

An Introduction to Multi-Valued Model Checking



Georgios E. Fainekos

Department of CIS

University of Pennsylvania

<http://www.seas.upenn.edu/~fainekos>

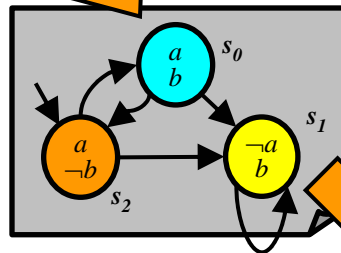
Written Preliminary Examination II

30th of June, 2005

Model Checking: Is the system correct??



Extract model



Formalize
Specification

$$A[Ga \Rightarrow (Xb \vee \neg a)]$$

Model Checker

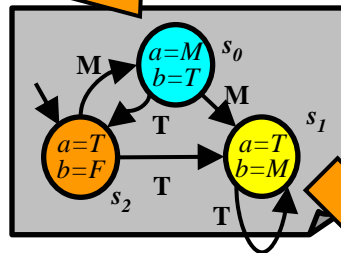
YES
Witness

NO
Counter Example

Multi-Valued Model Checking: In what "degree" is the system correct??



Extract model



Formalize
Specification

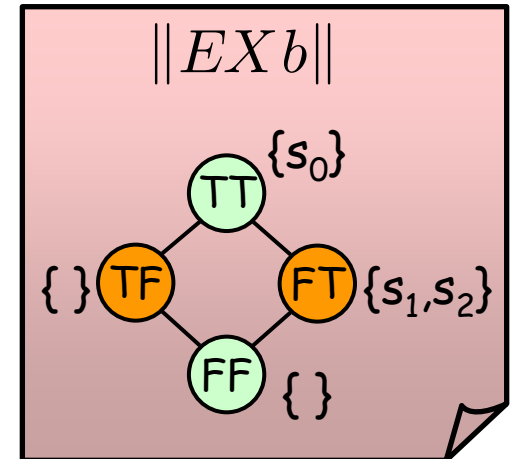
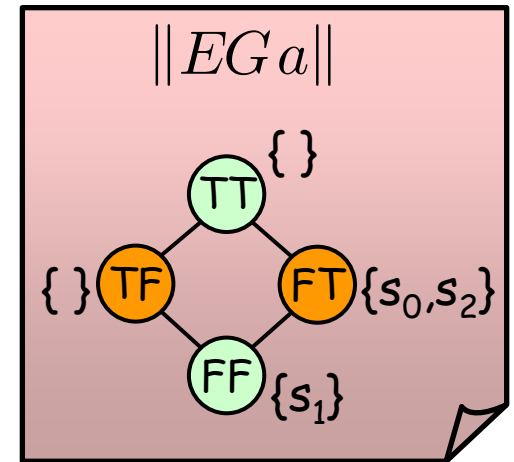
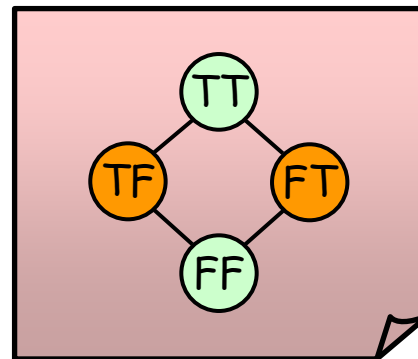
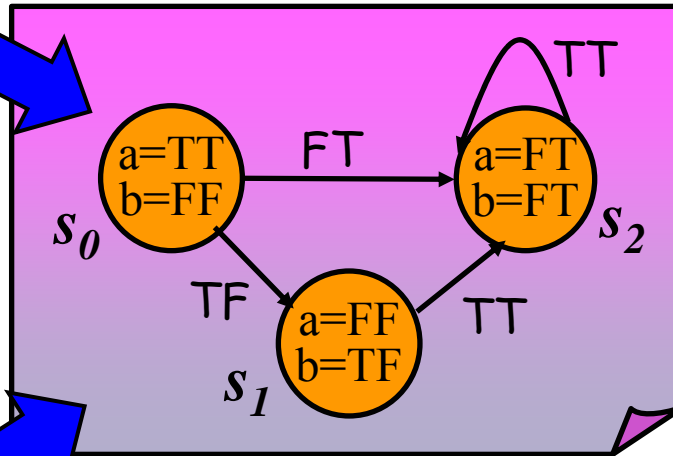
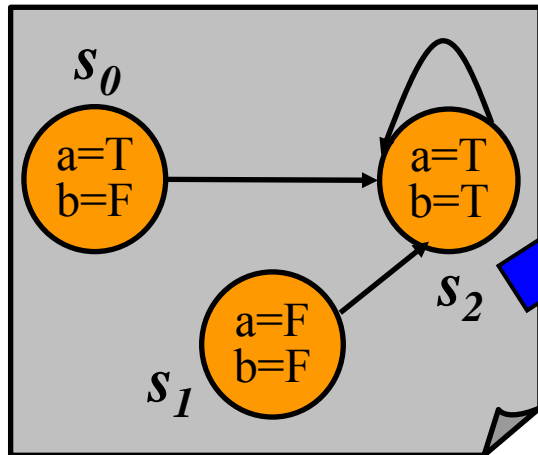
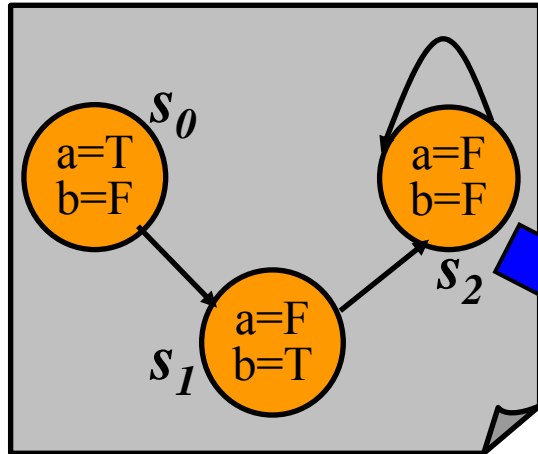
$$A[Ga \Rightarrow (Xb \vee \neg a)]$$

MV-Model Checker

The "degree" of satisfaction

Why multi-valued model checking?

Application 1: conflicting viewpoints



Why multi-valued model checking?

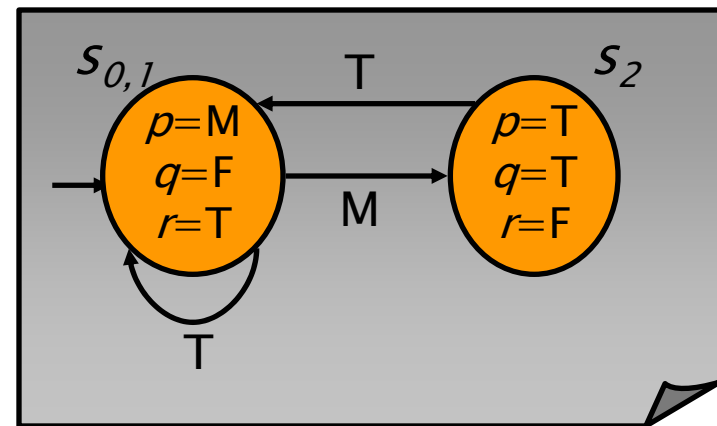
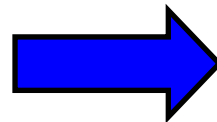
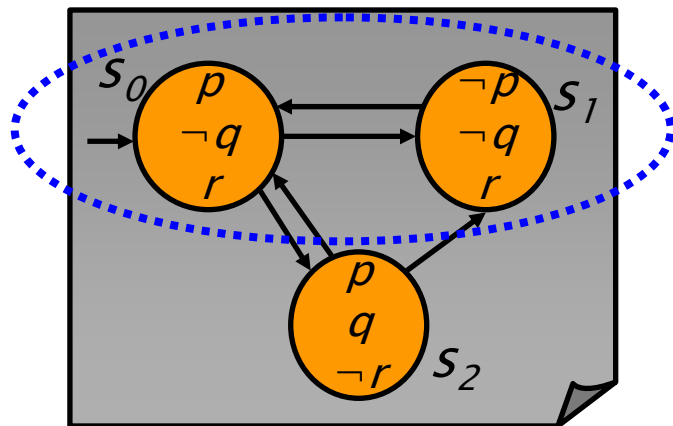
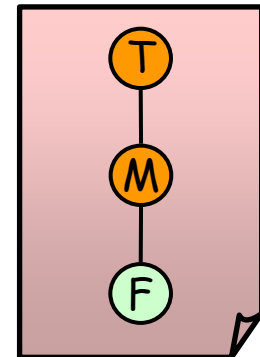
Application 2: Abstraction

Using 3-valued logic

- introduce new special value *Maybe* to stand for “unknown”

Advantages:

- No spurious counter-examples
 - ◆ result = T, F or M (unknown)
- Verification even using incomplete models



Why multi-valued model checking?

Application 3: Query Checking [Chan, CAV'00]

Goal: speed-up design understanding

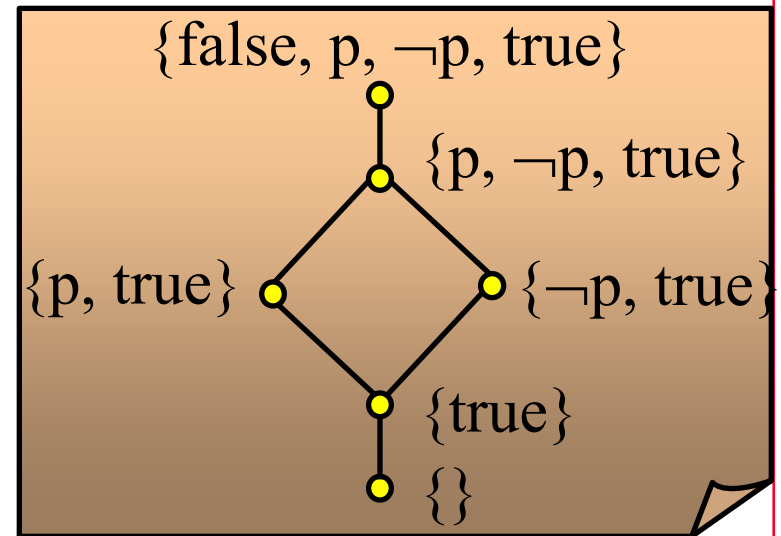
- discover properties not known *a priori*

Temporal logic query

- temporal logic formula with placeholders (unknowns)
 - ♦ e.g., $AG ?_x$, $AG(p \rightarrow ?_x)$
- evaluates to strongest propositional formula that makes query true.

