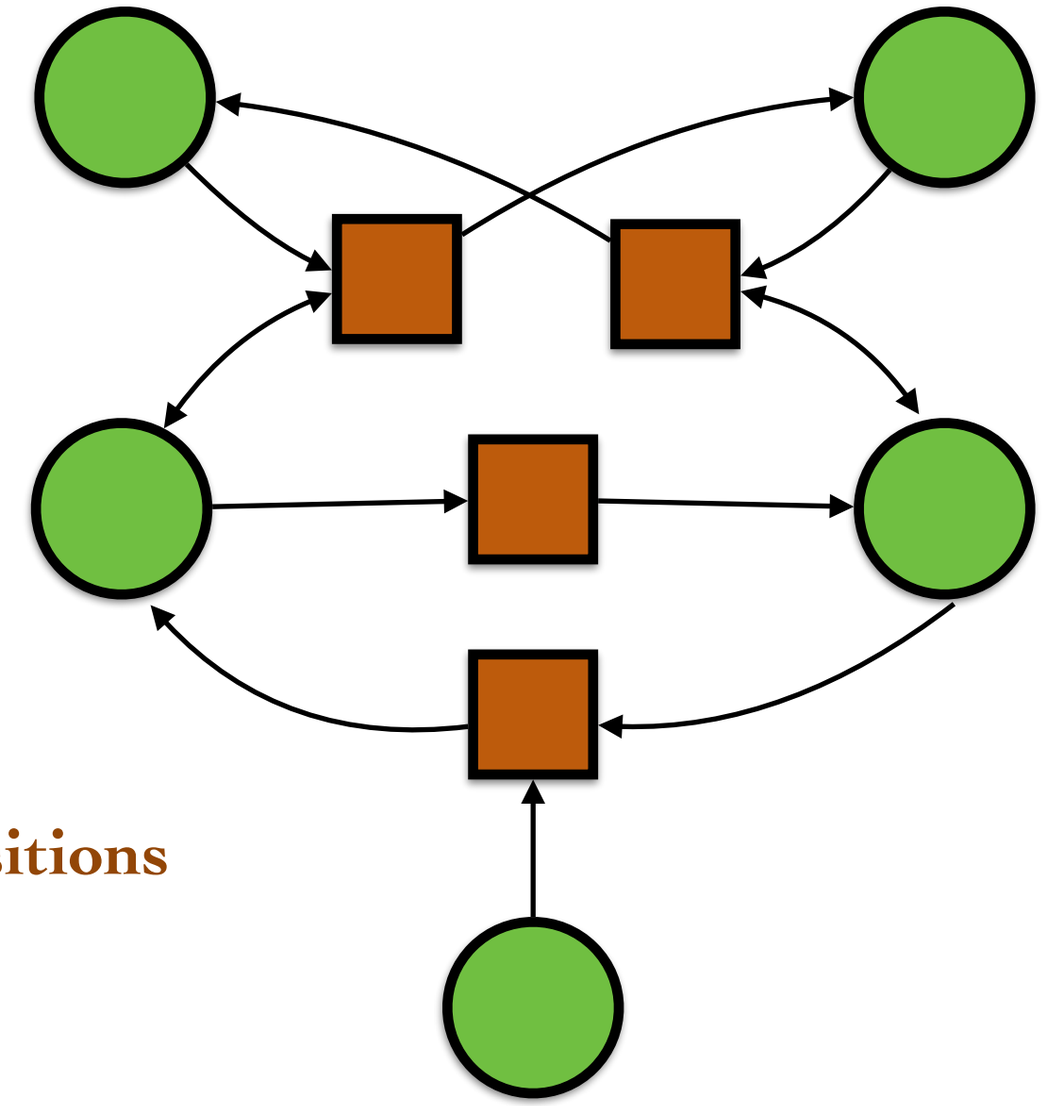


Decision problem:



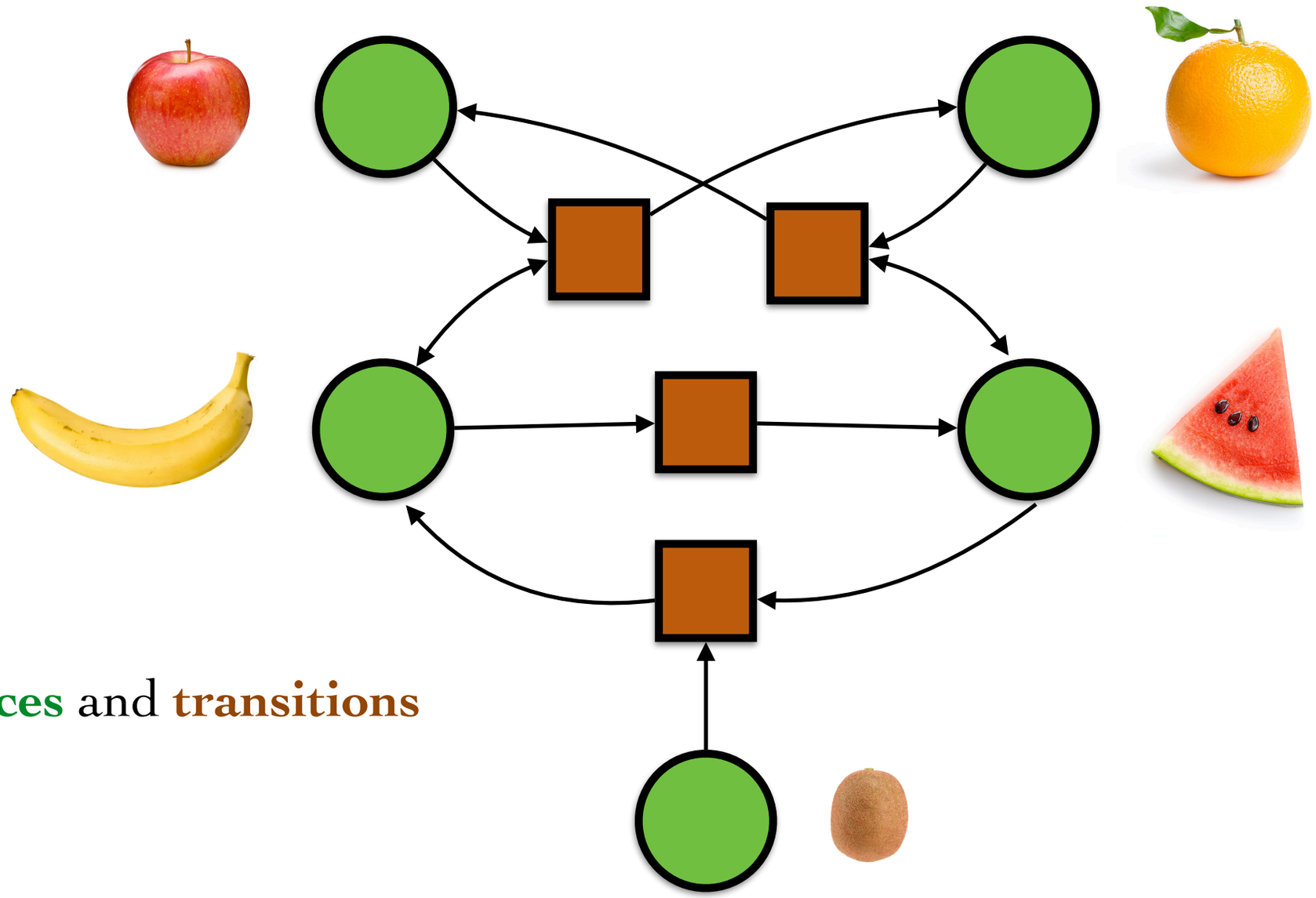
?

