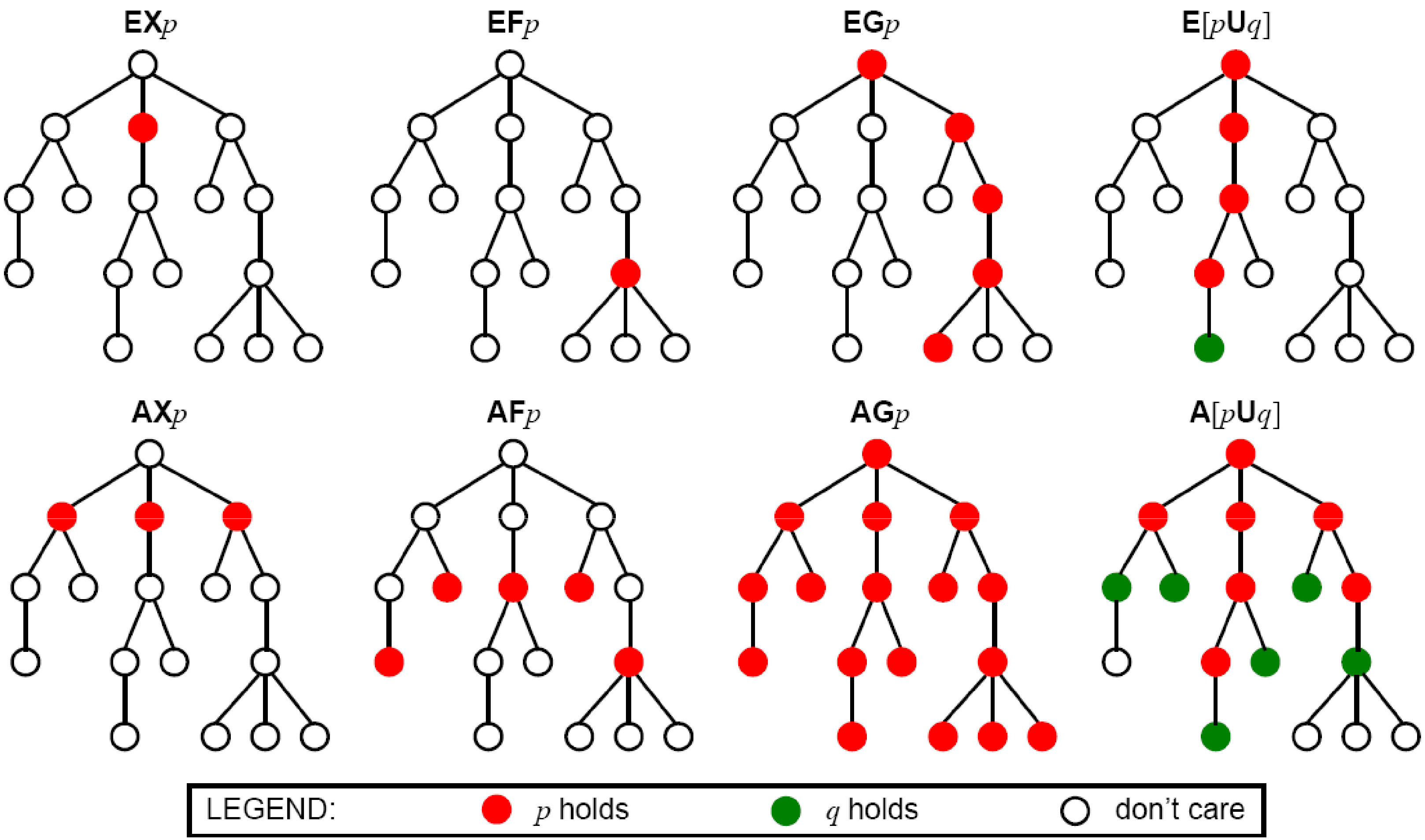
CTL Model checking

CTL semantics



EX, EU, and EG are a complete set of CTL operators, since:

$$\mathsf{AX}p = \neg\mathsf{EX}\neg p \qquad \mathsf{EF}p = \mathsf{E}[true \; \mathsf{U} \; p] \qquad \mathsf{E}[p\mathsf{R}q] = \neg\mathsf{A}[\neg p \mathsf{U} \neg q]$$

$$\mathsf{AF}p = \neg\mathsf{EG}\neg p \qquad \mathsf{A}[p \; \mathsf{U} \; q] = \neg\mathsf{E}[\neg q \; \mathsf{U} \; \neg p \; \wedge \; \neg q] \wedge \neg\mathsf{EG}\neg q \qquad \mathsf{A}[p\mathsf{R}q] = \neg\mathsf{E}[\neg p \mathsf{U} \neg q]$$

$$\mathsf{AG}p = \neg\mathsf{EF}\neg p$$

The easy cases

Case 1: $\alpha \equiv \neg p$ (with $p \in AP$)

$S_K(p)$ is given by $\{s \in S \mid p \in \ell(s)\}$ thus by definition

$$S_K(\neg p) = S \setminus S_K(p)$$

Case 2: $\alpha \equiv p \wedge q$ (with $p, q \in AP$)

$$S_K(p \wedge q) = S_K(p) \cap S_K(q)$$

Case 3: $\alpha \equiv EX\, p$ (with $p \in AP$)