Some applications

- provide partial explanation when property holds
 - ♦ e.g. instead of $AG(a \vee b)$, ask $AG ?_x\{a, b\}$
 - ♦ answer $a \wedge b$ is stronger!
- provide diagnostic information when property fails



Ordering objects

➤ A **partial-order** is a binary relation \sqsubseteq such that for all $x, y, z \in S$ the following properties hold:

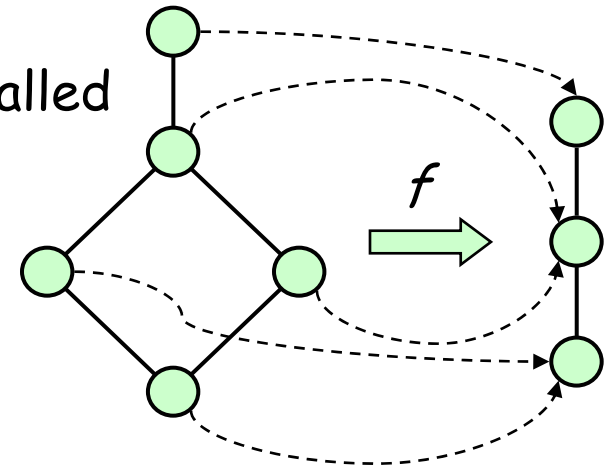
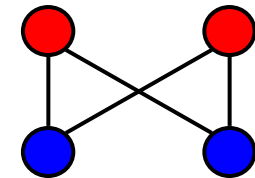
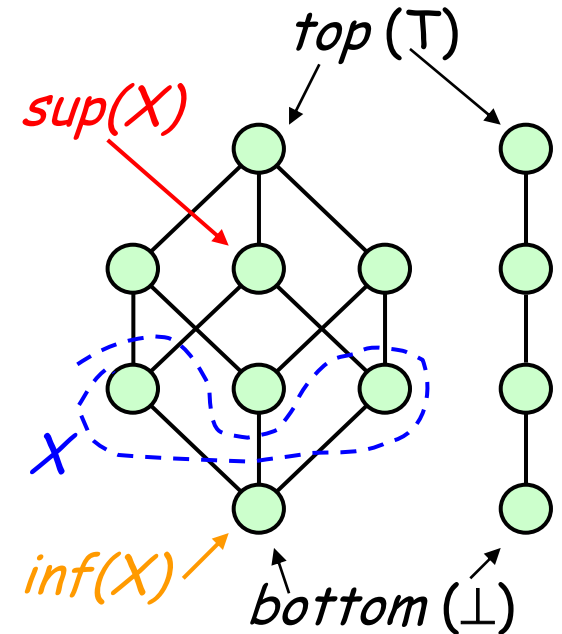
- **Reflexivity** $x \sqsubseteq x$
- **Transitivity** $x \sqsubseteq y$ and $y \sqsubseteq z$ imply $x \sqsubseteq z$
- **Antisymmetry** $x \sqsubseteq y$ and $y \sqsubseteq x$ imply $x = z$

➤ A **poset** is the pair: $S = (S, \sqsubseteq)$

➤ In a **linear order** all the elements are comparable.

➤ Let X, Y be posets, then a map $f: X \rightarrow Y$ is called **order-preserving** if:

$$(\forall x_1, x_2 \in X). (x_1 \sqsubseteq_X x_2 \rightarrow f(x_1) \sqsubseteq_Y f(x_2))$$



Lattices

- ❖ Define **join** \sqcup and **meet** \sqcap as:

$$x \sqcup y := \sup(\{x, y\}) \text{ and } x \sqcap y := \inf(\{x, y\})$$

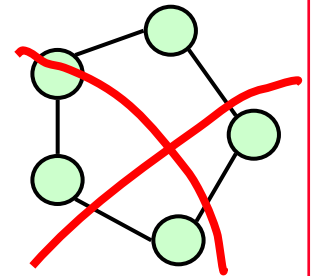
- ❖ **Lattice** \mathcal{L} is a poset (L, \sqsubseteq) where for all $x, y \in L$, $x \sqcap y$ and $x \sqcup y$ exist

- ❖ **Complete lattice** is a lattice where for all $X \subseteq L$, $\sqcap X$ and $\sqcup X$ exist

- ❖ **c-complete lattice** is a complete lattice with complement operator \sim such that $\sim T = \perp$ and $\sim \perp = T$

- ❖ A lattice is **distributive** iff it satisfies the distributive law

$$(\forall x, y, z \in L). (x \sqcap (y \sqcup z) = (x \sqcap y) \sqcup (x \sqcap z))$$



- ❖ Let \mathcal{X}, \mathcal{Y} be posets, then a map $f: \mathcal{X} \rightarrow \mathcal{Y}$ is called **continuous function** if for all non-empty directed sets $Z \subseteq \mathcal{X}$:

$$\sqcup f(Z) = f(\sqcup Z)$$

Some important lemmas

- The *join* and *meet* are **order preserving** functions, i.e. for all $x, y, z, w \in L$ $x \sqsubseteq y$ and $z \sqsubseteq w$ imply $x \sqcup z \sqsubseteq y \sqcup w$

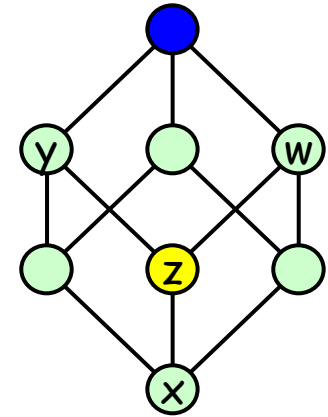
- The **connecting lemma**, for $x, y \in L$

$$x \sqsubseteq y \text{ iff } x \sqcup y = y \text{ iff } x \sqcap y = x$$

- Every **finite** lattice is **complete**

- Every **continuous** function is **order preserving**

- If X, Y are **finite** posets and $f: X \rightarrow Y$ is **order preserving**, then f is **continuous**



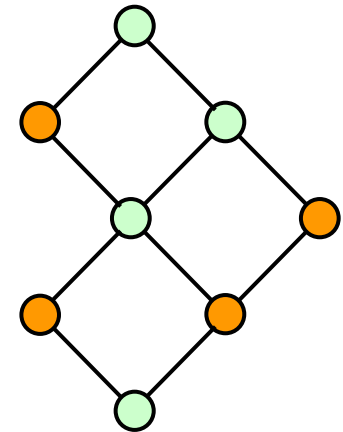
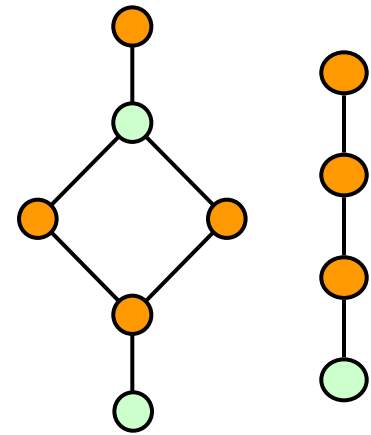
Join irreducible elements

- An element x of a lattice \mathcal{L} is **join irreducible** if
 - (i) $x \neq \perp$
 - (ii) $x = y \sqcup z$ implies $x = y$ or $x = z$ for all $y, z \in L$

- **Every element** of lattice L can be written as a **join of join irreducible** elements, for all $x \in L$:

$$x = \bigsqcup \{y \in J(L) \mid y \sqsubseteq x\}$$

- If L is **distributive** lattice then, the following are equivalent:
 - x is join-irreducible
 - if $y, z \in L$ and $x \sqsubseteq y \sqcup z$ then $x \sqsubseteq y$ or $x \sqsubseteq z$



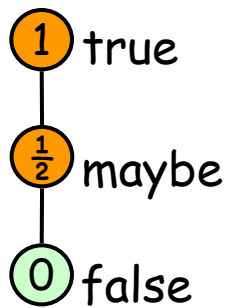
Quasi-Boolean and Boolean Algebras

- A **quasi-Boolean algebra** \mathcal{B} is a structure $\mathcal{B}=(\mathbf{B},\sqcap,\sqcup,\sim,\perp,\top)$; where \top and \perp are the greatest and least elements, $(\mathbf{B},\sqcap,\sqcup)$ is a distributive lattice and \sim is an unary operation of period 2 s.t. for every $x \in \mathbf{B}$ there exists unique $\sim x \in \mathbf{B}$ satisfying:
 - De Morgan laws: $\sim(x \sqcap y) = \sim x \sqcup \sim y$ $\sim(x \sqcup y) = \sim x \sqcap \sim y$
 - Antimonotonic: $x \sqsubseteq y$ iff $\sim y \sqsubseteq \sim x$
 - Involution: $\sim \sim x = x$
- A **Boolean algebra** \mathcal{B} is a quasi-Boolean algebra where for each element $x \in \mathbf{B}$ the following hold:
 - Law of non-contradiction $x \sqcap \sim x = \perp$
 - Law of excluded middle $x \sqcup \sim x = \top$

Quasi-Boolean and Boolean Algebras (examples)

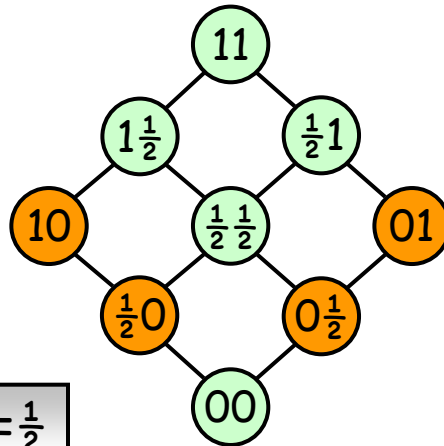
Quasi-Boolean Algebras

$$\mathcal{B}_3 = (\{0, \frac{1}{2}, 1\}, \leq)$$



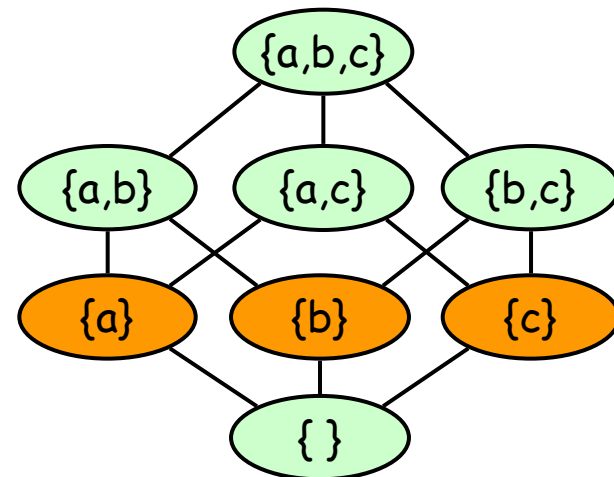
$$\sim 1 = 0, \sim 0 = 1, \sim \frac{1}{2} = \frac{1}{2}$$