Petri nets



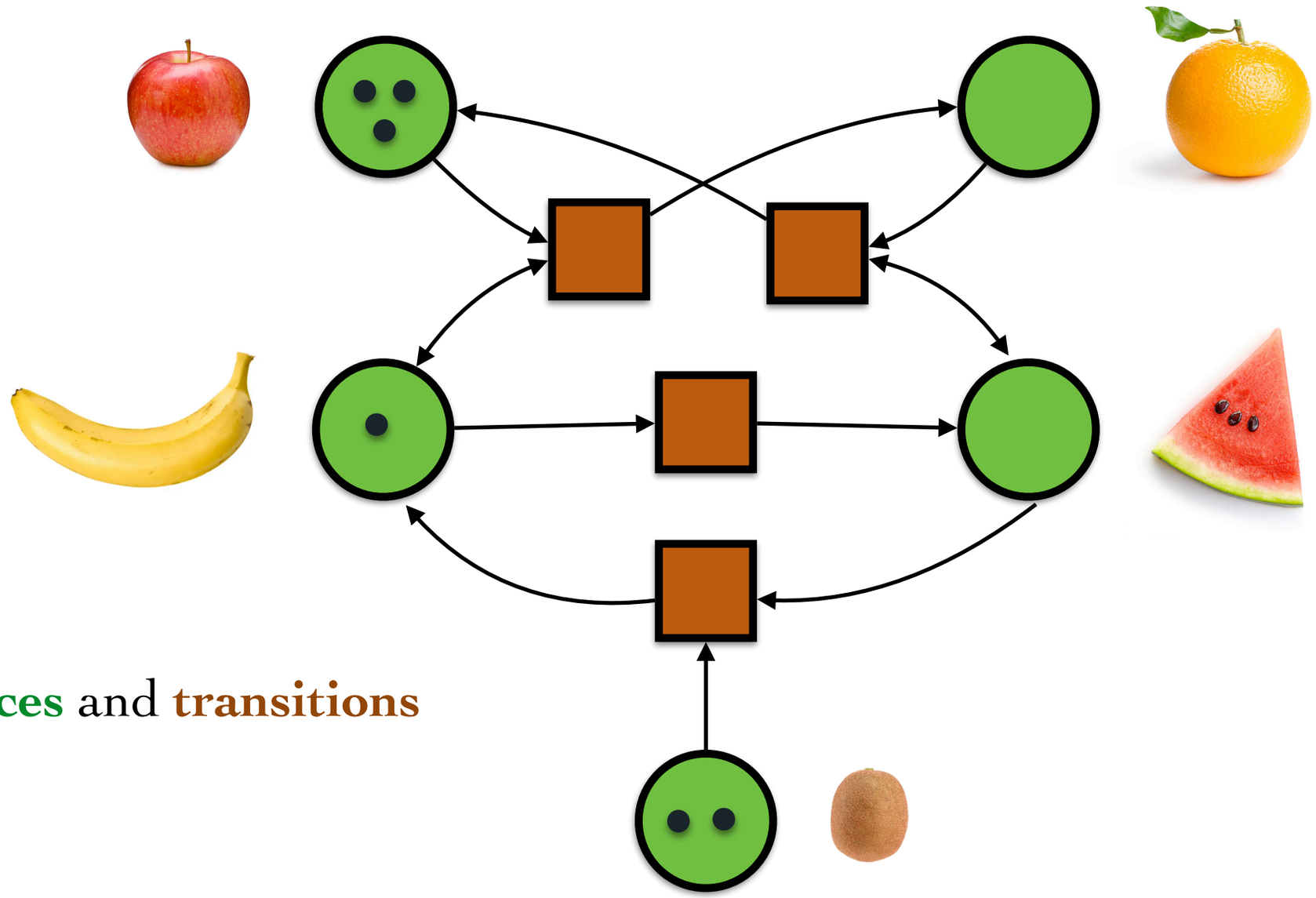
places and **transitions**

Petri nets



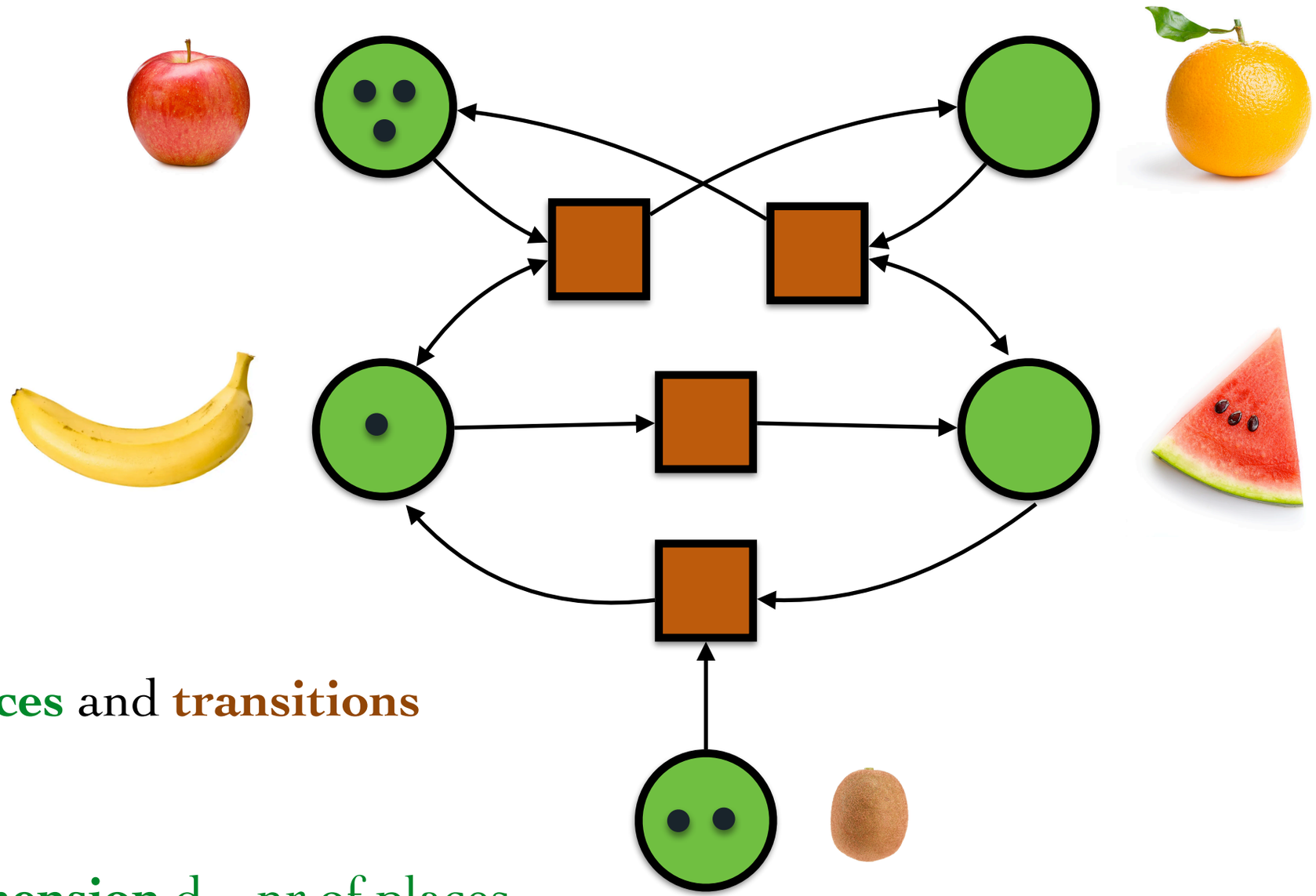
places and **transitions**

Petri nets



places and **transitions**

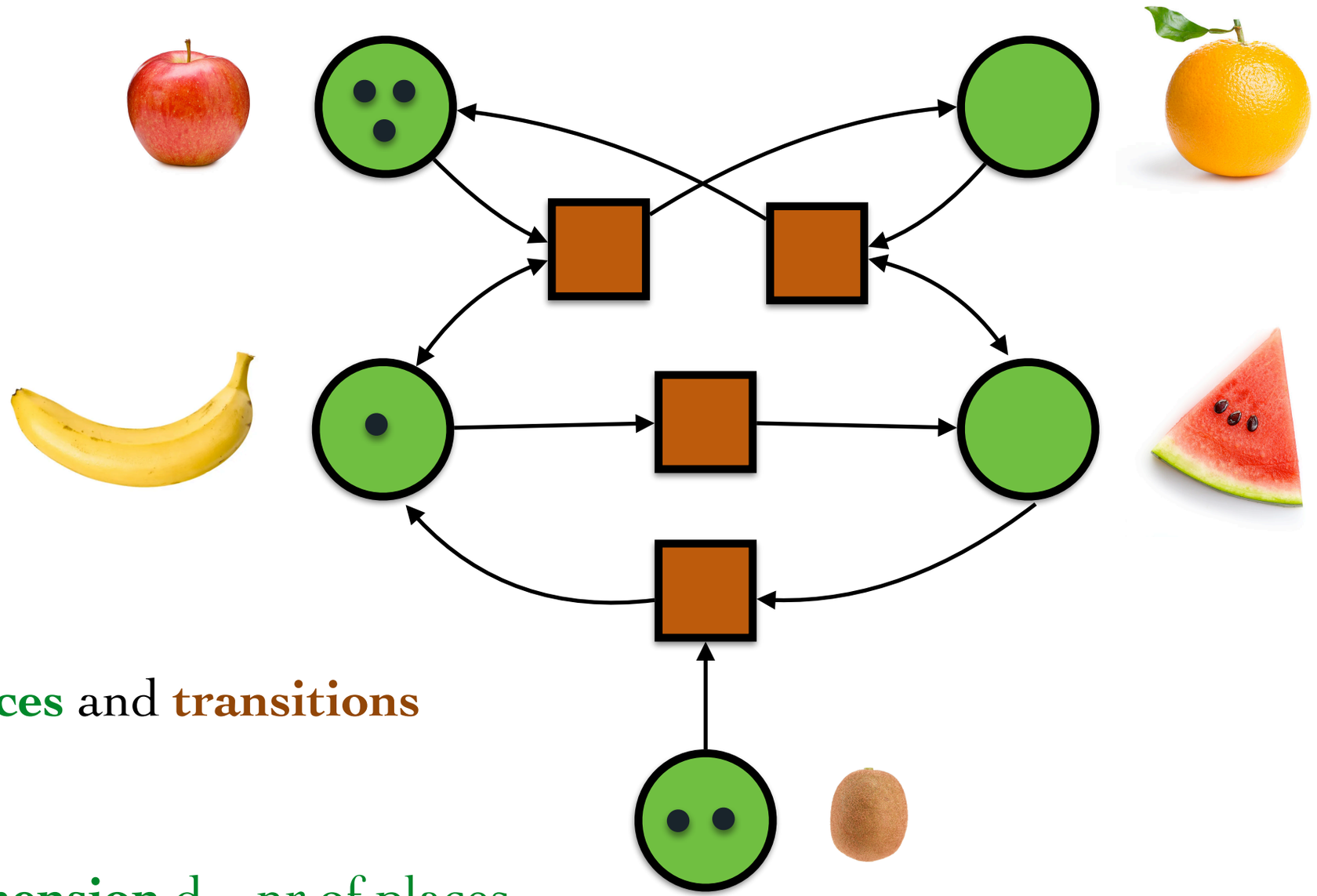
Petri nets



places and **transitions**

dimension d = nr of places

Petri nets

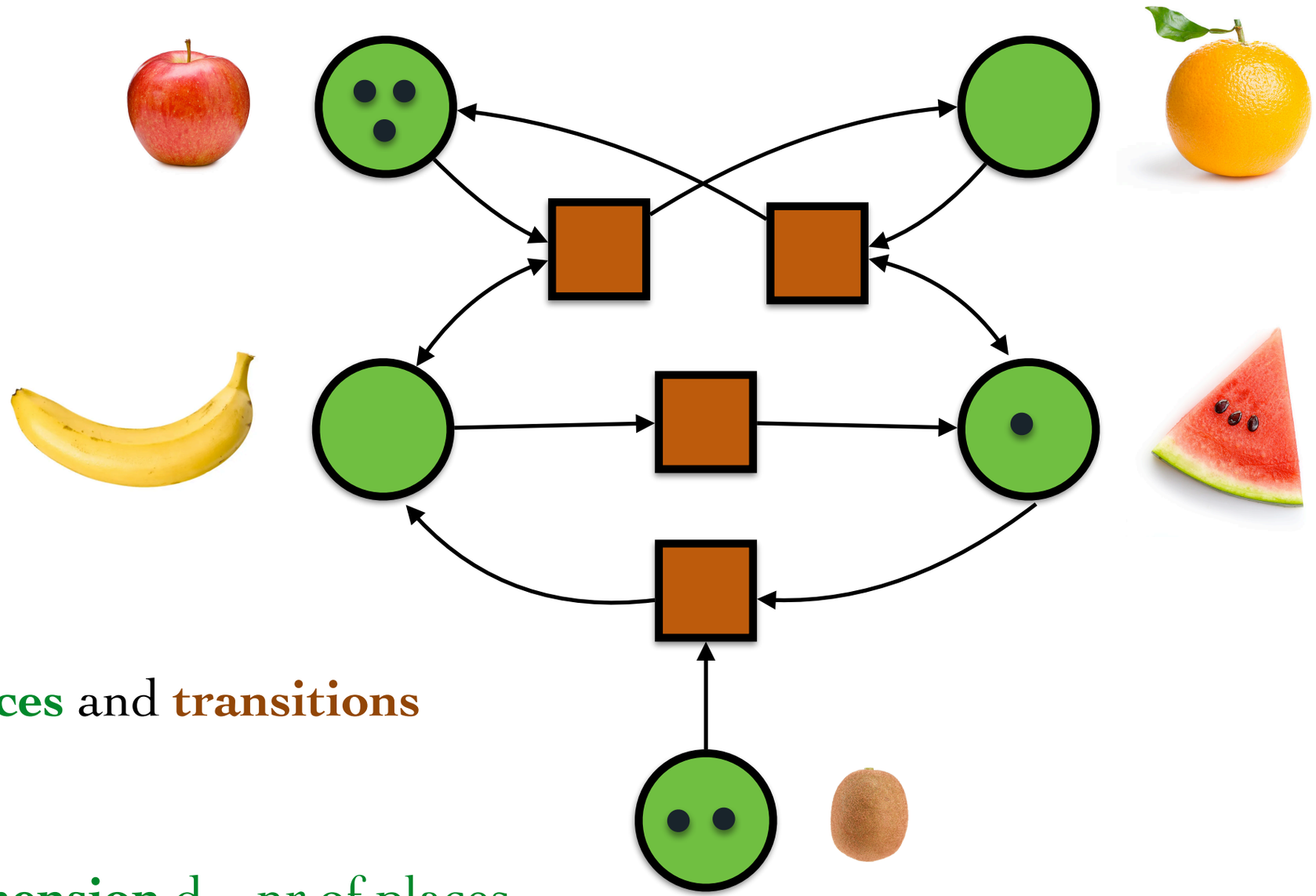


places and **transitions**

dimension d = nr of places

configuration : places $\rightarrow \mathbb{N} \equiv \mathbb{N}^d$

Petri nets



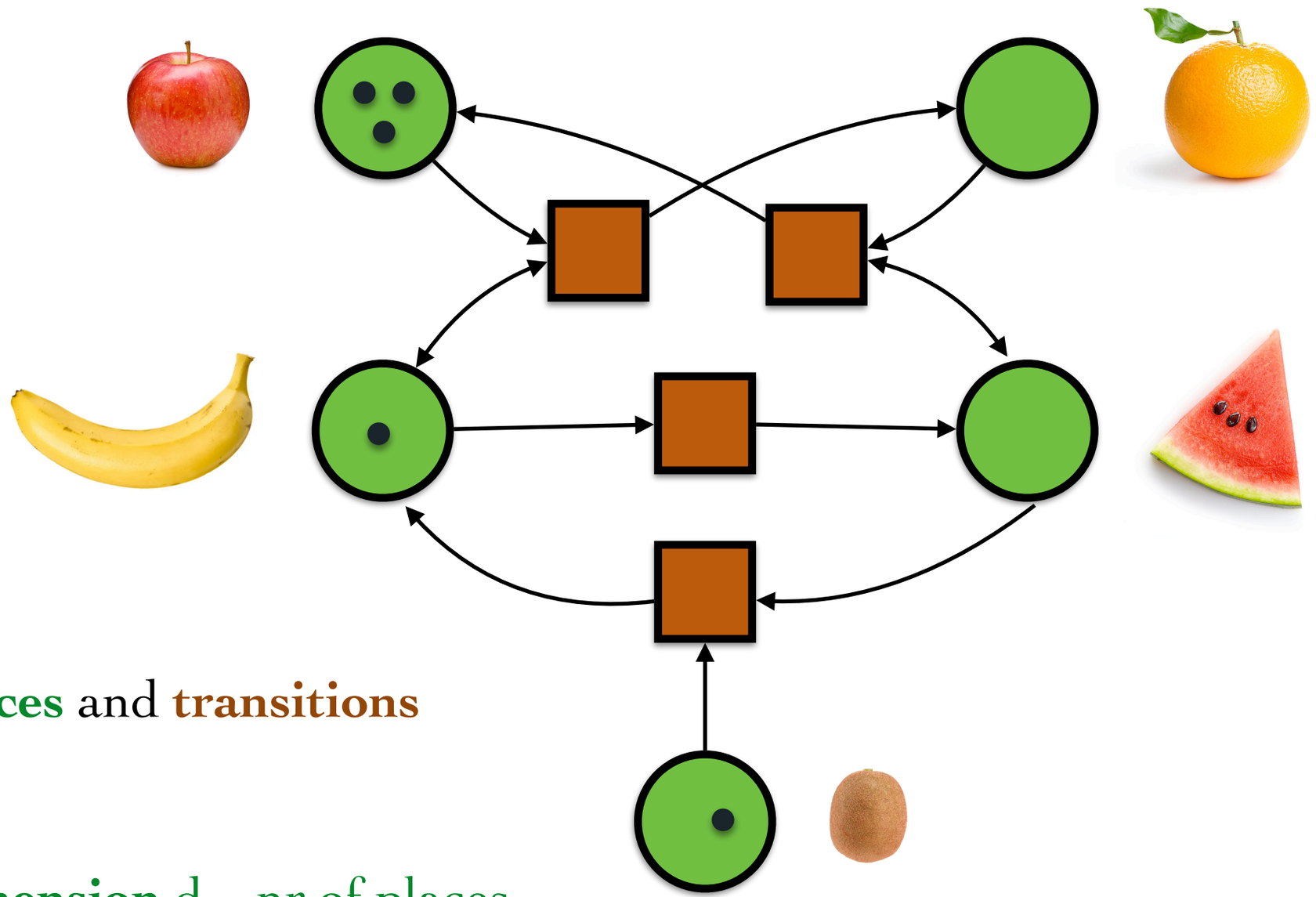
places and **transitions**

dimension d = nr of places

configuration : places $\rightarrow \mathbb{N} \equiv \mathbb{N}^d$

step relation between configurations

Petri nets



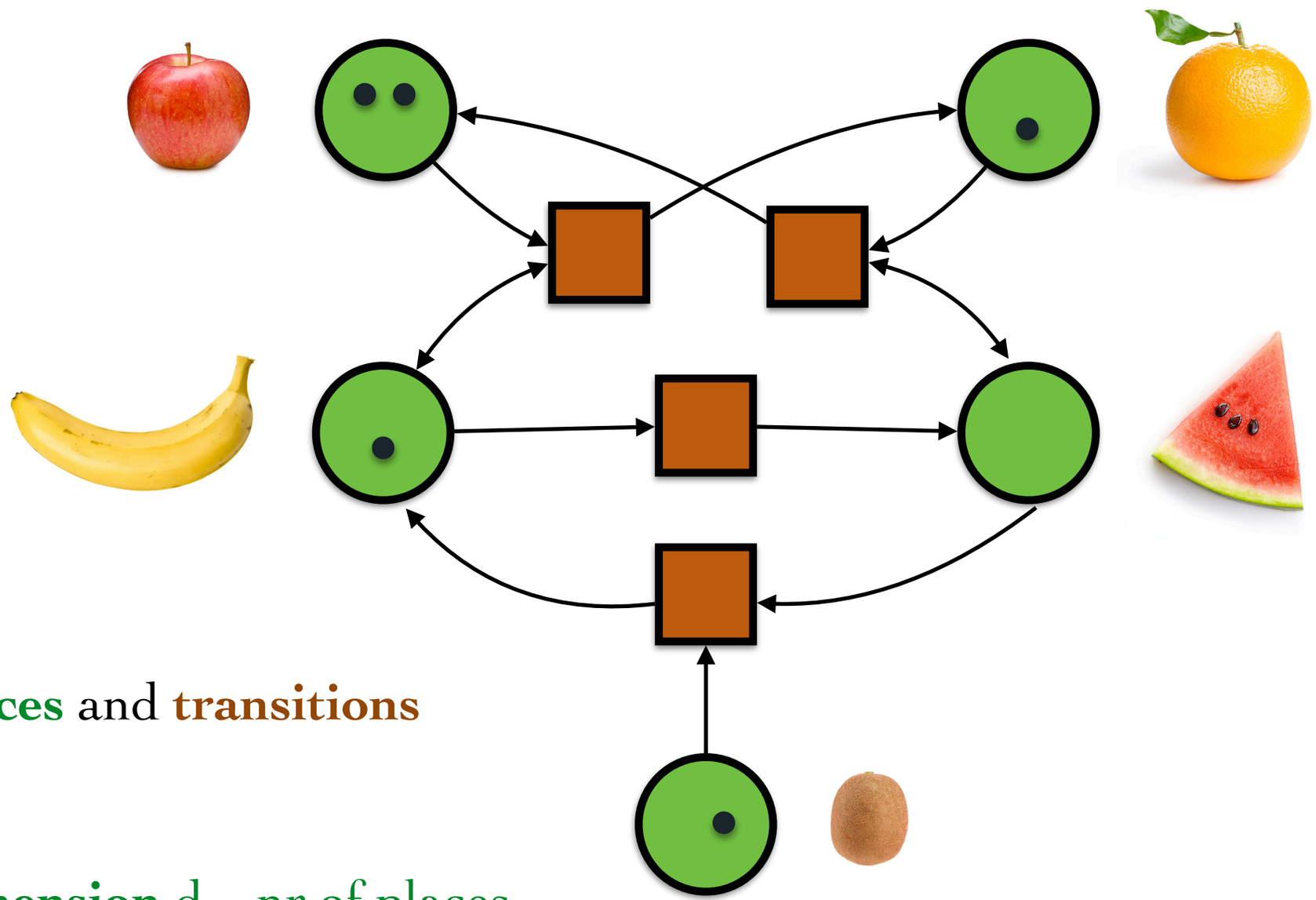
places and **transitions**

dimension d = nr of places

configuration : places $\rightarrow \mathbb{N} \equiv \mathbb{N}^d$

step relation between configurations

Petri nets



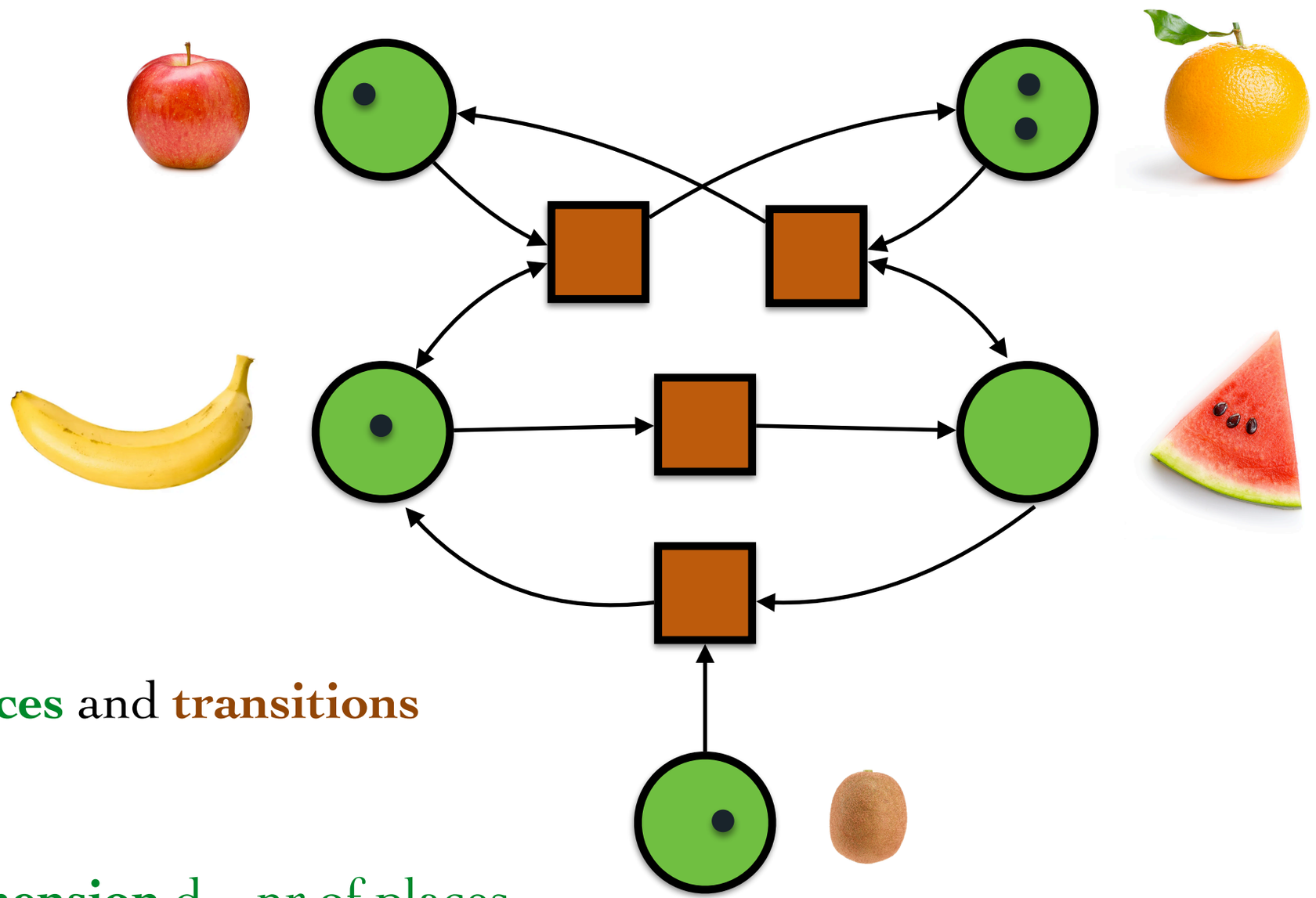
places and **transitions**

dimension d = nr of places

configuration : places $\rightarrow \mathbb{N} \equiv \mathbb{N}^d$

step relation between configurations

Petri nets



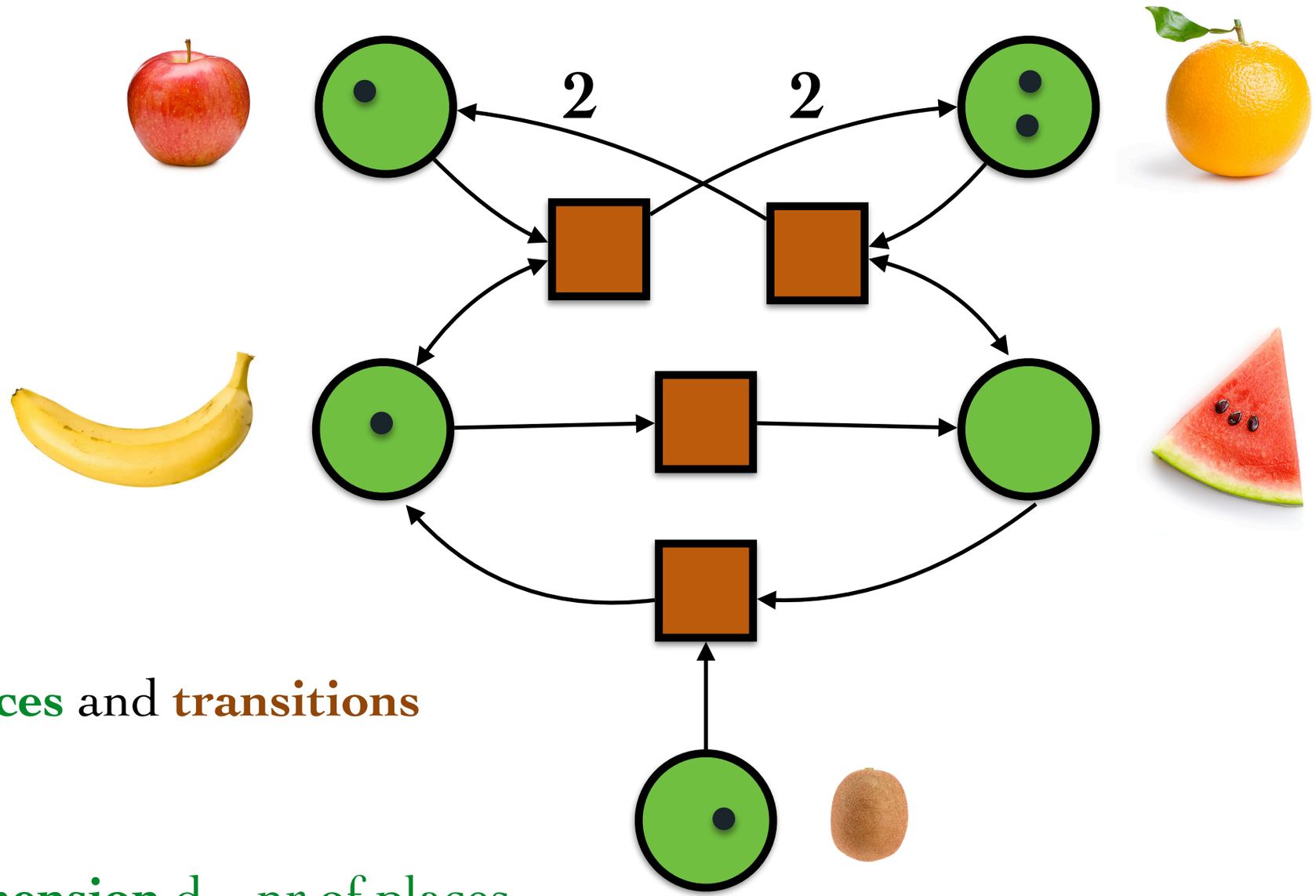
places and **transitions**

dimension d = nr of places

configuration : places $\rightarrow \mathbb{N} \equiv \mathbb{N}^d$

step relation between configurations

Petri nets



arcs weighted by $\mathbb{N}$

places and **transitions**

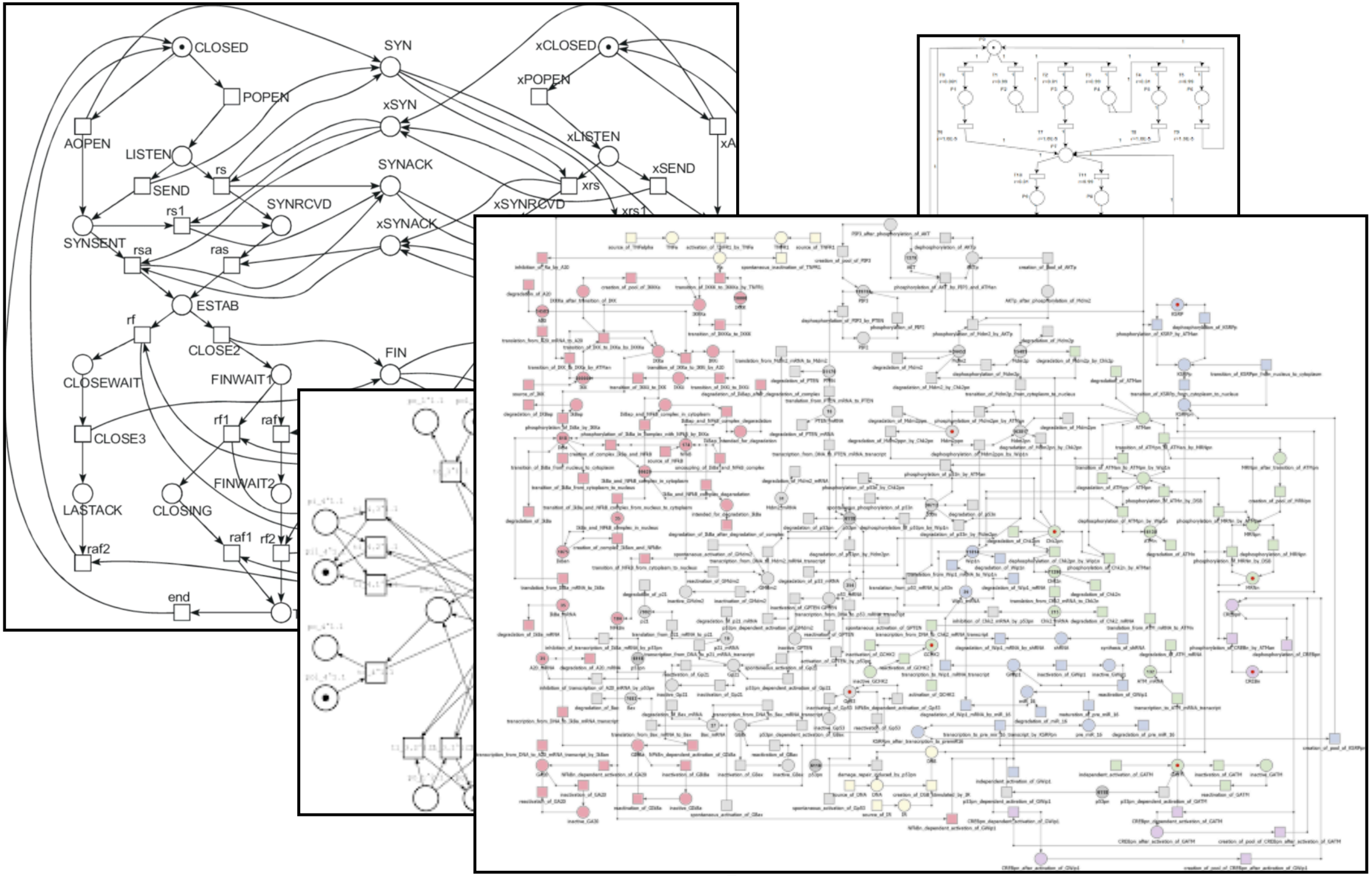
dimension d = nr of places

configuration : places $\rightarrow \mathbb{N} \equiv \mathbb{N}^d$

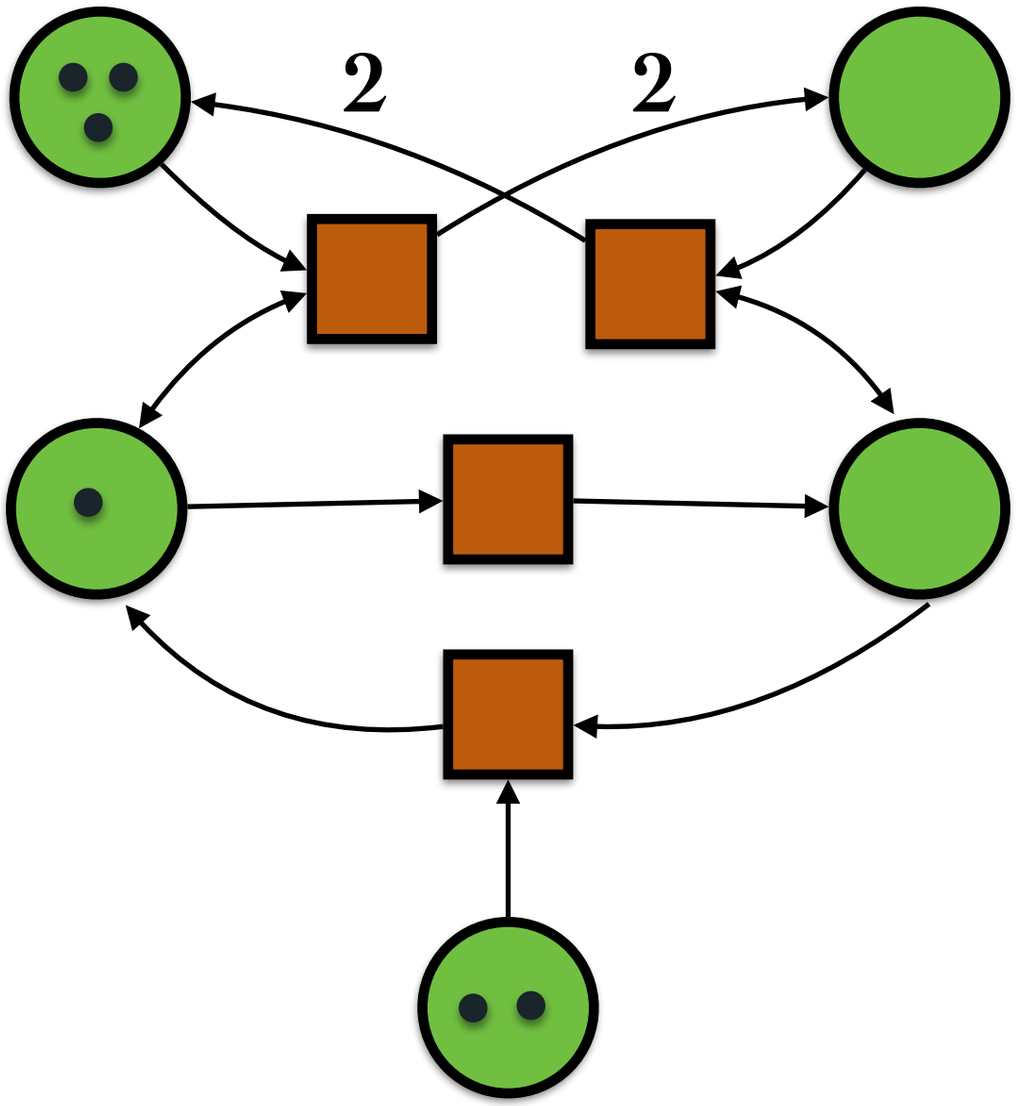
step relation between configurations

Useful model

Useful model

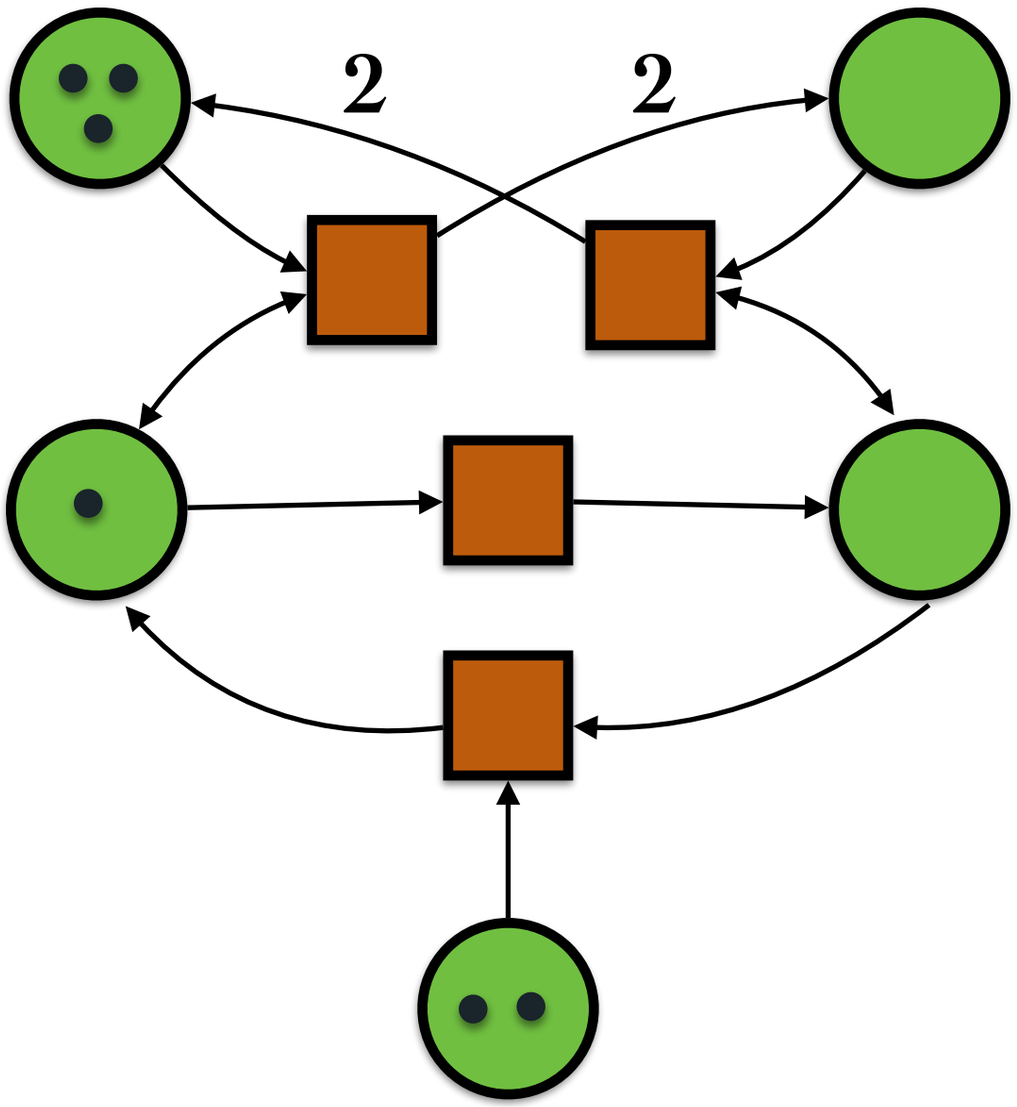


Reachability problem in Petri nets



configuration : $\mathbb{N}^d$

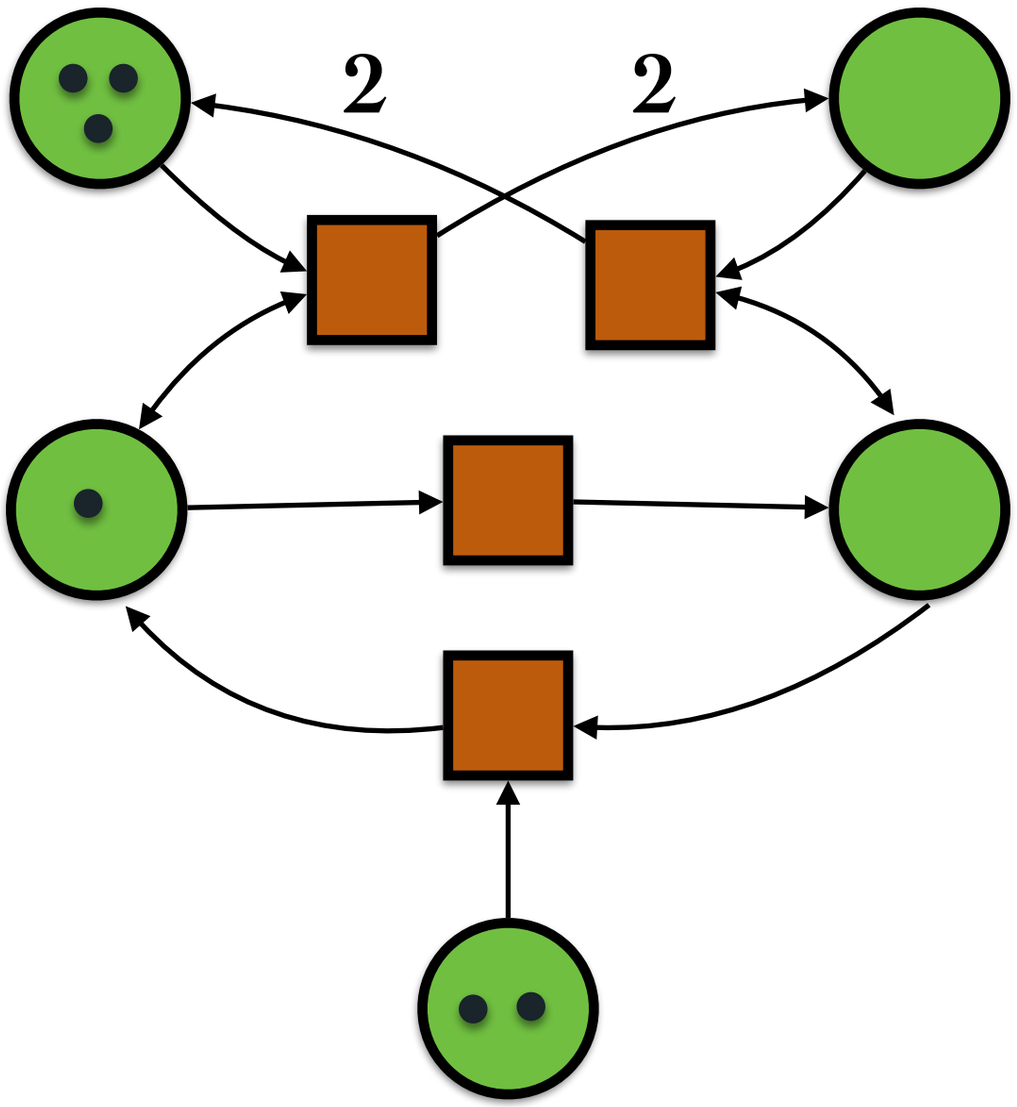
Reachability problem in Petri nets



Decision problem:

configuration : $\mathbb{N}^d$

Reachability problem in Petri nets



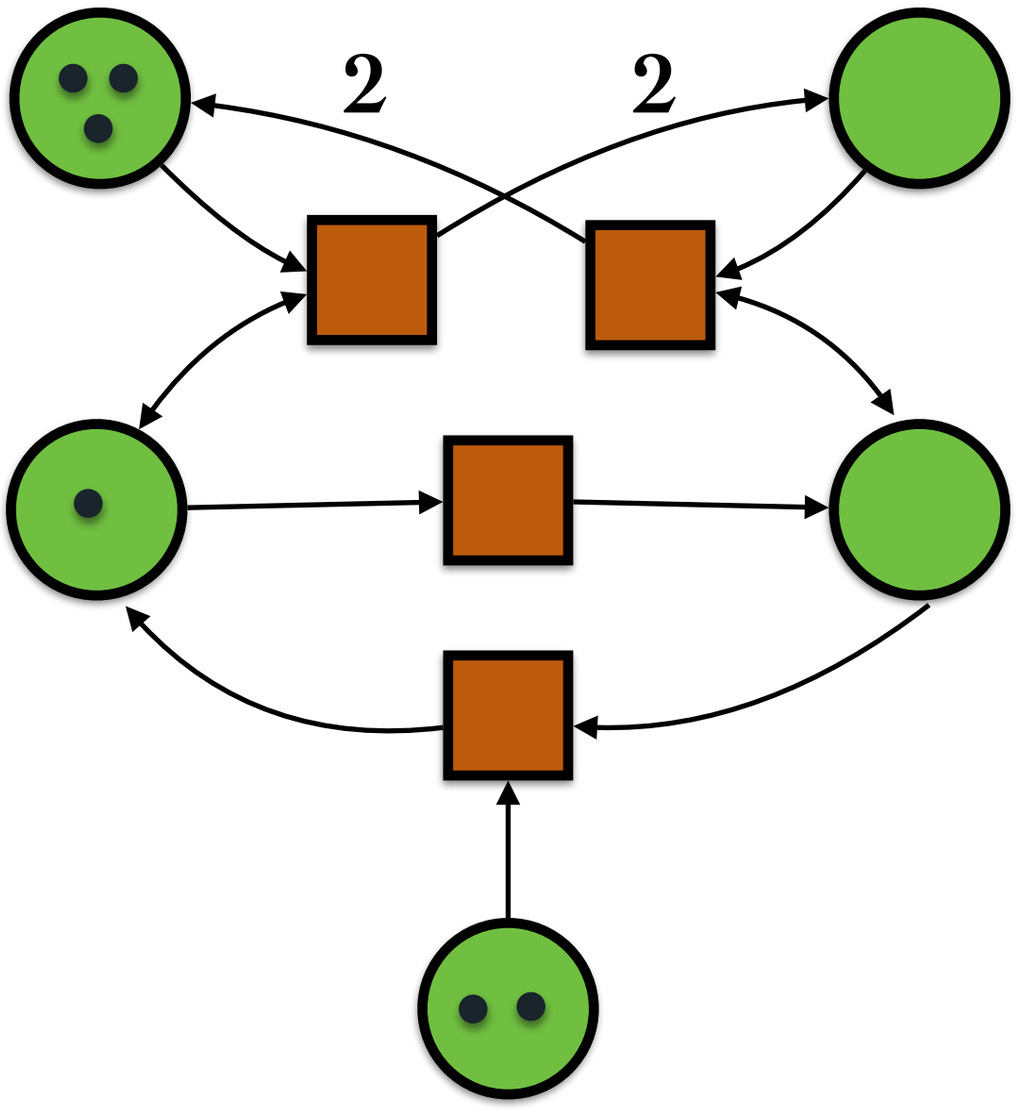
Decision problem:

given

- Petri net
- source configuration
- target configuration

configuration : $\mathbb{N}^d$

Reachability problem in Petri nets



configuration : $\mathbb{N}^d$

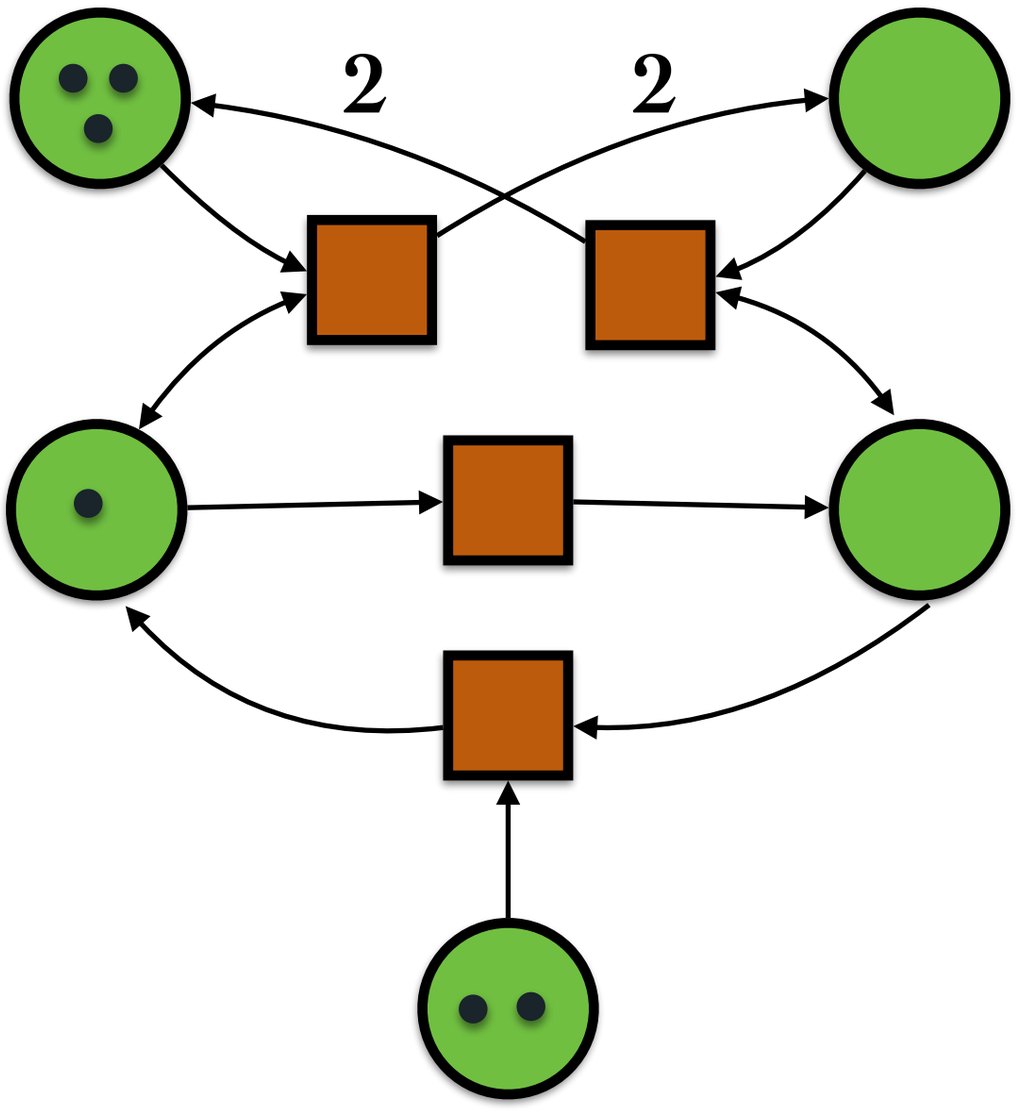
Decision problem:

given

- Petri net
- source configuration
- target configuration

check if there is a sequence of steps (**run**) from source to target

~~Reachability~~ Coverability problem in Petri nets



configuration : $\mathbb{N}^d$

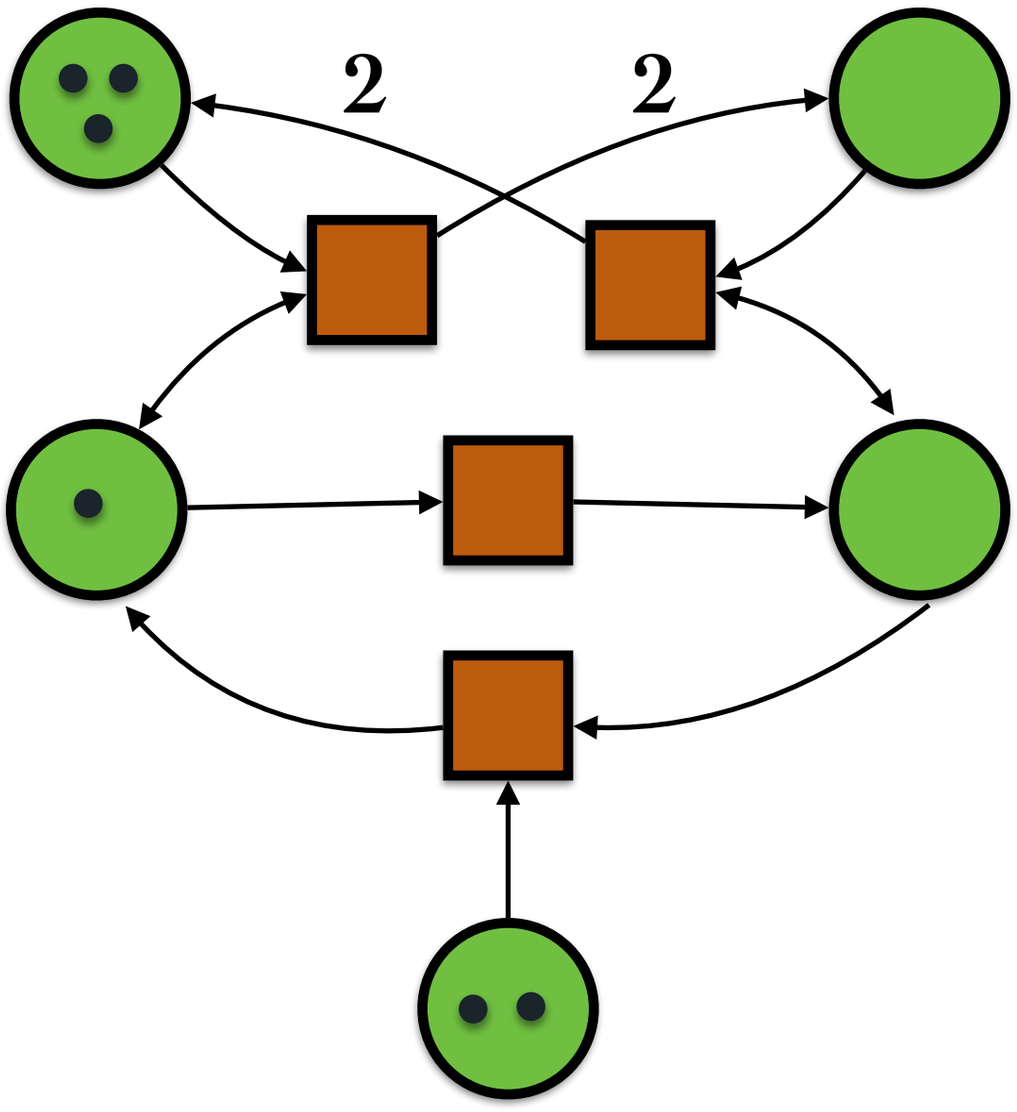
Decision problem:

given

- Petri net
- source configuration
- target configuration

check if there is a sequence of steps (**run**) from source to target

~~Reachability~~ Coverability problem in Petri nets



configuration : $\mathbb{N}^d$

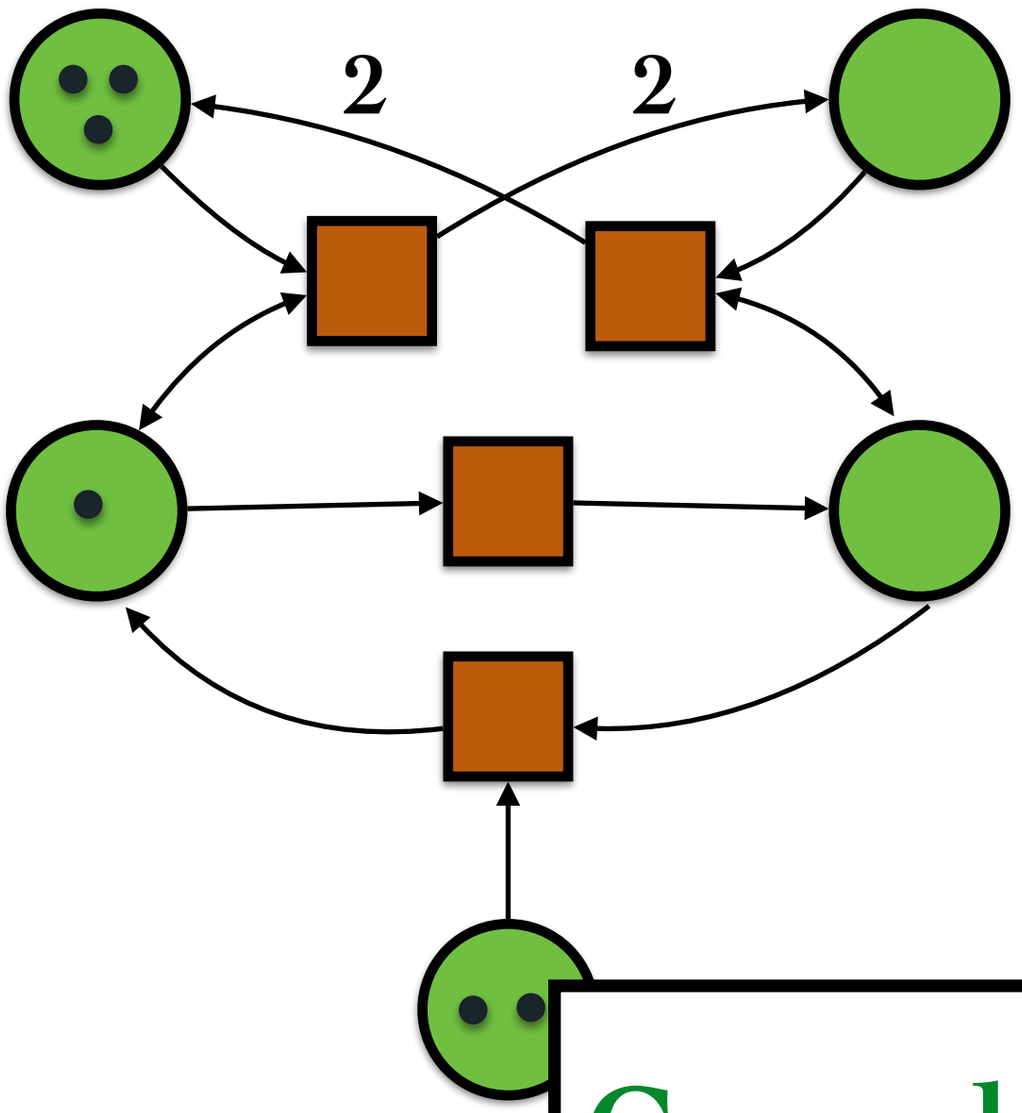
Decision problem:

given

- Petri net
- source configuration
- target configuration

check if there is a sequence of steps (**run**) from source to ~~target~~ **≥ target**

~~Reachability~~ Coverability problem in Petri nets



Decision problem:

given

- Petri net
- source configuration
- target configuration

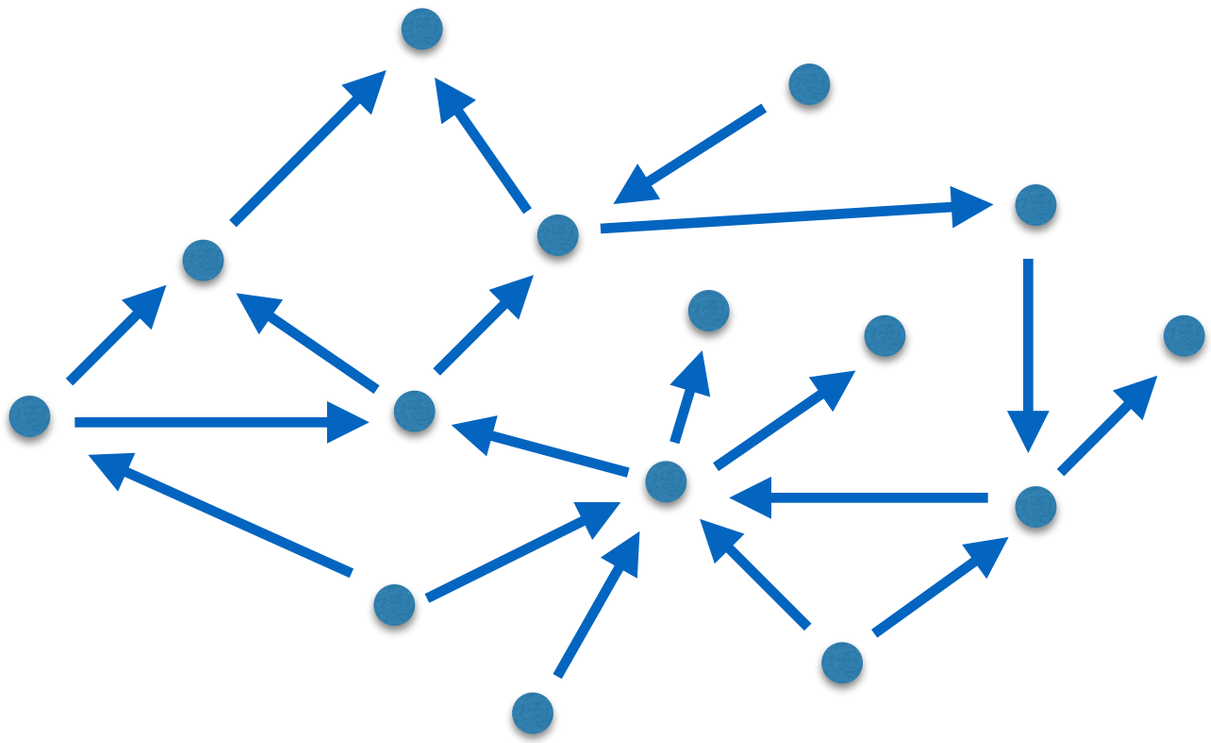
check if there is a sequence of steps (**run**) from source to ~~target~~ **≥ target**

configuration : $\mathbb{N}^d$

Coverability $\rightsquigarrow$ **Reachability**

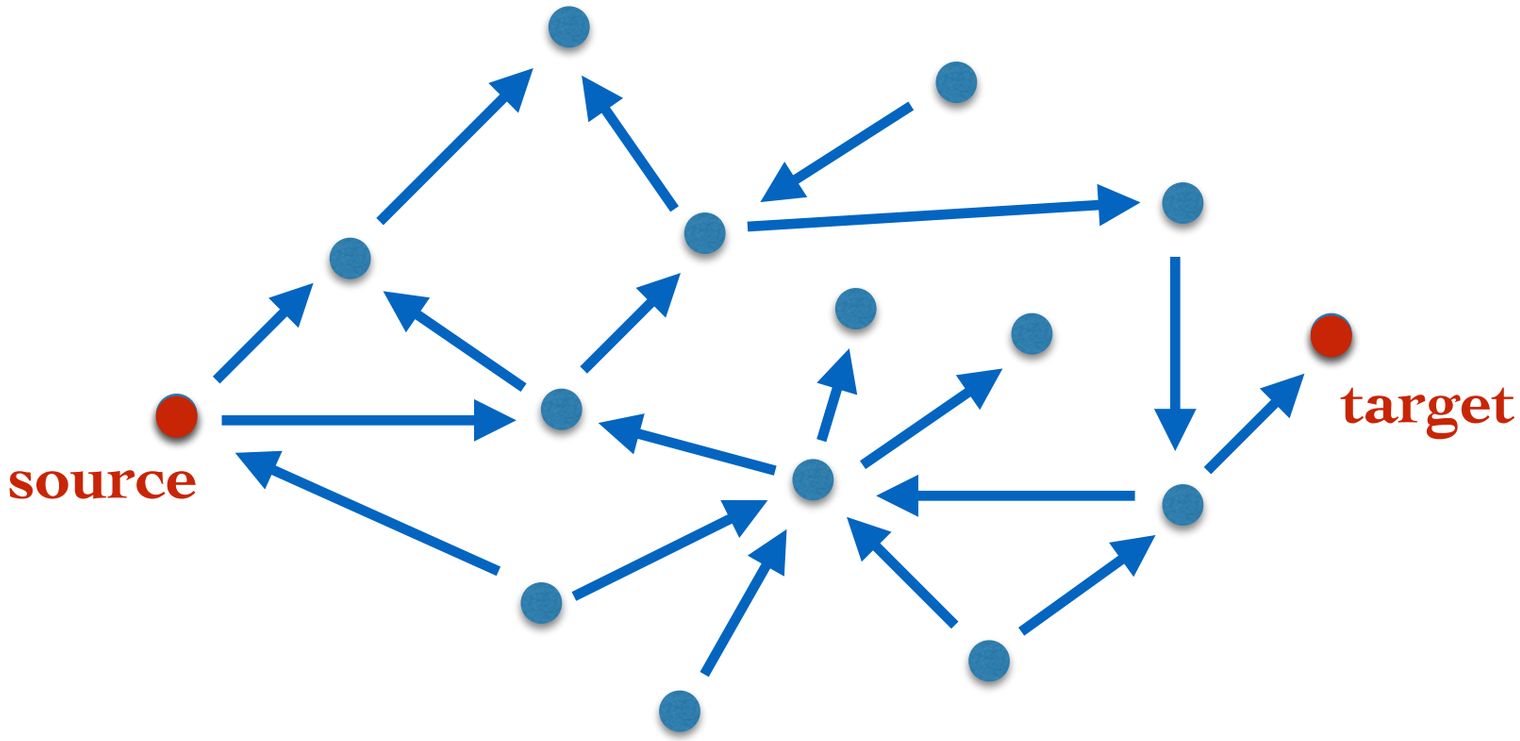
Reachability problem in Petri nets

configuration graph: configurations and steps



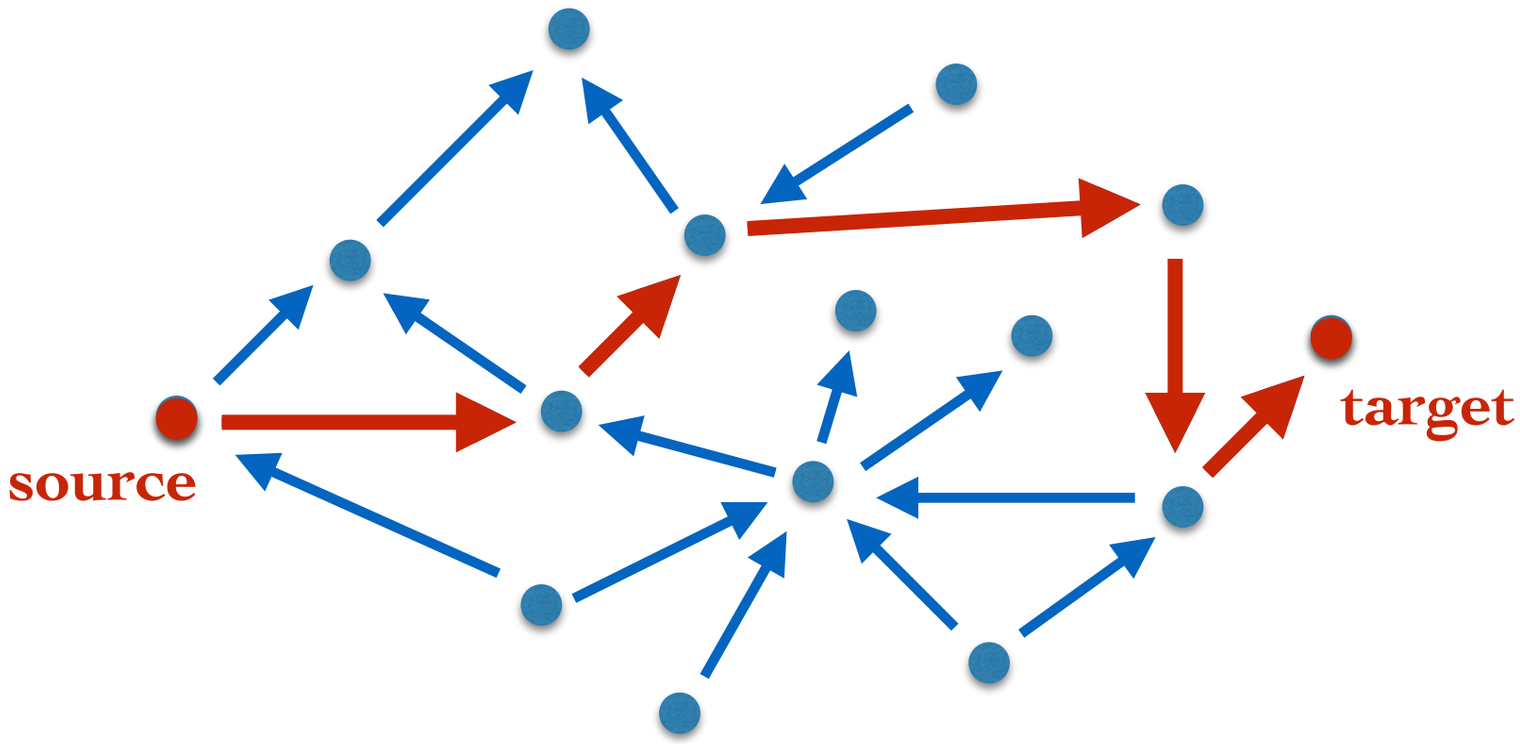
Reachability problem in Petri nets

configuration graph: configurations and steps



Reachability problem in Petri nets

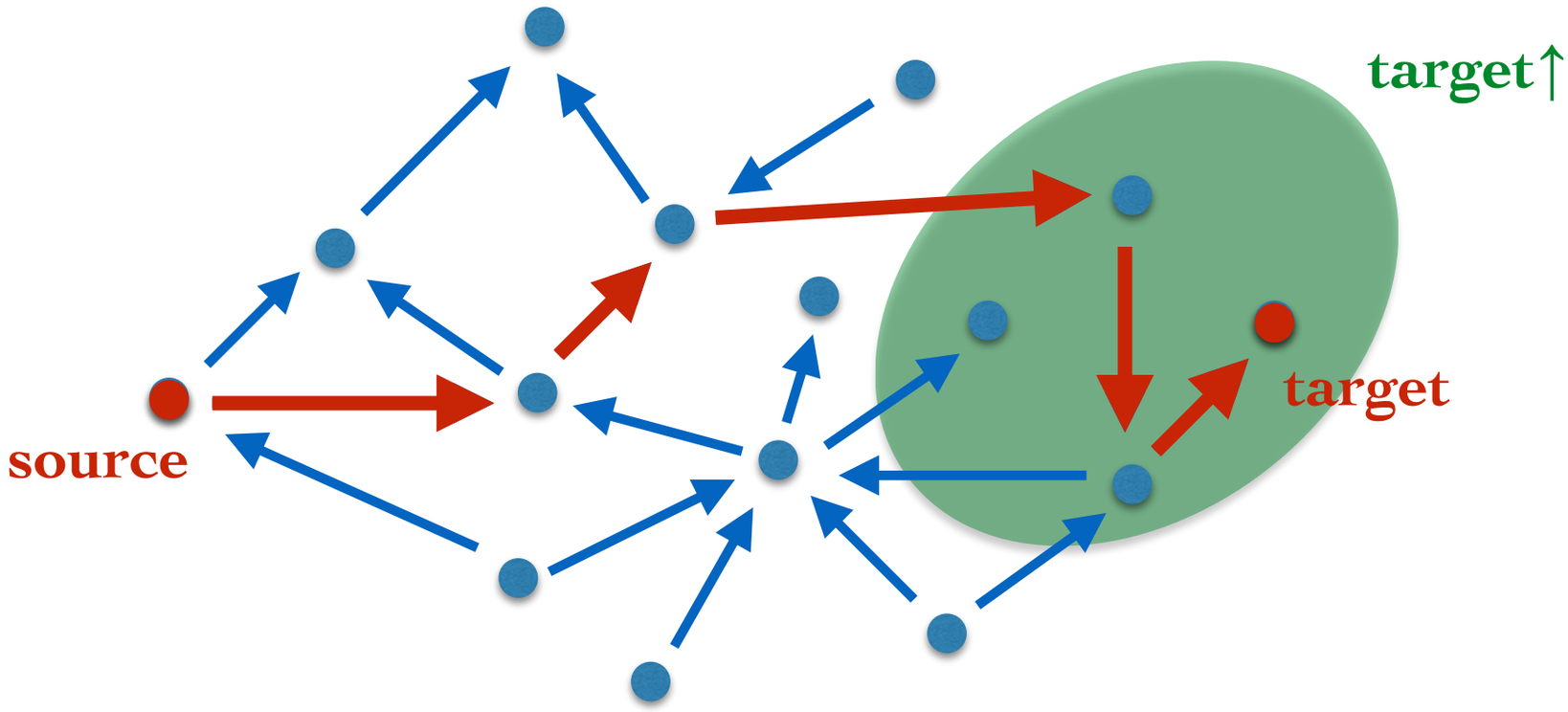
configuration graph: configurations and steps



Reachability: is there a path (**run**) from source to target ?

~~Reachability~~ Coverability problem in Petri nets

configuration graph: configurations and steps



Reachability: is there a path (**run**) from source to target ?

Coverability: is there a path (**run**) from source to target↑ ?

Why is it important?

Why is it important?

- core verification problem

Why is it important?