For the following, let us define $pre(X)$, where $X \subseteq S$, as the set $pre(X) = \{ s \in S \mid \exists\, t \in X : s \rightarrow t \}$.

$$S_K(EX\, p) = pre(S_K(p))$$

BDD based algorithm for EX(f)

- Let F be the set of states satisfying “f”
  - F can be built by selecting states from the full state space
- EX(F)
  - S := $Next^{-1}$ (F)
  - Return S

The *EG* operator

Case 4: $\alpha \equiv EG\, p$ (with $p \in AP$)

$S_K(EG\, p)$ is the greatest solution (w.r.t. $\subseteq$) of the equation

$$X = S_K(p) \cap pre(X)$$

$S_K(EG\, p)$ is the fixed point of the sequence

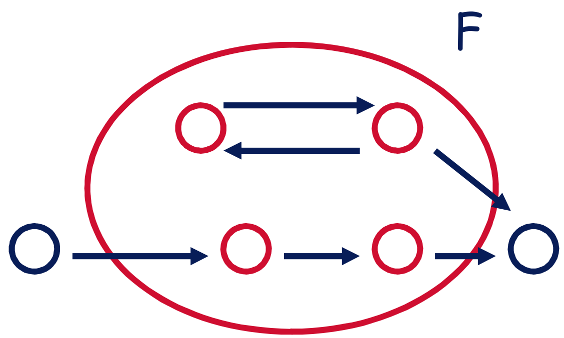
$$S,\ \pi(S),\ \pi(\pi(S)),\ \ldots \text{ where } \pi(X) = S_K(p) \cap pre(X)$$

Operator EG for EG (f)

- Let F be the set of states satisfying “f”
- EG(F)
  - S := F
  - N := 0
  - While (N != S)
    - N := S
    - $S := S \cap \text{Next}^{-1}(S)$
  - Return S

Initialize with states that verify f
Potentially all these states verify Gf

Remove some potential candidates state
If s verifies Gf, s verifies f
and successor verifies “f”

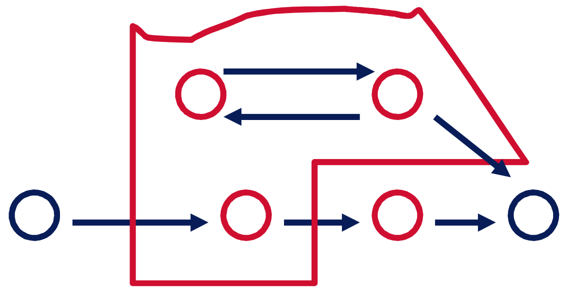


Operator EG for EG (f)

- Let F be the set of states satisfying “f”
- EG(F)
  - S := F
  - N := 0
  - While (N != S)
    - N := S
    - $S := S \cap Next^{-1}(S)$
  - Return S

Initialize with states that verify f
Potentially all these states verify Gf

Remove some potential candidates state
If s verifies Gf, s verifies f
and successor verifies “f”



Operator EG for EG (f)

- Let F be the set of states satisfying "f"
- EG(F)
  - S := F
  - N := 0
  - While (N != S)
    - N := S
    - S := S ∩ $Next^{-1}$ (S)
  - Return S

Initialize with states that verify f
Potentially all these states verify Gf

Remove some potential candidates state
If s verifies Gf, s verifies f
and successor verifies "f"