$$\mathcal{B}_{3,3} = \mathcal{B}_3 \times \mathcal{B}_3$$



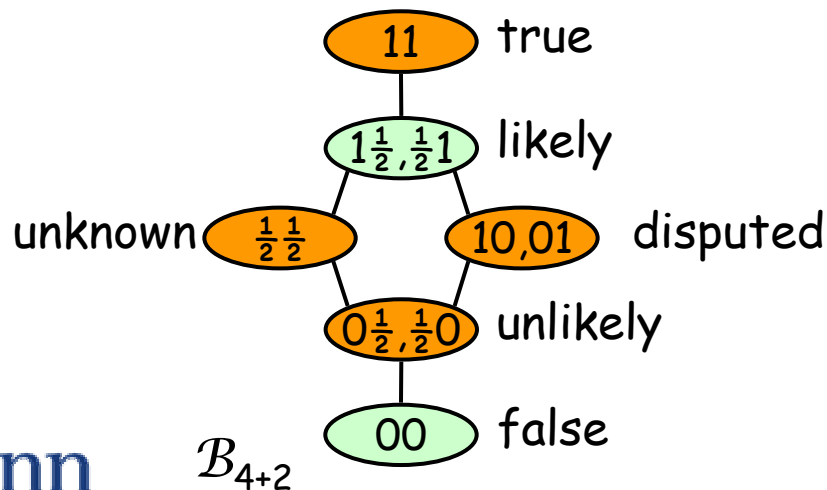
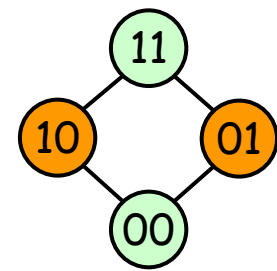
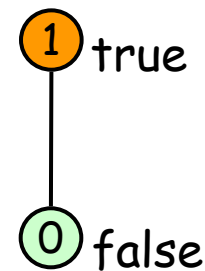
Boolean Algebras

$$\mathcal{B}_S = (2^S, \subseteq), S = \{a, b, c\}$$



$$\mathcal{B}_2 = (\{0, 1\}, \leq)$$

$$\mathcal{B}_{2,2} = \mathcal{B}_2 \times \mathcal{B}_2$$



\mathcal{B}_{4+2}

Tarski-Knaster Fixpoint Theorem

- Let L be a **complete lattice** and $f : L \rightarrow L$ be an **order-preserving function**, then f has fixpoints, i.e. $f(x) = x$. The **least** and **greatest fixpoints** are characterized as follows:

$$\mu x.f(x) = \bigsqcap \{x \in L \mid f(x) = x\} = \bigsqcap \{x \in L \mid f(x) \sqsubseteq x\}$$

$$\nu x.f(x) = \bigsqcup \{x \in L \mid f(x) = x\} = \bigsqcup \{x \in L \mid x \sqsubseteq f(x)\}$$

- Let y, z in L such that **$y \sqsubseteq f(y)$, $y \sqsubseteq \mu x.f(x)$, $f(z) \sqsubseteq z$, $\nu x.f(x) \sqsubseteq z$** and, let f to be **continuous**, then the iteration:

y_i defined as $y_0 := y$ and $y_{i+1} := f(y_i)$ converges to $\mu x.f(x)$

z_i defined as $z_0 := z$ and $z_{i+1} := f(z_i)$ converges to $\nu x.f(x)$

Multi-valued sets and relations

- A **multi-valued set** \mathbb{S} is a total function from the objects of a **set** S to the elements of a **lattice** \mathcal{L} , i.e. $\mathbb{S} : S \rightarrow \mathcal{L}$
 - Intuitively, expresses the "degree" that an object s belongs to a set S
 - Actually, in the two-valued case, i.e. when $\mathcal{L} = \mathcal{B}_2$, it reduces to the characteristic function of the set S
- A **multi-valued relation** \mathbb{R} on sets S and T over a lattice \mathcal{L} is a function $\mathbb{R} : S \times T \rightarrow \mathcal{L}$.

$$\begin{aligned}(\mathbb{S} \cap_L \mathbb{S}') (x) &:= \mathbb{S}(x) \sqcap \mathbb{S}'(x) && \text{(mv-intersection)} \\(\mathbb{S} \cup_L \mathbb{S}') (x) &:= \mathbb{S}(x) \sqcup \mathbb{S}'(x) && \text{(mv-union)} \\ \mathbb{S} \subseteq_L \mathbb{S}' &:= (\forall x). (\mathbb{S}(x) \sqsubseteq \mathbb{S}'(x)) && \text{(mv set inclusion)} \\ \mathbb{S} =_L \mathbb{S}' &:= (\forall x). (\mathbb{S}(x) = \mathbb{S}'(x)) && \text{(mv-equality)}\end{aligned}$$

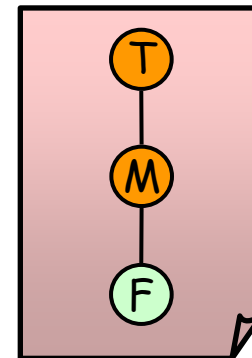
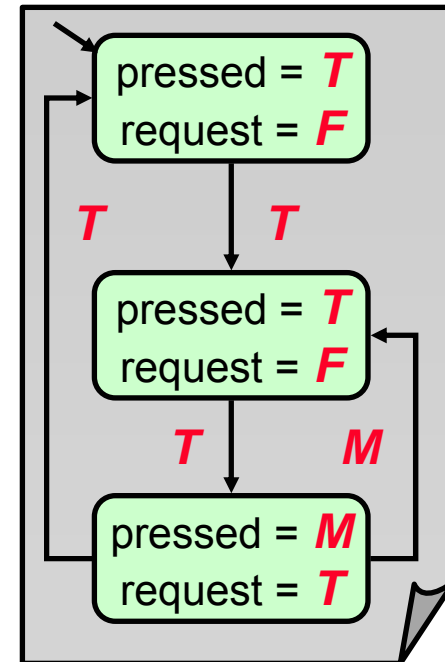
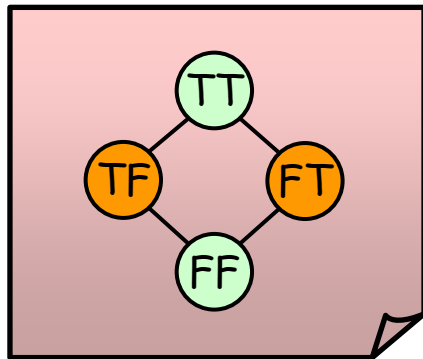
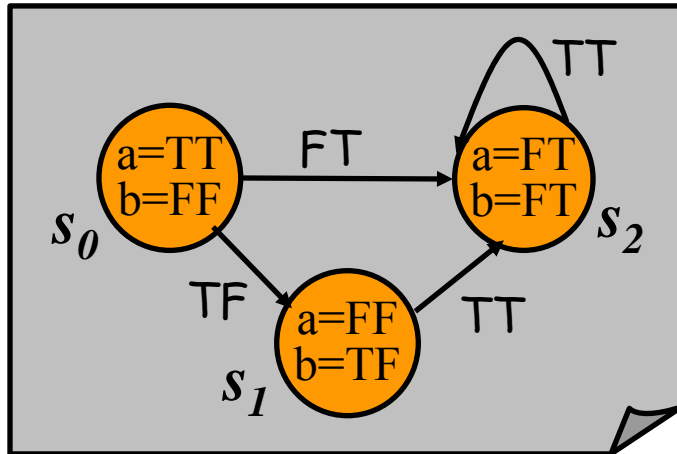
$$\begin{aligned}\overline{\mathbb{S}}(x) &:= \sim (\mathbb{S}(x)) && \text{(mv-complementation)} \\ \overline{\mathbb{S} \cap_B \mathbb{S}'} &= \overline{\mathbb{S}} \cup_B \overline{\mathbb{S}'} && \text{(De-Morgan)} \\ \overline{\mathbb{S} \cup_B \mathbb{S}'} &= \overline{\mathbb{S}} \cap_B \overline{\mathbb{S}'} \\ \mathbb{S} \subseteq_B \mathbb{S}' &= \overline{\mathbb{S}'} \subseteq_B \overline{\mathbb{S}} && \text{(antimonotonicity)}\end{aligned}$$

mv-Kripke Structures

An mv Kripke structure is a tuple $M = (S, S_0, \mathbb{R}, AP, \mathbb{O}, \mathcal{L}, D)$

- S is a (finite) set of states
- S_0 is a set of initial states ($S_0 \subseteq S$)
- $\mathbb{R} : S \times S \rightarrow \mathcal{L}$ is an mv-transition relation
 $(\forall s \in S). (\exists s' \in S). ((s, s') \in \text{Dom}(\mathbb{R}) \wedge \mathbb{R}(s, s') \in D)$
- AP is a (finite) set of atomic propositions
- $\mathbb{O} : S \times AP \rightarrow \mathcal{L}$ is a total labelling function that maps a pair of a state s and an atomic proposition a to an element of the lattice \mathcal{L}
- \mathcal{L} is a lattice or an algebra
- D is the set of designated values

mv-Kripke Structures (Examples)



Predecessor mv-sets

- The **existential predecessor set**:

$$pre_{\exists}^{\mathbb{R}}(Q)(s_1) := \bigsqcup_{s_2 \in S_2} (\mathbb{R}(s_1, s_2) \sqcap Q(s_2))$$

- The **universal predecessor set**:

- Bruns & Godefroid and Chechik et. al.

$$pre_{\forall}^{\mathbb{R}}(Q)(s_1) := \prod_{s_2 \in S_2} (\mathbb{R}(s_1, s_2) \Rightarrow Q(s_2)) \quad (\text{def. 1})$$

- Konikowska & Penczek

$$pre_{\forall}^{\mathbb{R}}(Q)(s_1) := \prod_{\{s_2 \in S_2 \mid \mathbb{R}(s_1, s_2) \in D\}} (\mathbb{R}(s_1, s_2) \sqcap Q(s_2)) \quad (\text{def. 2})$$

- Compare with **classical definition**:

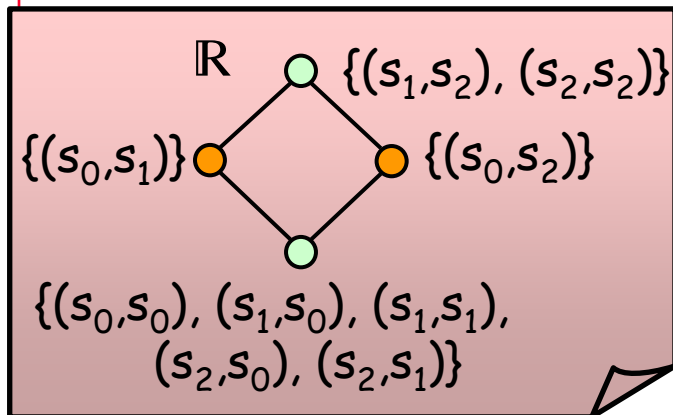
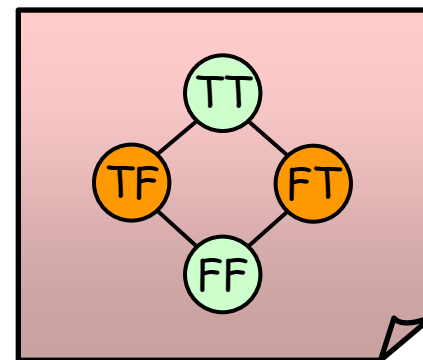
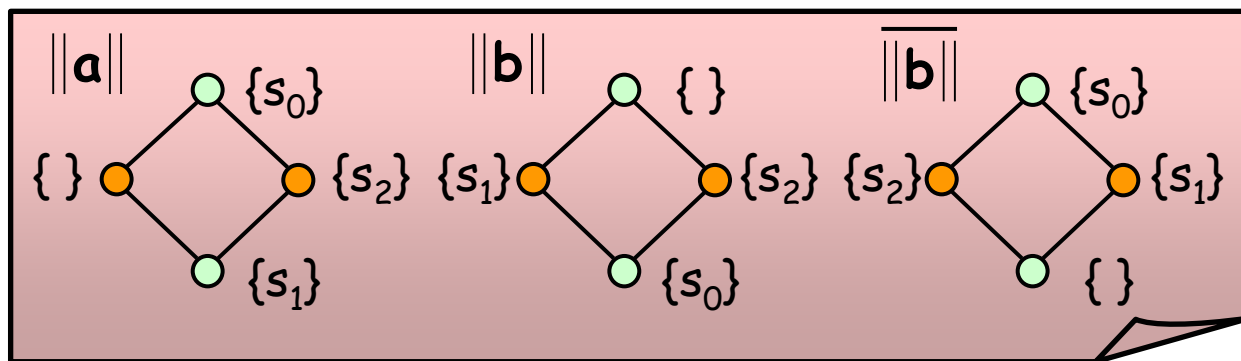
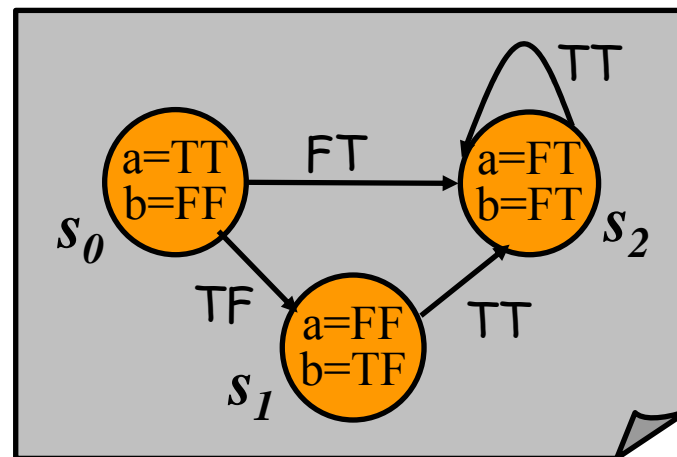
$$pre_{\exists}^R(Q) := \{s_1 \in S_1 \mid (\exists s_2).((s_1, s_2) \in R \wedge s_2 \in Q)\}$$