- core verification problem
- equivalent to many other problems in
  - concurrency,
  - process algebra,
  - logic,
  - automata theory,
  - language theory,
  - linear algebra,
  - …

1969 decidability of coverability **[Karp, Miller]**

1970

1980

1990

2000

2010

2020

1969 — decidability of coverability **[Karp, Miller]**

1970

1976 — EXPSPACE lower bound for coverability **[Lipton]**

1980

1990

2000

2010

2020

1969 — decidability of coverability **[Karp, Miller]**

1970

1976 — EXPSPACE lower bound for coverability **[Lipton]**

1978 — EXPSPACE algorithm for coverability **[Rackoff]**

1980

1990

2000

2010

2020

1969 — decidability of coverability **[Karp, Miller]**

1970

1976 — EXPSPACE lower bound for coverability **[Lipton]**

1978 — EXPSPACE algorithm for coverability **[Rackoff]**

1980

1981 — decidability of reachability **[Mayr]**

1990

2000

2010

2020

1969 — decidability of coverability **[Karp, Miller]**

1970

1976 — EXPSPACE lower bound for coverability **[Lipton]**

1978 — EXPSPACE algorithm for coverability **[Rackoff]**

1980

1981 — decidability of reachability **[Mayr]**

1982 — decidability of reachability - simplified proof **[Kosaraju]**

1990

2000

2010

2020

1969 — decidability of coverability **[Karp, Miller]**

1970

1976 — EXPSPACE lower bound for coverability **[Lipton]**

1978 — EXPSPACE algorithm for coverability **[Rackoff]**

1980

1981 — decidability of reachability **[Mayr]**

1982 — decidability of reachability - simplified proof **[Kosaraju]**

1990

1992 — decidability of reachability - refined data structure **[Lambert]**

KLM decomposition

2000

2010

2020

1969 — decidability of coverability **[Karp, Miller]**

1970

1976 — EXPSPACE lower bound for coverability **[Lipton]**

1978 — EXPSPACE algorithm for coverability **[Rackoff]**

1980

1981 — decidability of reachability **[Mayr]**

1982 — decidability of reachability - simplified proof **[Kosaraju]**

1990

1992 — decidability of reachability - refined data structure **[Lambert]**

2000

2010

2020

KLM decomposition

huge complexity gap!

1969 — decidability of coverability **[Karp, Miller]**

1970

1976 — EXPSPACE lower bound for coverability **[Lipton]**

1978 — EXPSPACE algorithm for coverability **[Rackoff]**

1980

1981 — decidability of reachability **[Mayr]**

1982 — decidability of reachability - simplified proof **[Kosaraju]**

1990

1992 — decidability of reachability - refined data structure **[Lambert]**

KLM decomposition

huge complexity gap!

2000

2010

2011 — decidability of reachability by semilinear invariants **[Leroux]**

2015 — upper bound $\mathbf{F}_{\omega^3}$ **[Leroux, Schmitz]**

2019 — Ackermannian upper bound $\mathbf{F}_{\omega}$ **[Leroux, Schmitz]**

2020

2019 — Ackermann upper bound $\mathbf{F}_\omega$ **[Leroux, Schmitz]**

2020

2019 — Ackermann upper bound F_ω **[Leroux, Schmitz]**

2019 — TOWER lower bound F_3 **[Czerwiński, L., Lazic, Leroux, Mazowiecki]**

2020

$$2^{2^{2^{\cdots^{2}}}\Big\}n}$$

2019 — Ackermann upper bound $\mathbf{F}_\omega$ **[Leroux, Schmitz]**

2019 — TOWER lower bound $\mathbf{F}_3$ **[Czerwiński, L., Lazic, Leroux, Mazowiecki]**

2020

$$2^{2^{2^{\cdots^{2}}}} \Big\} n$$

2021 — super-TOWER lower bound **[Czerwiński, L., Orlikowski]**

$$2^{2^{2^{\cdots^{2}}}} \Big\} 2^n$$

2019 — Ackermann upper bound F_ω **[Leroux, Schmitz]**

2019 — TOWER lower bound F_3 **[Czerwiński, L., Lazic, Leroux, Mazowiecki]**

2020

$$\left.2^{2^{2^{\cdots^{2}}}}\right\} n$$

2021 — super-TOWER lower bound **[Czerwiński, L., Orlikowski]**

$$\left.2^{2^{2^{\cdots^{2}}}}\right\} 2^n$$

gap closed!

2021 — Ackermann lower bound F_ω **[Czerwiński, Orlikowski] [Leroux]**

2019 — Ackermann upper bound F_ω **[Leroux, Schmitz]**

2019 — TOWER lower bound F_3 **[Czerwiński, L., Lazic, Leroux, Mazowiecki]**

2020

$$\left.2^{2^{2^{\cdots^2}}}\right\}n$$

2021 — super-TOWER lower bound **[Czerwiński, L., Orlikowski]**

$$\left.2^{2^{2^{\cdots^2}}}\right\}2^n$$

gap closed!

2021 — Ackermann lower bound F_ω **[Czerwiński, Orlikowski] [Leroux]**

2022 — $\mathsf{F}_\mathbf{d}$-hardness in dimension 3d+$O(1)$ **[L.]**

2019 — Ackermann upper bound F_ω **[Leroux, Schmitz]**

2019 — TOWER lower bound F_3 **[Czerwiński, L., Lazic, Leroux, Mazowiecki]**

2020

$$\left.2^{2^{2^{\cdots^2}}}\right\} n$$

2021 — super-TOWER lower bound **[Czerwiński, L., Orlikowski]**

$$\left.2^{2^{2^{\cdots^2}}}\right\} 2^n$$

gap closed!

2021 — Ackermann lower bound F_ω **[Czerwiński, Orlikowski] [Leroux]**

2022 — $\mathsf{F}_\mathbf{d}$-hardness in dimension 3d+$O(1)$ **[L.]**

2023 — $\mathsf{F}_\mathbf{d}$-hardness in dimension 2d+$O(1)$ **[Czerwiński, Jecker, L., Leroux, Orlikowski]**

2019 — Ackermann upper bound F_ω **[Leroux, Schmitz]**

2019 — TOWER lower bound F_3 **[Czerwiński, L., Lazic, Leroux, Mazowiecki]**

2020

$$\left.2^{2^{2^{\cdots^2}}}\right\}n$$

2021 — super-TOWER lower bound **[Czerwiński, L., Orlikowski]**

$$\left.2^{2^{2^{\cdots^2}}}\right\}2^n$$

gap closed!

2021 — Ackermann lower bound F_ω **[Czerwiński, Orlikowski] [Leroux]**

2022 — $\mathsf{F}_\mathbf{d}$-hardness in dimension 3d+$O(1)$ **[L.]**

2023 — $\mathsf{F}_\mathbf{d}$-hardness in dimension 2d+$O(1)$ **[Czerwiński, Jecker, L., Leroux, Orlikowski]**

2024 — $\mathsf{F}_\mathbf{d}$-upper bound in dimension d **[Fu, Yang, Zheng]**

2019 — Ackermann upper bound F_ω [Leroux, Schmitz]

2019 — TOWER lower bound F_3 [Czerwiński, L., Lazic, Leroux, Mazowiecki]

2020

$$2^{2^{2^{\cdots^2}}} \Big\} n$$

2021 — super-TOWER lower bound [Czerwiński, L., Orlikowski]

$$2^{2^{2^{\cdots^2}}} \Big\} 2^n$$

Fast growing functions and the induced complexity classes

gap closed!

2021 — Ackermann lower bound F_ω [Czerwiński, Orlikowski] [Leroux]

2022 — $F_{\mathbf{d}}$-hardness in dimension 3d+$O(1)$ [L.]

2023 — $F_{\mathbf{d}}$-hardness in dimension 2d+$O(1)$ [Czerwiński, Jecker, L., Leroux, Orlikowski]

2024 — $F_{\mathbf{d}}$-upper bound in dimension d [Fu, Yang, Zheng]

Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^{n}(1)$$

Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^{n}(1)$$

$$A_2(n) = 2^n$$

Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$$A_2(n) = 2^n$$

$$A_3(n) = \text{tower}(n) = \left.\begin{matrix} 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} \end{matrix}\right\} n$$

Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$$A_2(n) = 2^n$$

$$A_3(n) = \text{tower}(n) = \left.\begin{matrix} 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} \end{matrix}\right\} n$$

$$A_4(n) =$$

Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$$A_2(n) = 2^n$$

$$A_3(n) = \text{tower}(n) = \left.\begin{matrix} 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} \end{matrix}\right\} n$$

$$A_4(4) = 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}$$

Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$$A_2(n) = 2^n$$

$$A_3(n) = \text{tower}(n) = \left.\begin{matrix} 2^{2^{\cdots^2}} \end{matrix}\right\} n$$

$$A_4(4) = 2^{2^2}$$

Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$$A_2(n) = 2^n$$

$$A_3(n) = \text{tower}(n) = \left.\begin{matrix} 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} \end{matrix}\right\} n$$

$$A_4(4) = 2^{2^{2^2}}$$

Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$$A_2(n) = 2^n$$

$$A_3(n) = \text{tower}(n) = 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} \Big\}\, n$$

$$A_4(4) = 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} \Big\}\, 2^{2^{2^2}}$$

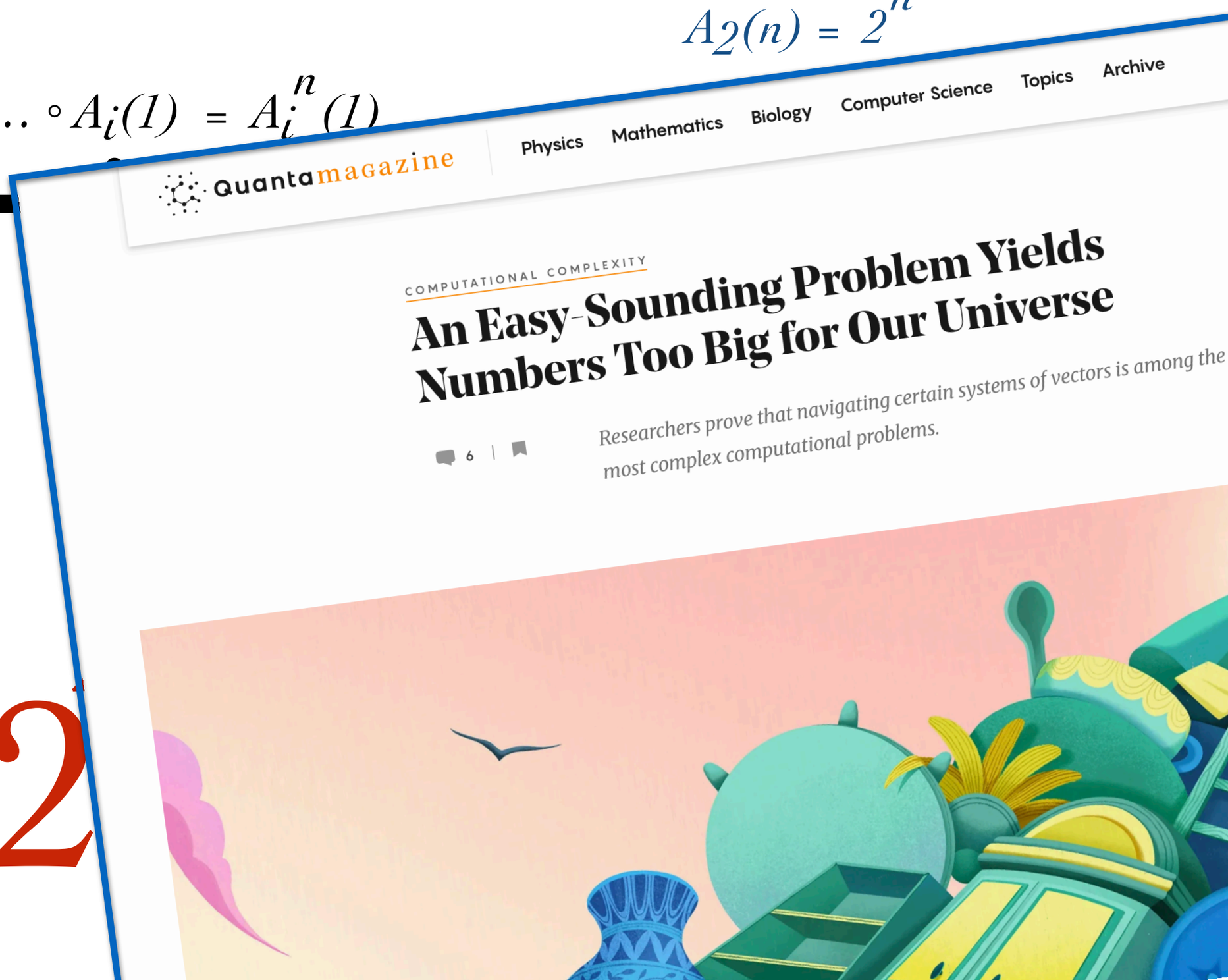
Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$$A_2(n) = 2^n$$

$$A_4(4) = 2^{\cdot^{\cdot^{\cdot^{2}}}}$$



Fast growing functions and the induced complexity classes

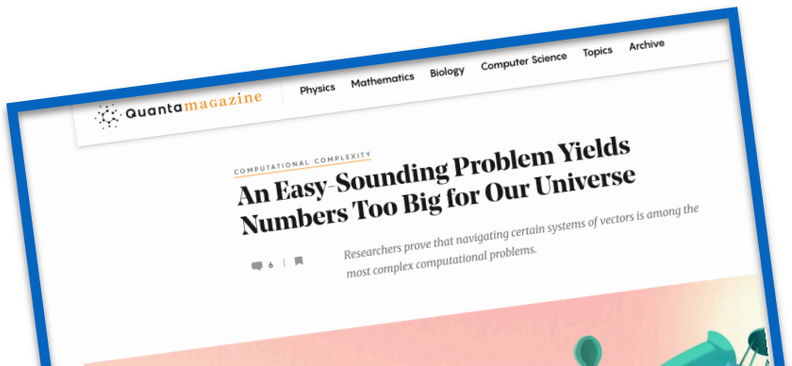
$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$$A_2(n) = 2^n$$

$$A_3(n) = \text{tower}(n) = \left.\begin{matrix} 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} \end{matrix}\right\} n$$

$$A_4(n) = \ldots$$



Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

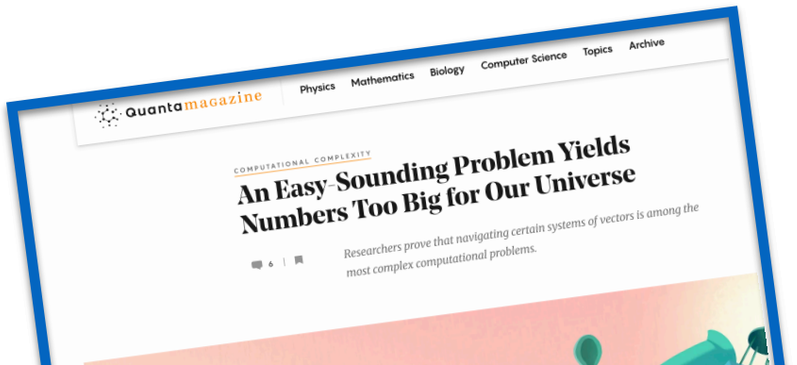
$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$$A_\omega(n) = A_n(n)$$ Ackermann function

$$A_2(n) = 2^n$$

$$A_3(n) = \text{tower}(n) = \left.\begin{matrix} 2^{2^{\cdots^2}} \end{matrix}\right\} n$$

$$A_4(n) = \ldots$$



Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

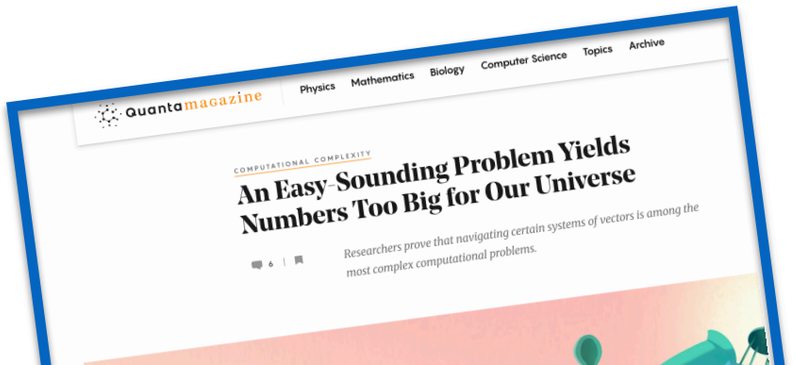
$$A_2(n) = 2^n$$

$$A_3(n) = \text{tower}(n) = 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} \Big\}\, n$$

$$A_4(n) = \ldots$$

$$A_\omega(n) = A_n(n) \quad \text{Ackermann function}$$

$$F_i = \bigcup_{j_1 \ldots j_m < i} \text{DTIME}(A_i \circ A_{j_1} \circ \ldots \circ A_{j_m})$$



Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$$A_\omega(n) = A_n(n)$$ Ackermann function

$$A_2(n) = 2^n$$

$$A_3(n) = \text{tower}(n) = 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} \Big\}\, n$$

$$A_4(n) = \ldots$$

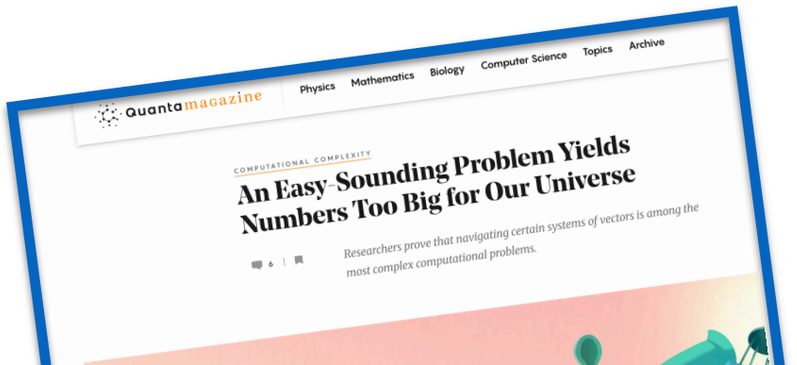
$$F_i = \bigcup_{j_1 \ldots j_m < i} \text{DTIME}(A_i \circ A_{j_1} \circ \ldots \circ A_{j_m})$$

$$F_2 = \text{DTIME}(2^{O(n)})$$

$$F_3 = \text{TOWER}$$

$$\ldots$$

$$F_\omega = \text{ACKERMANN}$$



Fast growing functions and the induced complexity classes

$$A_1(n) = 2n$$

$$A_{i+1}(n) = \underbrace{A_i \circ A_i \circ \ldots \circ A_i}_{n}(1) = A_i^n(1)$$

$A_\omega(n) = A_n(n)$ Ackermann function

$$A_2(n) = 2^n$$

$A_3(n) =$ tower(n)

$$= 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}} \Big\}\, n$$

$A_4(n) = \ldots$

$$F_i = \bigcup_{j_1 \ldots j_m < i} \mathrm{DTIME}(A_i \circ A_{j_1} \circ \ldots \circ A_{j_m})$$

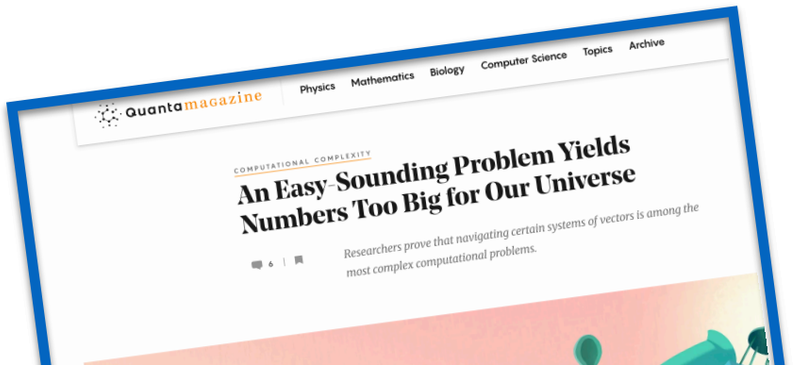
$F_2 = \mathrm{DTIME}(2^{O(n)})$

$F_3 = \mathrm{TOWER}$

…

$F_\omega = \mathrm{ACKERMANN}$

Hierarchy goes beyond ω !



Other decision problems

- boundedness
- place boundedness
- termination
- (in)evitability
- repeated coverability

- deadlock-freedom
- liveness
- semilinearity
- regular separability

- regularity / context freeness
- model-checking
- equivalence checking

Other decision problems

- boundedness
- place boundedness
- termination
- (in)evitability
- repeated coverability

EXPSPACE-complete

- deadlock-freedom
- liveness
- semilinearity
- regular separability

Ackermann-hard

- regularity / context freeness
- model-checking
- equivalence checking

undecidable

Other decision problems

- boundedness
- place boundedne
- ter
- (in)
- repe

- deadlo
- liveness
- semiline
- regular s

- regularity /
- model-check
- equivalence

reachability

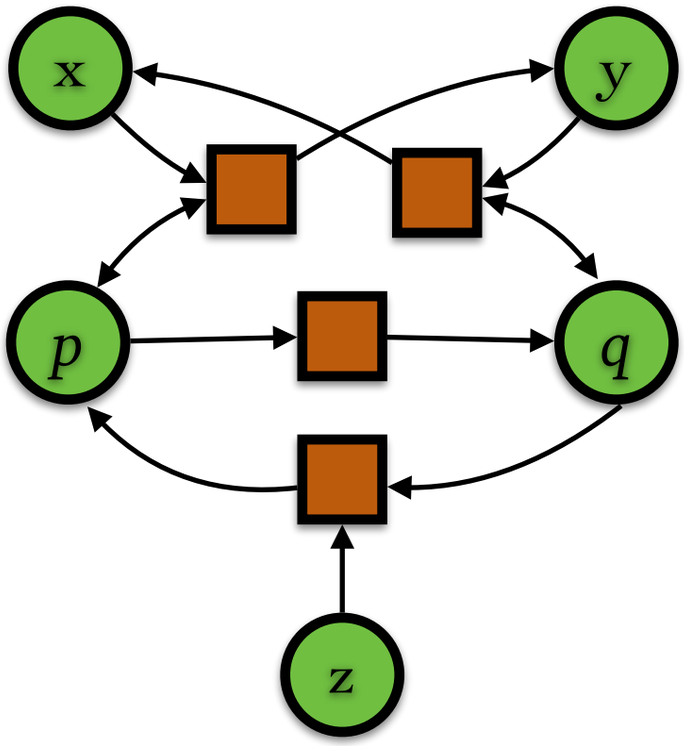
and

coverability

only

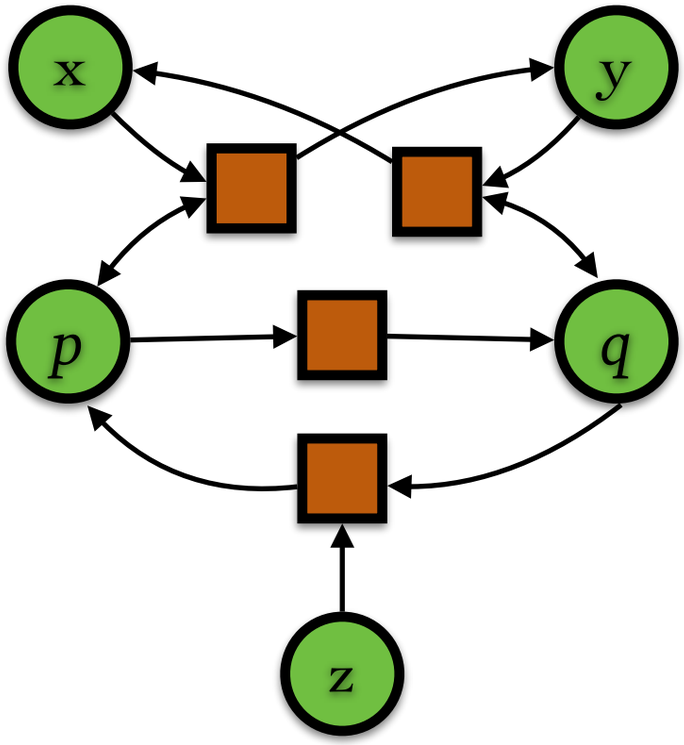
Many faces of Petri nets

- Petri nets:

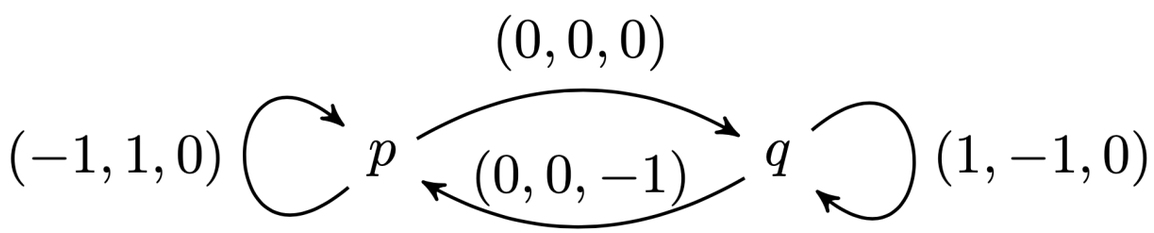


Many faces of Petri nets

- Petri nets:

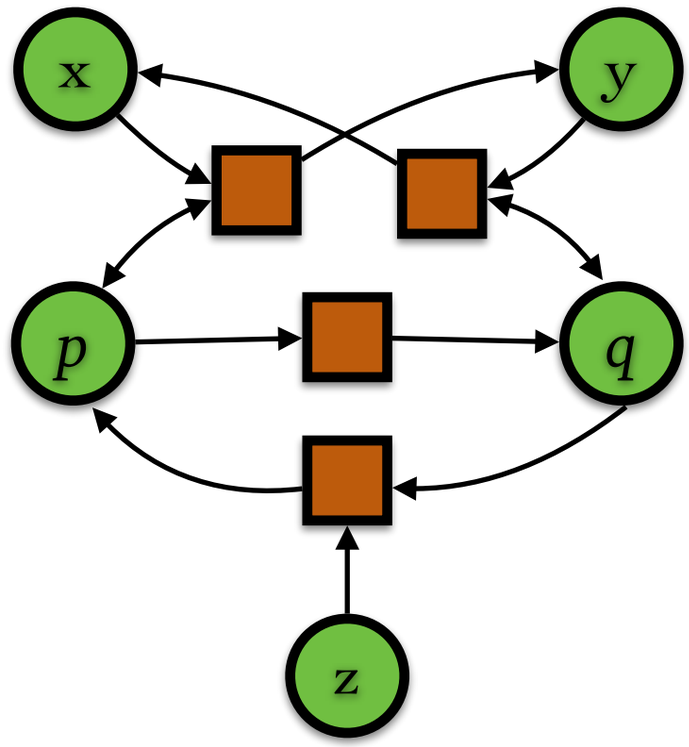


- vector addition systems with states (VASS):



Many faces of Petri nets

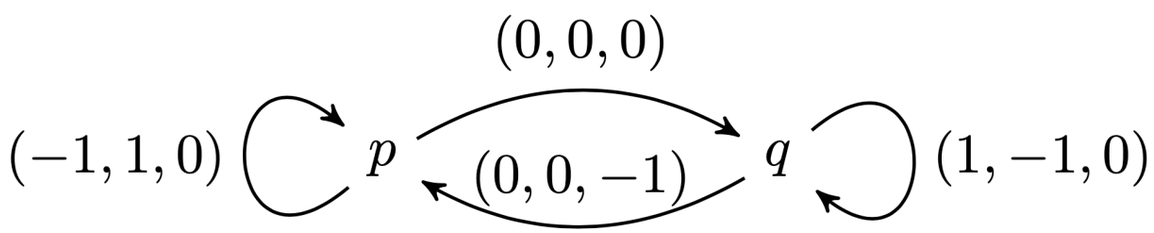
- Petri nets:



- counter programs **without zero-tests**:

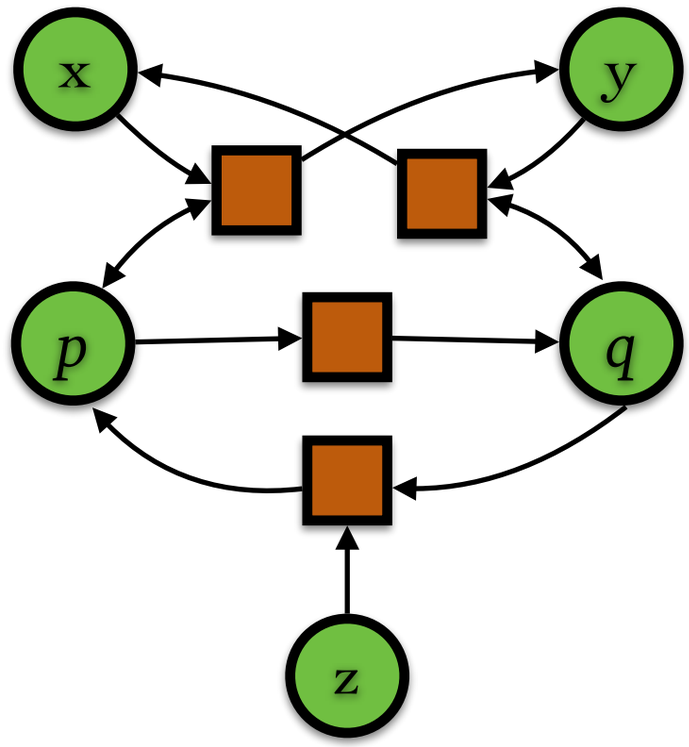
```
1: loop
2:     loop
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):



Many faces of Petri nets

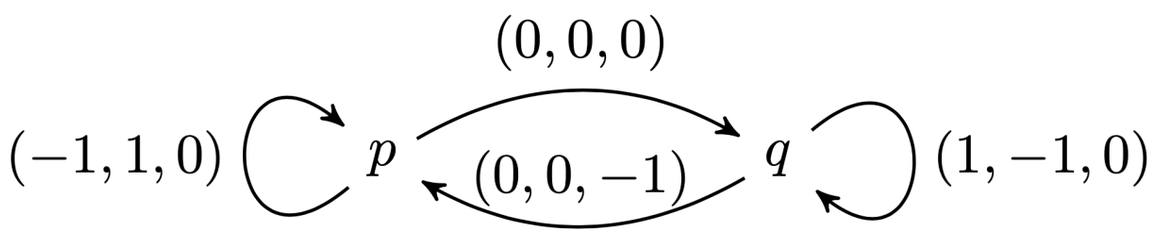
- Petri nets:



- counter programs **without zero-tests**:

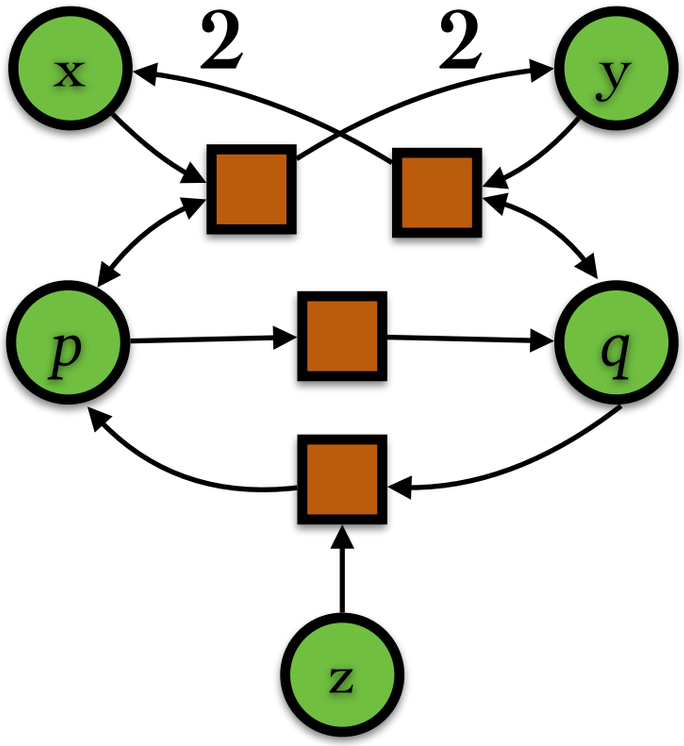
```
1:  goto 2 or 10
2:  goto 3 or 5
3:  x −= 1    y += 1
4:  goto 2
5:  goto 6 or 8
6:  x += 1    y −= 1
7:  goto 5
8:  z −= 1
9:  goto 1
10: ...
```

- vector addition systems with states (VASS):



Many faces of Petri nets

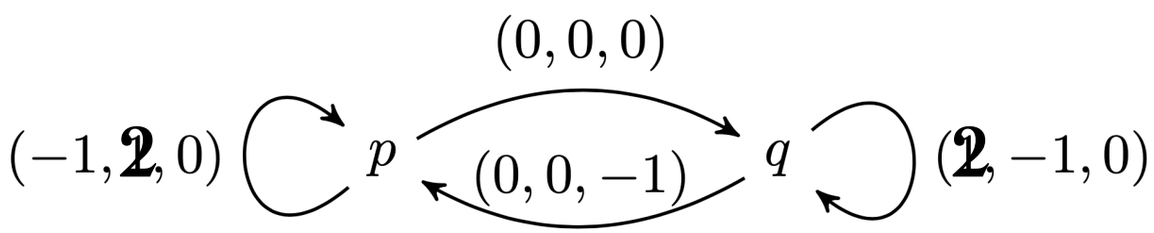
- Petri nets:



- counter programs **without zero-tests**:

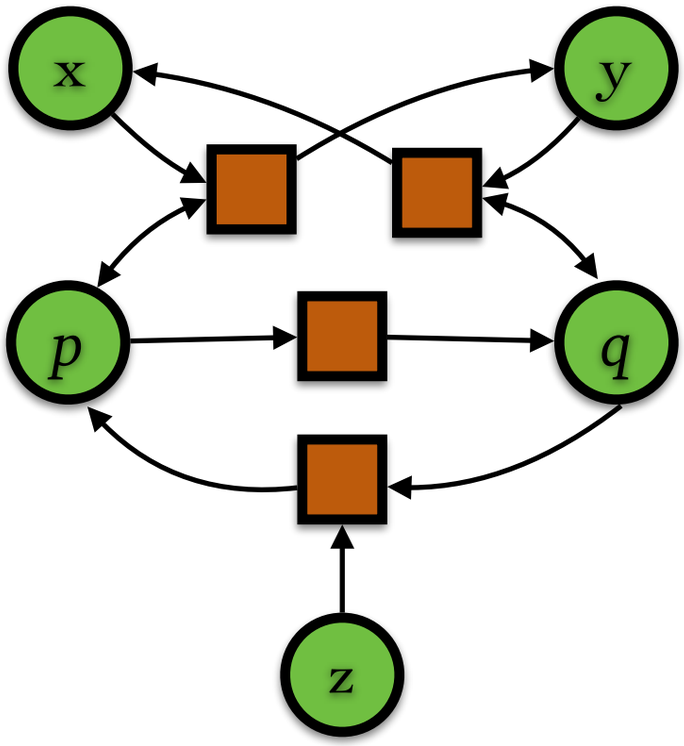
```
1: goto 2 or 10
2: goto 3 or 5
3: x −= 1    y += 2
4: goto 2
5: goto 6 or 8
6: x += 2    y −= 1
7: goto 5
8: z −= 1
9: goto 1
10: ...
```

- vector addition systems with states (VASS):



Many faces of Petri nets

- Petri nets:

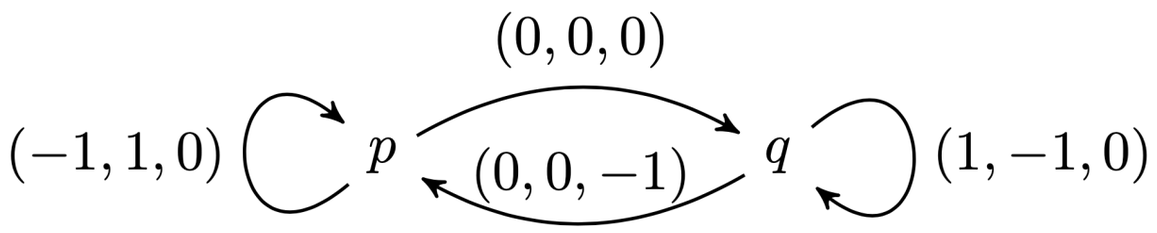


- counter programs **without zero-tests**:

```
1:  goto 2 or 10
2:  goto 3 or 5
3:  x −= 1    y += 1
4:  goto 2
5:  goto 6 or
6:  x += 1    y
7:  go
8:  z −= 1
9:  goto 1
10: ...
```

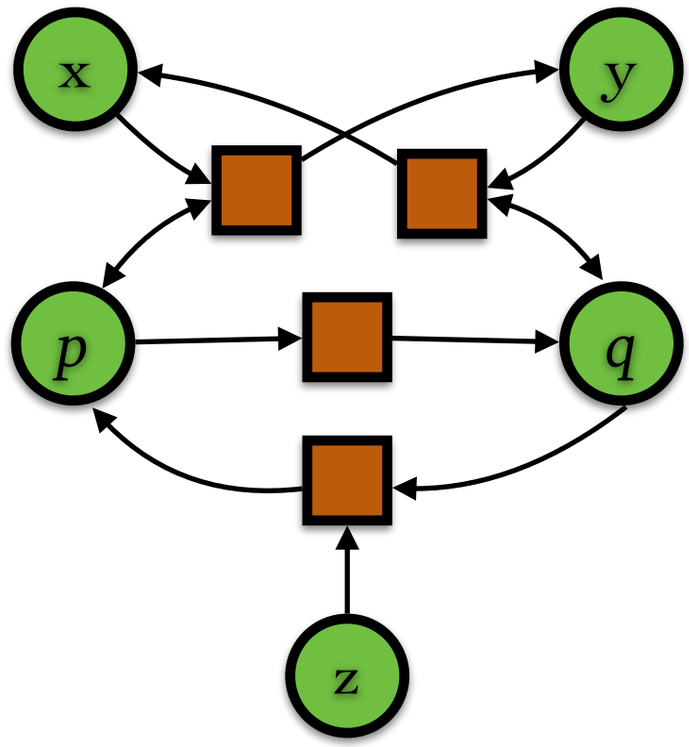
counter programs with zero-tests are Turing-complete

- vector addition systems with states (VASS):



Many faces of Petri nets

- Petri nets:



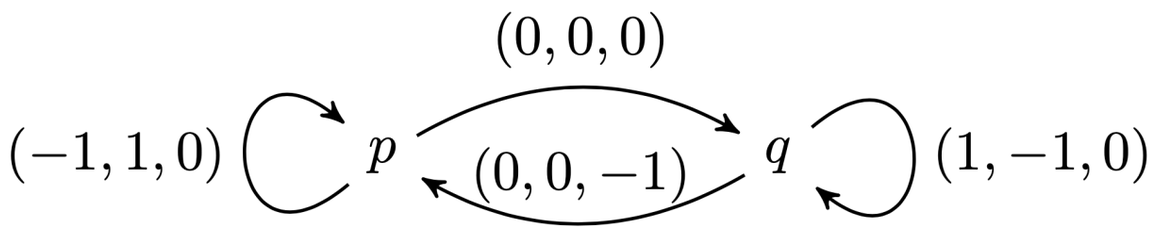
- counter programs **without zero-tests**:

```
1:  goto 2 or 10
2:  goto 3 or 5
3:  x −= 1    y += 1
4:  goto 2
5:  goto 6 or
6:  x += 1    y
7:  g
8:  z −= 1
9:  goto 1
10: ...
```

counter programs with zero-tests are Turing-complete

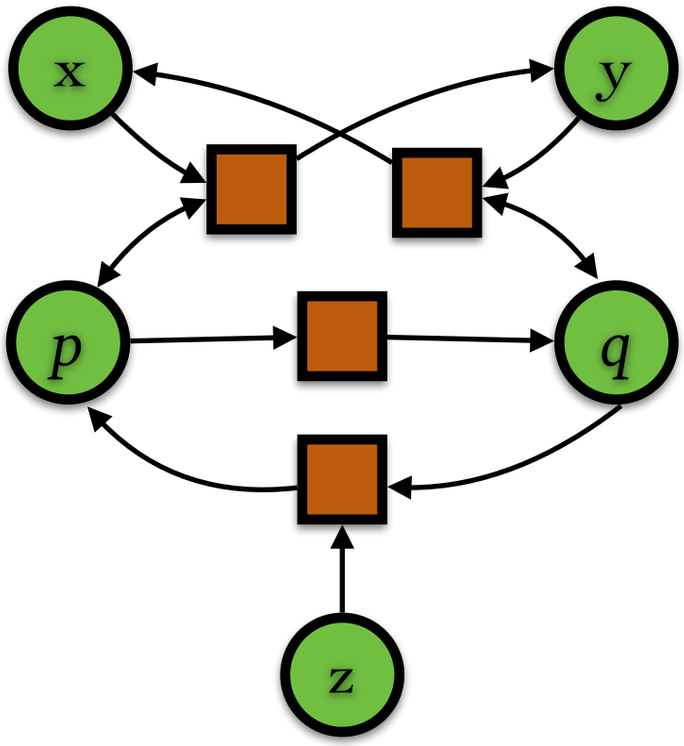
zero-test at the end only

- vector addition systems with states (VASS):



Many faces of Petri nets

- Petri nets:



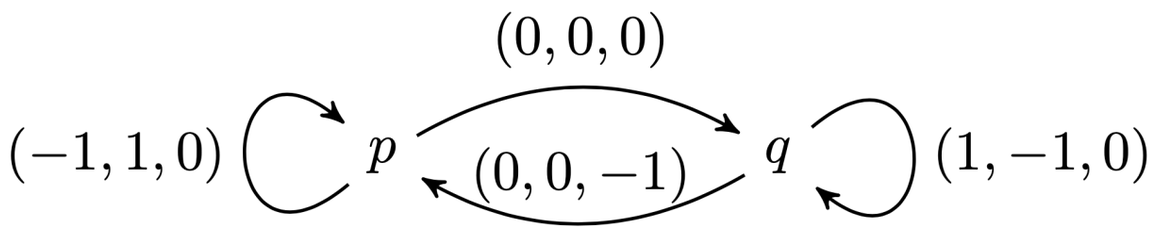
- counter programs **without zero-tests**:

```
1:  goto 2 or 10
2:  goto 3 or 5
3:  x −= 1     y += 1
4:  goto 2
5:  goto 6 or
6:  x += 1     y
7:  go
8:  z −= 1
9:  goto 1
10: ...
```

counter programs with zero-tests are Turing-complete

zero-test at the end only

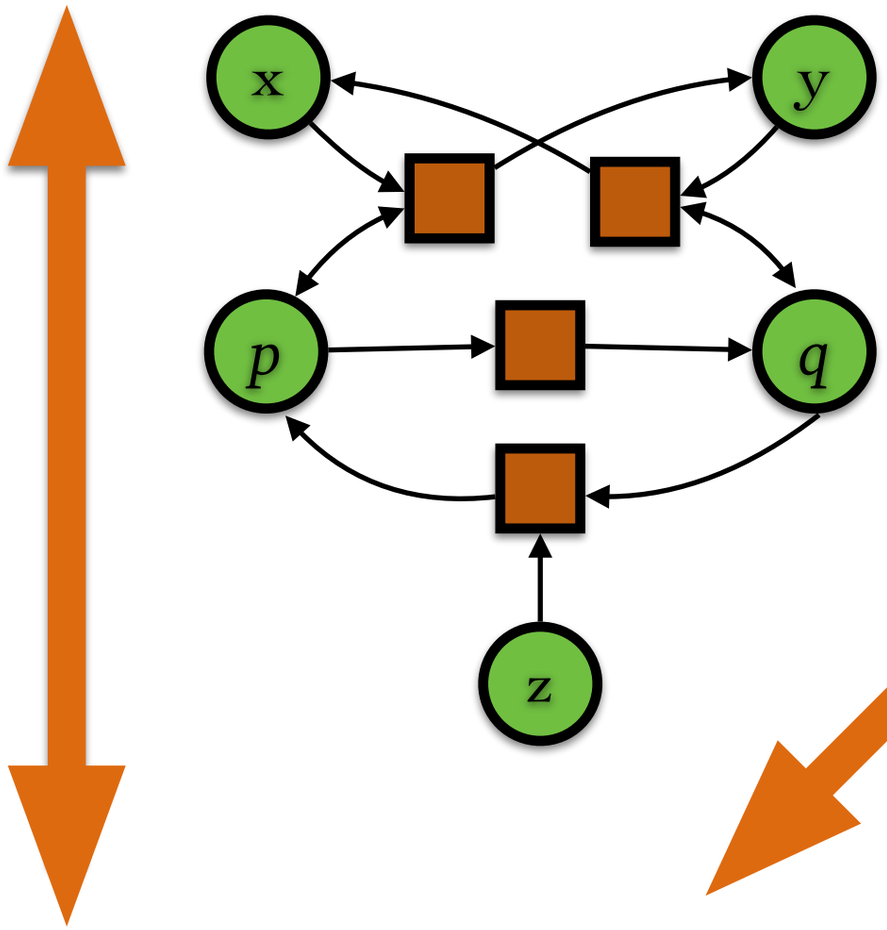
- vector addition systems with states (VASS):



- vector addition systems
- counter automata **without zero-tests**
- ground multiset rewriting
- …

Many faces of Petri nets

- Petri nets:



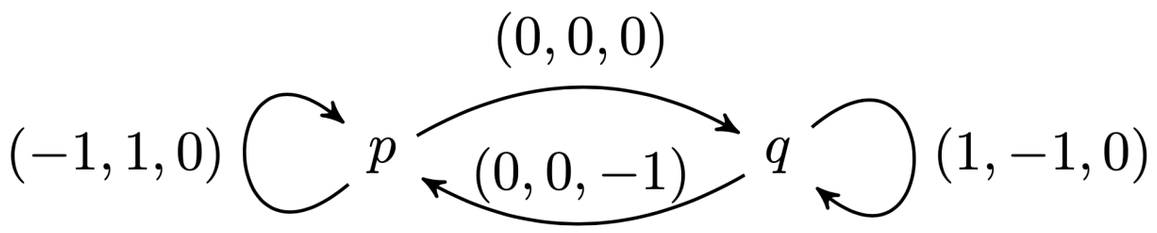
- counter programs **without zero-tests**:

```
1: goto 2 or 10
2: goto 3 or 5
3: x −= 1    y += 1
4: goto 2
5: goto 6 or
6: x += 1    y
7: g
8: z −= 1
9: goto 1
10: ...
```

counter programs with zero-tests are Turing-complete

zero-test at the end only

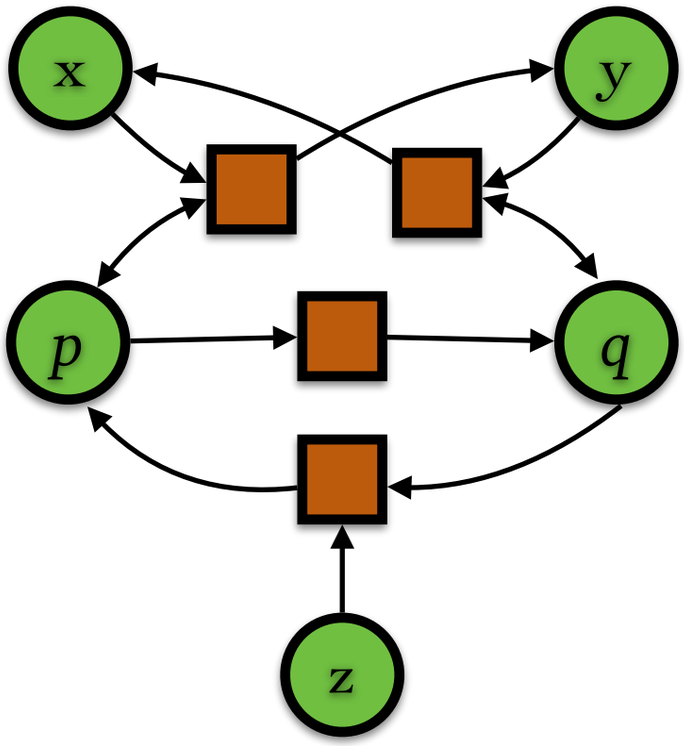
- vector addition systems with states (VASS):



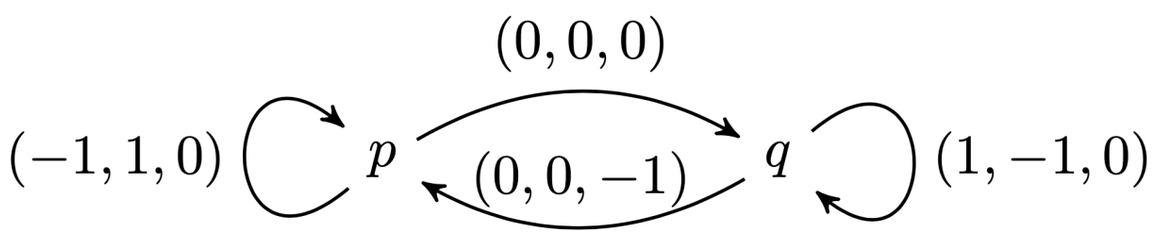
- vector addition systems
- counter automata **without zero-tests**
- ground multiset rewriting
- …

Petri nets $\leftrightarrows$ VASS

- Petri nets:

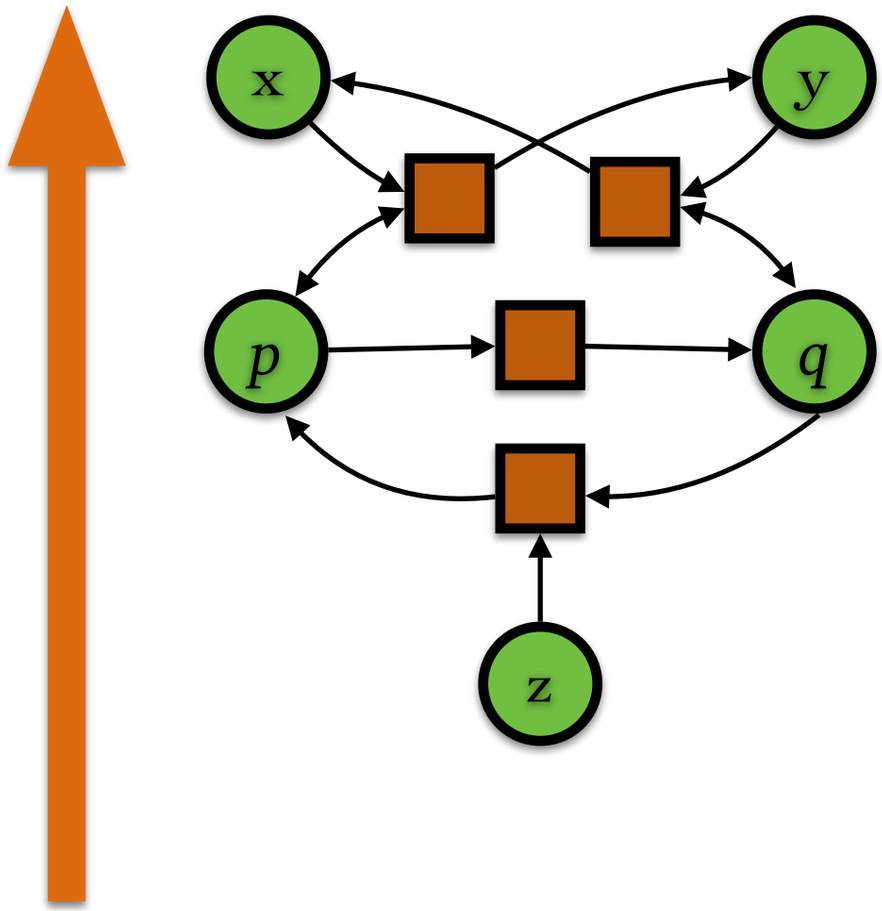


- vector addition systems with states (VASS):

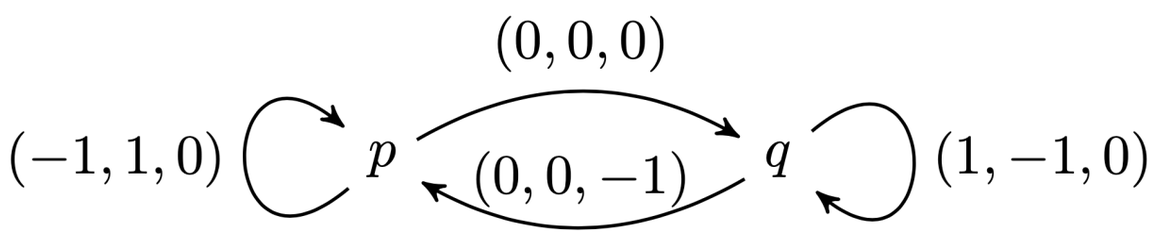


Petri nets $\leftrightarrows$ VASS

- Petri nets:

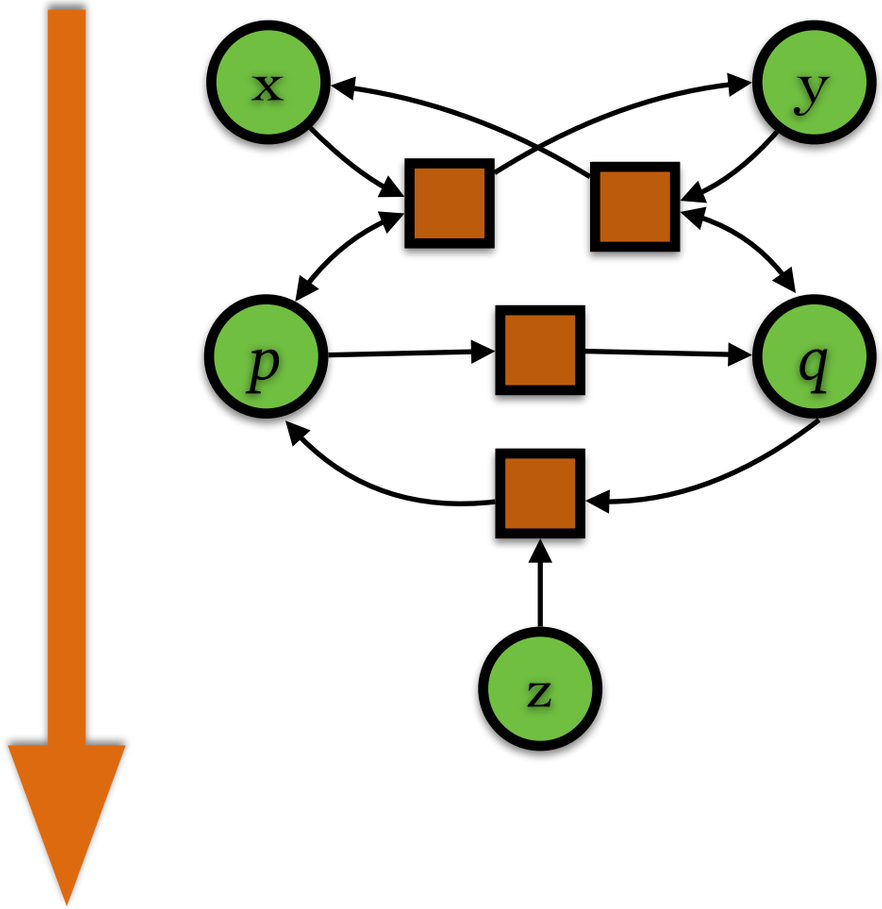


- vector addition systems with states (VASS):

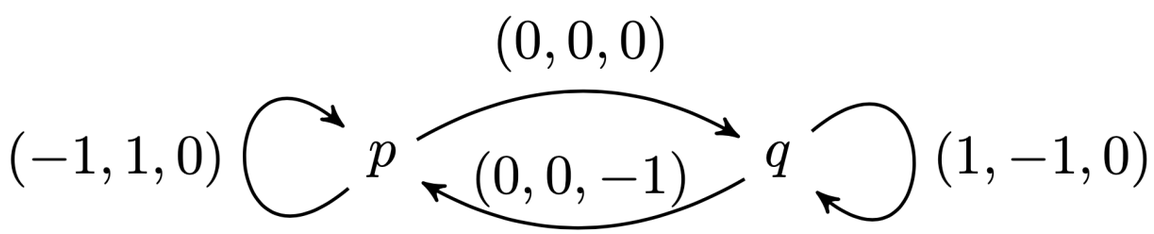


Petri nets $\leftrightarrows$ VASS

- Petri nets:

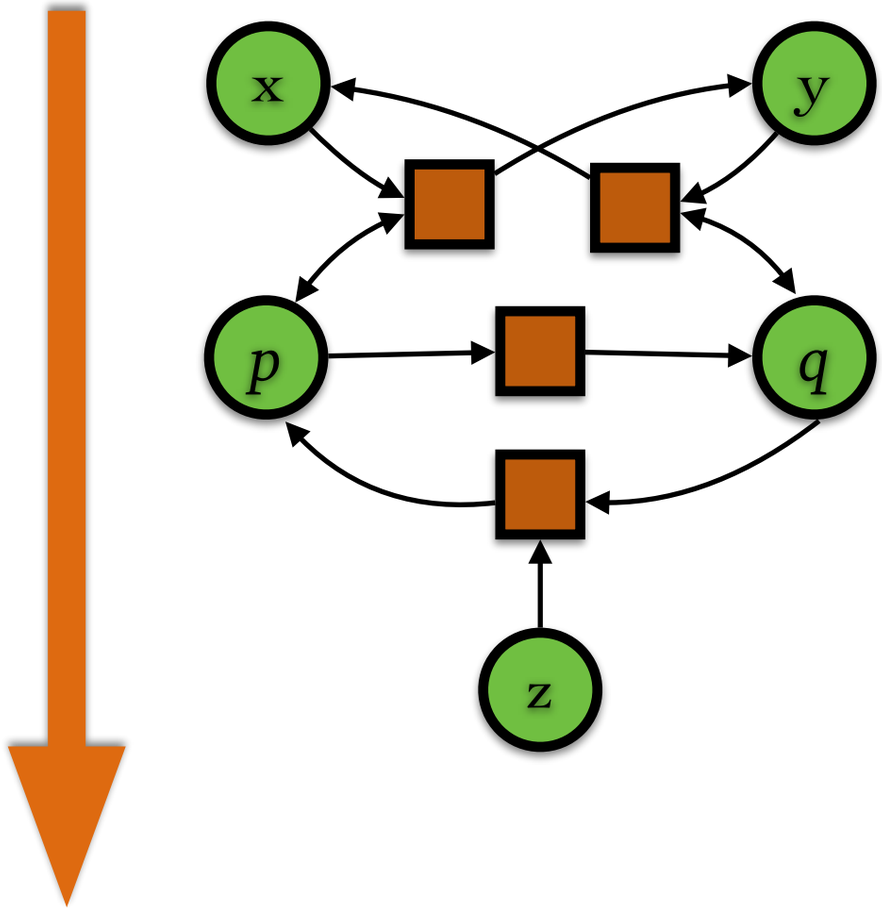


- vector addition systems with states (VASS):

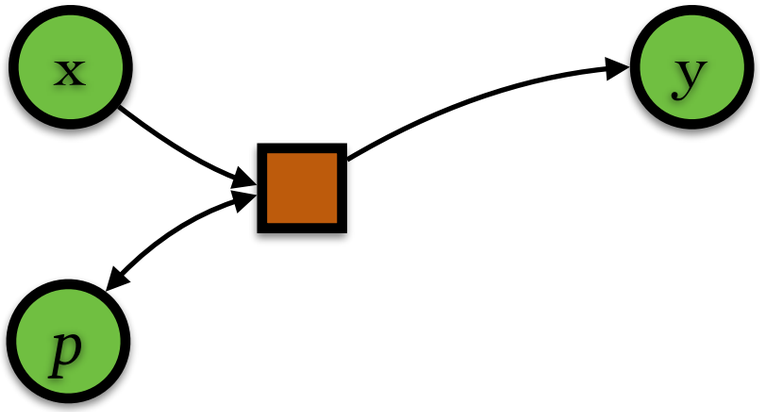


Petri nets $\leftrightarrows$ VASS

- Petri nets:

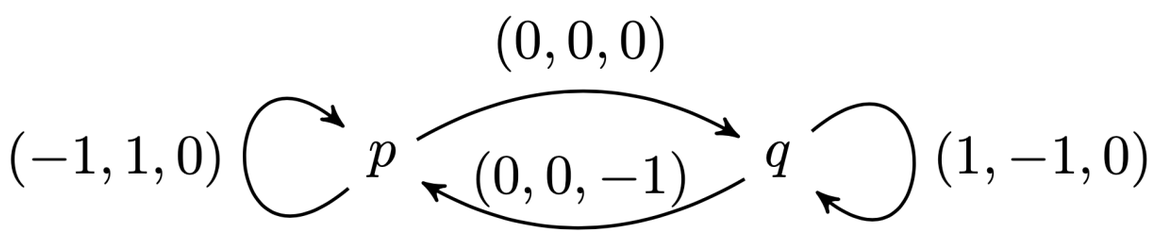


split every transition



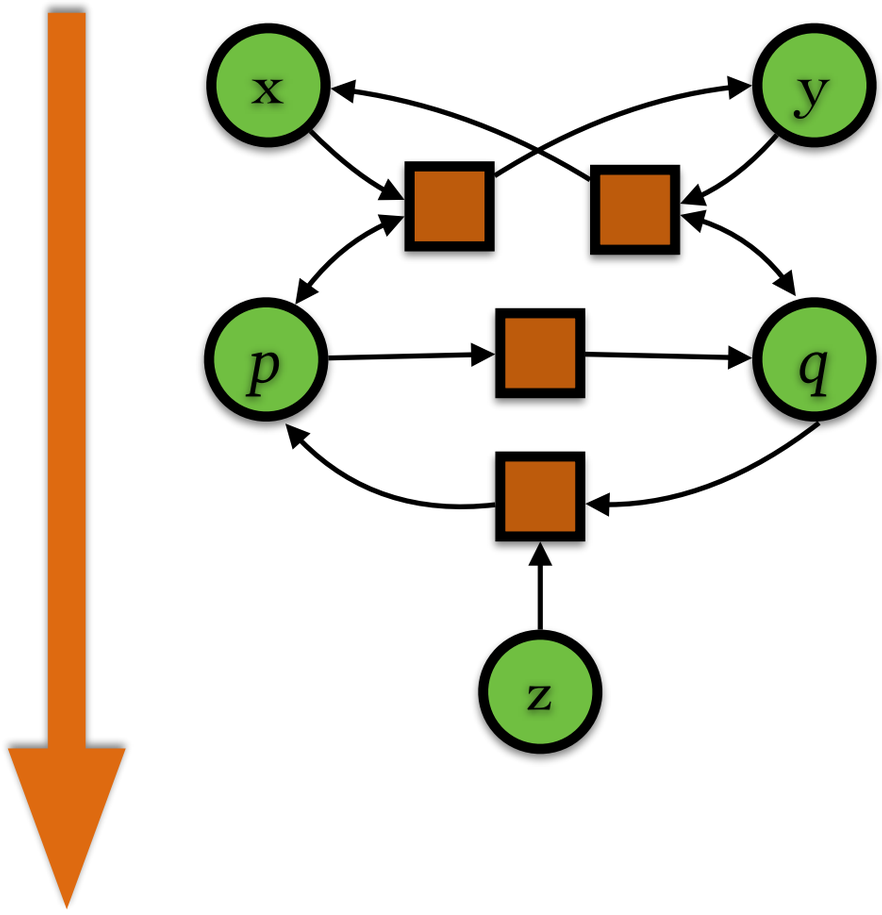
into input and output:

- vector addition systems with states (VASS):

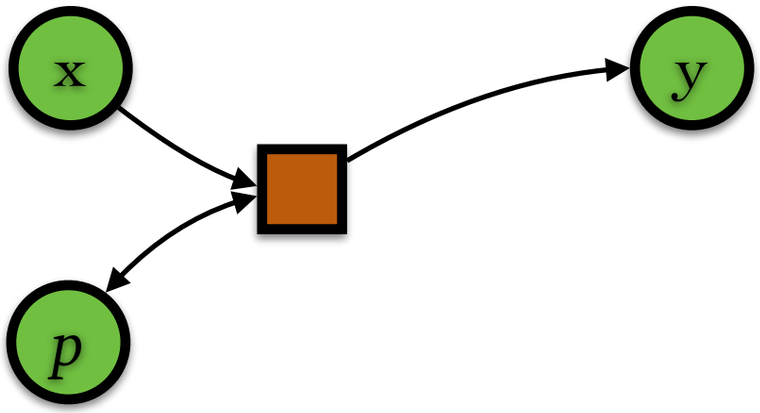


Petri nets $\leftrightarrows$ VASS

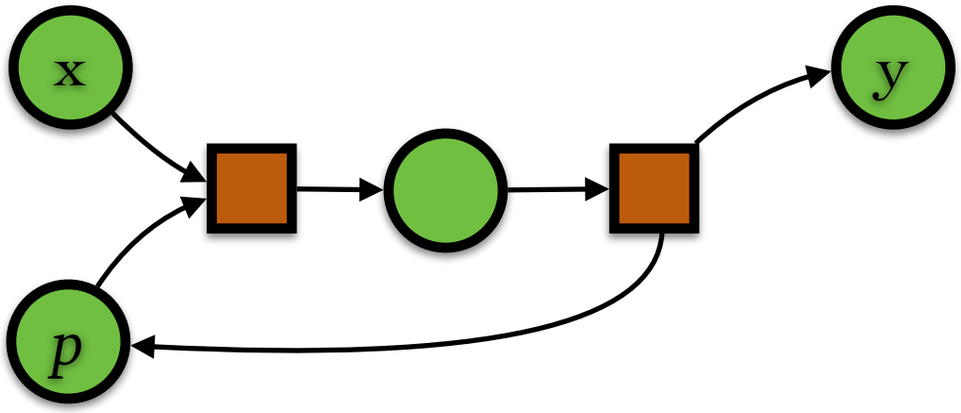
- Petri nets:



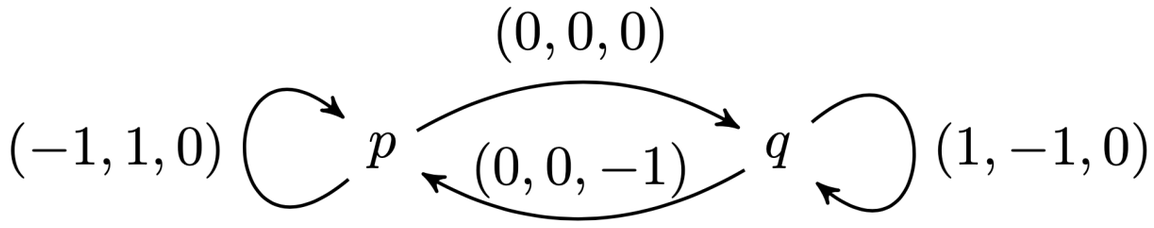
split every transition



into input and output:

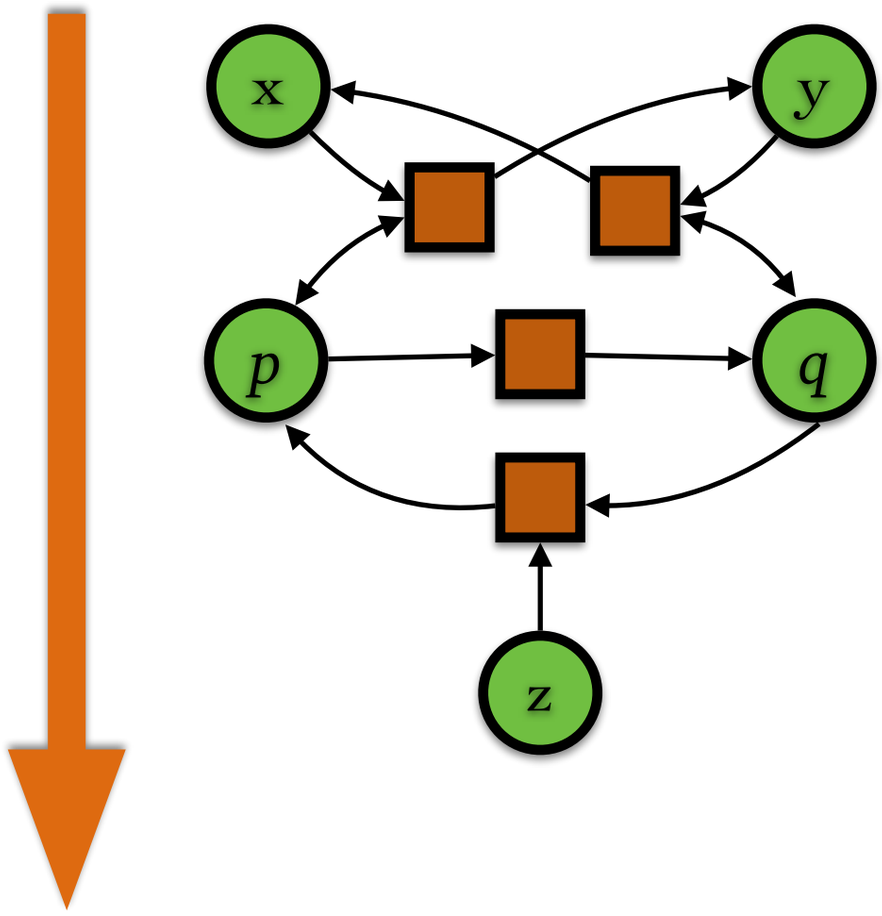


- vector addition systems with states (VASS):

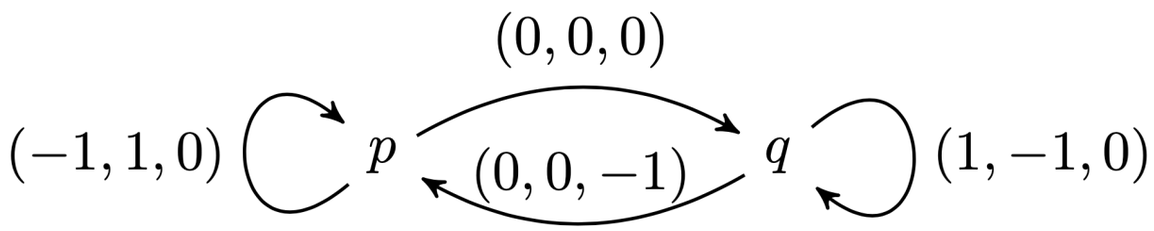


Petri nets $\leftrightarrows$ VASS

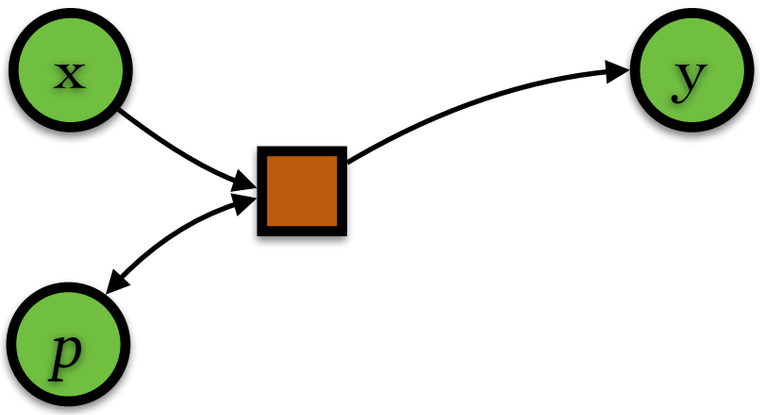
- Petri nets:



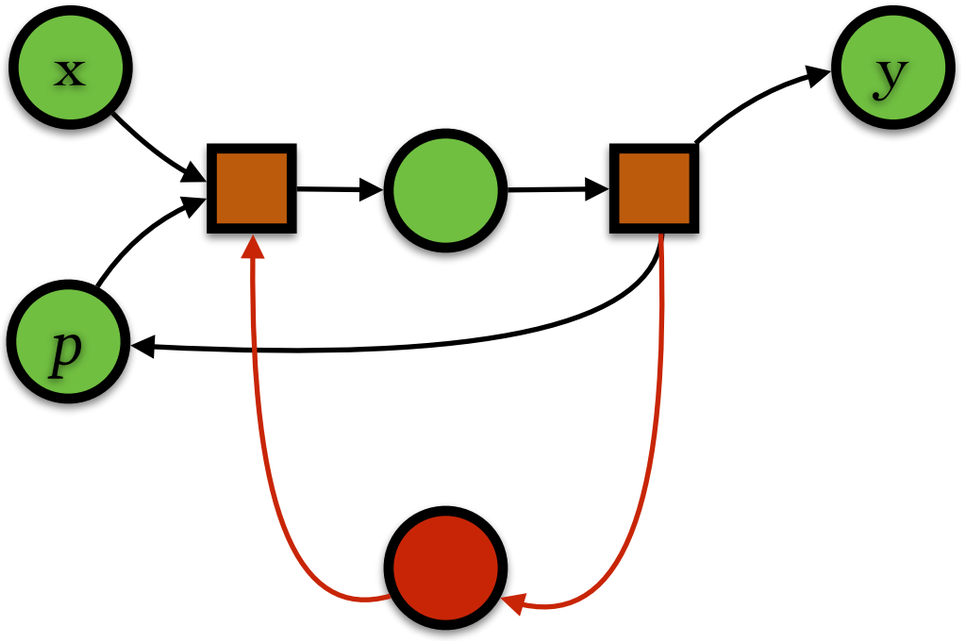
- vector addition systems with states (VASS):



split every transition



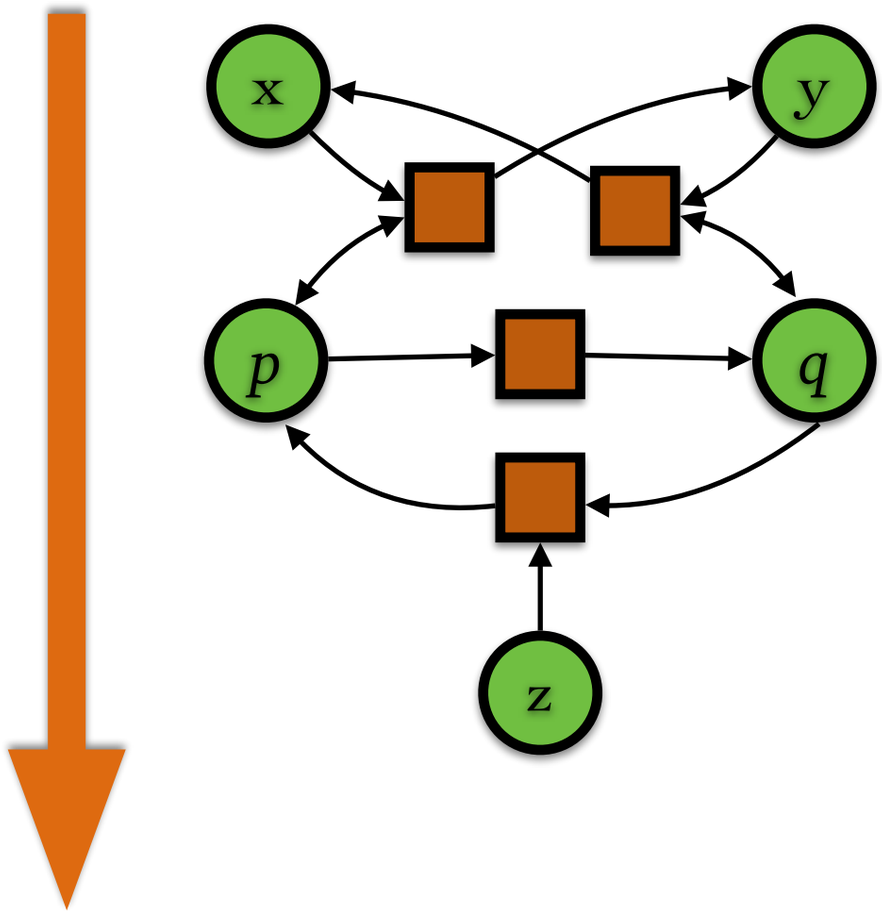
into input and output:



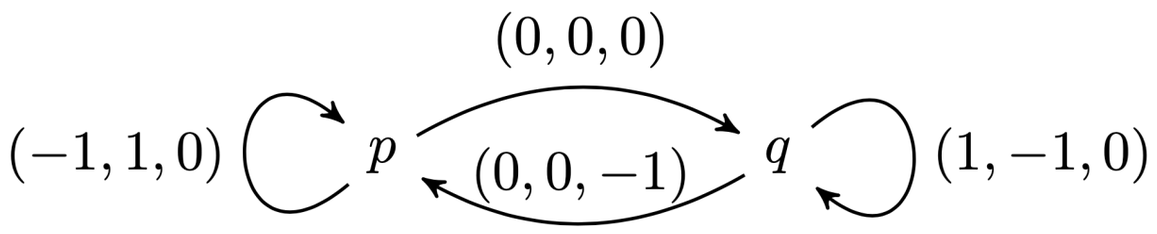
then add one more “global” place

Petri nets $\leftrightarrows$ VASS

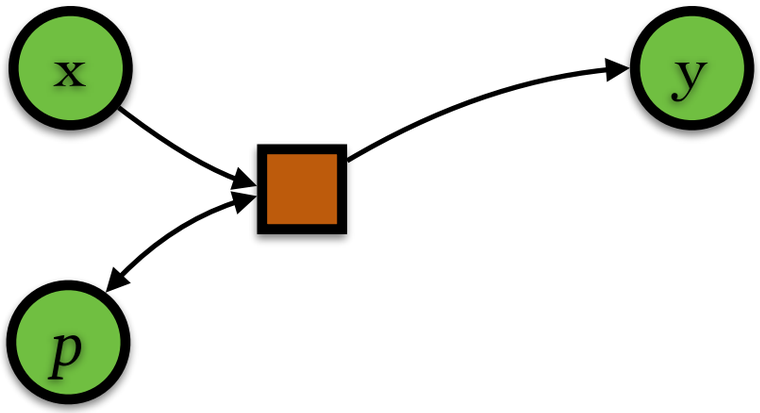
- Petri nets:



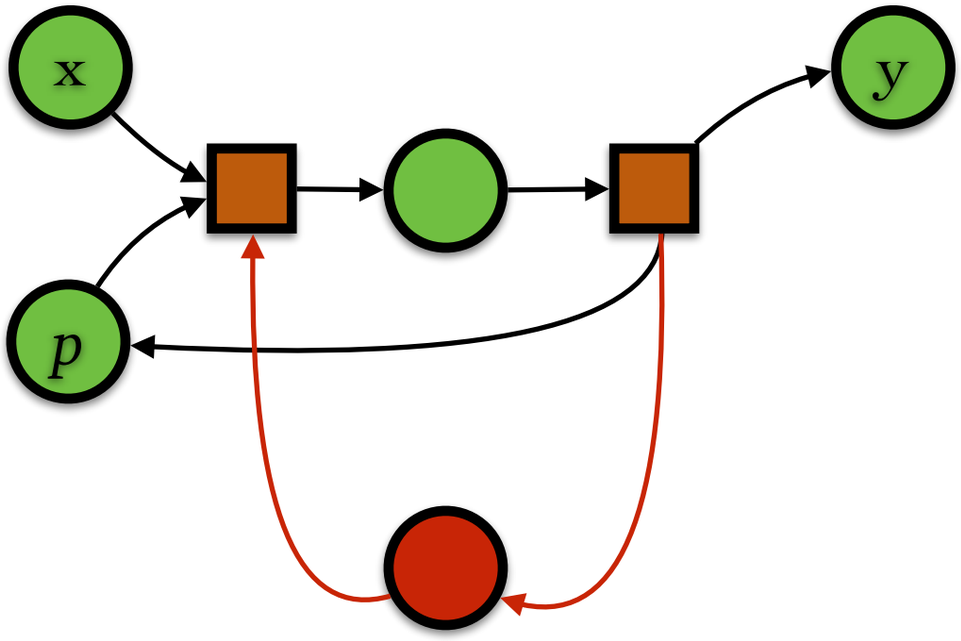
- vector addition systems with states (VASS):



split every transition



into input and output:



then add one more "global" place

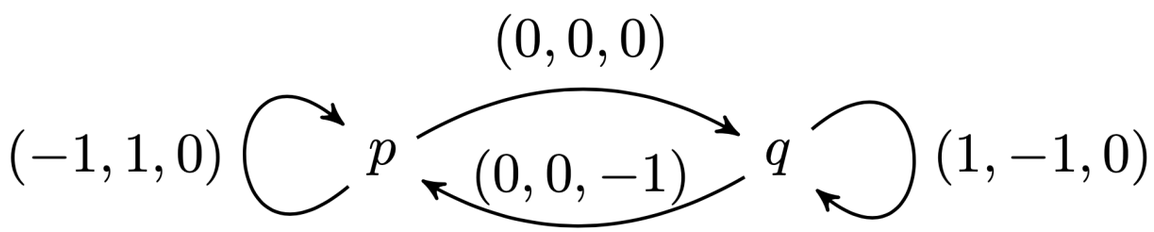
do we actually need  ?

Counter programs $\leftrightarrows$ VASS

- counter programs **without zero-tests**:

```
1:  goto 2 or 10
2:  goto 3 or 5
3:  x −= 1    y += 1
4:  goto 2
5:  goto 6 or 8
6:  x += 1    y −= 1
7:  goto 5
8:  z −= 1
9:  goto 1
10: ...
```

- vector addition systems with states (VASS):



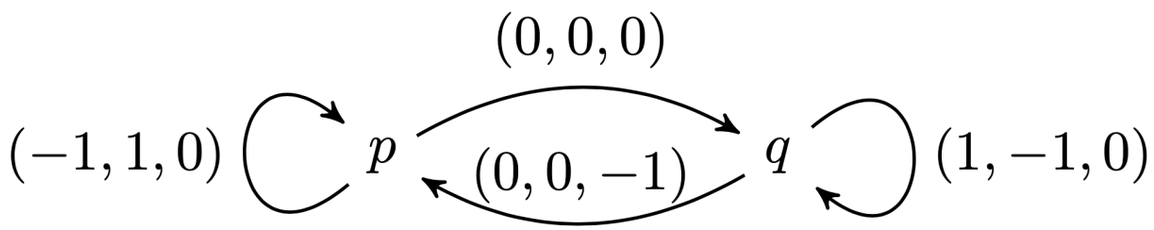
Counter programs $\leftrightarrows$ VASS

- dimension = number of counters
- control states = control locations
- transitions = commands

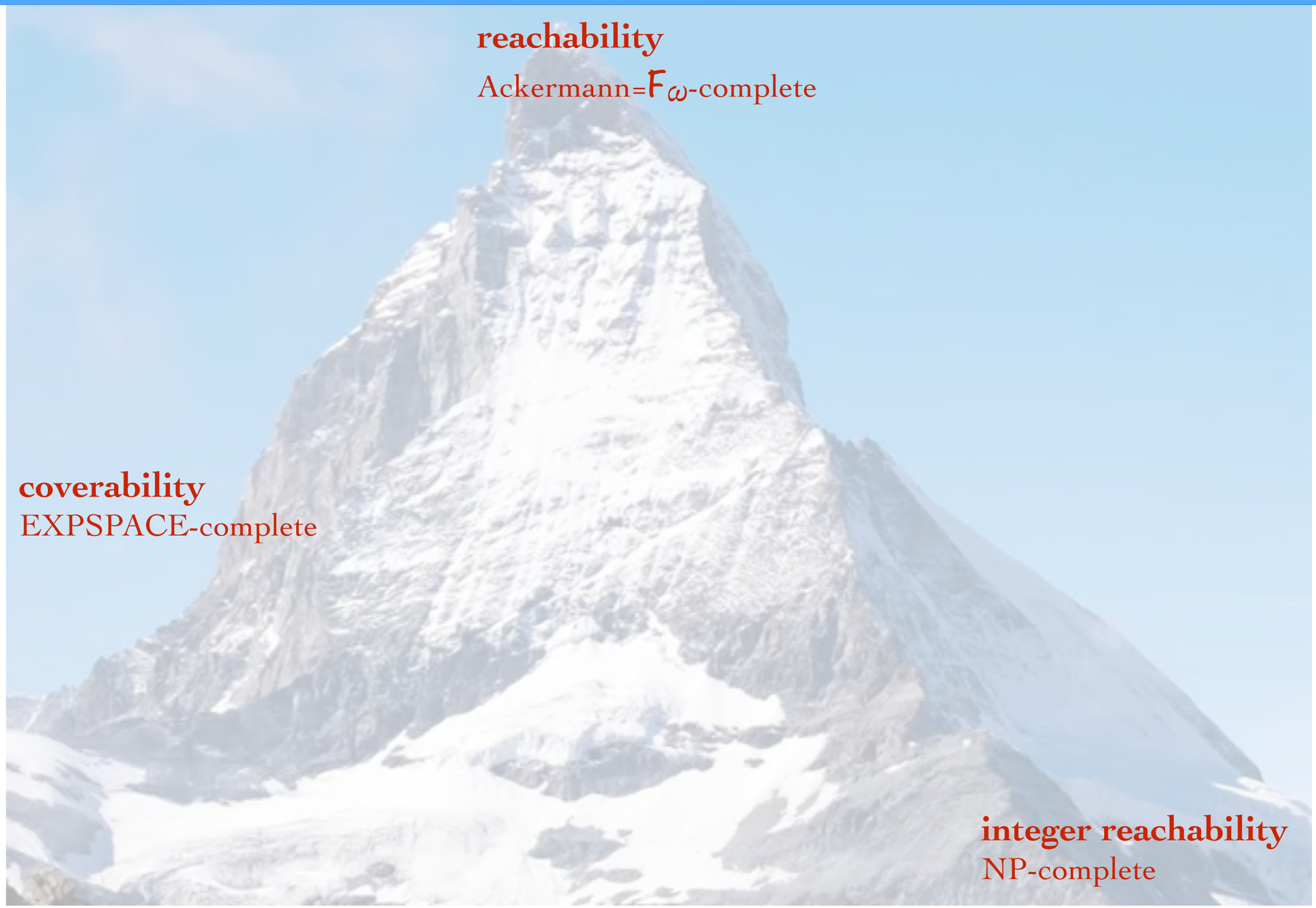
- counter programs **without zero-tests**:

```
 1: goto 2 or 10
 2: goto 3 or 5
 3: x −= 1    y += 1
 4: goto 2
 5: goto 6 or 8
 6: x += 1    y −= 1
 7: goto 5
 8: z −= 1
 9: goto 1
10: ...
```

- vector addition systems with states (VASS):



VASS


reachability
Ackermann=F_ω-complete
coverability
EXPSPACE-complete
integer reachability
NP-complete

VASS

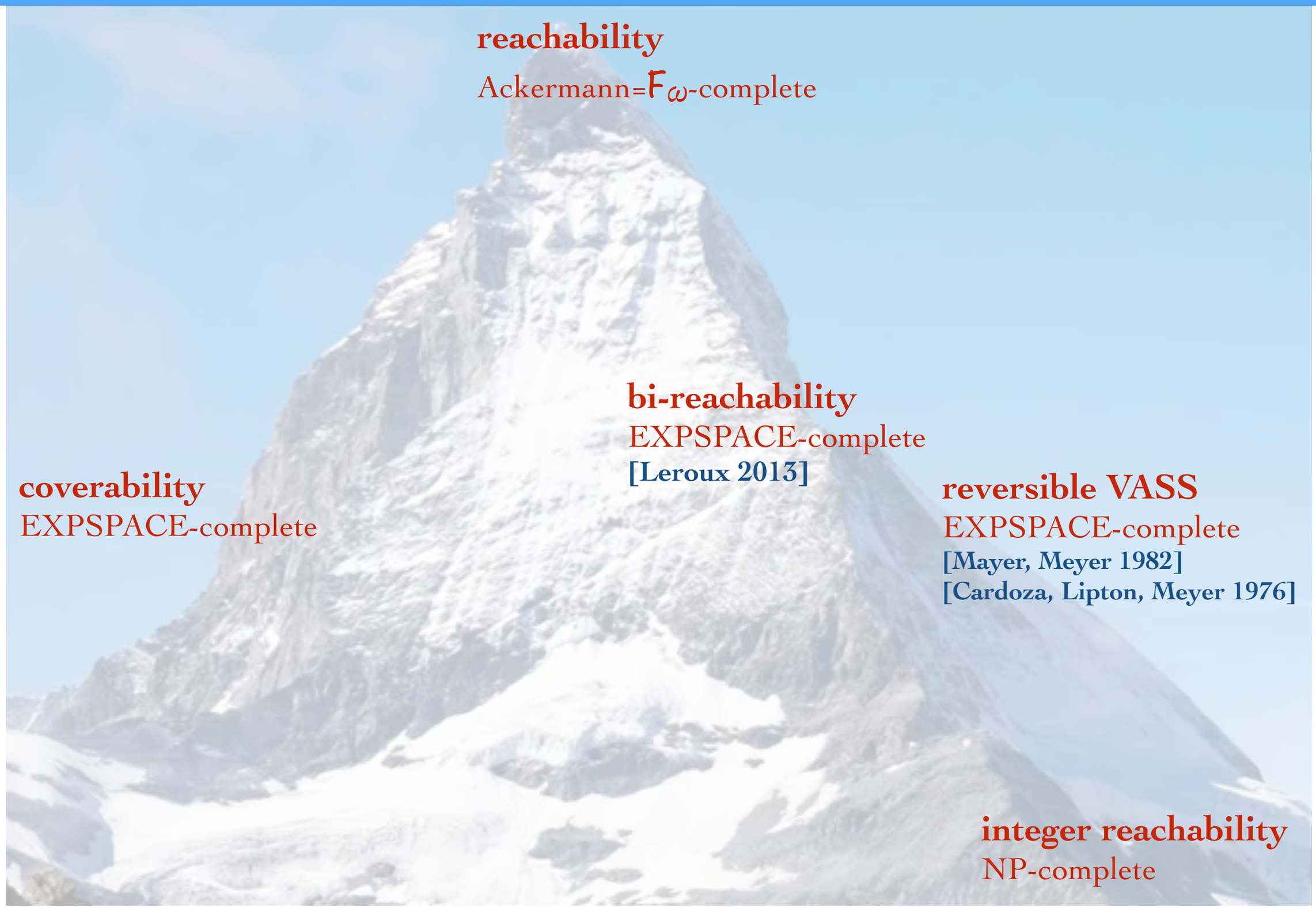
reachability
Ackermann=F_ω-complete

coverability
EXPSPACE-complete

reversible VASS
EXPSPACE-complete
[Mayer, Meyer 1982]
[Cardoza, Lipton, Meyer 1976]

integer reachability
NP-complete

VASS


reachability
Ackermann=F_ω-complete
bi-reachability
EXPSPACE-complete
[Leroux 2013]
coverability
EXPSPACE-complete
reversible VASS
EXPSPACE-complete
[Mayer, Meyer 1982]
[Cardoza, Lipton, Meyer 1976]
integer reachability
NP-complete

VASS

reachability
Ackermann=F_ω-complete

bi-reachability
EXPSPACE-complete
[Leroux 2013]

coverability
EXPSPACE-complete

reversible VASS
EXPSPACE-complete
[Mayer, Meyer 1982]
[Cardoza, Lipton, Meyer 1976]

dimension 1
NP-complete
[Haase, Kreutzer, Ouaknine, Joel 2009]

integer reachability
NP-complete

VASS

reachability
Ackermann=F_ω-complete

bi-reachability
EXPSPACE-complete
[Leroux 2013]

coverability
EXPSPACE-complete

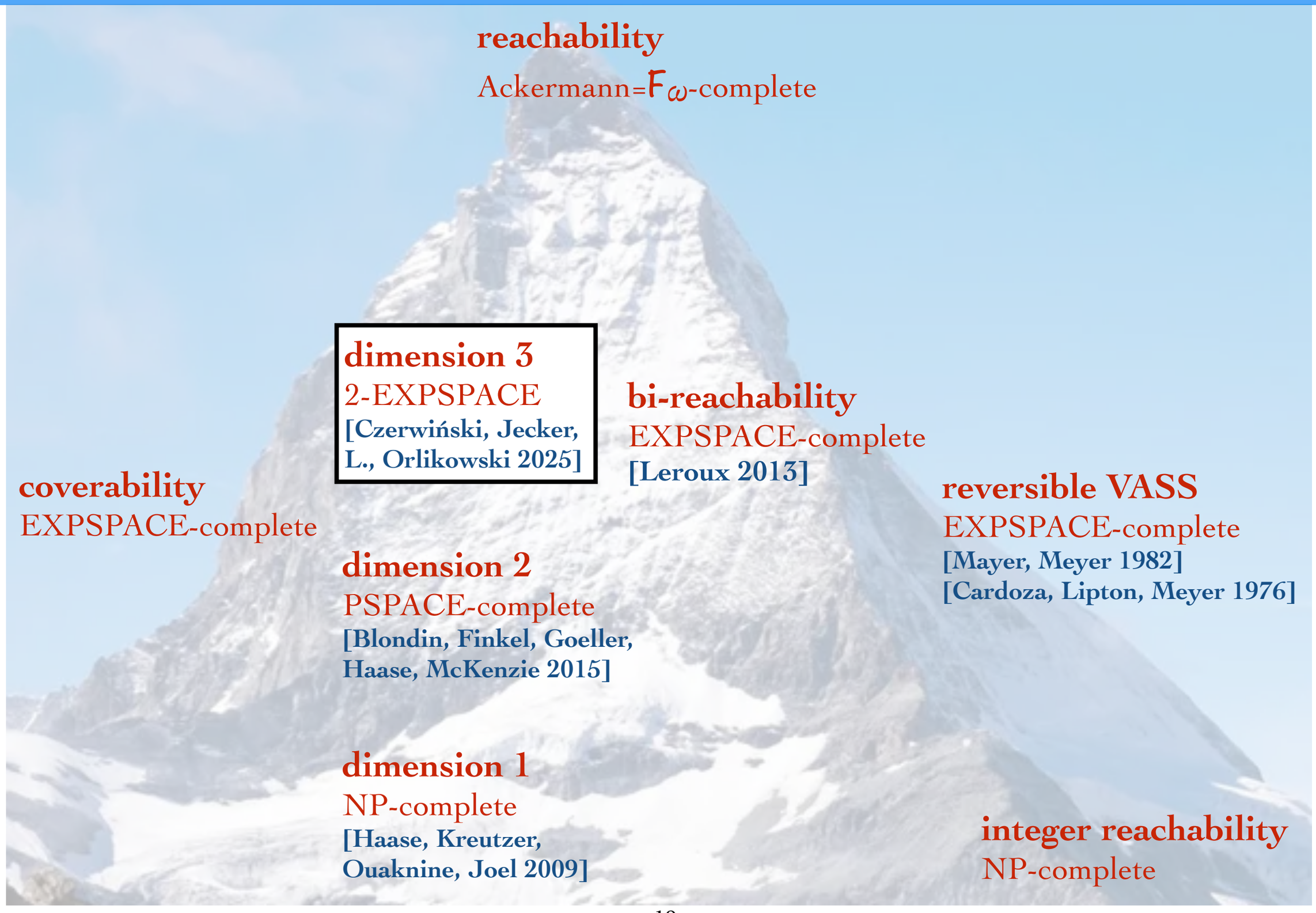
reversible VASS
EXPSPACE-complete
[Mayer, Meyer 1982]
[Cardoza, Lipton, Meyer 1976]

dimension 2
PSPACE-complete
[Blondin, Finkel, Goeller, Haase, McKenzie 2015]

dimension 1
NP-complete
[Haase, Kreutzer, Ouaknine, Joel 2009]

integer reachability
NP-complete

VASS


reachability
Ackermann=F_ω-complete
dimension 3
2-EXPSPACE
[Czerwiński, Jecker, L., Orlikowski 2025]
bi-reachability
EXPSPACE-complete
[Leroux 2013]
coverability
EXPSPACE-complete
reversible VASS
EXPSPACE-complete
[Mayer, Meyer 1982]
[Cardoza, Lipton, Meyer 1976]
dimension 2
PSPACE-complete
[Blondin, Finkel, Goeller, Haase, McKenzie 2015]
dimension 1
NP-complete
[Haase, Kreutzer, Ouaknine, Joel 2009]
integer reachability
NP-complete

VASS

reachability
Ackermann=F_ω-complete

dimension d
?

dimension 3
2-EXPSPACE
[Czerwiński, Jecker, L., Orlikowski 2025]

bi-reachability
EXPSPACE-complete
[Leroux 2013]

coverability
EXPSPACE-complete

reversible VASS
EXPSPACE-complete
[Mayer, Meyer 1982]
[Cardoza, Lipton, Meyer 1976]

dimension 2
PSPACE-complete
[Blondin, Finkel, Goeller, Haase, McKenzie 2015]

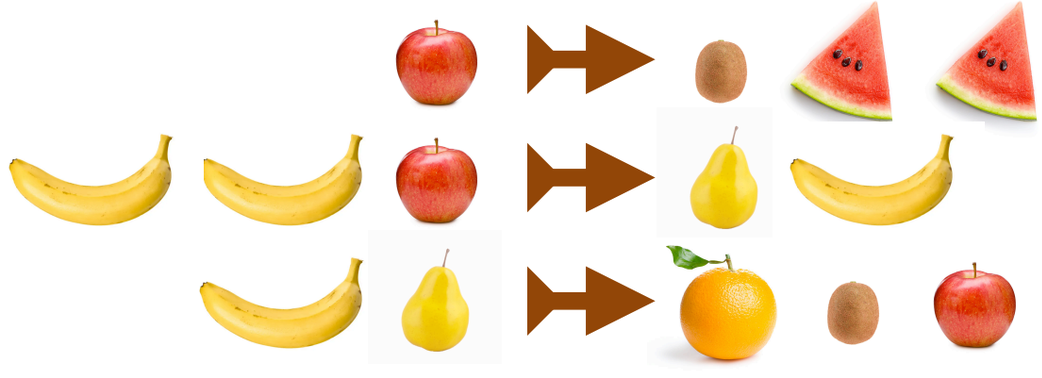
dimension 1
NP-complete
[Haase, Kreutzer, Ouaknine, Joel 2009]

integer reachability
NP-complete

I. Reachability in Petri nets

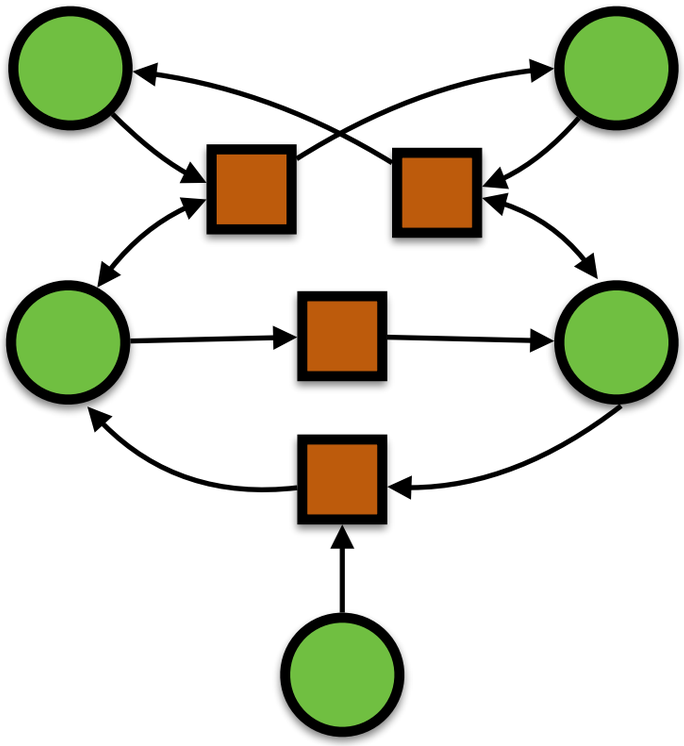
II. Extensions of Petri nets

- **pushdown stack**
- **branching**
- **colors**
- **arbitrary step relations**



Pushdown VASS

- Petri nets:



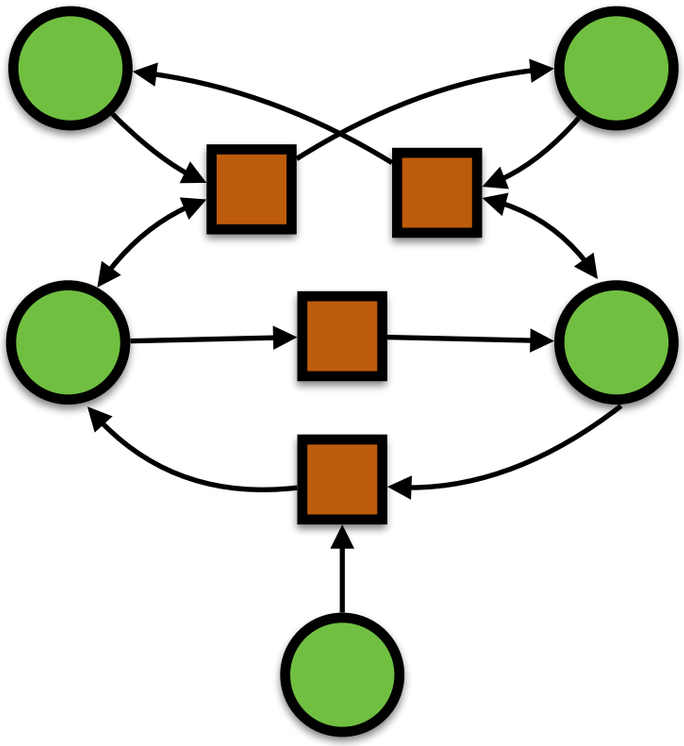
- counter programs **without zero-tests**:

```
1: loop
2:     loop
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):

Pushdown VASS

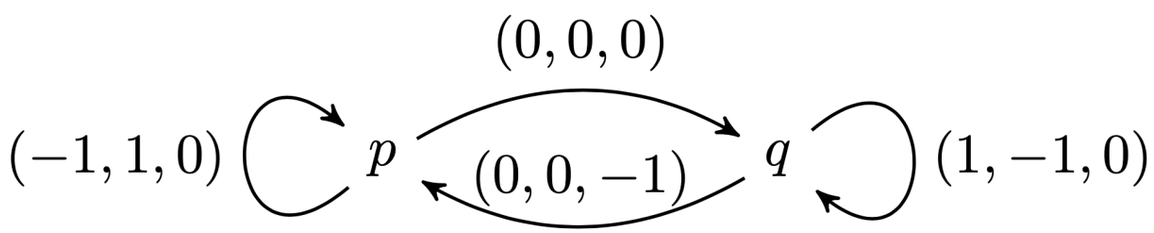
- Petri nets:



- counter programs **without zero-tests**:

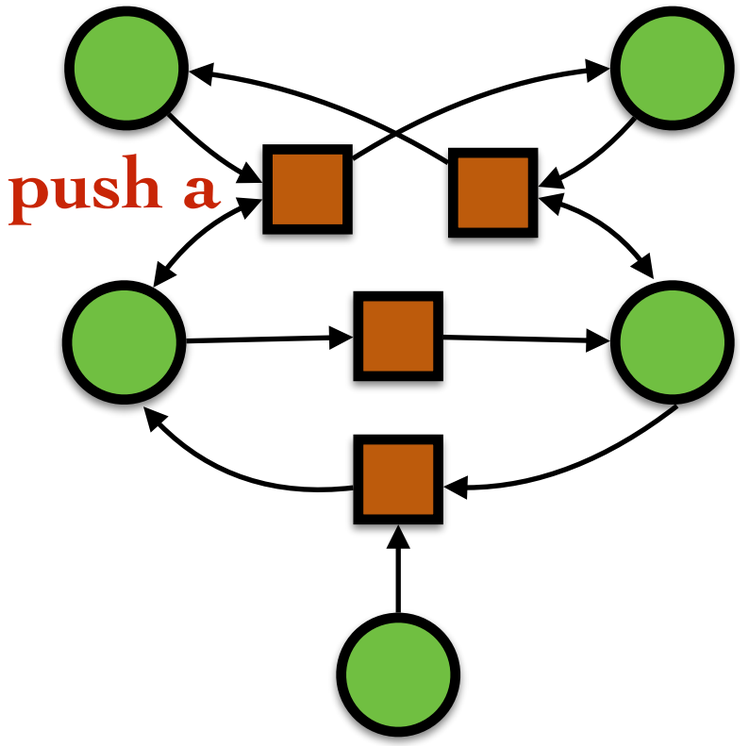
```
1: loop
2:     loop
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):



Pushdown VASS

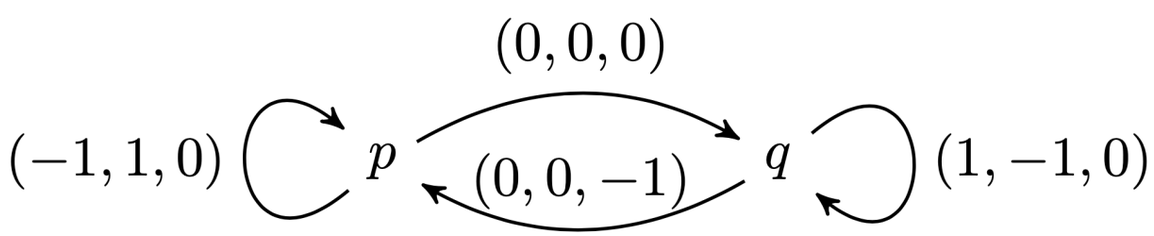
- Petri nets:



- counter programs **without zero-tests**:

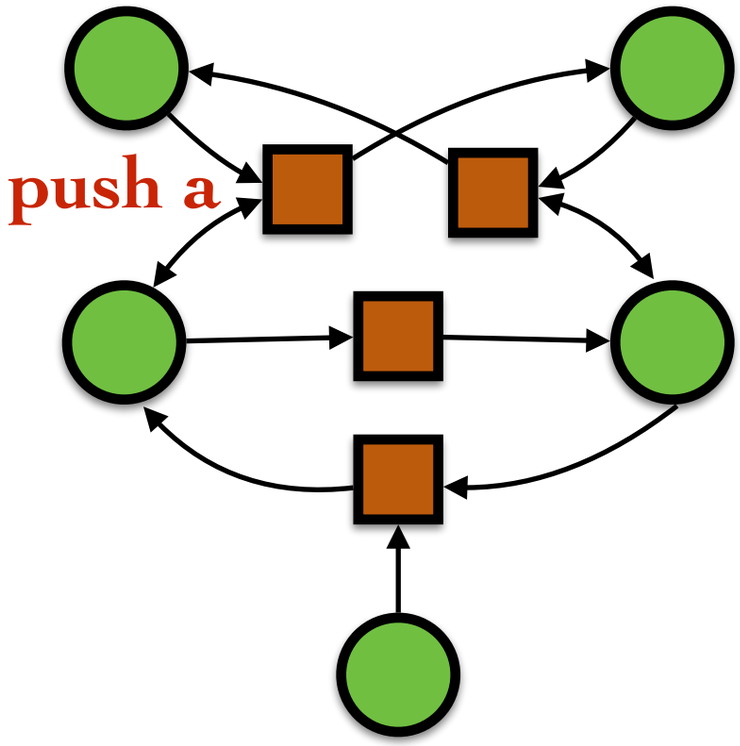
```
1: loop
2:     loop
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):



Pushdown VASS

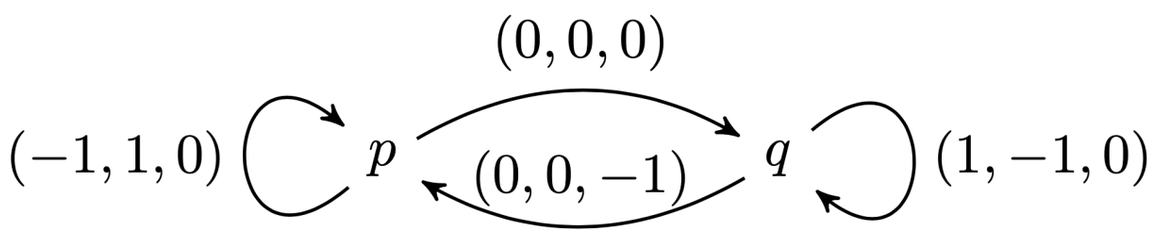
- Petri nets:



- counter programs **without zero-tests**:

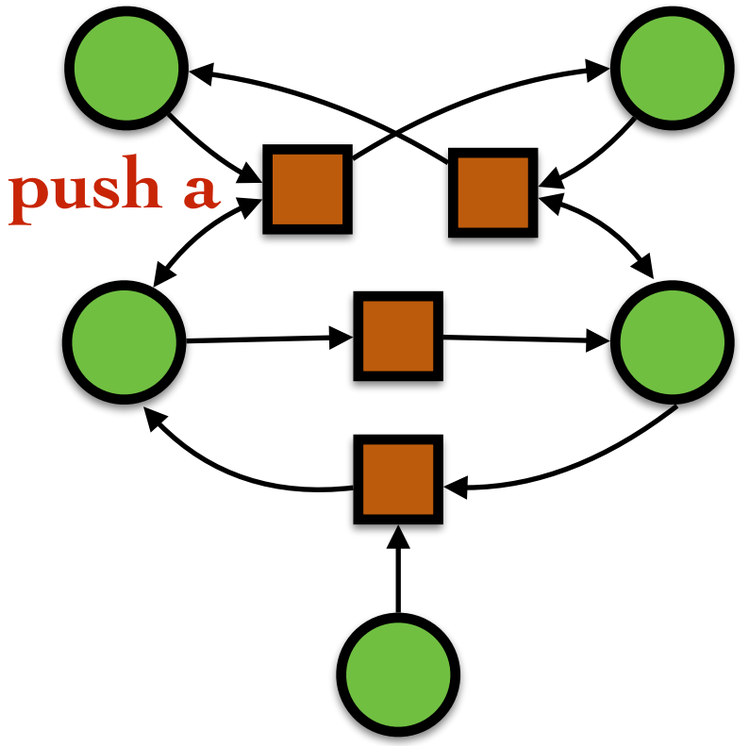
```
1: loop
2:     loop  push a
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):



Pushdown VASS

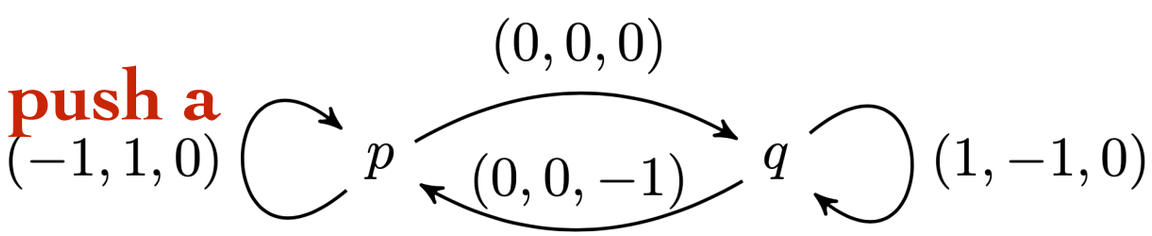
- Petri nets:



- counter programs **without zero-tests**:

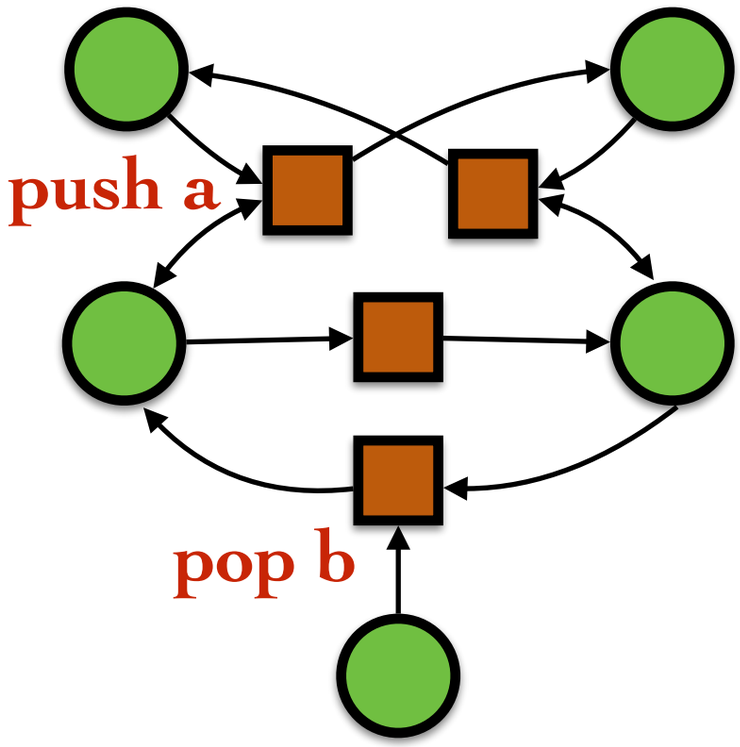
```
1: loop
2:     loop  push a
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):



Pushdown VASS

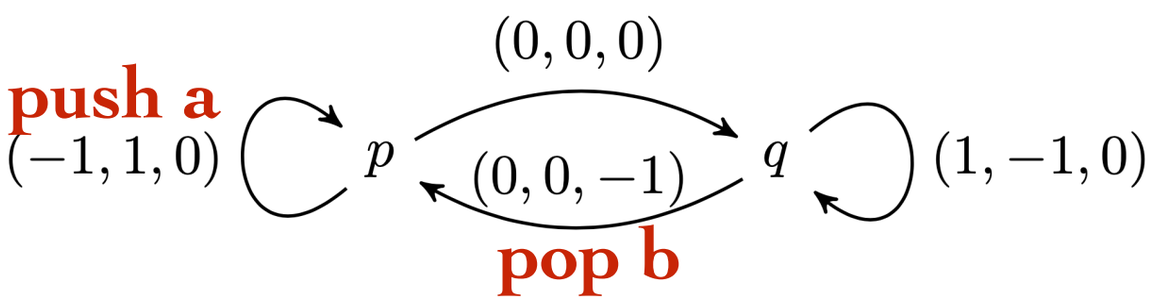
- Petri nets:



- counter programs **without zero-tests**:

```
1: loop
2:     loop  push a
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1  pop b
```

- vector addition systems with states (VASS):



Pushdown VASS reachability

Equivalent formulations:

Pushdown VASS reachability

Equivalent formulations:

- VASS reachability controlled by a context-free language

Pushdown VASS reachability

Equivalent formulations:

- VASS reachability controlled by a context-free language
- Context-free language $\cap$ VASS language $\neq \emptyset$

Pushdown VASS reachability

Equivalent formulations:

- VASS reachability controlled by a context-free language
- Context-free language $\cap$ VASS language $\neq \emptyset$
- Context-free language $\cap$ Commutative context-free language $\neq \emptyset$

VASS

reachability
Ackermann=F_ω-complete

dimension d
?

dimension 3
2-EXPSPACE
[Czerwiński, Jecker, L., Orlikowski 2025]

bi-reachability
EXPSPACE-complete
[Leroux 2013]

coverability
EXPSPACE-complete

reversible VASS
EXPSPACE-complete
[Mayer, Meyer 1982]
[Cardoza, Lipton, Meyer 1976]

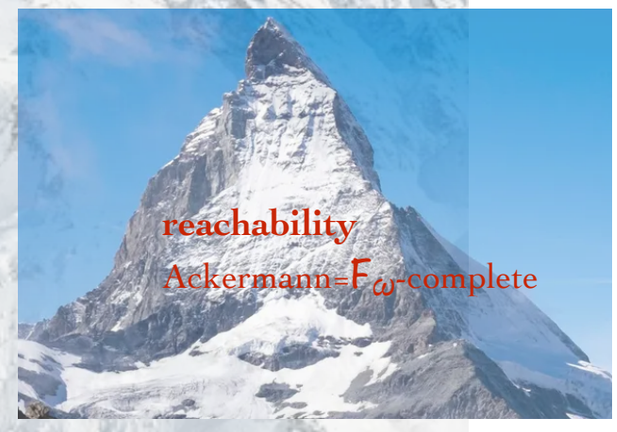
dimension 2
PSPACE-complete
[Blondin, Finkel, Goeller, Haase, McKenzie 2015]

dimension 1
NP-complete
[Haase, Kreutzer, Ouaknine, Joel 2009]

integer reachability
NP-complete

Pushdown VASS

reachability



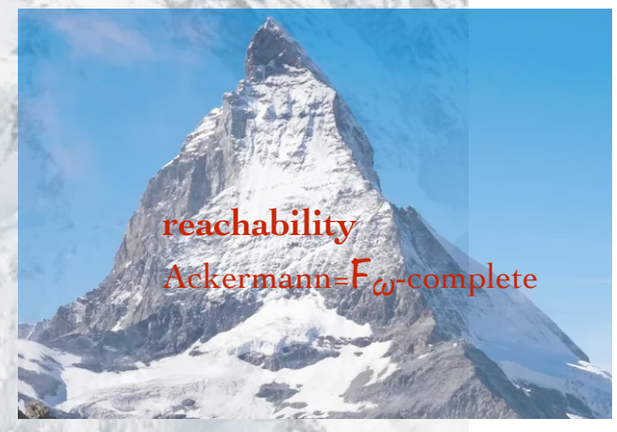
coverability - dimension 1
EXPSPACE
[Leroux, Sutre, Totzke 2015]

integer reachability
NP-complete
[Hague, Widjaja Lin 2011]

Pushdown VASS

reachability

finite-index PVASS
decidable
[Fauozi, Ganty 2011]



coverability - dimension 1
EXPSPACE
[Leroux, Sutre, Totzke 2015]

integer reachability
NP-complete
[Hague, Widjaja Lin 2011]

Pushdown VASS

reachability

VASS with nested 0-tests ⟷ **finite-index PVASS**

F_ω-complete

[Reinhardt 2008]

[Bonnet 2011]

[Czerwiński, L, Orlikowski 2025]

decidable

[Fauozi, Ganty 2011]

coverability - dimension 1

EXPSPACE

[Leroux, Sutre, Totzke 2015]

integer reachability

NP-complete

[Hague, Widjaja Lin 2011]

Pushdown VASS

reachability

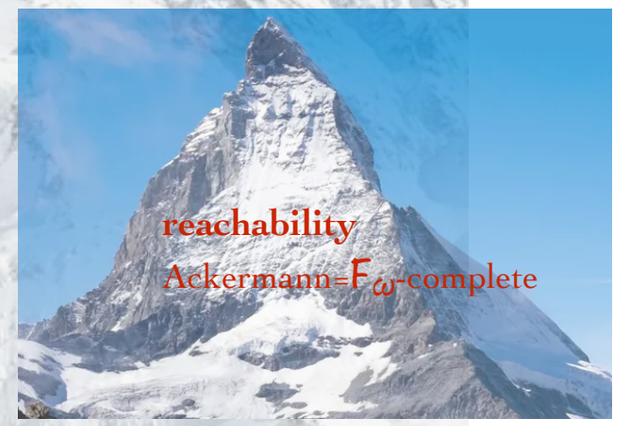
VASS with nested 0-tests ⟷ **finite-index PVASS** ⟷ **reversible PVASS**

F_ω-complete — decidable — F_ω

[Reinhardt 2008] [Fauozi, Ganty 2011] [Ganardi, Majumdar, Zetzsche 2022]

[Bonnet 2011]

[Czerwiński, L, Orlikowski 2025]



coverability - dimension 1

EXPSPACE

[Leroux, Sutre, Totzke 2015]

integer reachability

NP-complete

[Hague, Widjaja Lin 2011]

Pushdown VASS

reachability

VASS with nested 0-tests ⟷ **finite-index PVASS**

F_ω-complete

[Reinhardt 2008]

[Bonnet 2011]

[Czerwiński, L, Orlikowski 2025]

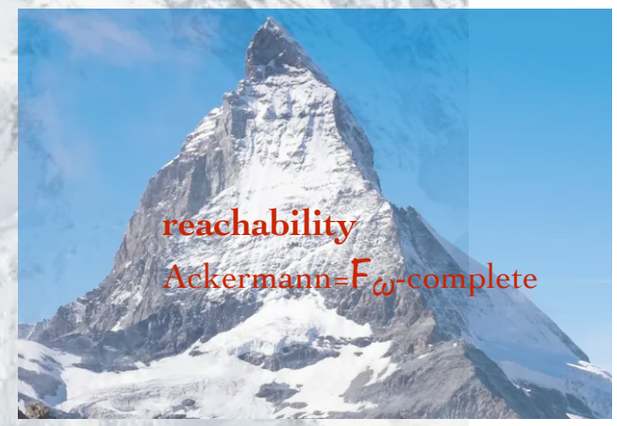
decidable

[Fauozi, Ganty 2011]

reversible PVASS

F_ω

[Ganardi, Majumdar, Zetzsche 2022]



reachability - dimension 1

F_ω

[Biziere, Czerwiński 2025]

coverability - dimension 1

EXPSPACE

[Leroux, Sutre, Totzke 2015]

integer reachability

NP-complete

[Hague, Widjaja Lin 2011]

Pushdown VASS

reachability

VASS with nested 0-tests ⟷ **finite-index PVASS** ⟷ **reversible PVASS**

F_ω-complete

[Reinhardt 2008]
[Bonnet 2011]
[Czerwiński, L, Orlikowski 2025]

decidable

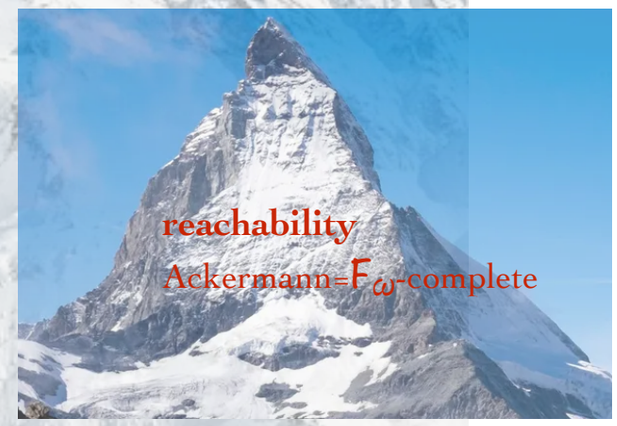
[Fauozi, Ganty 2011]

F_ω

[Ganardi, Majumdar, Zetzsche 2022]

coverability - dimension 2

?



reachability - dimension 1

F_ω

[Biziere, Czerwiński 2025]

coverability - dimension 1

EXPSPACE

[Leroux, Sutre, Totzke 2015]

integer reachability

NP-complete

[Hague, Widjaja Lin 2011]

Pushdown VASS

coverability ⟷ reachability

VASS with nested 0-tests ⟷ **finite-index PVASS** ⟷ **reversible PVASS**

F_ω-complete — decidable — F_ω

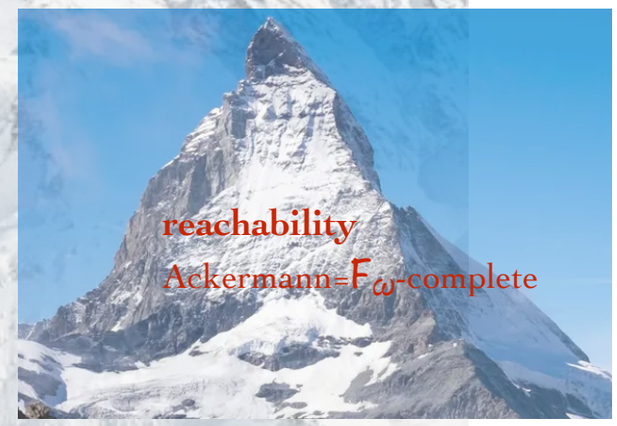
[Reinhardt 2008] — [Fauozi, Ganty 2011] — [Ganardi, Majumdar, Zetzsche 2022]

[Bonnet 2011]

[Czerwiński, L, Orlikowski 2025]

coverability - dimension 2

?



reachability - dimension 1

F_ω

[Biziere, Czerwiński 2025]

coverability - dimension 1

EXPSPACE

[Leroux, Sutre, Totzke 2015]

integer reachability

NP-complete

[Hague, Widjaja Lin 2011]

Pushdown VASS

coverability ⟷ reachability

HyperAckermann F_{ω^ω}

[Guttenberg, Keskin, Meyer 2025]

VASS with nested 0-tests ⟷ **finite-index PVASS** | **reversible PVASS**

F_ω-complete | decidable | F_ω

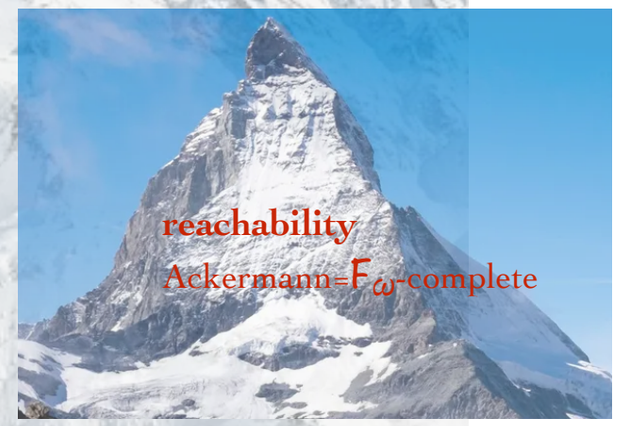
[Reinhardt 2008] | [Fauozi, Ganty 2011] | [Ganardi, Majumdar, Zetzsche 2022]

[Bonnet 2011]

[Czerwiński, L, Orlikowski 2025]

coverability - dimension 2

?



reachability - dimension 1

F_ω

[Biziere, Czerwiński 2025]

coverability - dimension 1

EXPSPACE

[Leroux, Sutre, Totzke 2015]

integer reachability

NP-complete

[Hague, Widjaja Lin 2011]

Pushdown VASS

coverability ⟷ reachability

HyperAckermann F_{ω^ω}

[Guttenberg, Keskin, Meyer 2025]

VASS with nested 0-tests ⟷ **finite-index PVASS** ⟷ **reversible PVASS**

F_ω-complete

[Reinhardt 2008]

[Bonnet 2011]

[Czerwiński, L, Orlikowski 2025]

decidable

[Fauozi, Ganty 2011]