$$(Next^{-1} \cap Id)^{*} \circ F$$

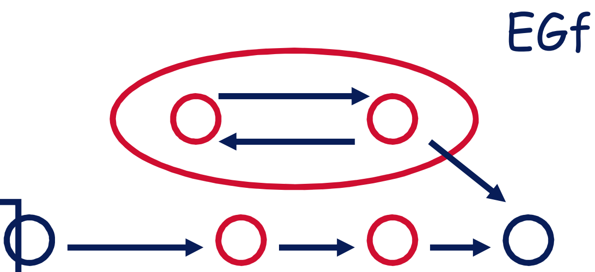
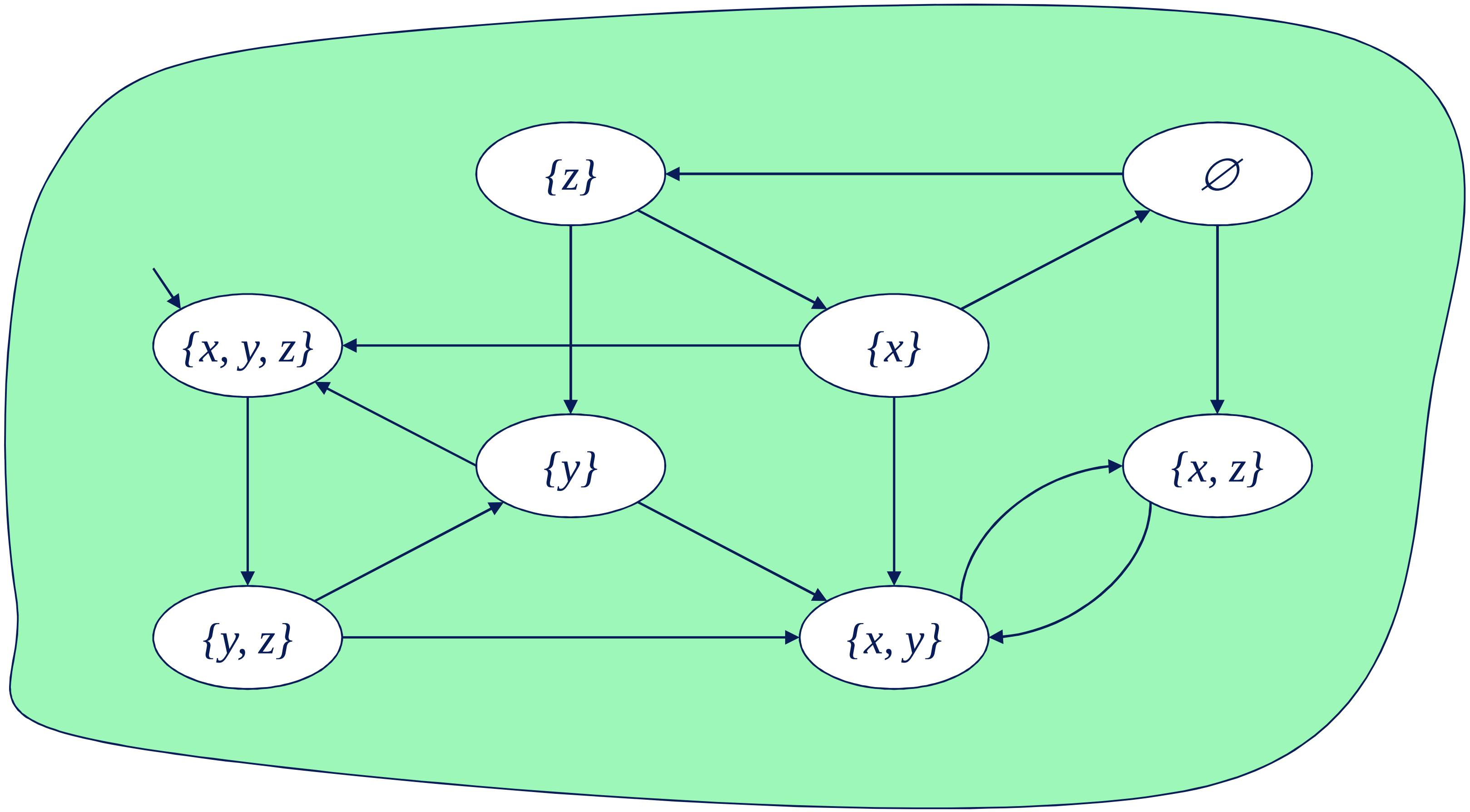
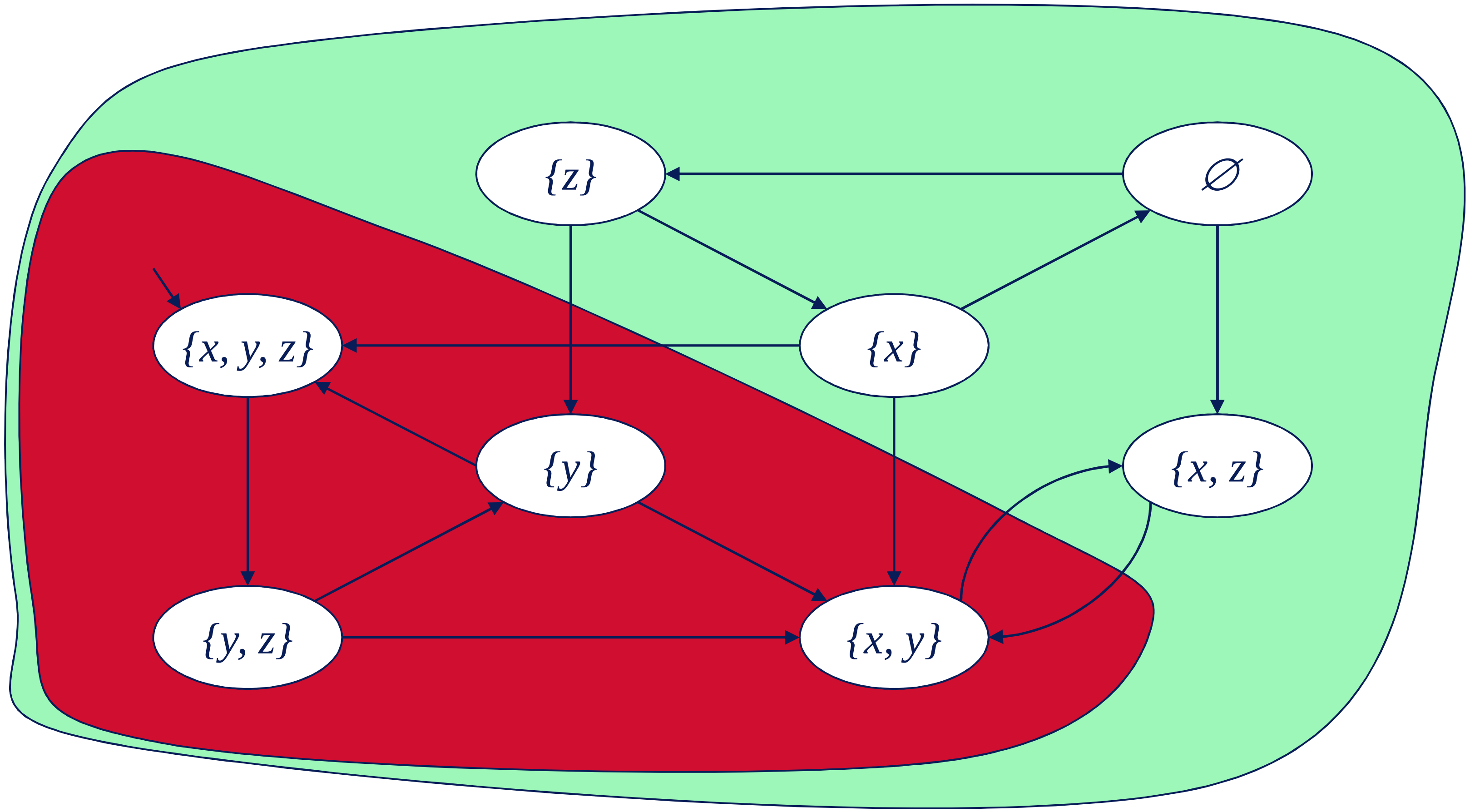