$$pre_{\forall}^R(Q) := \{s_1 \in S_1 \mid (\forall s_2).((s_1, s_2) \in R \rightarrow s_2 \in Q)\}$$

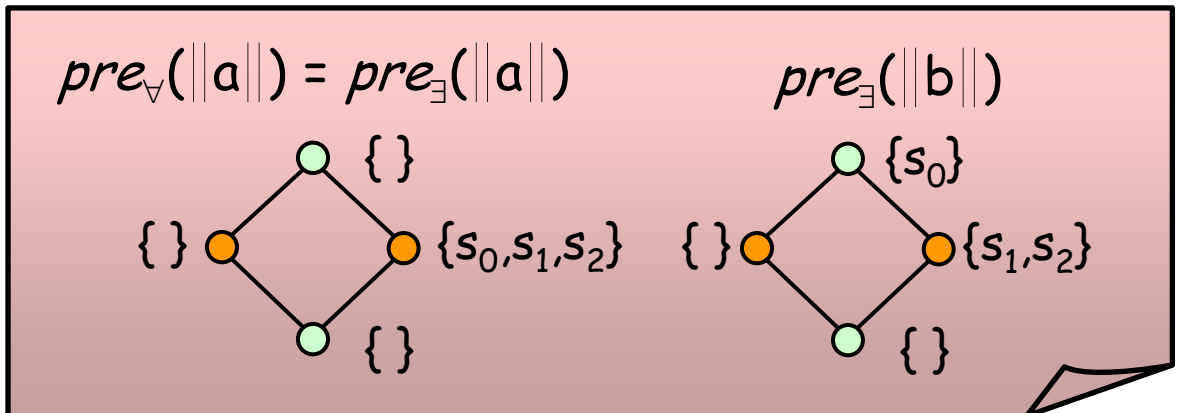
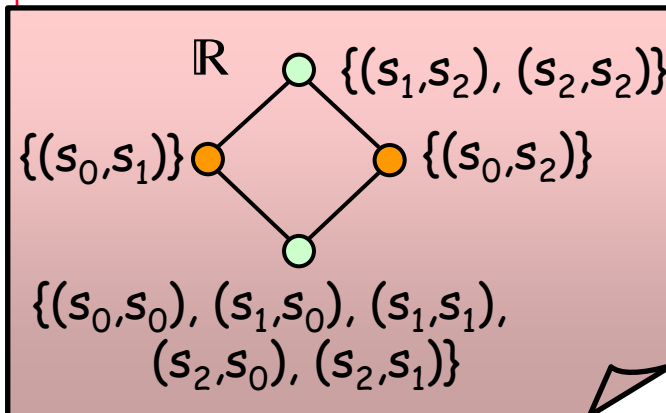
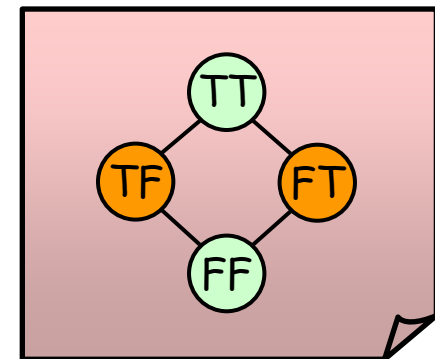
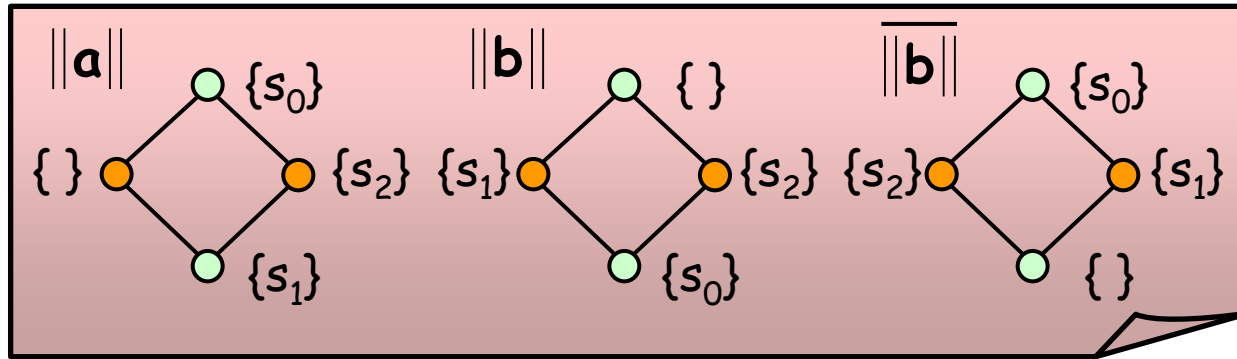
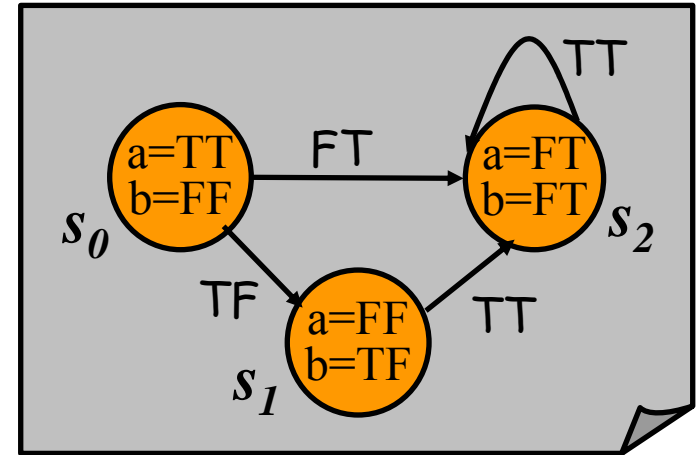
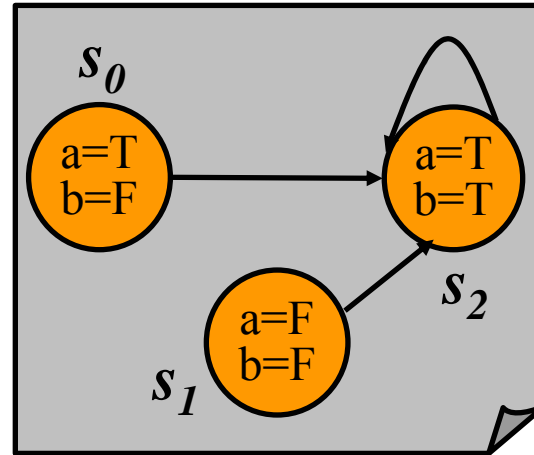
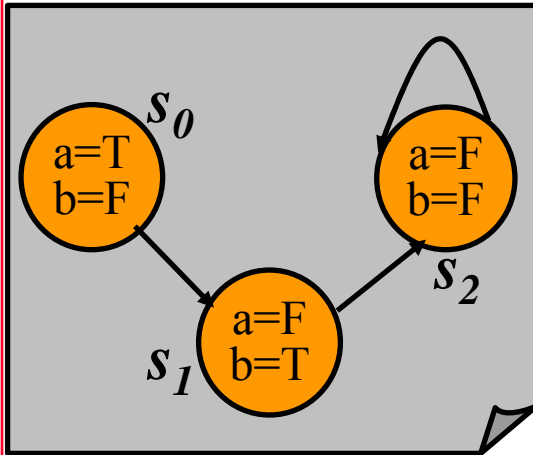
Example

- For any $a \in AP$, we denote by $\|a\| : S \rightarrow L$ the mv-set that represents the "degree" that the proposition a is satisfied in some state s
- The mv-set $\|a\|$ introduces a partition of the state space

Example from Chechik et al



Example from Chechik et al



The multi-valued model checking problem

Given multi-valued system $M = (S, S_0, \mathbb{R}, AP, \mathbb{O}, \mathcal{L}, D)$ and a specification φ

Multi-valued model checking problem

$$(\forall s \in S_0). (\|\varphi\|_M(s) \in D)$$

Alternative:

Given multi-valued system $M = (S, S_0, \mathbb{R}, AP, \mathbb{O}, \mathcal{L}, D)$, state s in S and specification φ determine $\|\varphi\|_M(s)$

The multi-valued model checking problem

Two main approaches

- Reduction methods to classical model checking
 - ❖ [Bruns and Godefroid] Reduction for multi-valued μ -calculus
 - ❖ [Chechik et. Al.] Reductions for multi-valued LTL, μ -calculus
 - ❖ [Konikowska and Penczek] Reduction methods for
 - ❖ mv-CTL* using designated values
 - ❖ mv-CTL* for FLO and specific lattices ($L_{2,2}, L_{4+2}, \text{etc}$)
 - ❖ μ -calculus

- Direct methods
 - ❖ [Bruns and Godefroid] Extended alternating automata
 - ❖ [Chechik et. Al.] Multi-valued CTL symbolic model checking

Temporal Logics (1)

CTL* syntax

$$\phi_s ::= a \mid \neg\phi_s \mid \phi_s \vee \phi_s \mid E[\phi_p]$$

$$\phi_p ::= \phi_s \mid \neg\phi_p \mid \phi_p \vee \phi_p \mid X\phi_p \mid [\phi_p \mathcal{U}\phi_p]$$