F_ω

[Ganardi, Majumdar, Zetzsche 2022]

coverability - dimension 2

?

reachability - dimension 1

F_ω

[Biziere, Czerwiński 2025]

coverability - dimension 1

EXPSPACE

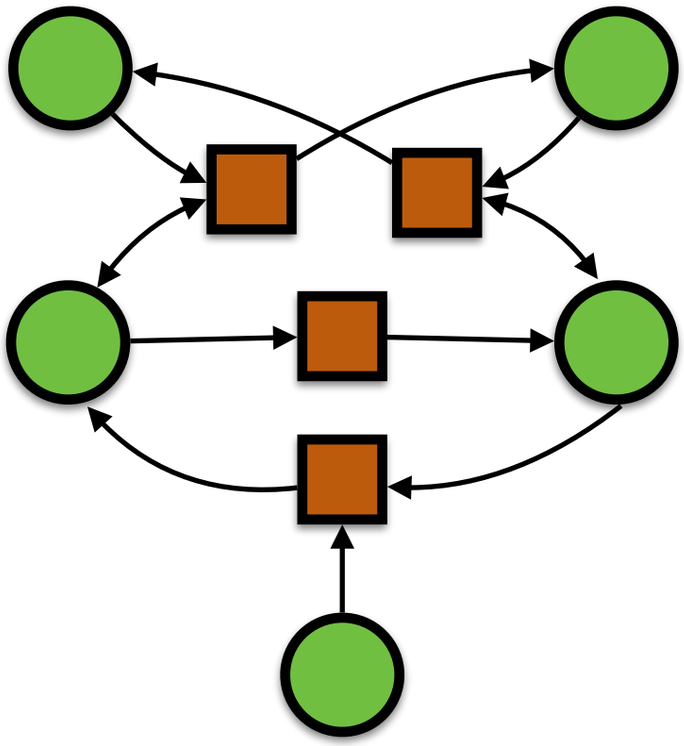
[Leroux, Sutre, Totzke 2015]

integer reachability

NP-complete

[Hague, Widjaja Lin 2011]

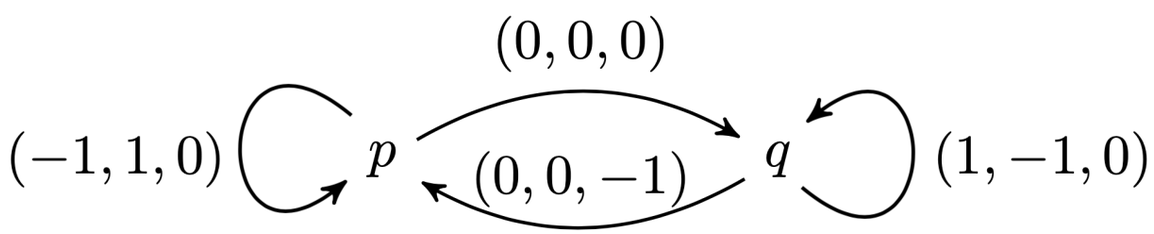
Branching VASS



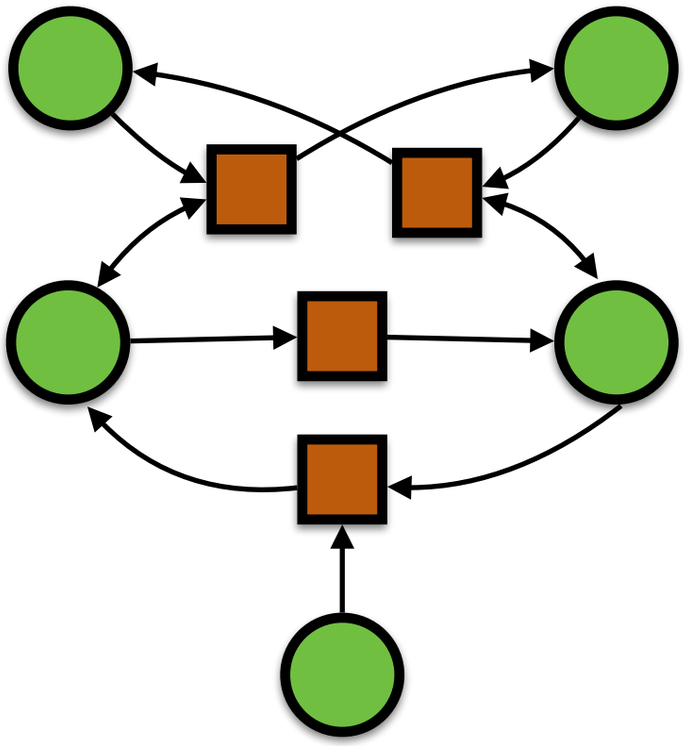
- counter programs **without zero-tests**:

```
1: loop
2:     loop
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):



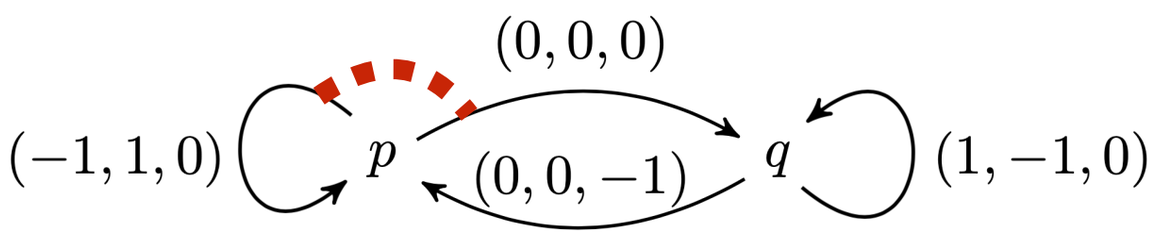
Branching VASS



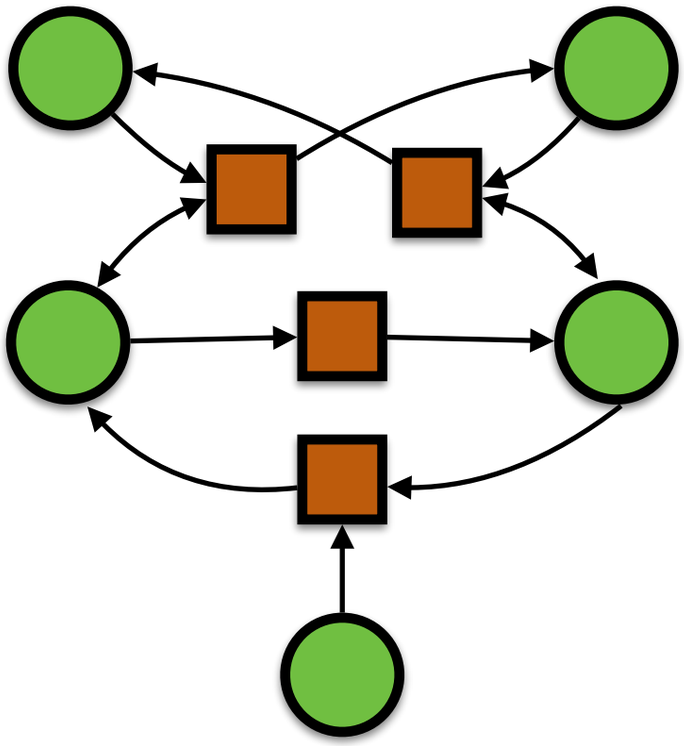
- counter programs **without zero-tests**:

```
1: loop
2:     loop
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):



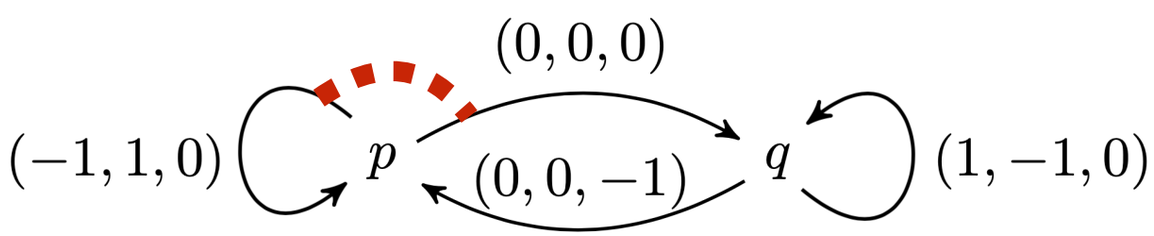
Branching VASS



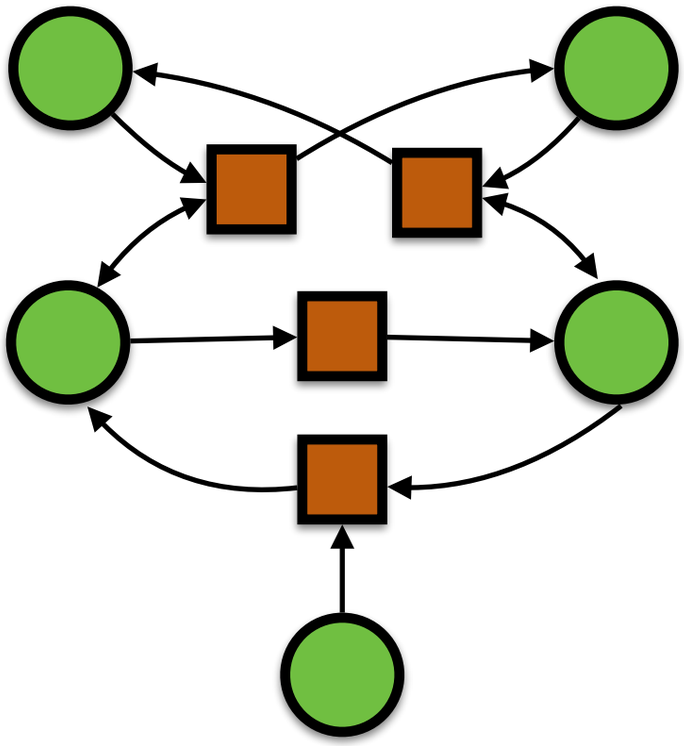
- counter programs **without zero-tests**:

```
1: loop fork 2 4
2:     loop
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):



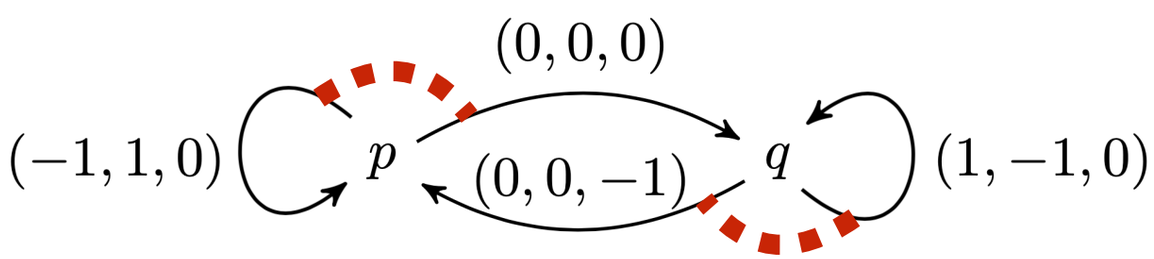
Branching VASS



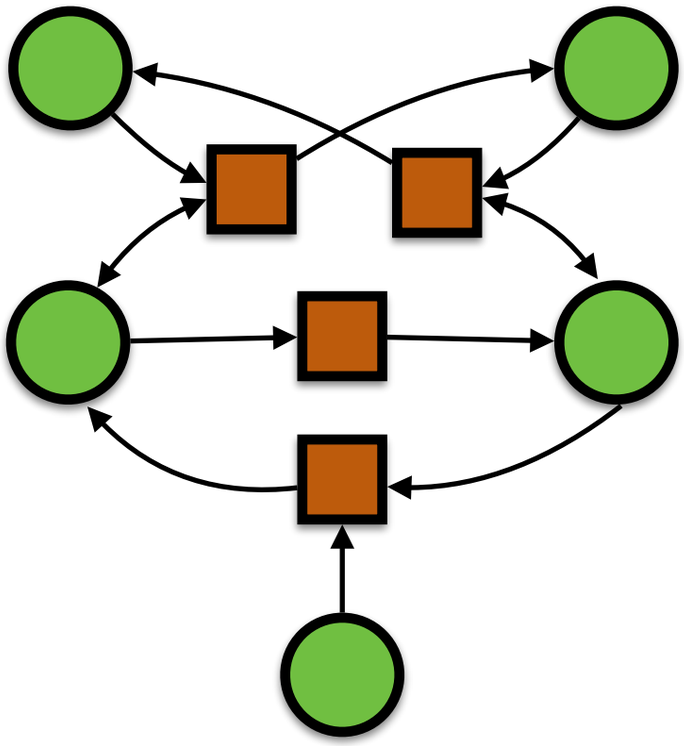
- counter programs **without zero-tests**:

```
1: loop fork 2 4
2:     loop
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):



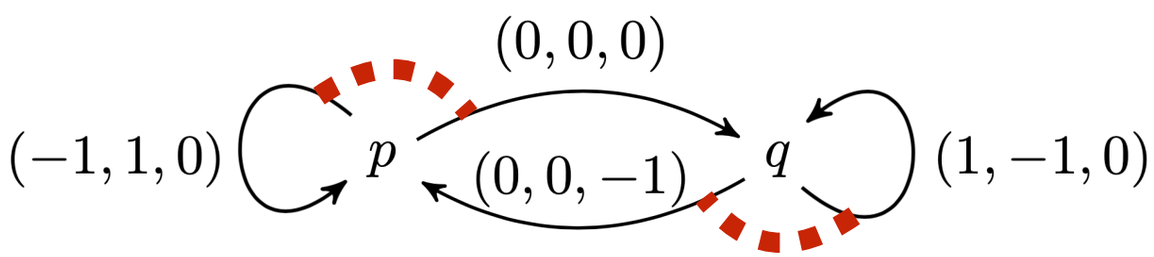
Branching VASS



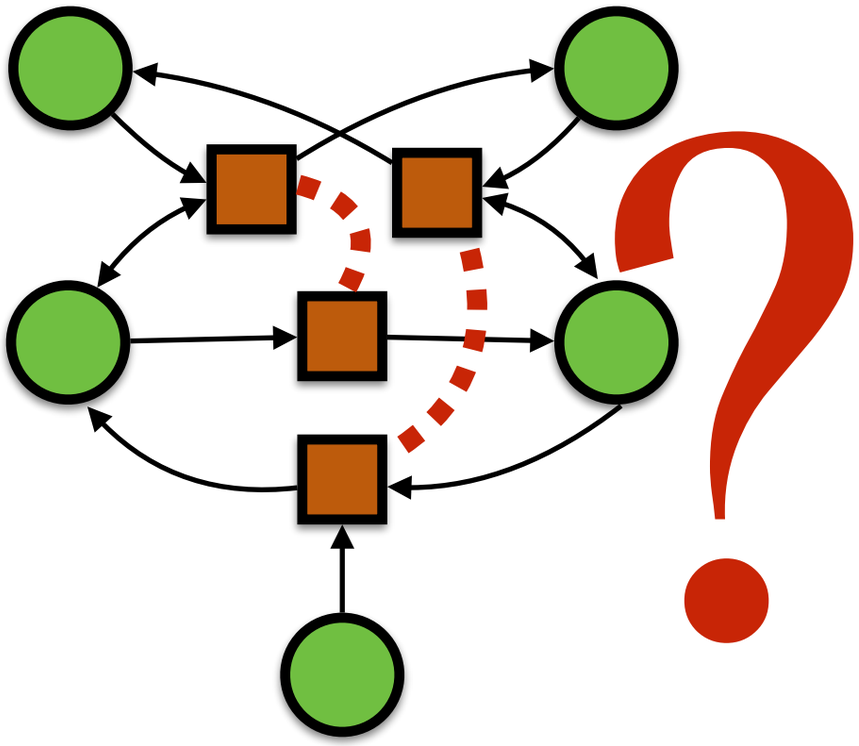
- counter programs **without zero-tests**:

```
1: loop fork 2 4
2:     loop
3:         x −= 1    y += 1
       fork 2 4
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):



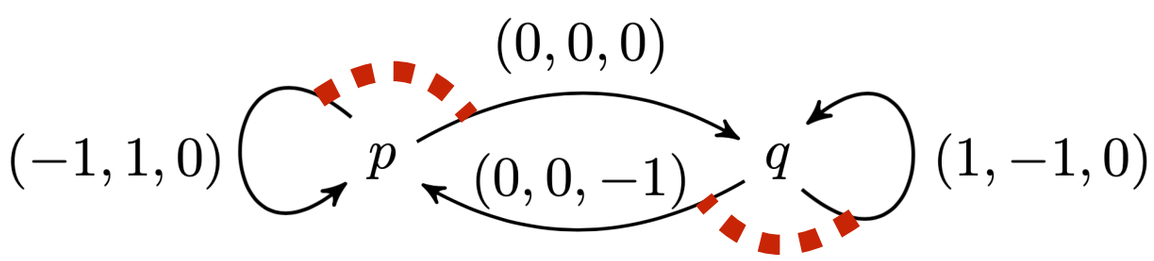
Branching VASS



- counter programs **without zero-tests**:

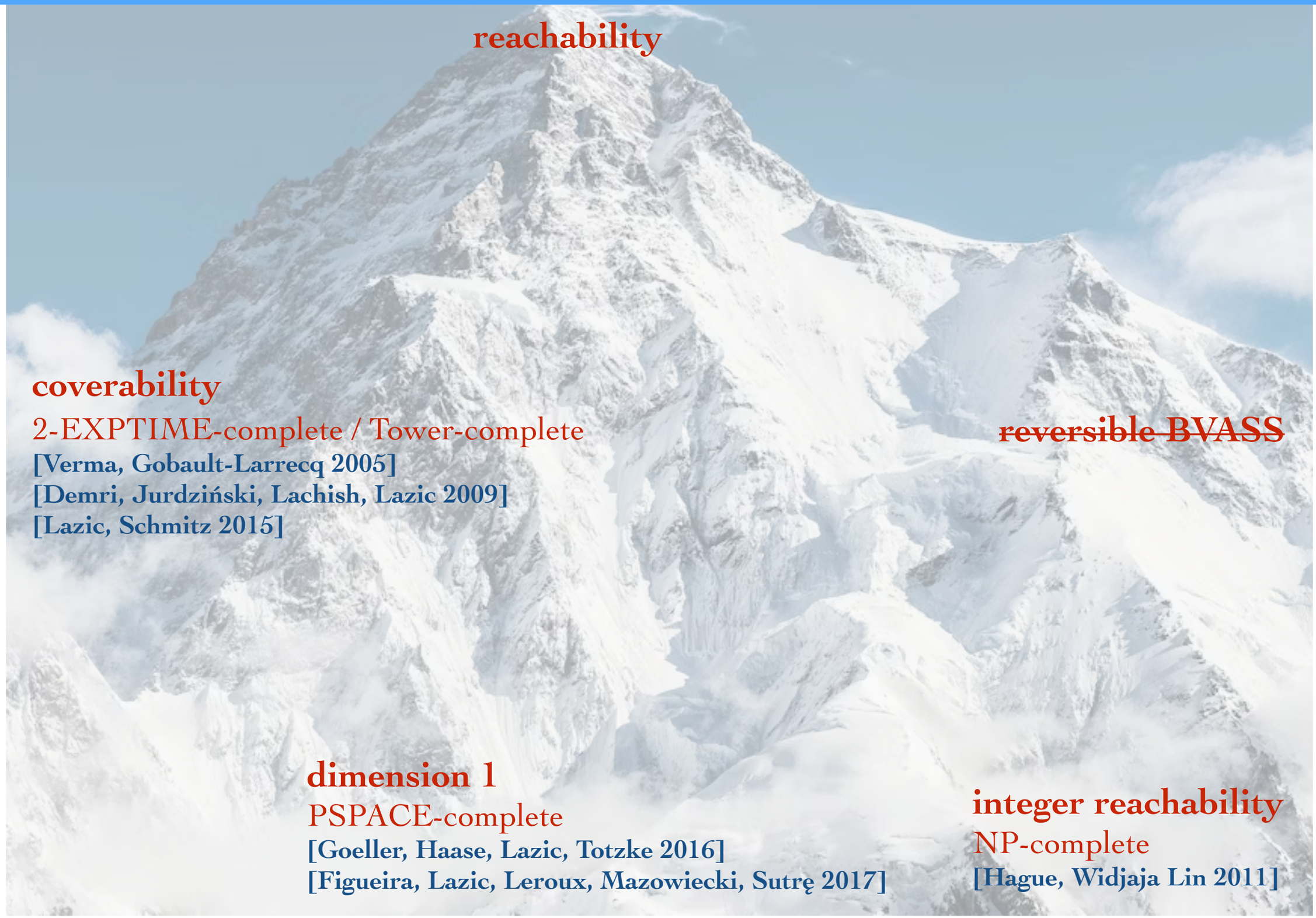
```
1: loop fork 2 4
2:     loop
3:         x −= 1    y += 1
       fork 2 4
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

- vector addition systems with states (VASS):

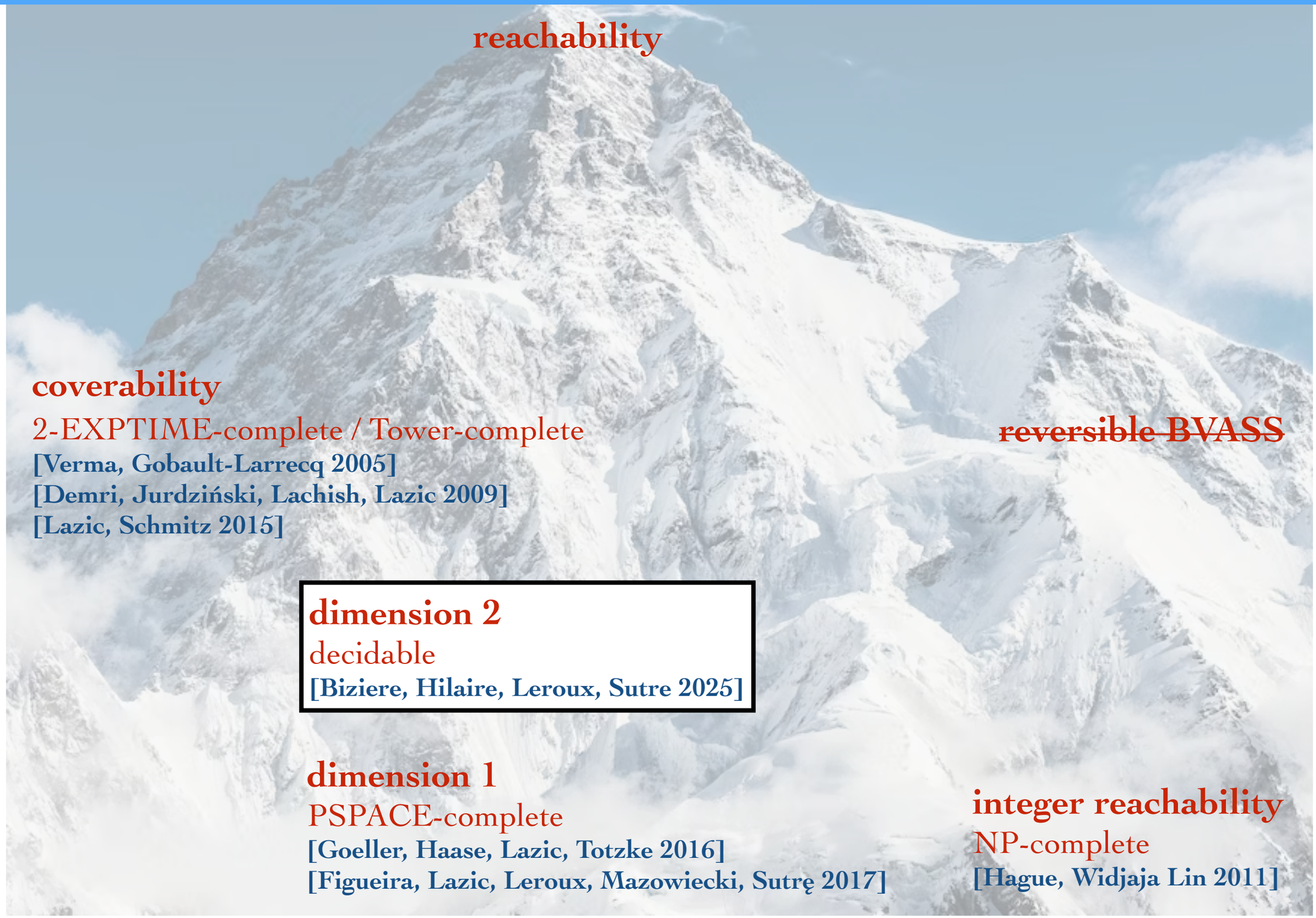


Branching VASS

Branching VASS



Branching VASS



reachability

coverability

2-EXPTIME-complete / Tower-complete

[Verma, Gobault-Larrecq 2005]

[Demri, Jurdziński, Lachish, Lazic 2009]

[Lazic, Schmitz 2015]

~~reversible BVASS~~

dimension 2

decidable

[Biziere, Hilaire, Leroux, Sutre 2025]

dimension 1

PSPACE-complete

[Goeller, Haase, Lazic, Totzke 2016]

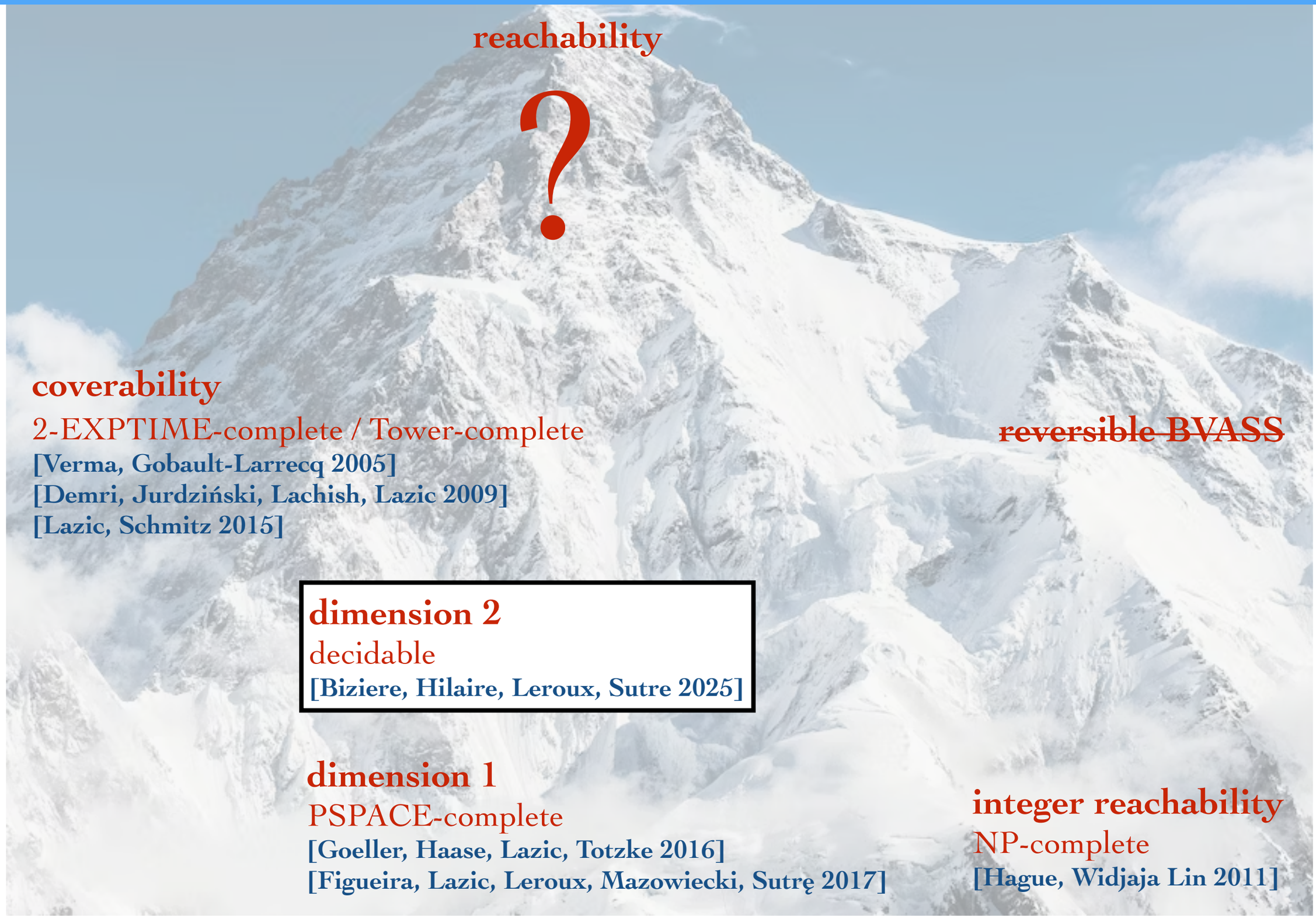
[Figueira, Lazic, Leroux, Mazowiecki, Sutrę 2017]

integer reachability

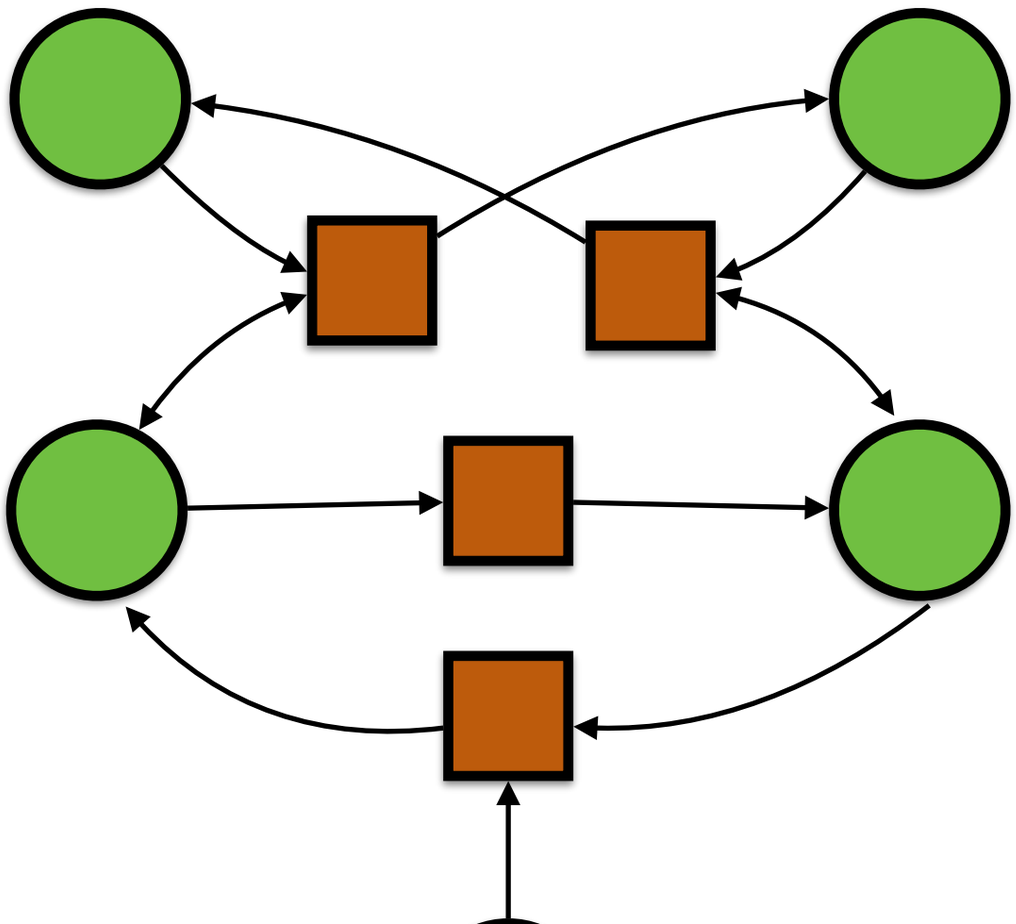
NP-complete

[Hague, Widjaja Lin 2011]