Illustration for *EG y*



$\pi^0(S) = S$

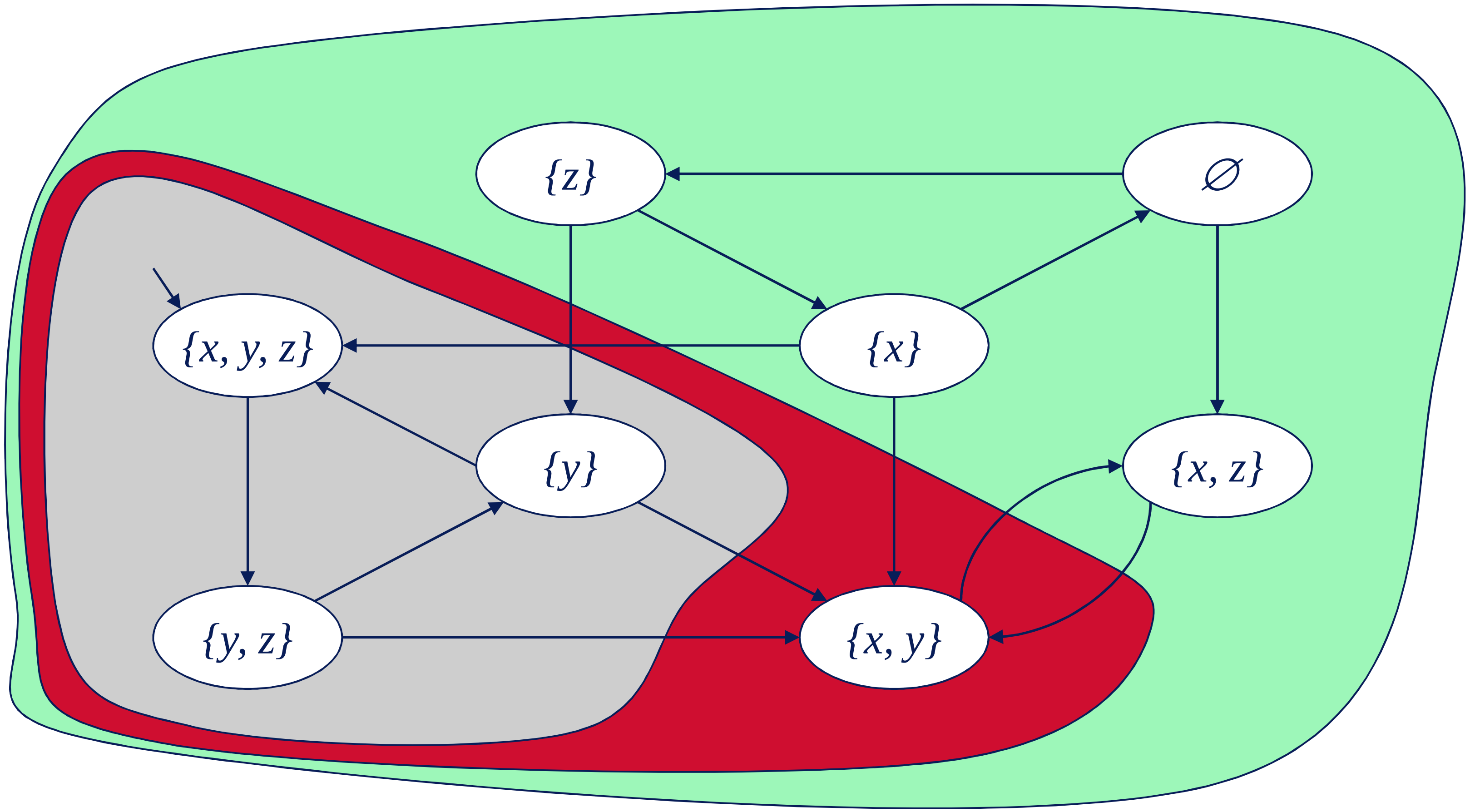
Illustration for *EG y*



$\pi^1(S) = S_K(y) \cap pre(S)$

States not satisfying *y* have been excluded

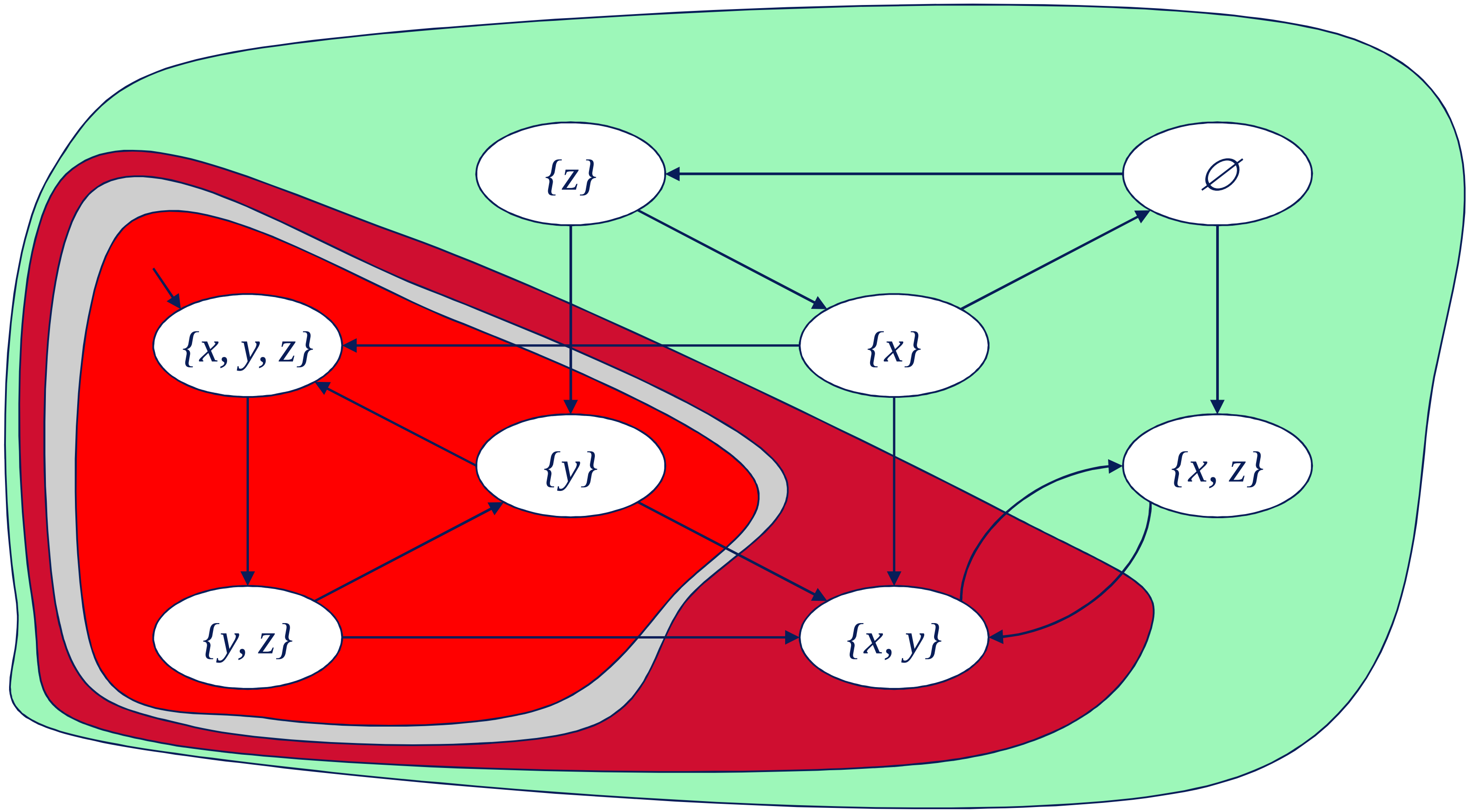
Illustration for *EG y*



$\pi^2(S) = S_K(y) \cap pre(\pi^1(S))$

States having all its successors outside π^1 have been excluded

Illustration for *EG y*



$\pi^3(S) = S_K(y) \cap pre(\pi^2(S))$

The fixed point has been reached

The *EU* operator

Case 5: $\alpha \equiv p$ *EU* p (with $p, q \in AP$)

$S_K(p$ *EU* $q)$ is the smallest solution (w.r.t. $\subseteq$) of the equation

$$X = S_K(q) \cup (S_K(p) \cap pre(X))$$

$S_K($*EG* $p)$ is the fixed point of the sequence

$\emptyset, \xi(\emptyset), \xi(\xi(\emptyset)), \ldots$ where $\xi(X) = S_K(q) \cup (S_K(p) \cap pre(X))$

Operator EU for E(f U g)