Derived operators

$$\phi_1 \rightarrow \phi_2 := \neg\phi_1 \vee \phi_2 \quad \phi_1 \leftrightarrow \phi_2 := \phi_1 \rightarrow \phi_2 \wedge \phi_2 \rightarrow \phi_1$$

$$A[\phi] = \neg E[\neg\phi]$$

$$\text{Eventually: } F\phi = [1 \mathcal{U}\phi]$$

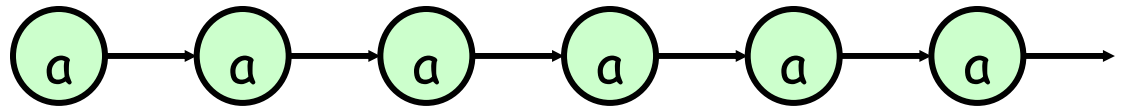
$$\text{Before (weak): } [\phi_1 \mathcal{B}\phi_2] = \neg[\neg\phi_1 \mathcal{U}\phi_2]$$

$$\text{Release: } [\phi_1 \mathcal{R}\phi_2] = \neg[\neg\phi_1 \mathcal{U}\neg\phi_2]$$

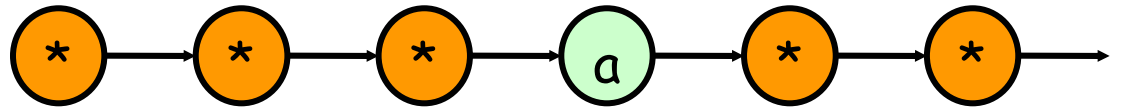
$$\text{Always: } G\phi = [0 \mathcal{B}\neg\phi] \text{ or } G\phi = [0 \mathcal{R}\phi]$$

Temporal logics (2): Semantic Intuition of Linear time properties

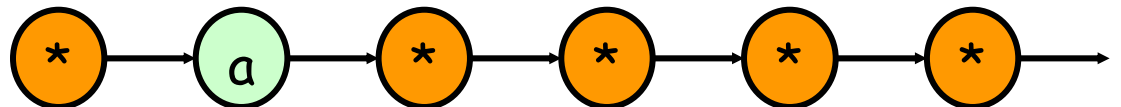
$G a$ - always a



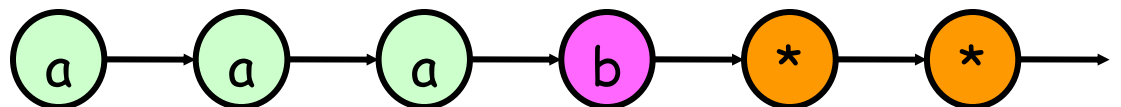
$F a$ - eventually a



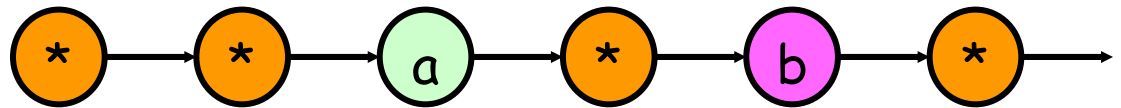
$X a$ - next state a



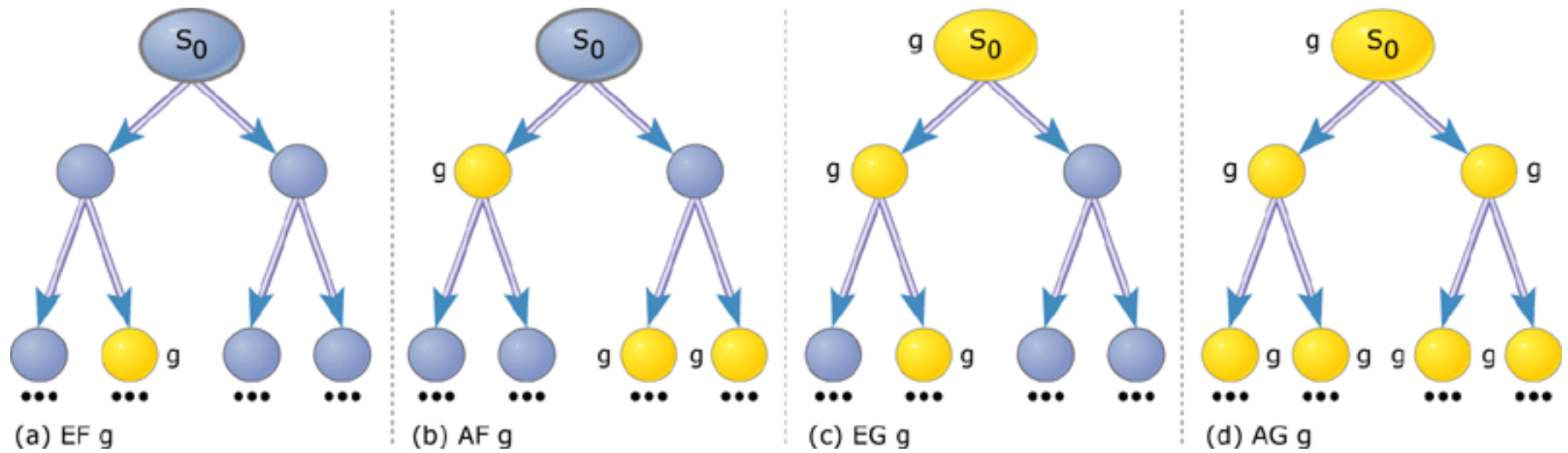
$a U b$ - a until b



$a B b$ - a before b



Temporal Logics (3): Semantic intuition of branching temporal properties



Mv-CTL* model checking using designated values (1)

Semantics of mv-CTL* in ~~Negation Normal Form (NINF)~~

$$AP^+ = AP \cup \overline{AP}$$

$$\overline{AP} = \{\neg a \mid a \in AP\}$$

$$a \in AP^+$$

State formulas

$$\|\neg\phi\| := \sim \|\phi\|$$

$$\|l\|(s) := l$$

$$\|a\| := \mathbb{O}_a$$

$$\|\phi_s \vee \psi_s\| := \|\phi_s\| \cup_L \|\psi_s\|$$

$$\|\phi_s \wedge \psi_s\| := \|\phi_s\| \cap_L \|\psi_s\|$$

$$\|E\phi_p\|(s) := \bigsqcup_{\pi \in Paths_{\mathcal{M}}(s)} \|\phi_p\|(\pi[0])$$

$$\|A\phi_p\|(s) := \bigsqcap_{\pi \in Paths_{\mathcal{M}}(s)} \|\phi_p\|(\pi[0])$$

Path formulas

$$\|\phi_s\|(\pi[i]) := \|\phi_s\|(\pi(i))$$

$$\|\phi_p \vee \psi_p\|(\pi[i]) := \|\phi_p\|(\pi[i]) \sqcup \|\psi_p\|(\pi[i])$$

$$\|\phi_p \wedge \psi_p\|(\pi[i]) := \|\phi_p\|(\pi[i]) \sqcap \|\psi_p\|(\pi[i])$$

$$\|X\phi_p\|(\pi[i]) := \mathbb{R}(\pi(i), \pi(i+1)) \sqcap \|\phi_p\|(\pi[i+1])$$

$$\|\phi_p \mathcal{U} \psi_p\|(\pi[i]) := \|\psi_p\|(\pi[i]) \sqcup \bigsqcup_{j>i} (\|\phi_p\|(\pi[i]) \sqcap \bigsqcap_{i<k<j} (\mathbb{R}(\pi(k-1), \pi(k)) \sqcap \|\phi_p\|(\pi[k])) \sqcap \bigsqcap_{i<k<j} (\mathbb{R}(\pi(j-1), \pi(j)) \sqcap \|\psi_p\|(\pi[j])))$$

$$\|\phi_p \mathcal{R} \psi_p\|(\pi[i]) := \|\psi_p\|(\pi[i]) \sqcap \bigsqcap_{j>i} (\|\phi_p\|(\pi[i]) \sqcup \bigsqcup_{i<k<j} (\mathbb{R}(\pi(k-1), \pi(k)) \sqcap \|\phi_p\|(\pi[k])) \sqcup \bigsqcup_{i<k<j} (\mathbb{R}(\pi(j-1), \pi(j)) \sqcap \|\psi_p\|(\pi[j])))$$

Mv-CTL* model checking using designated values (2)

❖ **Theorem 1 (Reduction from NNF mv-CTL* to CTL* using Designated Values)** Assume that L is a c -complete lattice.

- ❖ Let the designated values D and non-designated values N be closed under arbitrary bounds.
- ❖ Define $\tau : M = (S, S_0, R, AP^+, \mathcal{O}, \mathcal{L}, D) \rightarrow K = (S, S_0, R, AP^+, \mathcal{O})$

such that:

- 1) $R = \{(s, s') \in S^2 \mid \mathbb{R}(s, s') \in D\}$
- 2) for any $a \in AP^+$ it is $\|a\|_{\mathcal{K}}(s) = 1$ iff $\|a\|_{\mathcal{M}}(s) \in D$

- ❖ Then for any state formula ϕ_s and any path formula ϕ_p of NNF mv-CTL* over the lattice L and any state s in S and path π in $\text{Paths}_{\mathcal{M}}(s)$ of M , we have:

$$\begin{aligned} \|\phi_s\|_{\mathcal{M}}(s) \in D & \quad \text{iff} \quad (\mathcal{K}, s) \models_s \phi_s \\ \|\phi_p\|_{\mathcal{M}}(\pi[0]) \in D & \quad \text{iff} \quad (\mathcal{K}, \pi[0]) \models_p \phi_p \end{aligned}$$

Mv-CTL* model checking using designated values (3)

Sketch of proof:

- ✓ Notice that the paths on M and K are the same
- ✓ For any subset L_s of L the following properties hold:

$$1) \bigsqcup L_s \in D \text{ iff } (\exists l \in L_s).(l \in D)$$

$$2) \prod L_s \in D \text{ iff } (\forall l \in L_s).(l \in D)$$

- ✓ Proof proceeds by **induction on the structure of φ** , some cases:

- ✓ $\varphi = a$, a in AP^+ , then holds by definition
- ✓ $\varphi = \varphi_1 \vee \varphi_2$, then $\|\varphi\|_M(s) = \|\varphi_1\|_M(s) \sqcup \|\varphi_2\|_M(s) \in D$ iff (property 1) $(\exists i).$
 $(\|\varphi_i\|_M(s) \in D)$ iff (IH) $(K, s) \models \varphi_i$ implies $(K, s) \models \varphi_1 \vee \varphi_2 = \varphi$
- ✓ $\varphi = [\varphi_1 \cup \varphi_2]$, then $\|\varphi\|_M(\pi[i]) \in D$ iff (property 1) there exists $j > i + 1$ s.t.
 $(\mathbb{R}(\pi(j-1), \pi(j)) \sqcap \|\varphi_2\|_M(\pi[j])) \in D$ iff (as $\mathbb{R}(\pi(j-1), \pi(j)) \in D$ and D is closed
under bounds) $\|\varphi_2\|_M(\pi[j]) \in D$ and (property 2) for all $0 < k < j$ $(\mathbb{R}(\pi(k-1),$
 $\pi(k)) \sqcap \|\varphi_1\|_M(\pi[k])) \in D$ iff $\|\varphi_1\|_M(\pi[k]) \in D$ iff (IH) on the same path π ,
 $(K, \pi[j]) \models \varphi_2$ and for all $0 < k < j$ $(K, \pi[k]) \models \varphi_1$ which by definition is
 $(K, \pi[0]) \models [\varphi_1 \cup \varphi_2] = \varphi$