Branching VASS



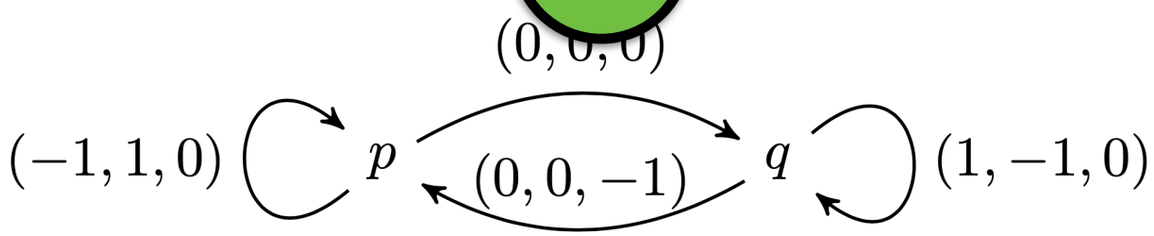
Coloured PN



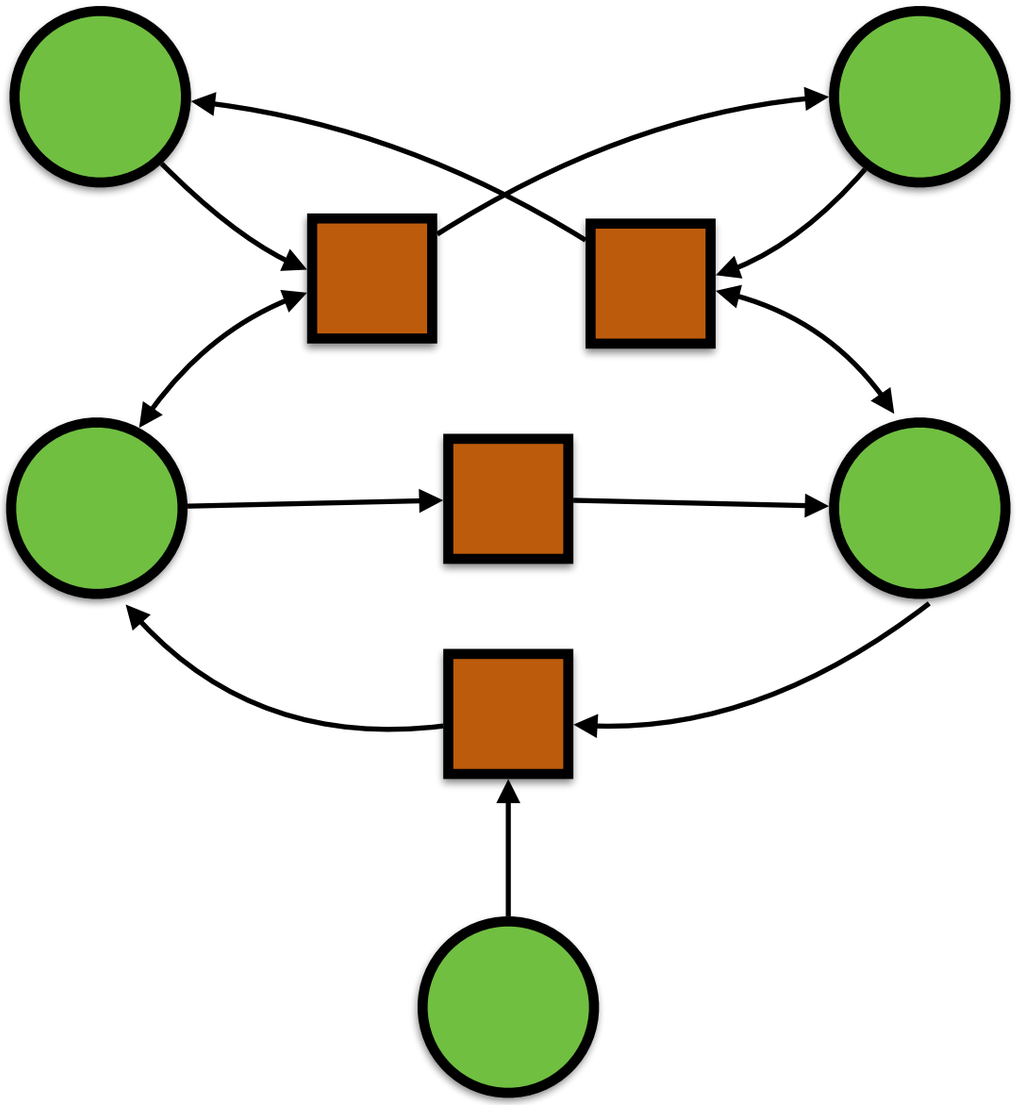
- counter programs **without zero-tests**:

```
1: loop
2:     loop
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

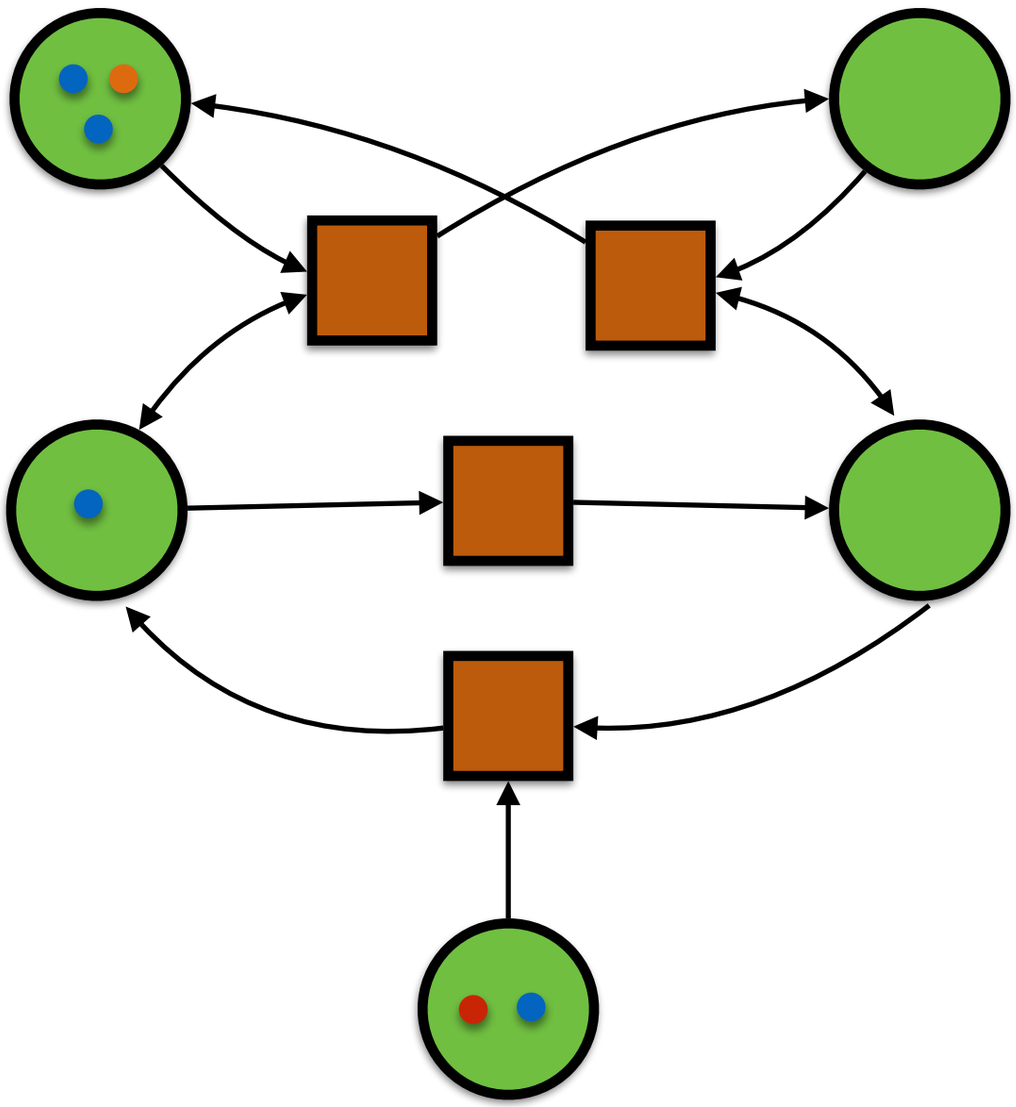
- vector addition sys[illegible]ith states (VASS):



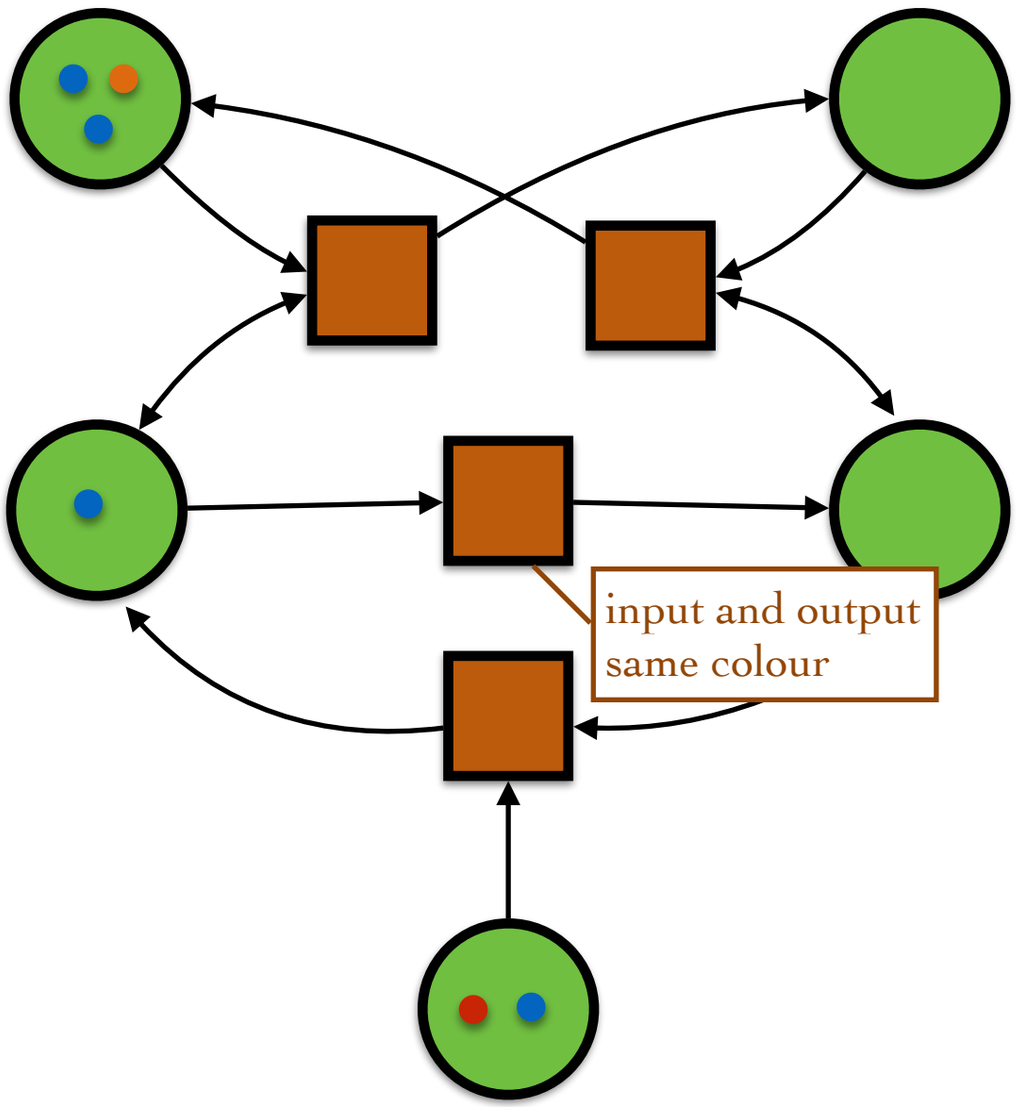
Coloured PN



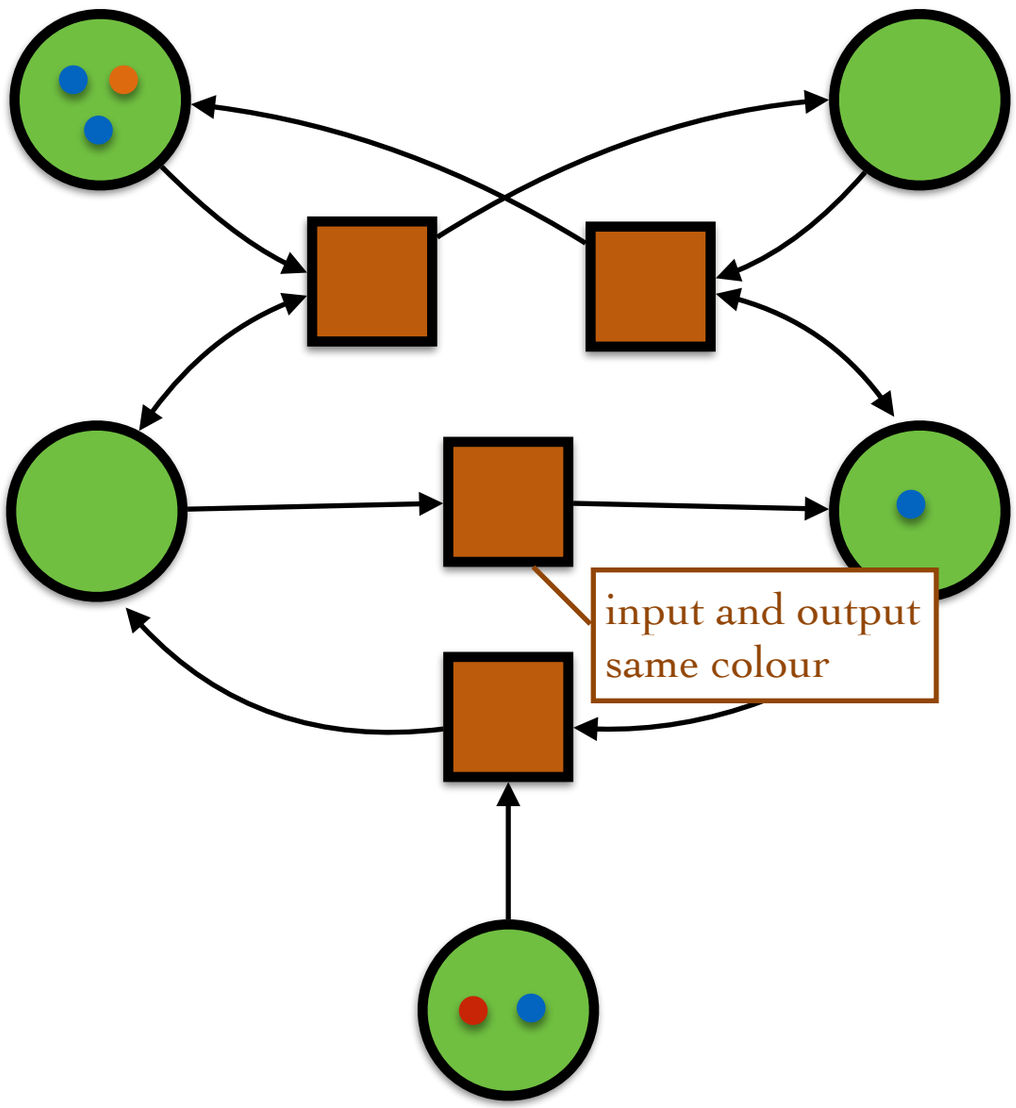
Coloured PN



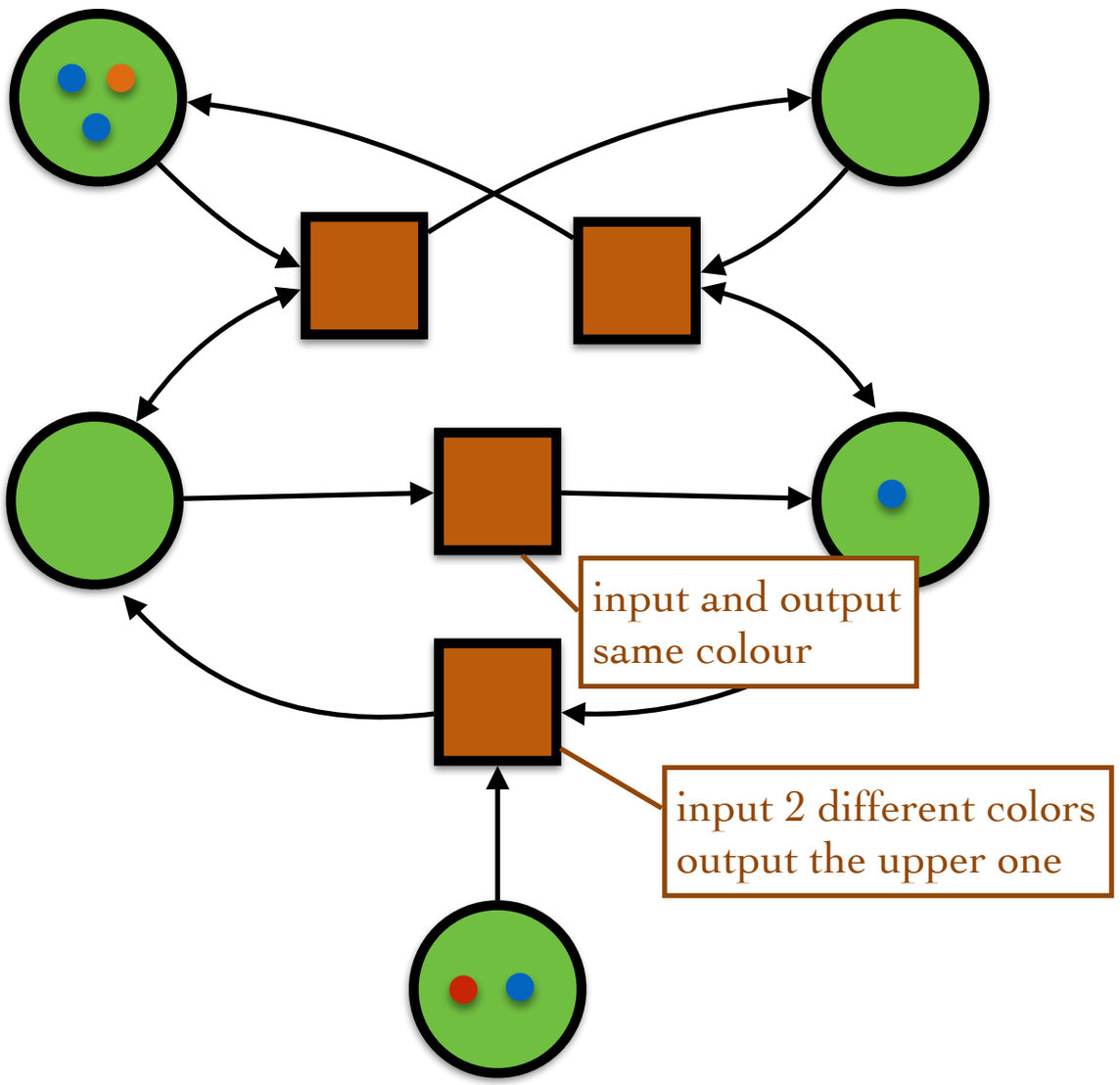
Coloured PN



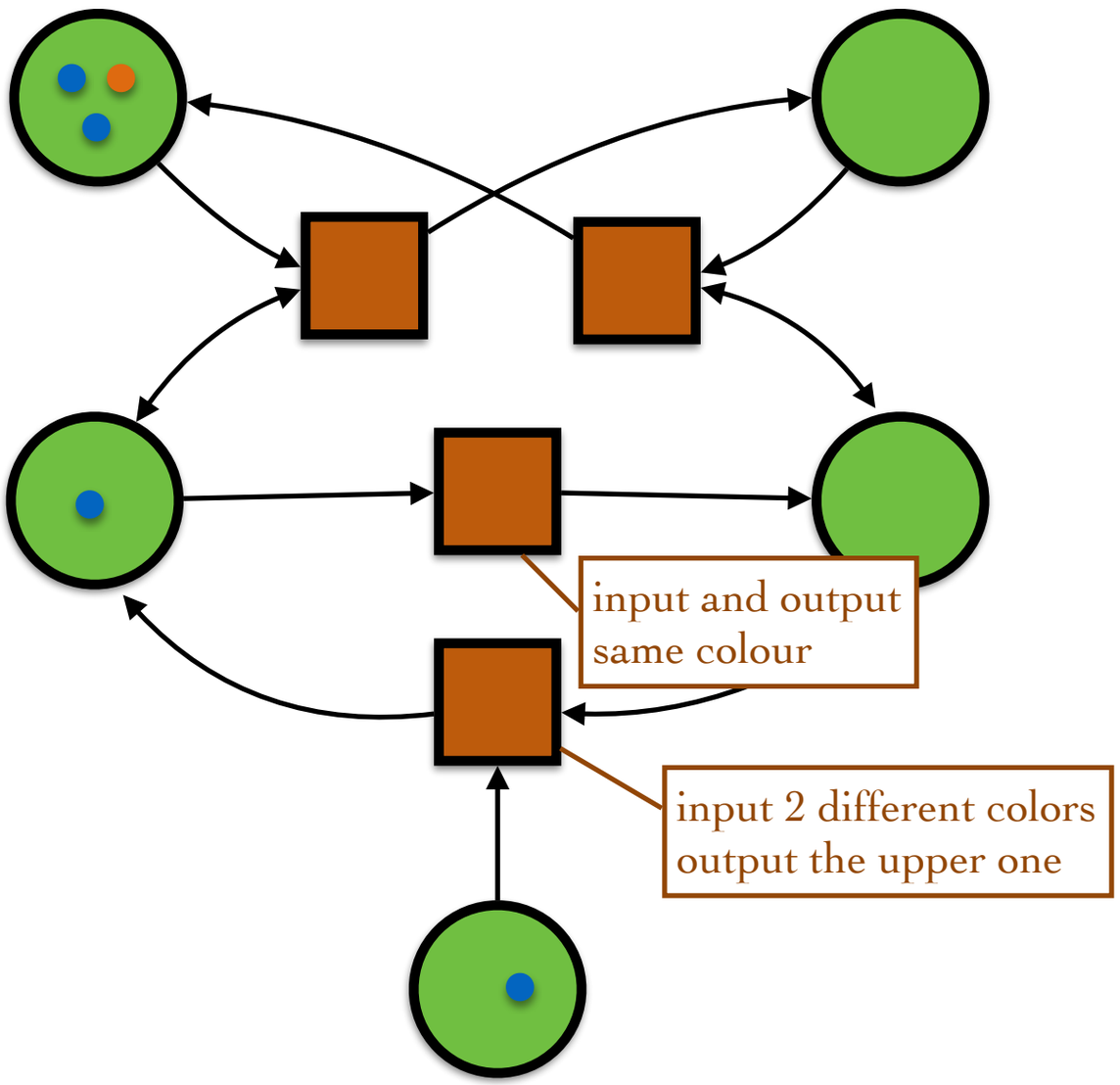
Coloured PN



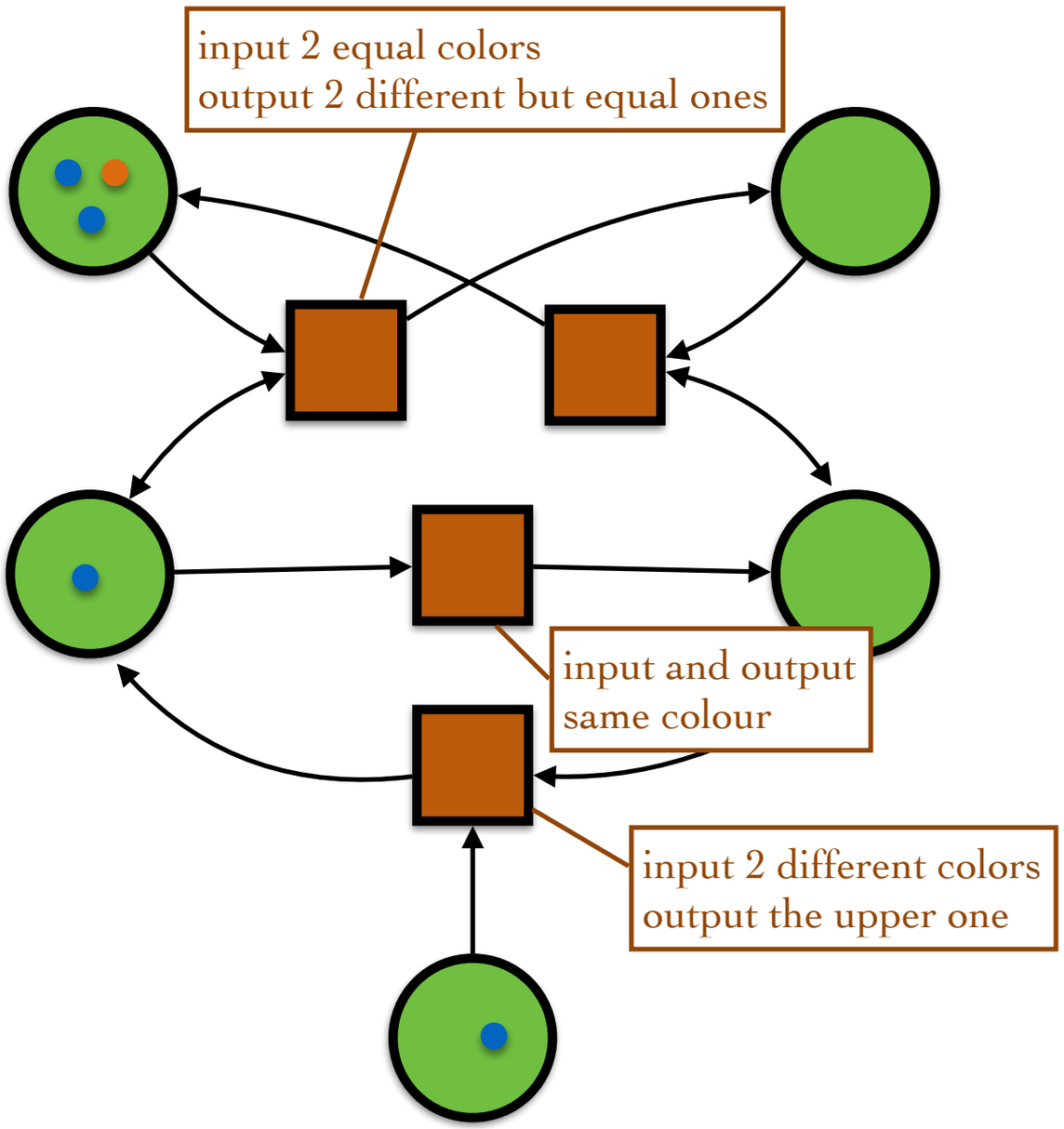
Coloured PN


input and output
same colour
input 2 different colors
output the upper one

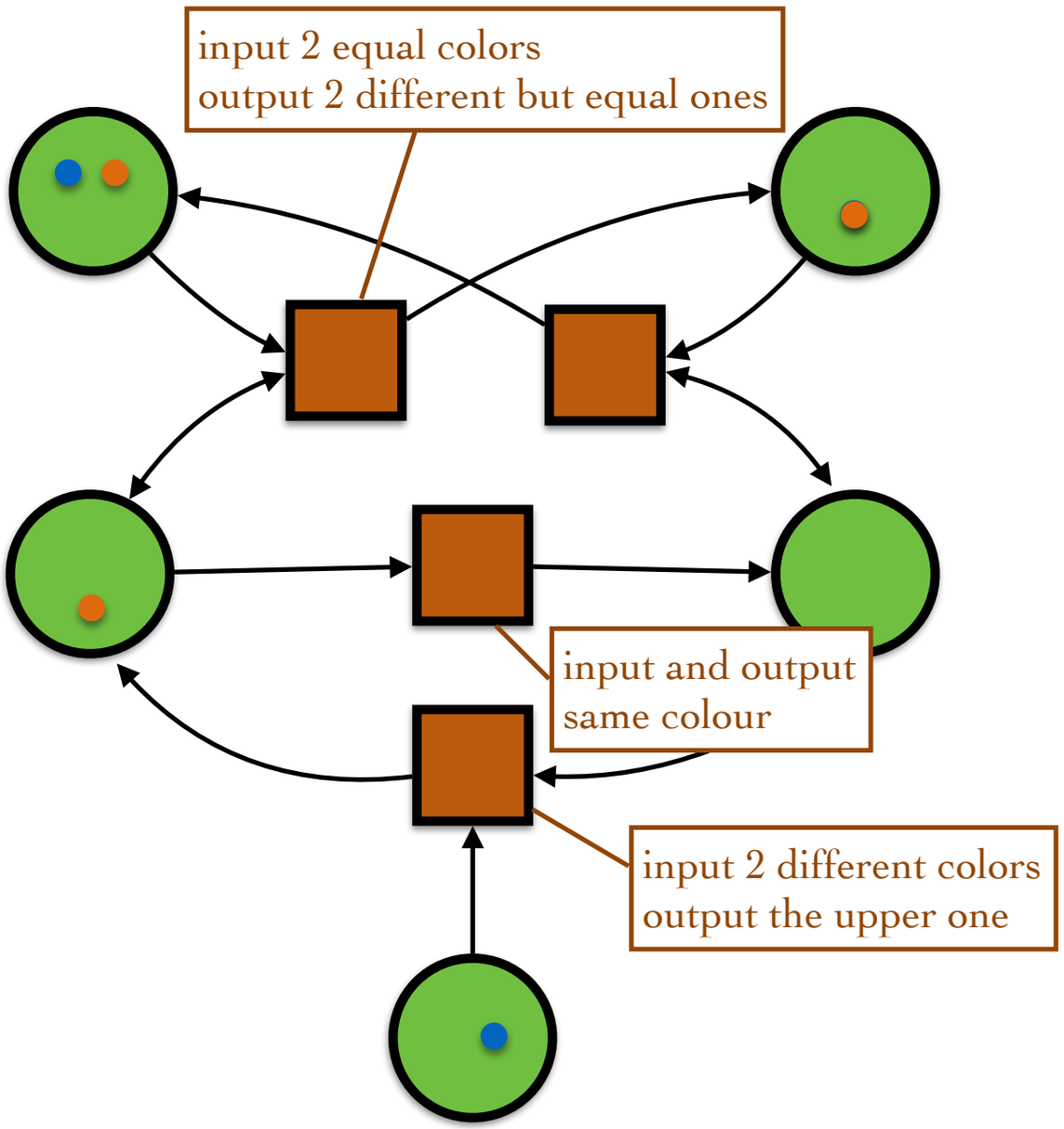
Coloured PN


input and output
same colour
input 2 different colors
output the upper one

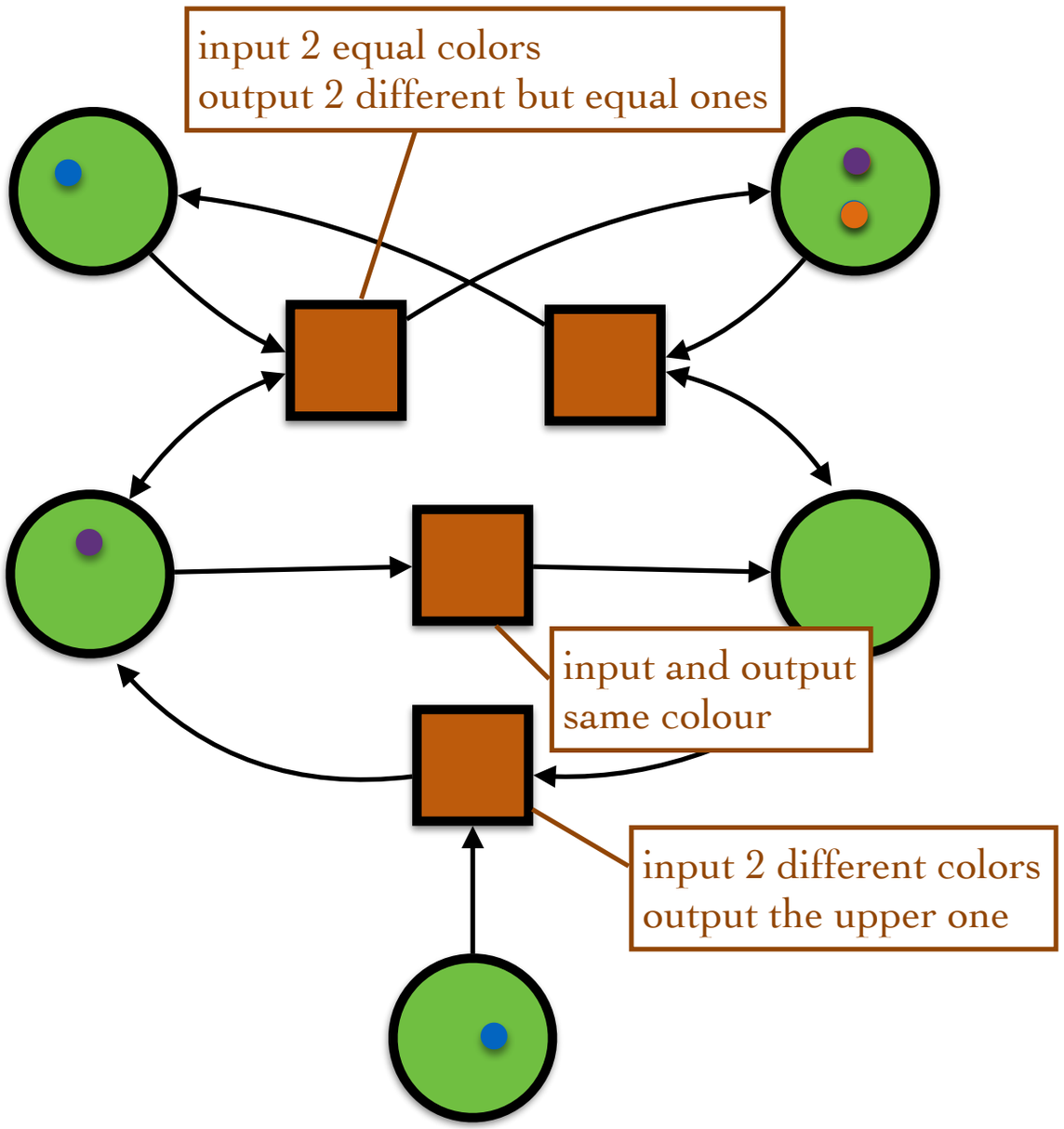
Coloured PN


input 2 equal colors
output 2 different but equal ones
input and output
same colour
input 2 different colors
output the upper one

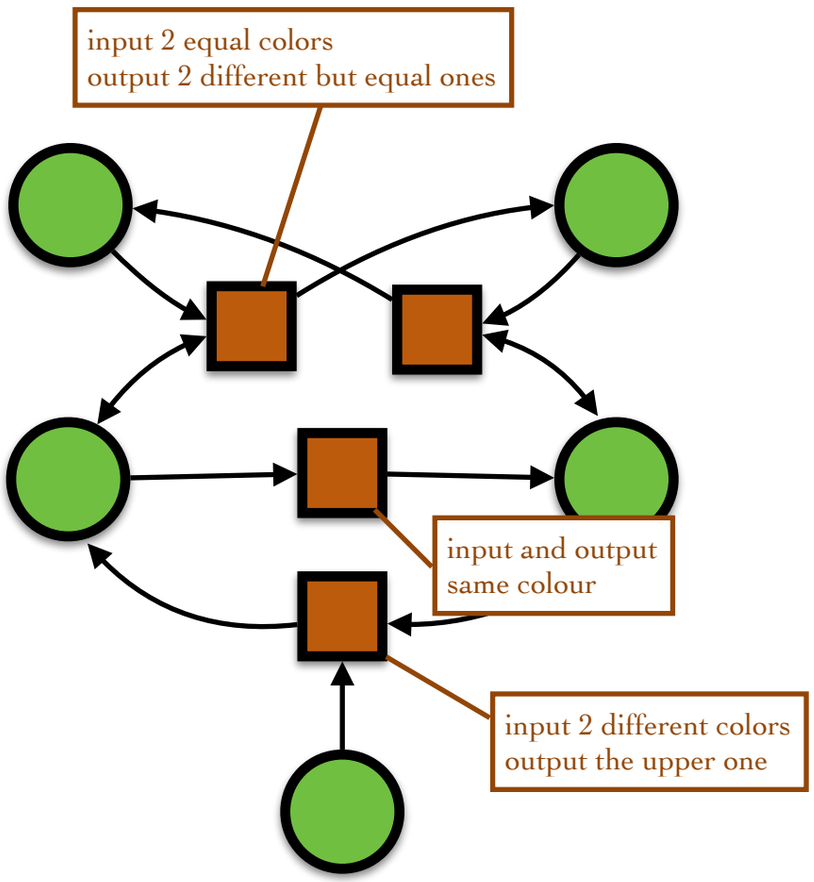
Coloured PN


input 2 equal colors
output 2 different but equal ones
input and output
same colour
input 2 different colors
output the upper one

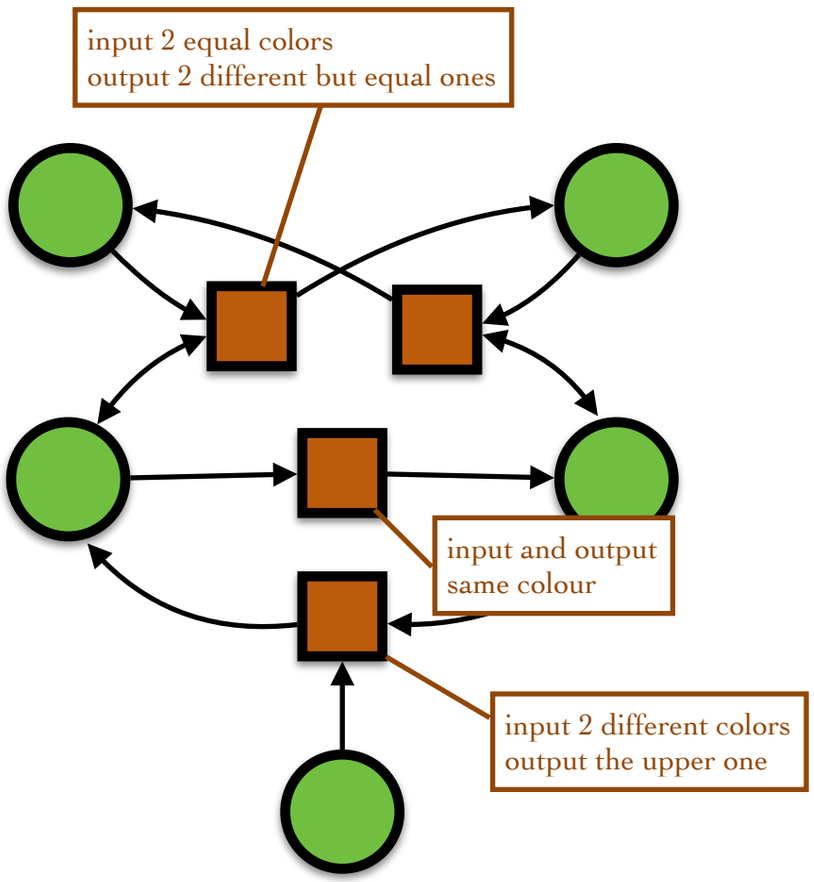
Coloured PN


input 2 equal colors
output 2 different but equal ones
input and output
same colour
input 2 different colors
output the upper one

Coloured PN


input 2 equal colors
output 2 different but equal ones
input and output
same colour
input 2 different colors
output the upper one

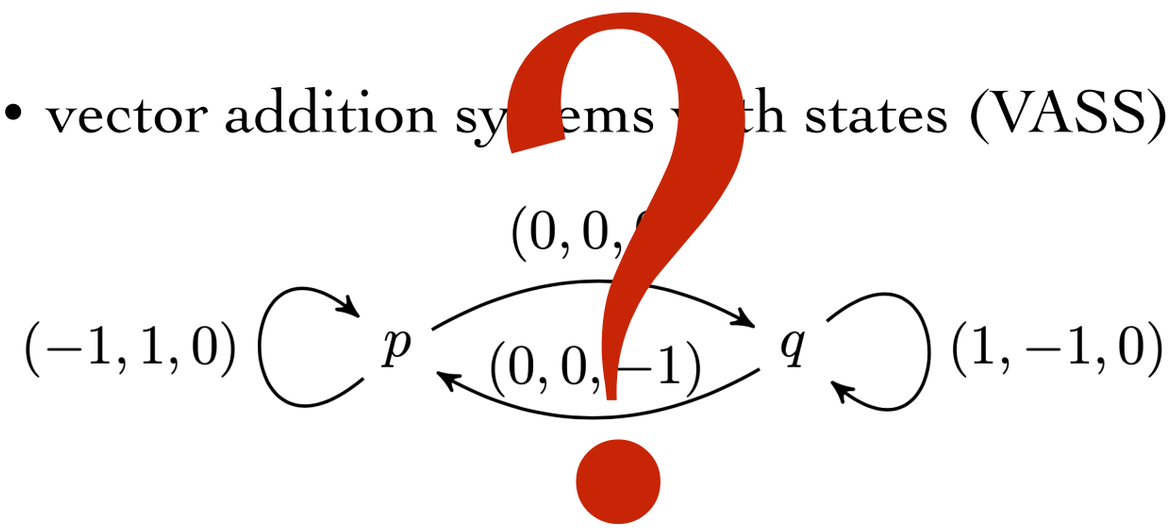
Coloured PN



- counter programs without zero-tests:

```
1: loop
2:     loop
3:         x −= 1    y += 1
4:     loop
5:         x += 1    y −= 1
6:     z −= 1
```

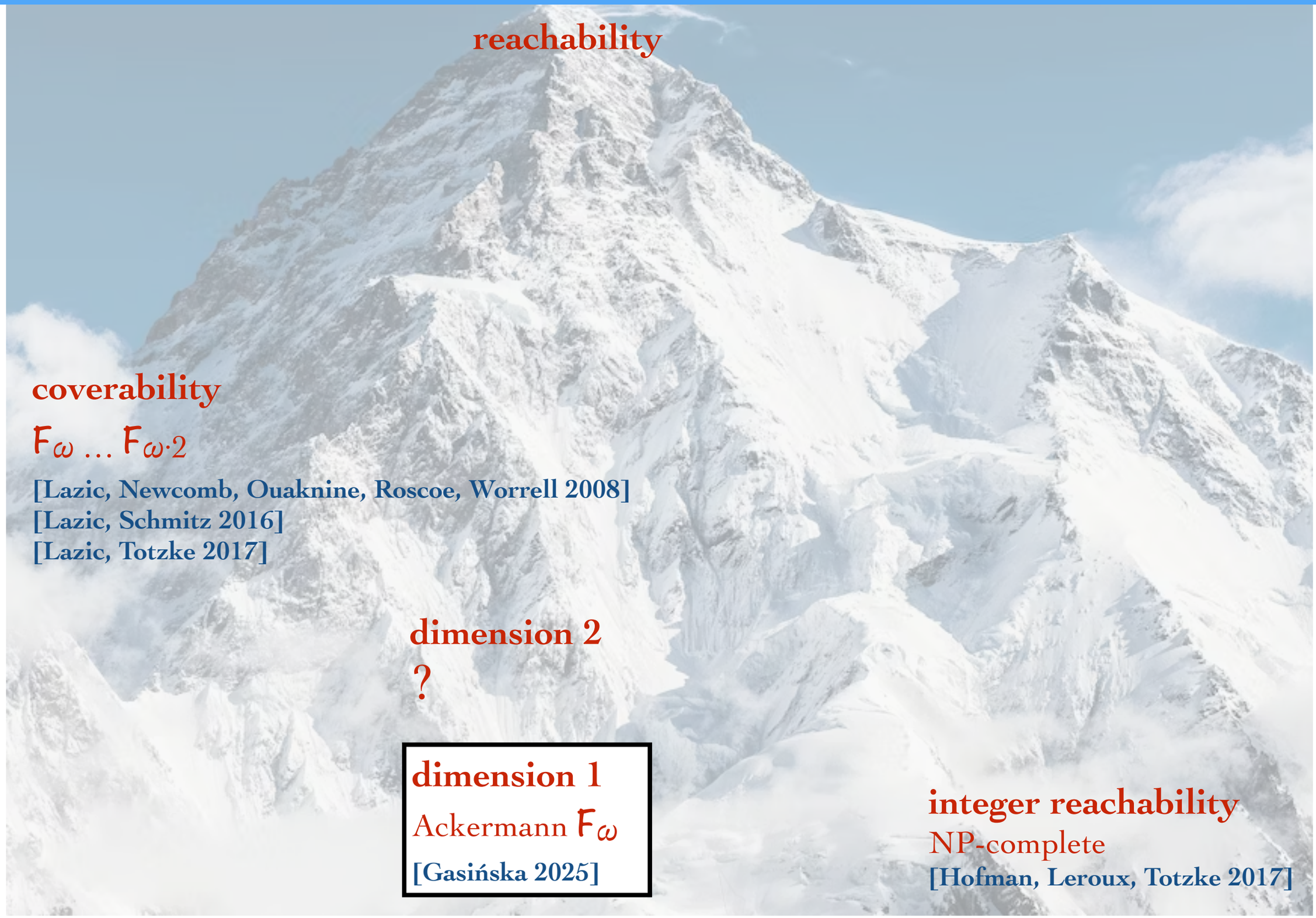
- vector addition systems with states (VASS):



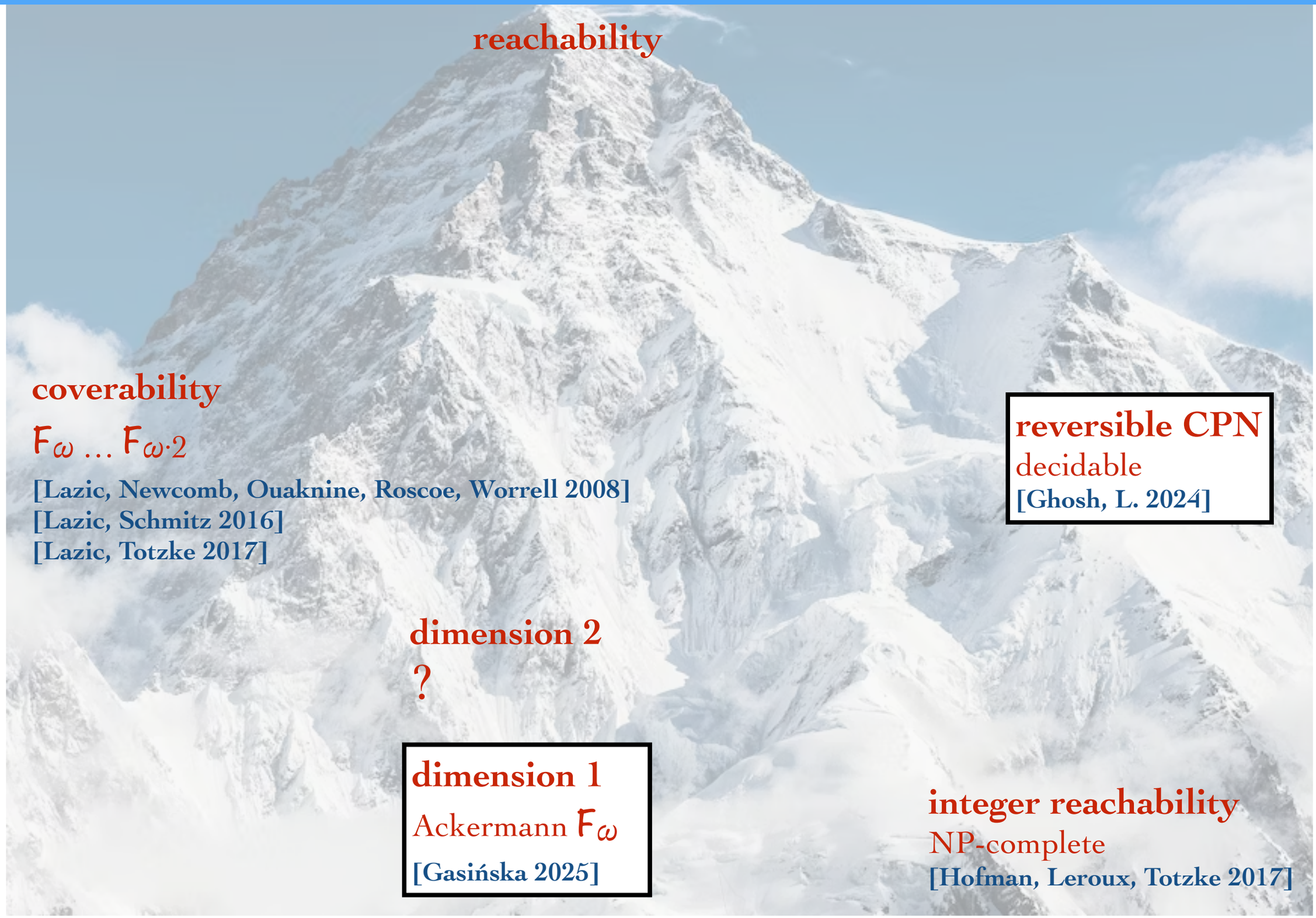
Coloured PN


reachability
coverability
$F_\omega \dots F_{\omega \cdot 2}$
[Lazic, Newcomb, Ouaknine, Roscoe, Worrell 2008]
[Lazic, Schmitz 2016]
[Lazic, Totzke 2017]
integer reachability
NP-complete
[Hofman, Leroux, Totzke 2017]

Coloured PN


reachability
coverability
$F_\omega \dots F_{\omega \cdot 2}$
[Lazic, Newcomb, Ouaknine, Roscoe, Worrell 2008]
[Lazic, Schmitz 2016]
[Lazic, Totzke 2017]
dimension 2
?
dimension 1
Ackermann F_ω
[Gasińska 2025]
integer reachability
NP-complete
[Hofman, Leroux, Totzke 2017]

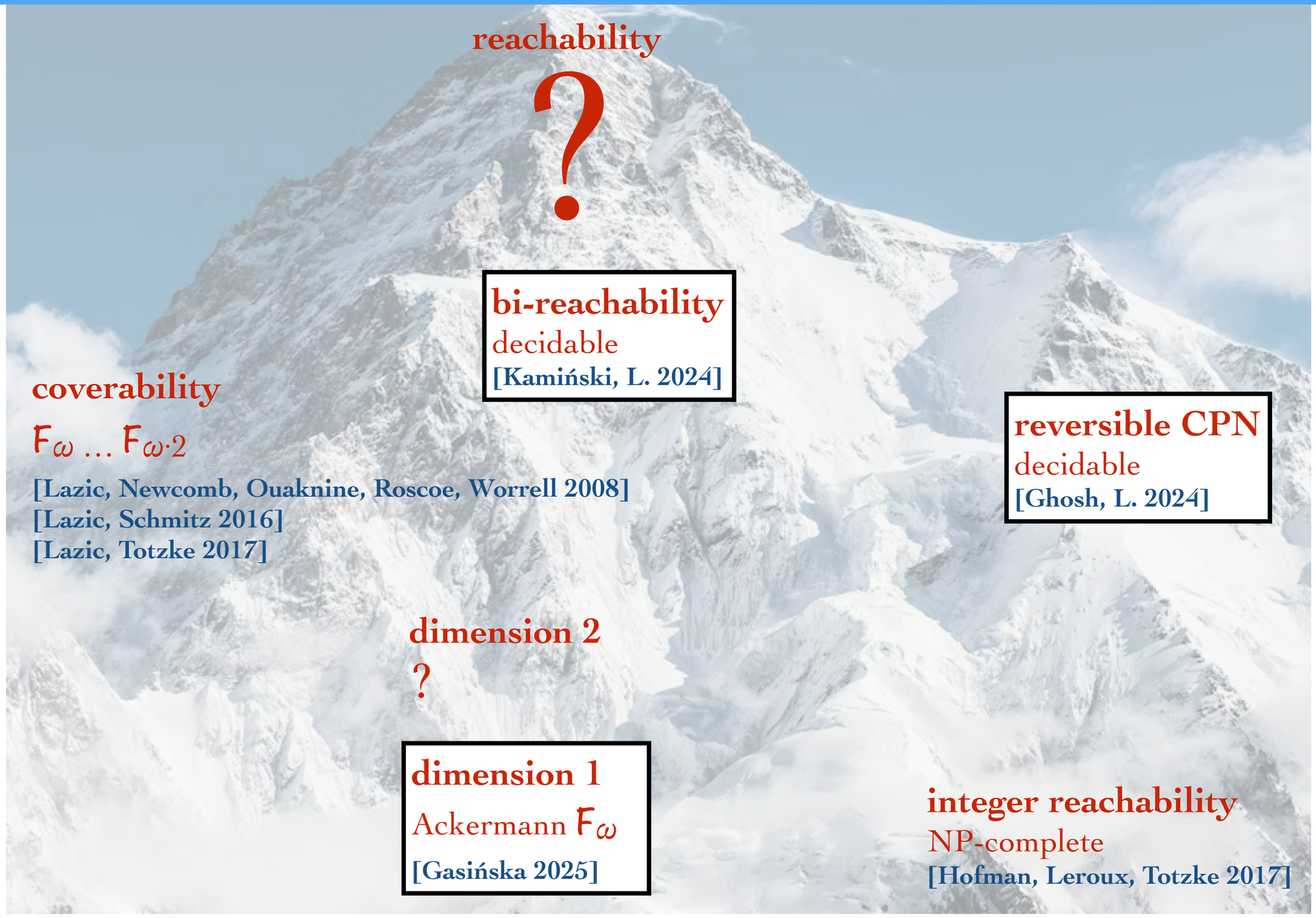
Coloured PN


reachability
coverability
F_ω … $F_{\omega \cdot 2}$
[Lazic, Newcomb, Ouaknine, Roscoe, Worrell 2008]
[Lazic, Schmitz 2016]
[Lazic, Totzke 2017]
reversible CPN
decidable
[Ghosh, L. 2024]
dimension 2
?
dimension 1
Ackermann F_ω
[Gasińska 2025]
integer reachability
NP-complete
[Hofman, Leroux, Totzke 2017]

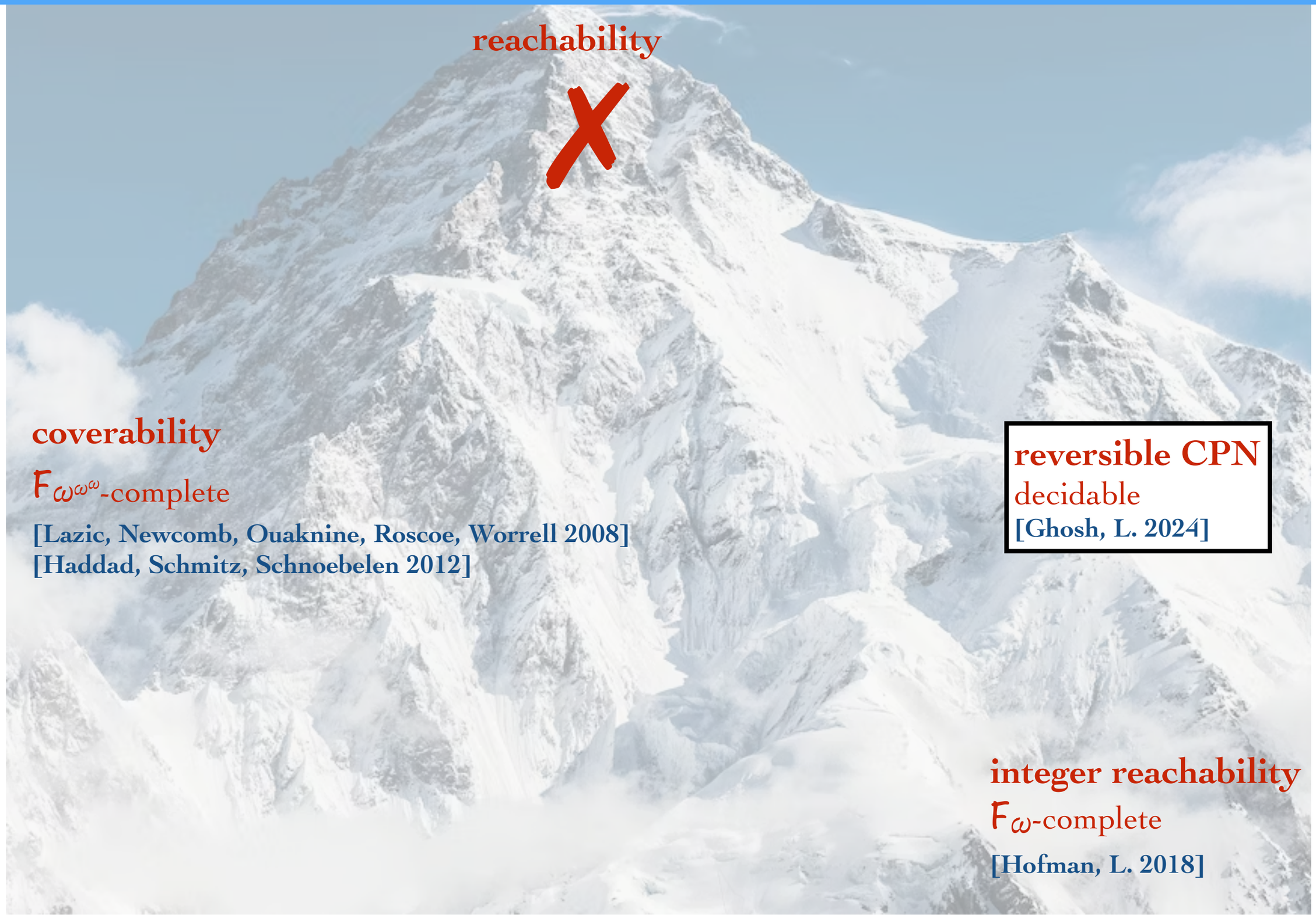
Coloured PN


reachability
bi-reachability
decidable
[Kamiński, L. 2024]
coverability
F_ω … $F_{\omega \cdot 2}$
[Lazic, Newcomb, Ouaknine, Roscoe, Worrell 2008]
[Lazic, Schmitz 2016]
[Lazic, Totzke 2017]
reversible CPN
decidable
[Ghosh, L. 2024]
dimension 2
?
dimension 1
Ackermann F_ω
[Gasińska 2025]
integer reachability
NP-complete
[Hofman, Leroux, Totzke 2017]

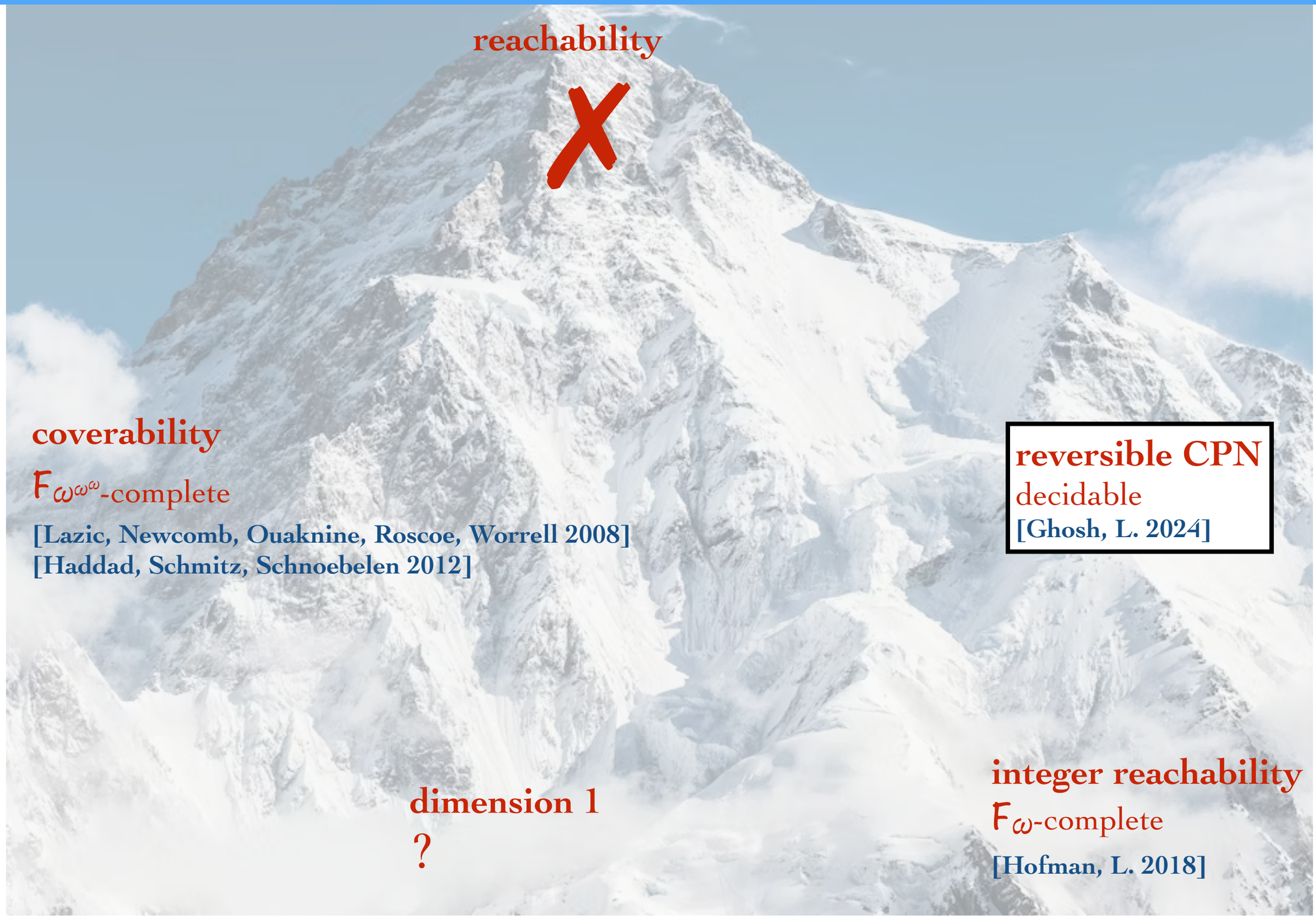
Coloured PN


reachability
?
bi-reachability
decidable
[Kamiński, L. 2024]
coverability
F_ω … $F_{\omega \cdot 2}$
[Lazic, Newcomb, Ouaknine, Roscoe, Worrell 2008]
[Lazic, Schmitz 2016]
[Lazic, Totzke 2017]
reversible CPN
decidable
[Ghosh, L. 2024]
dimension 2
?
dimension 1
Ackermann F_ω
[Gasińska 2025]
integer reachability
NP-complete
[Hofman, Leroux, Totzke 2017]

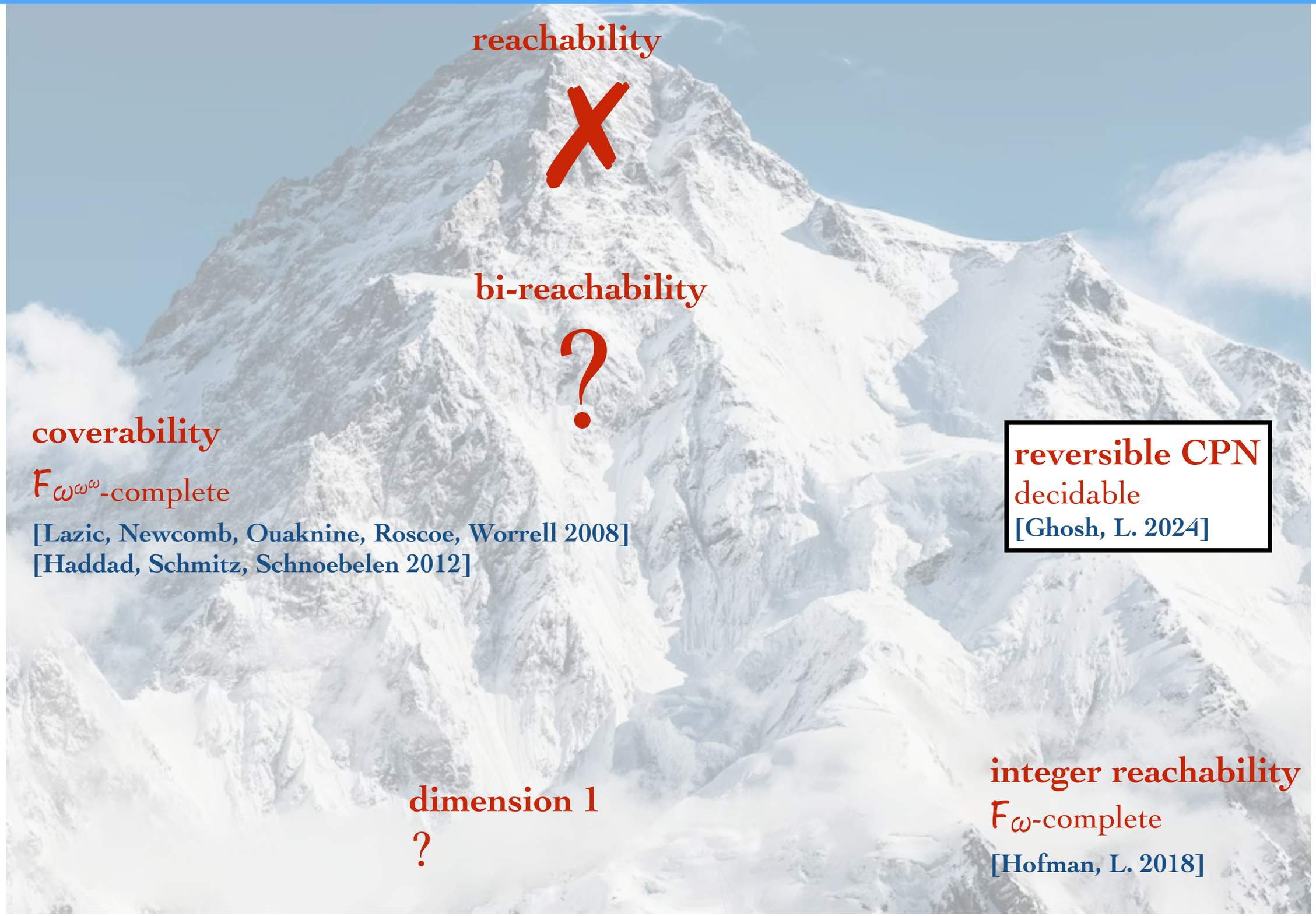
Coloured PN (ordered colours)



Coloured PN (ordered colours)



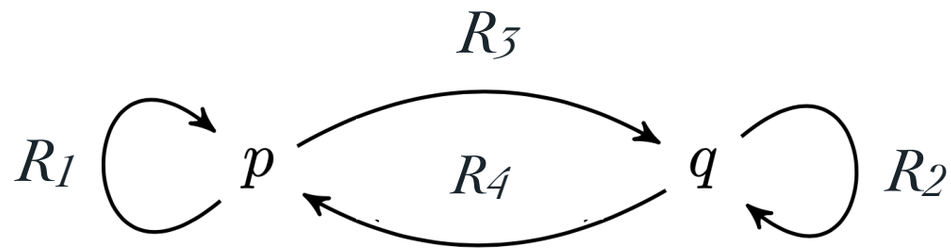
Coloured PN (ordered colours)



C-systems

$\mathcal{C} \subseteq \mathbb{N}^d \times \mathbb{N}^d$ step relations

admitting finite presentations

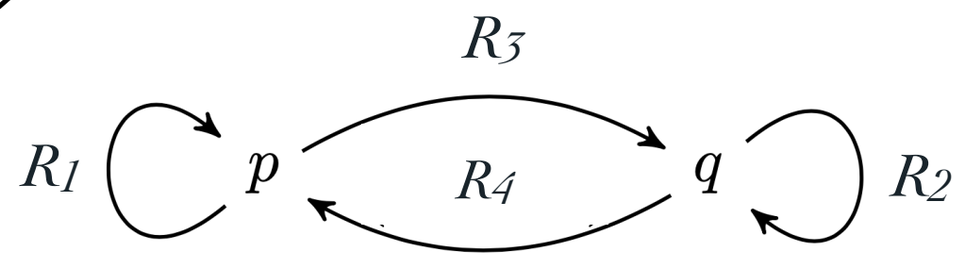


$R_1\ R_2\ R_3\ R_4\ \in \mathcal{C}$

C-systems

$\mathcal{C} \subseteq \mathbb{N}^d \times \mathbb{N}^d$ step relations

admitting finite presentations



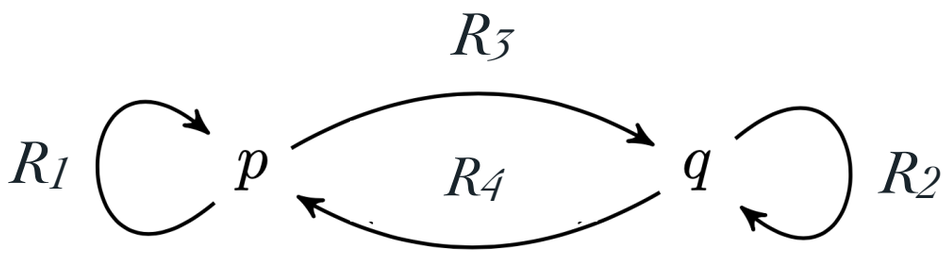
$R_1\ R_2\ R_3\ R_4\ \in \mathcal{C}$

C = Semilinear ?

C-systems

$\mathcal{C} \subseteq \mathbb{N}^d \times \mathbb{N}^d$ step relations

admitting finite presentations



$R_1\ R_2\ R_3\ R_4\ \in \mathcal{C}$

C = Semilinear ?

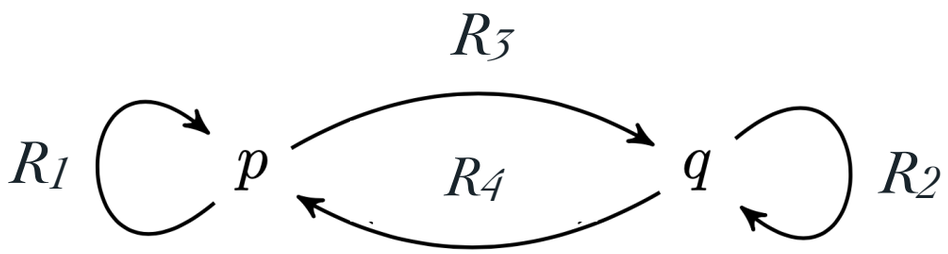
Semilinear-systems include counter machines !

Semilinear-systems are Turing-complete

monotone C-systems

$\mathcal{C} \subseteq \mathbb{N}^d \times \mathbb{N}^d$ step relations

admitting finite presentations



$R_1\ R_2\ R_3\ R_4\ \in \mathcal{C}$

C = Semilinear ?

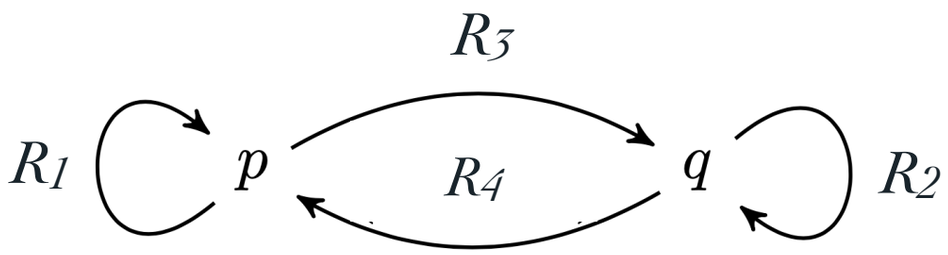
Semilinear-systems include counter machines !

Semilinear-systems are Turing-complete

monotone C-systems

$\mathcal{C} \subseteq \mathbb{N}^d \times \mathbb{N}^d$ step relations

admitting finite presentations



$R_1\ R_2\ R_3\ R_4\ \in \mathcal{C}$

and **monotone**

$(x, y) \in R,\ v \in \mathbb{N}^d$

$\Downarrow$

$(x + v, y + v) \in R$

C = Semilinear ?

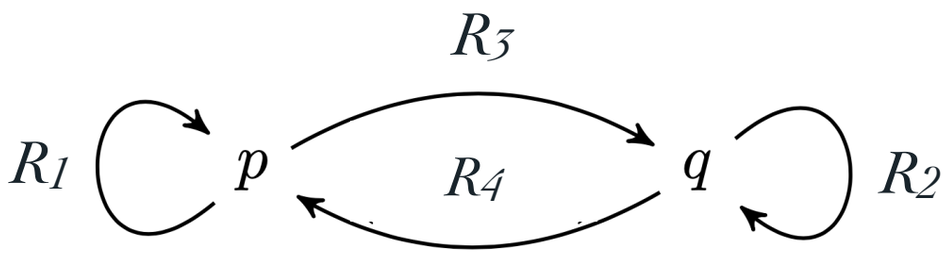
Semilinear-systems include counter machines !

Semilinear-systems are Turing-complete

monotone C-systems

$\mathcal{C} \subseteq \mathbb{N}^d \times \mathbb{N}^d$ step relations

admitting finite presentations



$R_1\ R_2\ R_3\ R_4\ \in \mathcal{C}$

and **monotone**

$(x, y) \in R,\ v \in \mathbb{N}^d$

$\Downarrow$

$(x + v, y + v) \in R$

C = Semilinear ?

Semilinear-systems include counter machines !

Semilinear-systems are Turing-complete

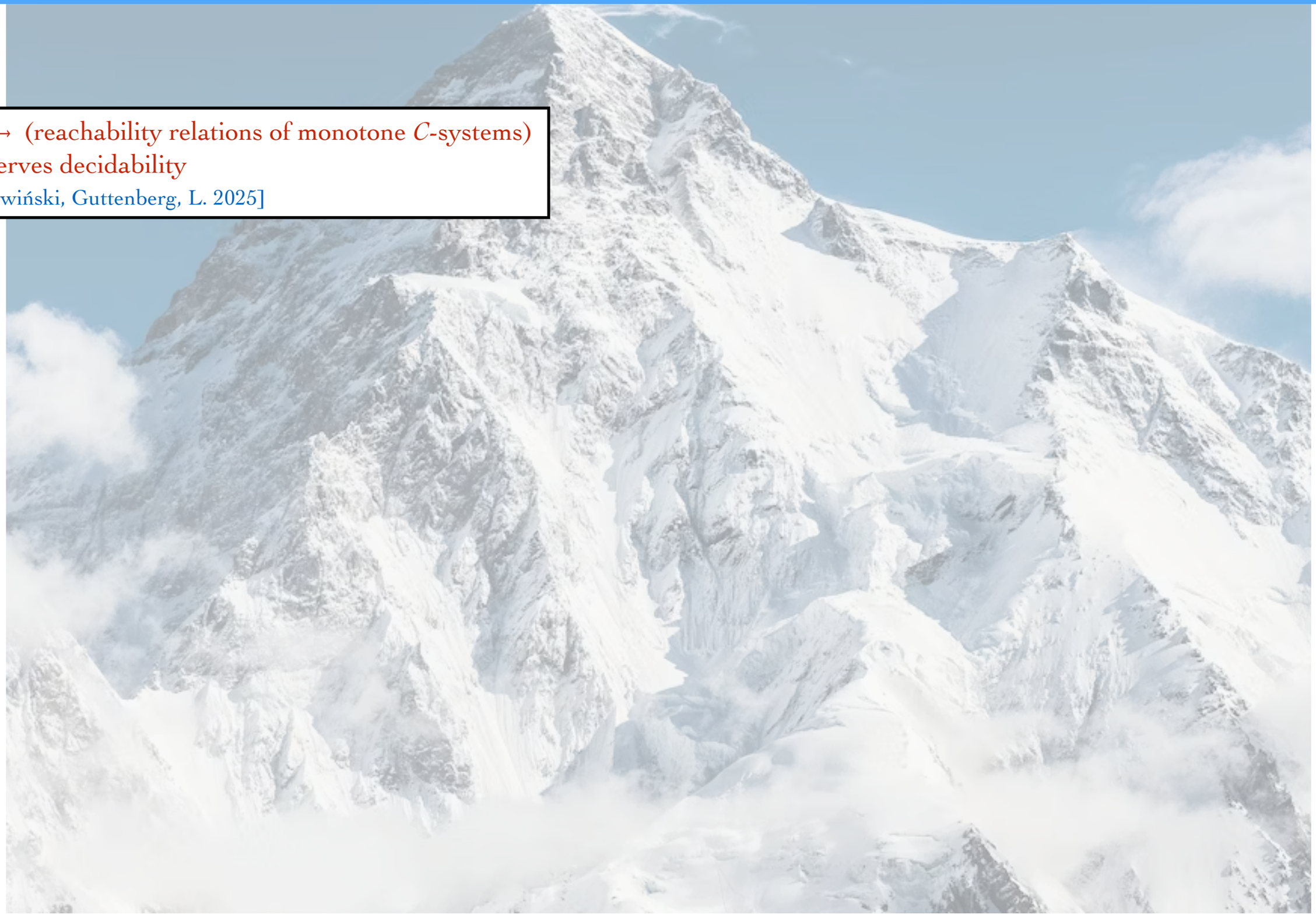
$C \longmapsto$ (reachability relations of monotone C-systems) preserves decidability

[Czerwiński, Guttenberg, L. 2025]

monotone C-systems - applications

$C \longmapsto$ (reachability relations of monotone C-systems) preserves decidability

[Czerwiński, Guttenberg, L. 2025]



monotone C-systems - applications

$C \longmapsto$ (reachability relations of monotone C-systems) preserves decidability

[Czerwiński, Guttenberg, L. 2025]

VASS with nested 0-tests

...

↑

VASS with 2 nested zero-tested counters

↑

VASS with 1 zero-tested counter

↑

VASS

↑

Semilinear

monotone C-systems - applications

$C \longmapsto$ (reachability relations of monotone C-systems) preserves decidability
[Czerwiński, Guttenberg, L. 2025]

Pushdown VASS

…

CFG-controlled VASS of nesting 2

CFG-controlled VASS of nesting 1

CFG-controlled VASS of nesting 0

VASS with nested 0-tests

…

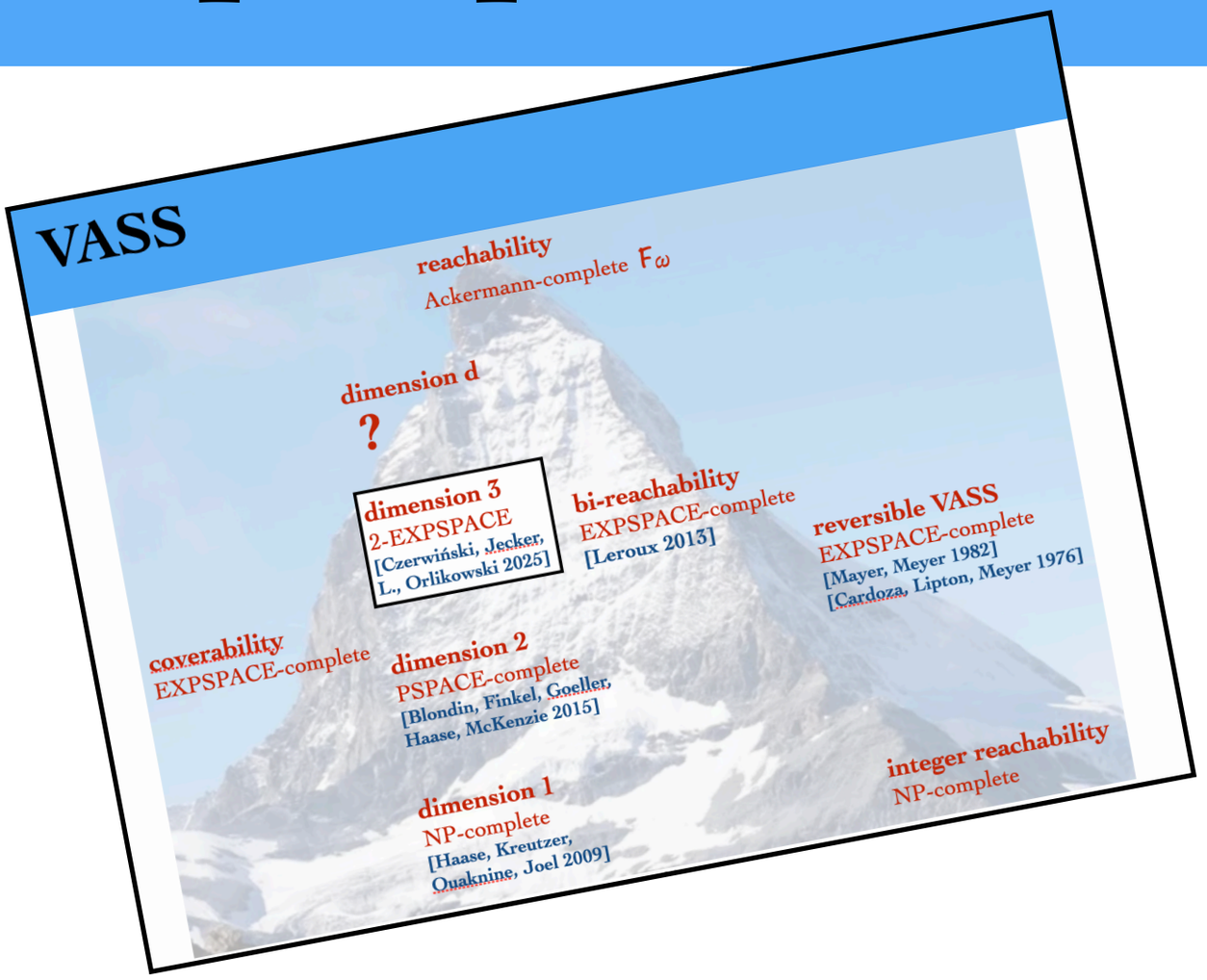
VASS with 2 nested zero-tested counters

VASS with 1 zero-tested counter

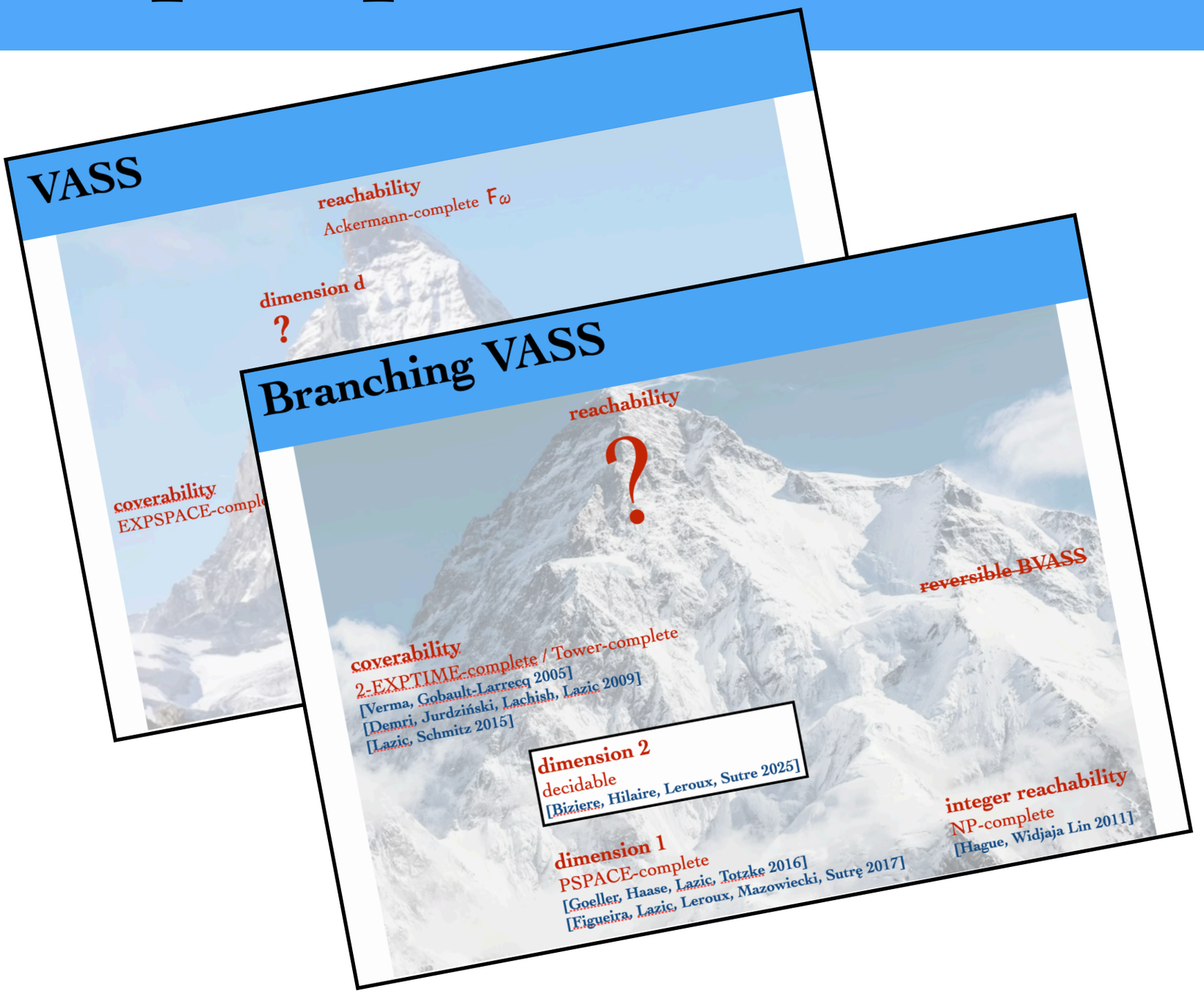
VASS

Semilinear

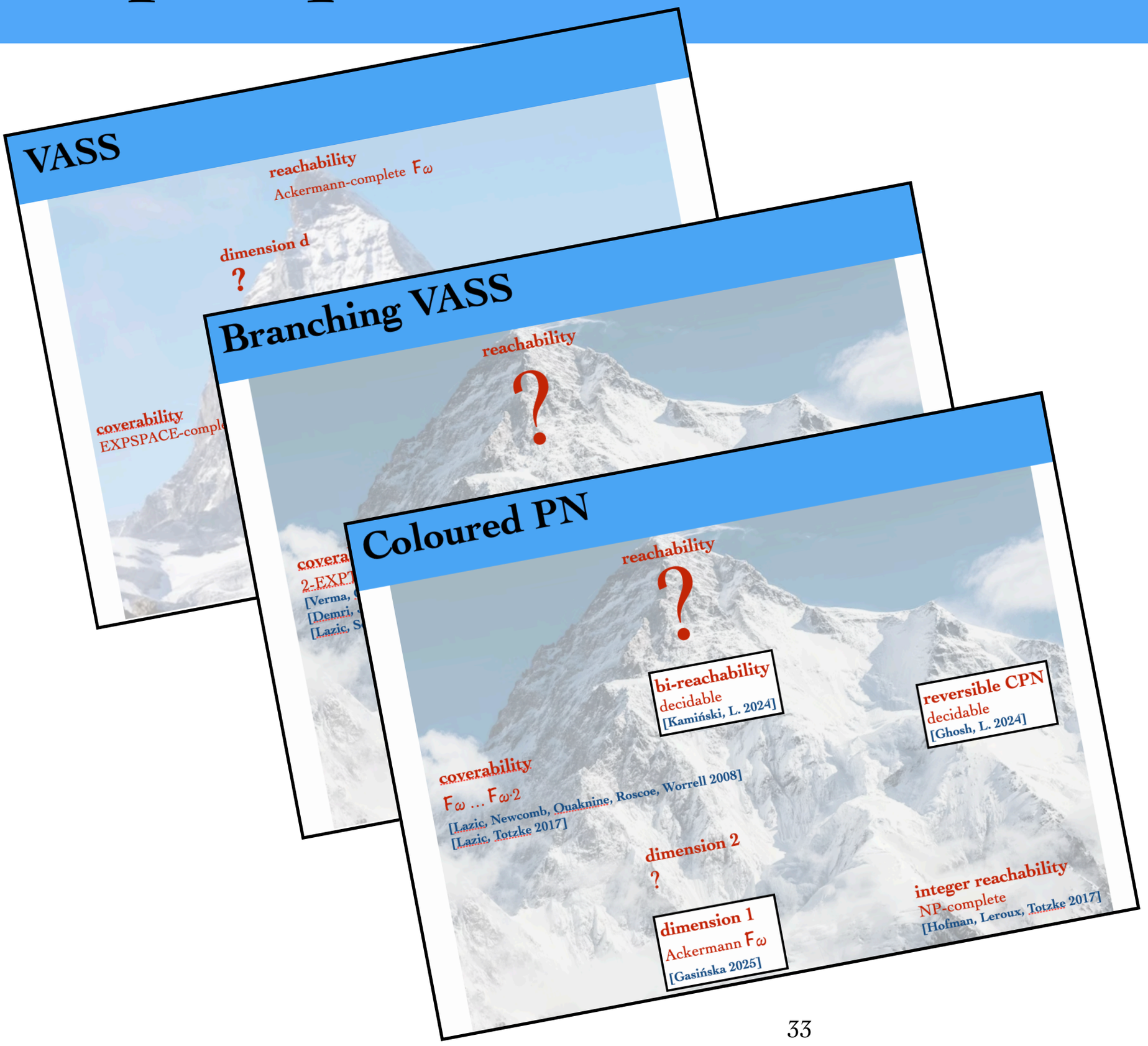
Open questions



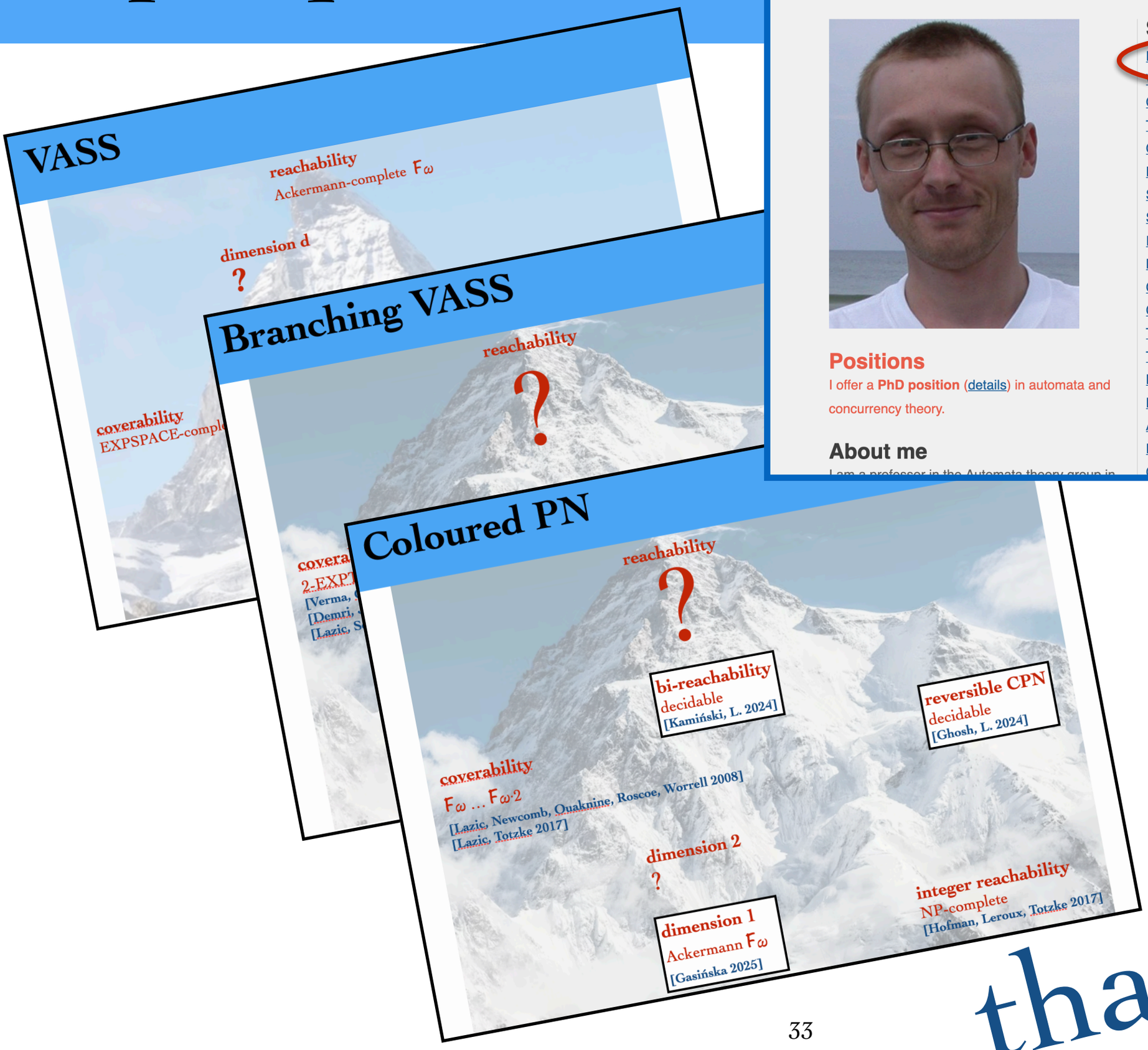
Open questions



Open questions



Open questions



thank you!