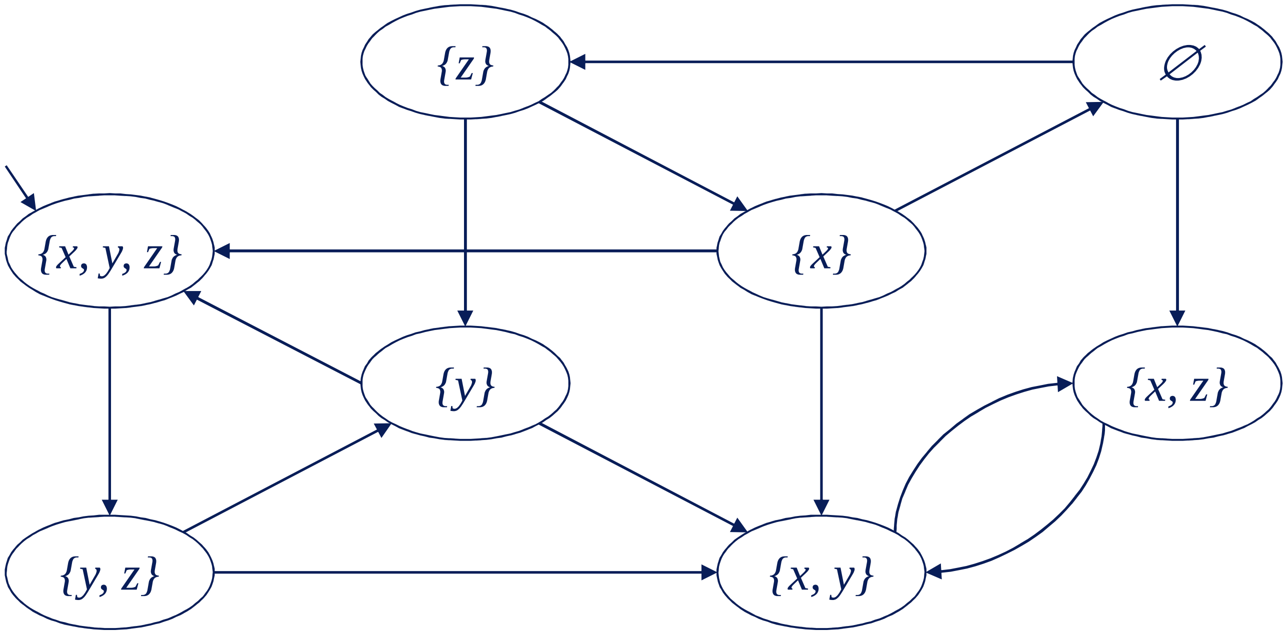
- Let F and G be the set of states satisfying "f" and "g"
- EU(F,G)
  - S := G
  - N := 0
  - While (N != S)
    - N := S
    - S := S ∪ (F ∩ Next^{-1} (S))
  - Return S

Initialize with states that verify g

Keep only predecessors that verify f

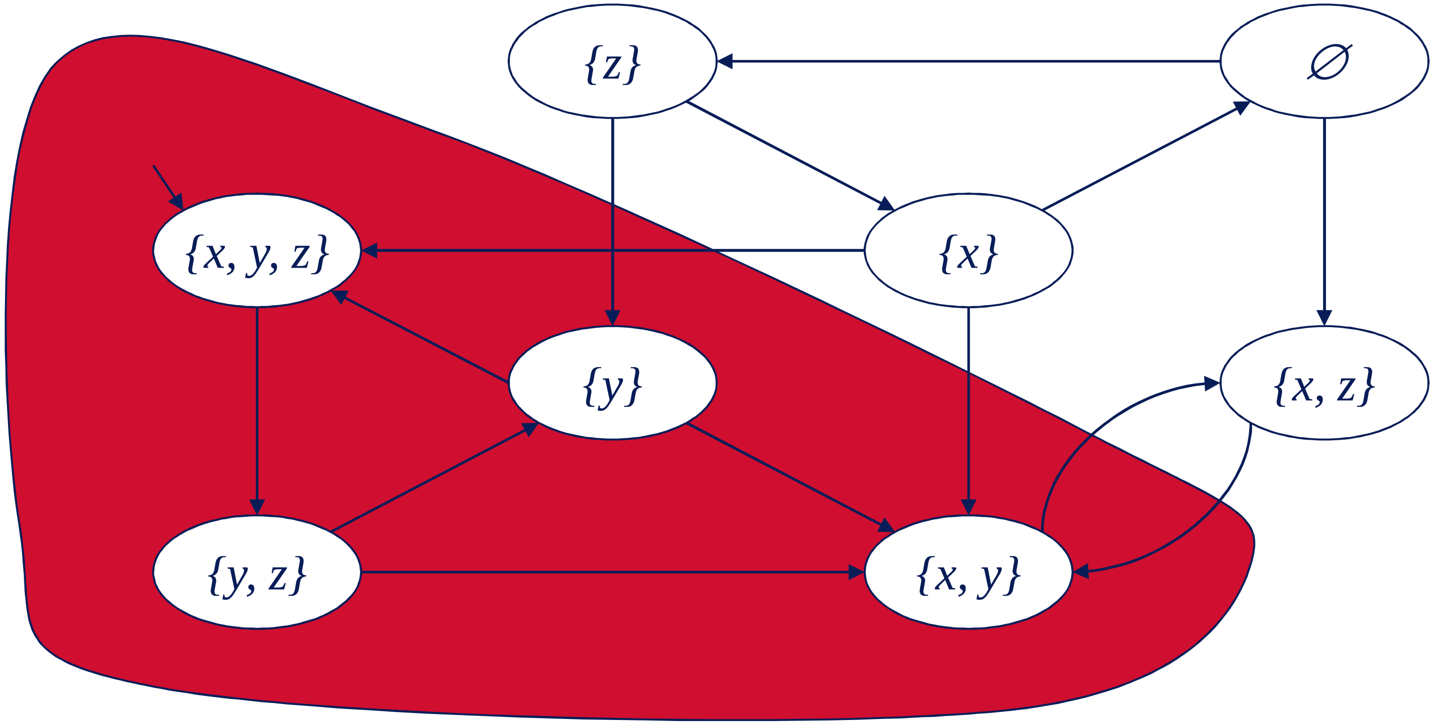
$$\boxed{(F \circ Next^{-1} + Id)^{*} \circ G}$$