Mv-CTL* model checking using designated values (4)

- ❖ **Theorem 2 (Reduction from mv-CTL* to CTL* using Designated Values)** Assume that L is a c -complete lattice.
 - ❖ Let the designated values D and non-designated values N be closed under arbitrary bounds.
 - ❖ $x \in D$ implies $\sim x \in N$ and $x \in N$ implies $\sim x \in D$
 - ❖ Define $\tau : M = (S, S_0, \mathbb{R}, AP, \mathbb{O}, \mathcal{L}, D) \rightarrow K = (S, S_0, R, AP, \mathbb{O})$

such that:

- 1) $R = \{(s, s') \in S^2 \mid \mathbb{R}(s, s') \in D\}$
- 2) for any $a \in AP$ it is $\|a\|_{\mathcal{K}}(s) = 1$ iff $\mathbb{O}_a(s) \in D$

- ❖ Then for any state formula ϕ_s and any path formula ϕ_p of NNF mv-CTL* over the lattice L and any state s in S and path π in $\text{Paths}_{\mathcal{M}}(s)$ of M , we have:

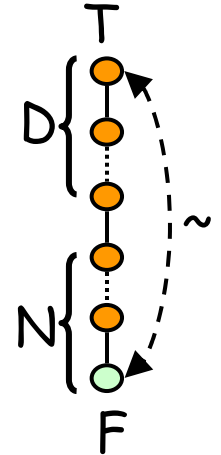
$$\begin{aligned} \|\phi_s\|_{\mathcal{M}}(s) \in D & \quad \text{iff} \quad (\mathcal{K}, s) \models_s \phi_s \\ \|\phi_p\|_{\mathcal{M}}(\pi[0]) \in D & \quad \text{iff} \quad (\mathcal{K}, \pi[0]) \models_p \phi_p \end{aligned}$$

- ❖ **Proof:** The only additional case is for the complementation

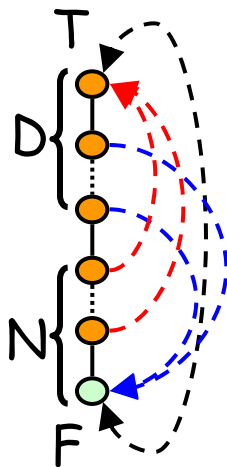
Mv-CTL* model checking using designated values (5)

Examples:

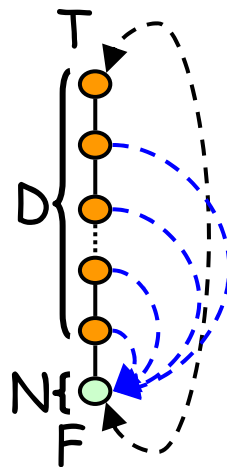
- **Theorem 1:** The condition that D and N should be closed under arbitrary bounds is satisfied by logics over finite linear orders
 - i.e. 3-valued Kleene logic, many-value Lukasiewicz logics etc
- **Theorem 2:** The conditions are satisfied by:
 - Logics over finite linear orders
 - Logic over the lattice $L_{2,2}$



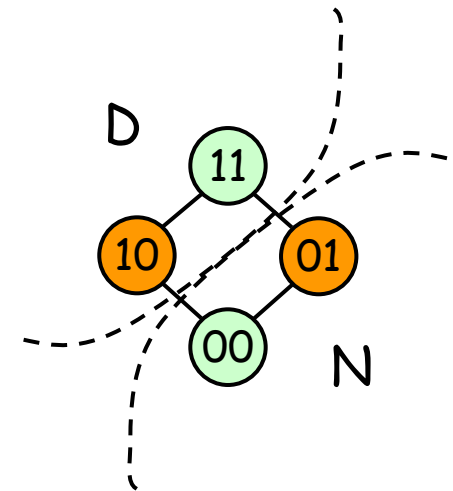
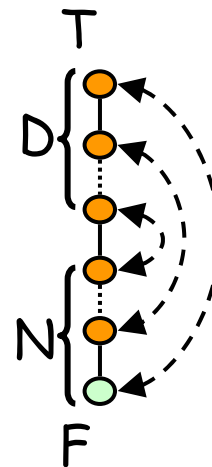
Rosser-Turquette



Gödel



Lukasiewicz



Mv-CTL* model checking using designated values (6)

Remarks

- ✓ The **complexity** of mv-CTL* model checking is the same as the two-valued case
- ✓ The complexity of CTL* model checking is $O(|K| \times 2^{|\phi|})$
 - ✓ A combination of LTL and CTL model checking algorithms
- ✓ Due to the construction a **counter-example** in K is a counter-example in M
- ✓ The approach is helpful as long as we do not care about the exact value
- ✓ If the conditions of theorem 2 are satisfied then the 2 definitions of the predecessor sets coincide for the designated values

The propositional two-valued μ -calculus

Syntax

$\phi ::= a \mid X \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \diamond\phi \mid \square\phi \mid \mu X.\phi \mid \nu X.\phi$

Semantics

$$\begin{aligned} \llbracket l \rrbracket_{\mathcal{K}e} &= l, \quad l \in \{0, 1\} \\ \llbracket a \rrbracket_{\mathcal{K}e} &= \{s \in S \mid a \in \mathcal{O}(s)\} \\ \llbracket X \rrbracket_{\mathcal{K}e} &= e(X) \\ \llbracket \neg\phi \rrbracket_{\mathcal{K}e} &= S \setminus \llbracket \phi \rrbracket_{\mathcal{K}e} = \overline{\llbracket \phi \rrbracket_{\mathcal{K}e}} \\ \llbracket \phi \vee \psi \rrbracket_{\mathcal{K}e} &= \llbracket \phi \rrbracket_{\mathcal{K}e} \cup \llbracket \psi \rrbracket_{\mathcal{K}e} \\ \llbracket \phi \wedge \psi \rrbracket_{\mathcal{K}e} &= \llbracket \phi \rrbracket_{\mathcal{K}e} \cap \llbracket \psi \rrbracket_{\mathcal{K}e} \\ \llbracket \diamond\phi \rrbracket_{\mathcal{K}e} &= \text{pre}_{\exists}^R(\llbracket \phi \rrbracket_{\mathcal{K}e}) \\ \llbracket \square\phi \rrbracket_{\mathcal{K}e} &= \text{pre}_{\forall}^R(\llbracket \phi \rrbracket_{\mathcal{K}e}) \\ \llbracket \mu X.\phi(X) \rrbracket_{\mathcal{K}e} &= \bigcap \{Q \in 2^S \mid \llbracket \phi(X) \rrbracket_{\mathcal{K}e}[X \leftarrow Q] \subseteq Q\} \\ \llbracket \nu X.\phi(X) \rrbracket_{\mathcal{K}e} &= \bigcup \{Q \in 2^S \mid Q \subseteq \llbracket \phi(X) \rrbracket_{\mathcal{K}e}[X \leftarrow Q]\} \end{aligned}$$

mv- μ -Calculus Model Checking by Reduction (1)

Semantics of mv- μ -calculus in NNF wrt to mv-model M

- Atomic propositions and mv-transition relation take values over a quasi-Boolean algebra B

$$\begin{aligned} \|l\|_{\mathcal{M}\varepsilon}(s) &:= l \\ \|a\|_{\mathcal{M}\varepsilon} &:= \mathbb{O}_a \quad a \in AP^+ \begin{cases} AP^+ = AP \cup \overline{AP} \\ \overline{AP} = \{\neg a \mid a \in AP\} \end{cases} \\ \|X\|_{\mathcal{M}\varepsilon} &:= \varepsilon(X) \\ \|\phi \vee \psi\|_{\mathcal{M}\varepsilon} &:= \|\phi\|_{\mathcal{M}\varepsilon} \cup_B \|\psi\|_{\mathcal{M}\varepsilon} \\ \|\phi \wedge \psi\|_{\mathcal{M}\varepsilon} &:= \|\phi\|_{\mathcal{M}\varepsilon} \cap_B \|\psi\|_{\mathcal{M}\varepsilon} \\ \|\diamond\phi\|_{\mathcal{M}\varepsilon} &:= pre_{\exists}^{\mathbb{R}}(\|\phi\|_{\mathcal{M}\varepsilon}) \\ \|\square\phi\|_{\mathcal{M}\varepsilon} &:= pre_{\forall}^{\mathbb{R}}(\|\phi\|_{\mathcal{M}\varepsilon}) \\ \|\mu X.\phi(X)\|_{\mathcal{M}\varepsilon} &:= \bigcap_B \{Q \in B^S \mid \|\phi(X)\|_{\mathcal{M}\varepsilon}[X \leftarrow Q] \subseteq_B Q\} \\ \|\nu X.\phi(X)\|_{\mathcal{M}\varepsilon} &:= \bigcup_B \{Q \in B^S \mid Q \subseteq_B \|\phi(X)\|_{\mathcal{M}\varepsilon}[X \leftarrow Q]\} \end{aligned}$$

mv- μ -Calculus Model Checking by Reduction (2)

- Assume that the transition relation \mathbb{R} is 2-valued (denoted by R)
- Define translation:
 - $\tau : M = (S, S_0, R, AP^+, \mathbb{O}, \mathfrak{B}, D) \rightarrow K_x = (S, S_0, R, AP^+, O_x)$
 - For all $s \in S$ and for some $x \in \mathfrak{B}$ $a \in O_x(s)$ iff $x \sqsubseteq O_a(s)$
- **Proposition:** Let M be a mv-Kripke structure over a finite distributive lattice L , φ an mv- μ -calculus formula in NNF, s in S and x, x' in L , then $(\|\varphi\|_{K_x} e)(s) = 1$ and $x' \sqsubseteq x$ imply $(\|\varphi\|_{K_{x'}} e)(s) = 1$.
 - **Proof:** Straightforward double induction on the **alternation depth** and the **structure** of the formula φ .
- **Main Result (Theorem):** Let M be a mv-Kripke structure over a finite distributive lattice L , φ an mv- μ -calculus formula in NNF, s in S , then $(\|\varphi\|_{M\varepsilon})(s) = \sqcup\{x \in J(L) \mid (\|\varphi\|_{K_x} e)(s) = 1\}$
 - **Proof:** Every element of lattice L can be written as a **join of join irreducible** elements, i.e. $(\|\varphi\|_{M\varepsilon})(s) = \sqcup\{x \in J(L) \mid x \sqsubseteq (\|\varphi\|_{M\varepsilon})(s)\}$

mv- μ -Calculus Model Checking by Reduction (3)

- **Lemma:** Let M be a mv-Kripke structure over a finite distributive lattice L , φ an mv- μ -calculus formula in NNF, s in S and x in $J(L)$, then $(\|\varphi\|_{Kx}e)(s) = 1$ iff $x \sqsubseteq (\|\varphi\|_{M\varepsilon})(s)$.

Proof:

- By induction on the **alternation depth n** of the formula φ . Let $n=0$, we proceed by induction on the **structure** of φ , case
 - $\varphi = a \in AP^+$ by definition
 - $\varphi = \varphi_1 \vee \varphi_2$, then $\|\varphi_1 \vee \varphi_2\|_{Kx}e(s) = 1$ iff $\|\varphi_1\|_{Kx}e(s) = 1$ or $\|\varphi_2\|_{Kx}e(s) = 1$ iff (IH) $x \sqsubseteq \|\varphi_1\|_{M\varepsilon}(s)$ or $x \sqsubseteq \|\varphi_2\|_{M\varepsilon}(s)$ **iff(*)** $x = x \sqcup x \sqsubseteq \|\varphi_1\|_{M\varepsilon}(s) \sqcup \|\varphi_2\|_{M\varepsilon}(s)$ iff $x \sqsubseteq \|\varphi_1 \vee \varphi_2\|_{M\varepsilon}(s)$
- Consider **alternation depth $n+1$** and proceed by induction on the **structure** of φ , case
 - $\varphi = \mu X. \psi(X)$, then $\|\mu X. \psi(X)\|_{Kx}e(s) = 1$ iff $s \in (f_{Kx, \psi})^{|\mathcal{S}|+1}(\emptyset)$. Also, $x \sqsubseteq \|\mu X. \psi(X)\|_{M\varepsilon}(s)$ iff $x \sqsubseteq (f_{M, \psi})^{|\mathcal{S}|+1}(\perp)$. By IH $s \in (f_{Kx, \psi})^{|\mathcal{S}|+1}(\emptyset)$ iff $x \sqsubseteq (f_{M, \psi})^{|\mathcal{S}|+1}(\perp)$.

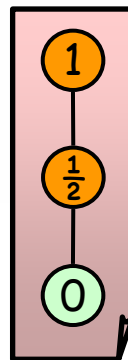
mv- μ -Calculus Model Checking by Reduction (4)

Reduction algorithm for mv- μ -calculus

```
1: procedure REDUCEMUCALC( $\mathcal{M}, \phi$ )
2:    $A \leftarrow \mathcal{J}(\mathcal{L})$ 
3:    $B \leftarrow \emptyset$ 
4:   while  $A \neq \emptyset$  do
5:      $x \leftarrow$  maximal element of  $A$ 
6:     if  $s \in \llbracket \phi \rrbracket_{\mathcal{K}_x} e$  then
7:        $C \leftarrow \{x' \in \mathcal{J}(\mathcal{L}) \mid x' \sqsubseteq x\}$ 
8:        $B \leftarrow B \cup C$ 
9:        $A \leftarrow A \setminus C$ 
10:    else
11:       $A \leftarrow A \setminus \{x\}$ 
12:    end if
13:  end while
14:  return  $B$ 
15: end procedure
```

Reduction method for the mv- μ -calculus calls at most $|\mathcal{J}(\mathcal{L})|$ times the μ -calculus model checker

The running time of the naive μ -calculus model checking algorithm is: $O(|\phi| \times |K| \times |S|^{\text{nest}(\phi)})$



Example: The Kripke structure K_1 expresses the pessimistic viewpoint that $\frac{1}{2}$ is false, while $K_{\frac{1}{2}}$ expresses the optimistic viewpoint that both the values 1 and $\frac{1}{2}$ are true. If K_1 satisfies ϕ then $(\llbracket \phi \rrbracket_{M\varepsilon})(s) = \sqcup\{1, \frac{1}{2}\} = 1$. If $K_{\frac{1}{2}}$ satisfies ϕ then $(\llbracket \phi \rrbracket_{M\varepsilon})(s) = \sqcup\{\frac{1}{2}\} = \frac{1}{2}$.

Direct mv-CTL Model Checking (1)

CTL Syntax

$$\phi_s ::= \pi \mid \neg\phi_s \mid \phi_s \vee \phi_s \mid EX\phi_s \mid EG\phi_s \mid E[\phi_s \mathcal{U}\phi_s]$$

Semantics of mv-CTL wrt mv-model M

- Atomic propositions and mv-transition relation take values over a quasi-Boolean algebra B

$$\|b\|(s) := b \text{ for } b \in B \text{ and } s \in S$$

$$\|a\| := \mathbb{O}_a \text{ for } a \in AP$$

$$\|\neg\phi\| := \overline{\|\phi\|}$$

$$\|\phi \vee \psi\| := \|\phi\| \cup_B \|\psi\|$$

$$\|EX\phi\| := pre_{\exists}^{\mathbb{R}}(\|\phi\|)$$

$$\|EG\phi\| := \nu\mathbb{Z}.\|\phi\| \cap_B \|EX\mathbb{Z}\|$$

$$\|E[\phi \mathcal{U}\psi]\| := \mu\mathbb{Z}.\|\psi\| \cup_B (\|\phi\| \cap_B \|EX\mathbb{Z}\|)$$

$$\text{where } \|EX\mathbb{Z}\| = pre_{\exists}^{\mathbb{R}}(\mathbb{Z})$$

Direct mv-CTL Model Checking (2)

mv-CTL symbolic model checking algorithm

```

1: procedure  $Check_{\mathcal{M}}(\phi)$ 
2:   Case  $\phi$ 
3:      $a \in AP$  return  $\|a\|_{\mathcal{M}}$ 
4:      $\neg\phi_1$  return  $\sim Check_{\mathcal{M}}(\phi_1)$ 
5:      $\phi_1 \vee \phi_2$  return  $Check_{\mathcal{M}}(\phi_1) \sqcup Check_{\mathcal{M}}(\phi_2)$ 
6:      $EX\phi_1$  return  $CheckEX(Check_{\mathcal{M}}(\phi_1))$ 
7:      $EG\phi_1$  return  $FxPoint(\top, f_{Check_{\mathcal{M}}(\phi_1)}(Z))$ 
8:      $E[\phi_1 \mathcal{U} \phi_2]$  return  $FxPoint(\perp, f_{Check_{\mathcal{M}}(\phi_1), Check_{\mathcal{M}}(\phi_2)}(Z))$ 
9:   End Case
10: end procedure

```

$\triangleright \mathcal{M} = (S, S_0, \mathbb{R}, AP, \odot, B, D)$ is the mv-Kripke Structure
 $\triangleright \|a\|_{\mathcal{M}}$ is in symbolic representation
 \triangleright symbolic complementation
 \triangleright symbolic join operation
 $\triangleright CheckEX(\|\phi\|) = pre_{\exists}^{\mathbb{R}}(\|\phi\|)$
 $\triangleright f_{\|\phi_1\|}(Z) = \|\phi_1\| \sqcap \|EX Z\|$
 $\triangleright f_{\|\phi_1\|, \|\phi_2\|}(Z) = \|\phi_2\| \sqcup (\|\phi_1\| \sqcap \|EX Z\|)$

```

1: procedure  $FXPOINT(x, f)$ 
2:    $x' = f(x)$ 
3:   while  $x \neq x'$  do
4:      $x = x'$ 
5:      $x' = f(x)$ 
6:   end while
7:   return  $x$ 
8: end procedure

```

The **running time** of the mv-CTL symbolic model checking algorithm is:

$$O(|\varphi| \times |S| \times |M| \times t_L)$$

Direct mv-CTL Model Checking (3)

Derived operators

$$\|\phi \wedge \psi\| := \overline{\overline{\|\phi\|} \cup_B \overline{\|\psi\|}} = \|\phi\| \cap_B \|\psi\|$$

$$\|AX\phi\| := pre_{\forall}^{\mathbb{R}}(\|\phi\|) = \overline{\|EX\neg\phi\|}$$

$$\|AF\phi\| := \|A[\top \mathcal{U}\phi]\|$$

$$\|EF\phi\| := \|E[\top \mathcal{U}\phi]\|$$

$$\|AG\phi\| := \overline{\|EF\neg\phi\|}$$

$$\|A[\phi \mathcal{U}\psi]\| := \overline{\|E[\neg\psi \mathcal{U}\neg\phi \wedge \neg\psi]\|} \cap_B \overline{\|EG\neg\psi\|}$$

$$\|E[\phi \mathcal{B}\psi]\| := \overline{\|A[\neg\phi \mathcal{U}\psi]\|}$$

$$\|A[\phi \mathcal{B}\psi]\| := \overline{\|E[\neg\phi \mathcal{U}\psi]\|}$$

Derived fixpoint properties

$$\|AF\phi\| = \mu Z. \|\phi\| \cup_B \|AX Z\|$$

$$\|EF\phi\| = \mu Z. \|\phi\| \cup_B \|EX Z\|$$

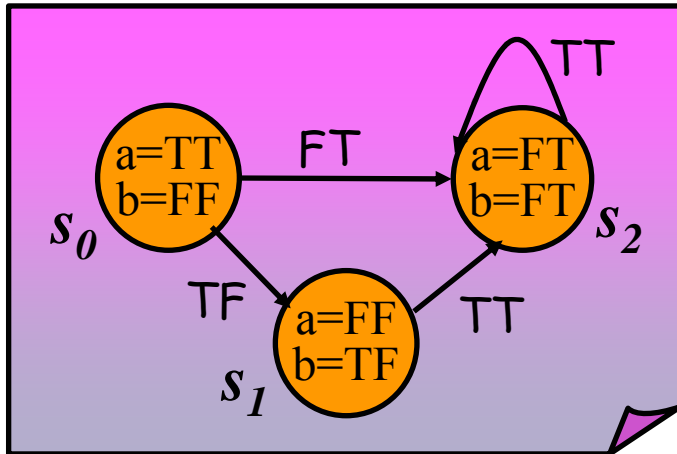
$$\|AG\phi\| = \nu Z. \|\phi\| \cap_B \|AX Z\|$$

$$\|A[\phi_1 \mathcal{U}\phi_2]\| = \mu Z. \|\phi_2\| \cup_B (\|\phi_1\| \cap_B \|AX Z\|)$$

$$\|A[\phi_1 \mathcal{B}\phi_2]\| = \nu Z. \overline{\|\phi_2\|} \cap_B (\|\phi_1\| \cup_B \|AX Z\|)$$

$$\|E[\phi_1 \mathcal{B}\phi_2]\| = \nu Z. \overline{\|\phi_2\|} \cap_B (\|\phi_1\| \cup_B \|EX Z\|)$$

Direct mv-CTL Model Checking (4)