Illustration for *z EU y*



$\xi^0(\varnothing) = \varnothing$

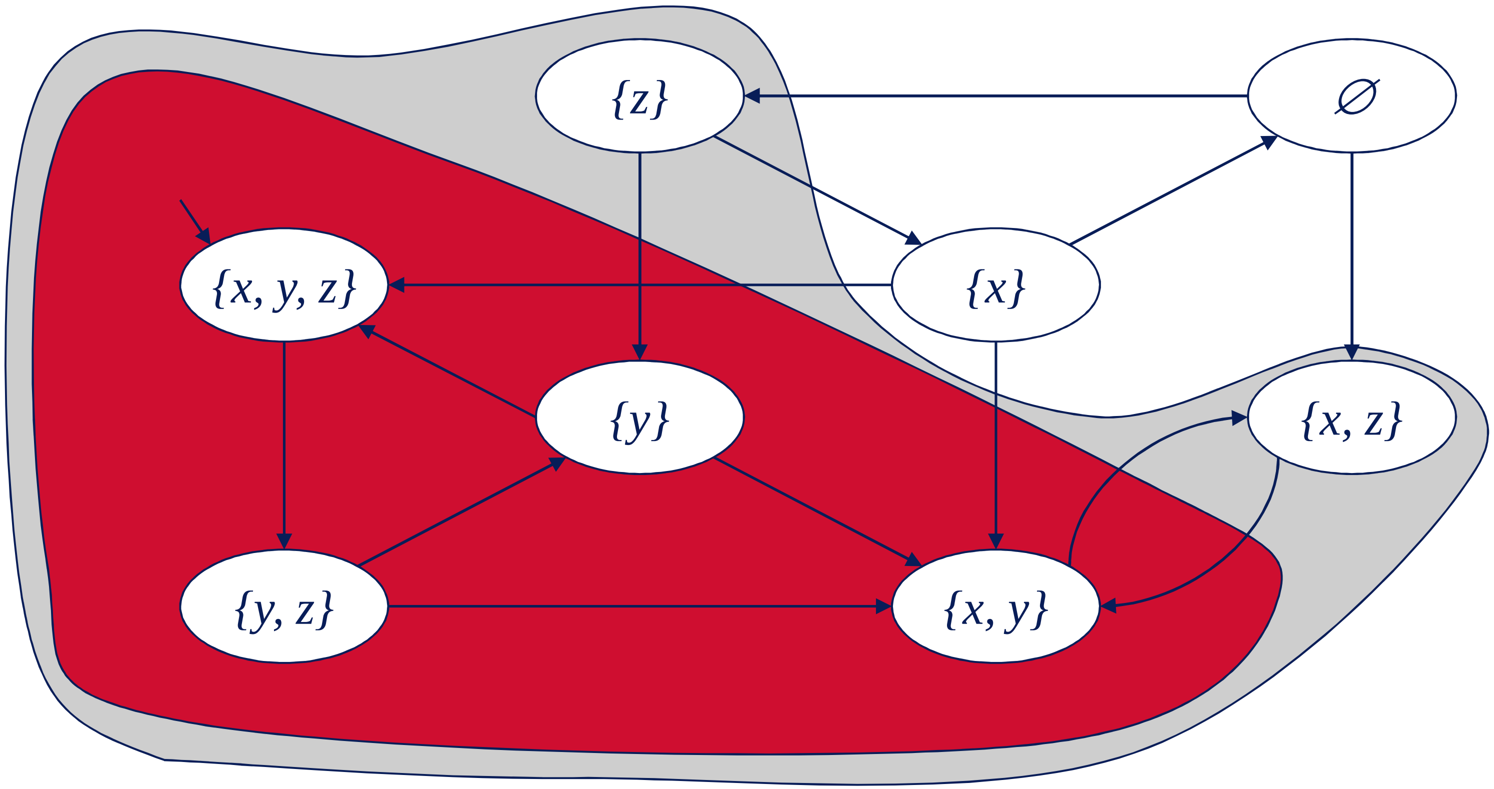
Illustration for *z* ***EU*** *y*



$$\xi^1(\varnothing) = S_K(y) \cup (S_K(z) \cap pre(\xi^0(\varnothing)))$$

States satisfying *y* have been added

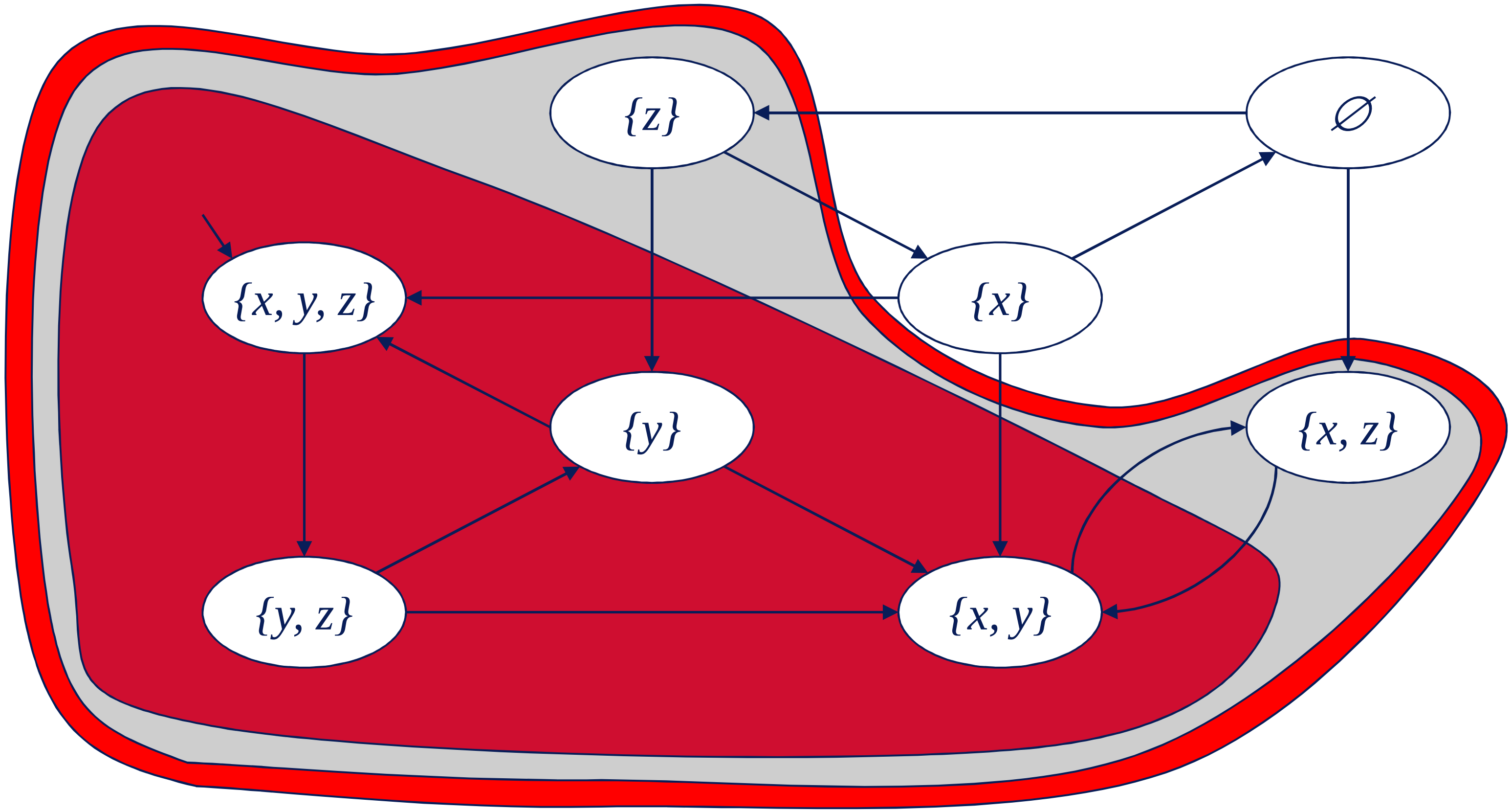
Illustration for z *EU* y



$$\xi^2(\emptyset) = S_K(y) \cup (S_K(z) \cap pre(\xi^1(\emptyset)))$$

States satisfying z and having at least a successor in ξ^1 have been added

Illustration for z *EU* y



$$\xi^3(\varnothing) = S_K(y) \cup (S_K(z) \cap pre(\xi^2(\varnothing)))$$

The fixed point has been reached

Conclusion on CTL

- CTL (Branching time) can specify safety properties and some liveness properties
- CTL can be efficiently implemented (linear complexity w.r.t. to the Kripke structure), provided a good management of sets of states.
- Fairness needs to augment the capability of CTL model checkers (SCC searches are needed).
- CTL fair model checkers can be used to verify CTL and also LTL formula.

- CTL does not provide a counter example when the property does not hold. The output is the set of states that satisfy the formula (maybe huge).
- CTL model checkers cannot answer before labeling the initial state with the truth value of the formula.