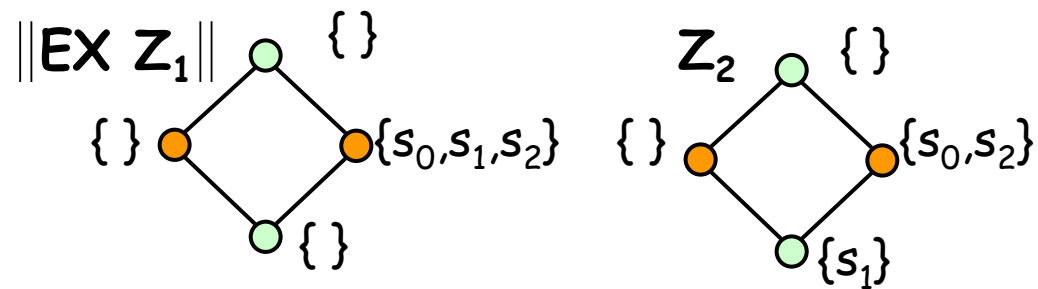
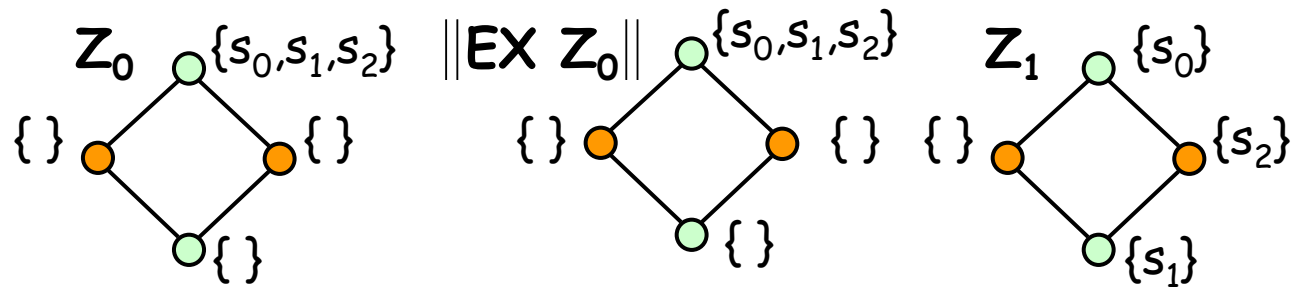
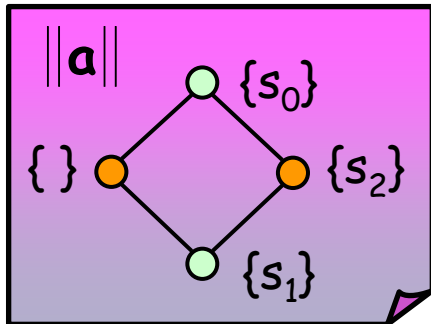


We want to model check the specification:

$$\|EG a\|_M$$

We use the fixpoint:

$$\|EG a\| = \nu Z. \|a\| \cap_B \|EX Z\|$$



Direct mv-CTL Model Checking (5)

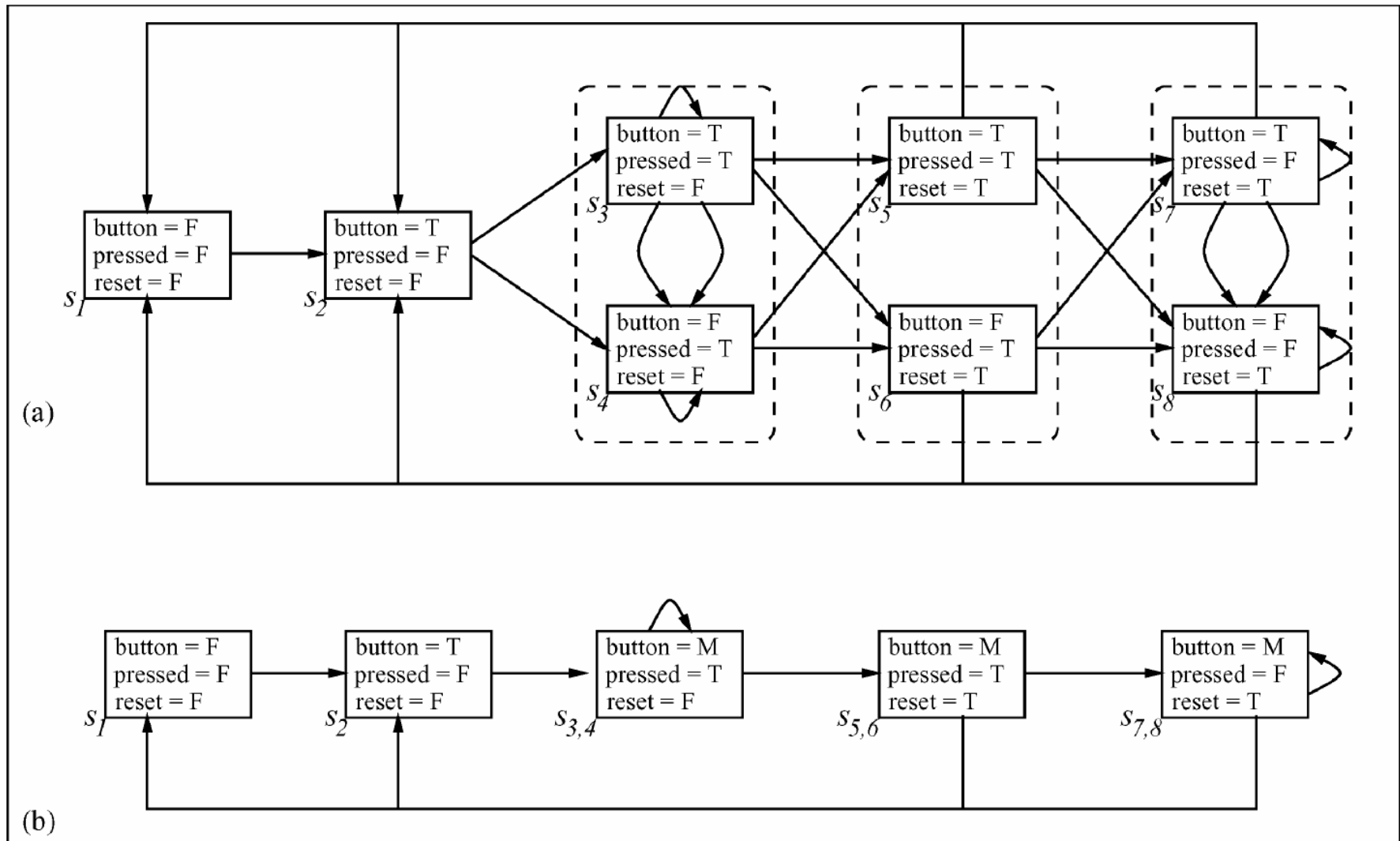
Remarks:

- **Fairness conditions**
 - Preserve values of fair paths, set unfair paths to \perp
 - Let fairness conditions $\{c_i\}$ then
 - ♦ $(\forall s \in S). (\|c_i\|_K(s) \in \{T, \perp\})$
 - A computation is fair if every computation comprising it is fair
 - ♦ i.e. when we consider composition of different viewpoints
 - $\|E_c G \varphi\|_K := \nu Z. \|\varphi\|_K \cap_B \bigcap_{B,i=1\dots n} \|EX E[\varphi U \varphi \wedge Z \wedge c_k]\|_K$
 - $\|E_c X \varphi\|_K := \|EX (\varphi \wedge (E_c G T \neq \perp))\|_K$
 - $\|E_c [\varphi U \psi]\|_K := \|E[\varphi U (\psi \wedge (E_c G T \neq \perp))]\|_K$
- Generation of proof like **counter-examples** and **witnesses**

mv-Model Checking in Practice (1)

- **Reduction methods:** just use existing model checkers
 - nuSMV, SPIN, CADP, EVALUATOR etc
- **Direct Methods:**
 - **x-Check:** mv-CTL model checker based on symbolic methods
- **An example to compare the two approaches:**
 - **Case study:** the SMV elevator example
 - ◆ Single Button Collective Control
 - ◆ 1 modified module Button per floor (outside elevator)
 - ◆ 1 module Lift (var: floor, door, direction, 1 button per floor)
 - Comparison using the same model checker **x-Check**
 - Pentium III, 850MHz, 256MB RAM, Linux

mv-Model Checking in Practice (2)



mv-Model Checking in Practice (3)

1. “If the door closes, it will eventually open”:
 $AG(\text{lift.door} = \text{closed} \rightarrow AF \text{lift.door} = \text{open})$
2. “Pressing a landing button guarantees that the lift will arrive at that landing and open its doors”:
 $AG(\text{landingBut2.button} \rightarrow AF(\text{lift.floor} = 2 \wedge \text{lift.door} = \text{open}))$
3. “If a button inside the lift is pressed, the lift will eventually arrive at the corresponding floor”:
 $AG(\text{lift.liftBut2.button} \rightarrow AF(\text{lift.floor} = 2 \wedge \text{lift.door} = \text{open}))$
4. “The lift may stop at floor 2 for landing calls when traveling downwards”:
 $\neg AG((\neg \text{lift.floor} = 2 \wedge \neg \text{lift.liftBut2.button} \wedge \text{lift.direction} = \text{down}) \rightarrow \text{lift.door} = \text{closed})$
5. “Whenever a button indicator is on, a button is being pressed”
 $AG(\text{lift.liftBut2.pressed} \rightarrow \text{lift.liftBut2.button})$

Model	CTL Property Number	Result	Bruns & Godefroid [2000]				XChek
			Pessimistic	Optimistic	Total	Best	
3-floor	1.	⊥	0.756 s	0.774 s	1.53 s	0.774 s	0.306 s
	2.	T	0.273 s	0.182 s	0.455 s	0.273 s	0.114 s
	3.	T	0.249 s	0.173 s	0.422 s	0.249 s	0.119 s
	4.	T	0.608 s	0.634 s	1.242 s	0.608 s	0.299 s
	5.	M	0.087 s	0.155 s	0.242 s	0.242 s	0.105 s
Size of trans. relation			2130	2153			954
4-floor	1.	⊥	4.638 s	4.594 s	9.232 s	4.594 s	2.29 s
	2.	T	0.936 s	0.942 s	1.878 s	0.936 s	0.463 s
	3.	T	0.869 s	1.044 s	1.913 s	0.869 s	0.494 s
	4.	T	3.767 s	3.698 s	7.465 s	3.767 s	2.122 s
	5.	M	0.047 s	0.502 s	0.549 s	0.549 s	0.298 s
Size of trans. relation			5249	5307			2367

Conclusions

- Both **reduction** and **direct** approaches to multi-valued model checking have their own advantages
- The additional **expressive power** of the mv-models permits the formal verification of problems that could not be handled before
- **One concern:** Hard to transfer these methods to industry
 - one has to be well versed to many-valued logics

Future Directions

- Reduction to CTL* using designated values
 - Built proof system
- mv-CTL symbolic model checker
 - Introduce types for the atomic propositions
 - Extend to mv-LTL model checking
 - Use property patterns
 - Investigate more realistic applications

References

- [1] G. Bruns and P. Godefroid, "Model checking with multi-valued logics." Bell Labs, Lucent Technologies, Tech. Rep. ITD-03-44535H, May 2003.
- [2] —, "Model checking with multi-valued logics." in *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP)*, ser. Lecture Notes in Computer Science, vol. 3142. Springer-Verlag, 2004, pp. 281-293.
- [3] M. Chechik, B. Devereux, S. Easterbrook, and A. Gurfinkel, "Multi-valued symbolic model-checking," *ACM Trans. Softw. Eng. Methodol.*, vol. 12, no. 4, pp. 1-38, Oct. 2004.
- [4] B. Konikowska and W. Penczek, "On designated values in multi-valued ctl* model checking," *Fundamenta Informaticae*, vol. 57, pp. 1-14, 2004.

Thank you!

Questions???