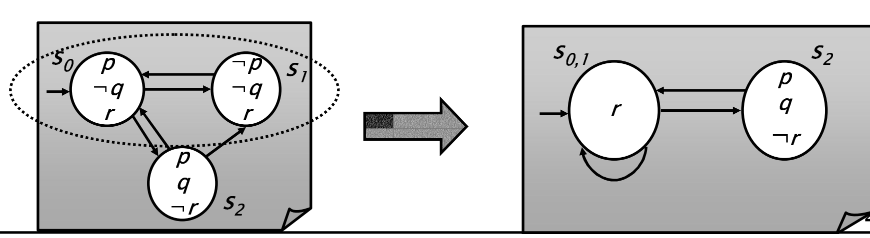
3-Valued Abstraction and 3-Valued Model-Checking

Abstraction

- **Abstraction:**
  - an effective technique to combat state explosion problem
    - approximate sets of concrete states by an abstract state
    - approximate sets of concrete transitions by an abstract transition
- **Using 2-valued logic (over-approximation)**
  - False variables represent "unknown" value
  - True transitions represent possible behaviour

$r(s_{0,1}) = T \Rightarrow r(s_0) = T$ $\quad\quad$ $p(s_{0,1}) = F \Rightarrow ?$



2

Abstraction, Cont'd

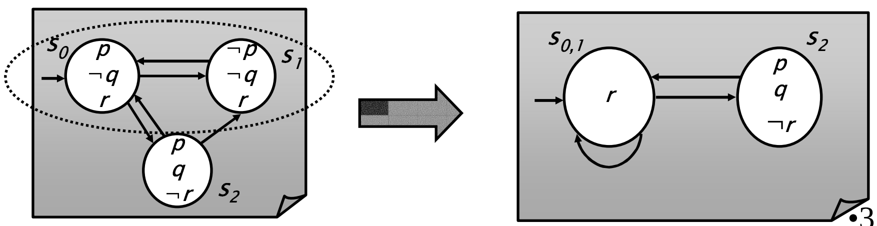
- **Using 2-valued logic**
  - False variables represent "unknown" value
  - True transitions represent possible behaviour

$AX\ (r \vee p)(s_{0,1})=T \Rightarrow AX\ (r \vee p)(s_0)=T$

$AX\ r\ (s_{0,1})=F \Rightarrow ?$

$EX\ (\neg r \wedge \neg p)(s_{0,1})=F \Rightarrow EX\ (\neg r \wedge \neg p)(s_0)=F$

$EX\ r(s_{0,1})=T \Rightarrow ?$



•3

Abstraction, Cont'd

- **Soundness:**
  - Only with respect to *True* universal properties
    - For existential properties – use under-approximation
  - For *False* properties:
    - play counter-example to determine whether spurious
    - Use counter-example-based abstraction refinement

•4

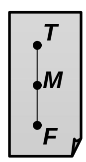
3-valued abstraction

- **Goals:**
  - Reason about mixed properties
  - Not have to tell which counterexamples are spurious
  - Not have an increase in statespace, when compared to 2-valued
  - Use counterexample for abstraction refinement
- **Outline:**
  - 3-valued logic, properties, models, model-checking
  - 3-valued abstractions
  - Abstraction refinement

•5

Logic: 3-valued Kleene logic

Logic order

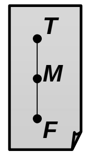


- **Properties:**
  - $F \sqsubseteq M, \ \ M \sqsubseteq T$
  - A $\wedge$ B = min (A, B)
  - A $\vee$ B = max (A, B)
  - $\neg T = F, \neg F = T, \neg M = M$
- **Preserves:**
  - Commutativity, associativity, idempotence, De Morgan laws
- **Does not preserve**
  - Law of excluded middle: $A \vee \neg A = T$ (top)
  - Law of non-contradiction: $A \wedge \neg A = \perp$ (bottom)

•6

Note

- **3-valued logic forms a lattice**

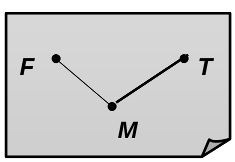


  - Ordering ⊑ : less than or equal
  - Meet operation ⊓ : min
  - Join operation ⊔ : max
  - Negation : horizontal symmetry
- **This is an example of a quasi-boolean algebra**
- **Equality and Identity are different!**
  - a ▯ b
  - a = b

•7

Logic

- **Information order**

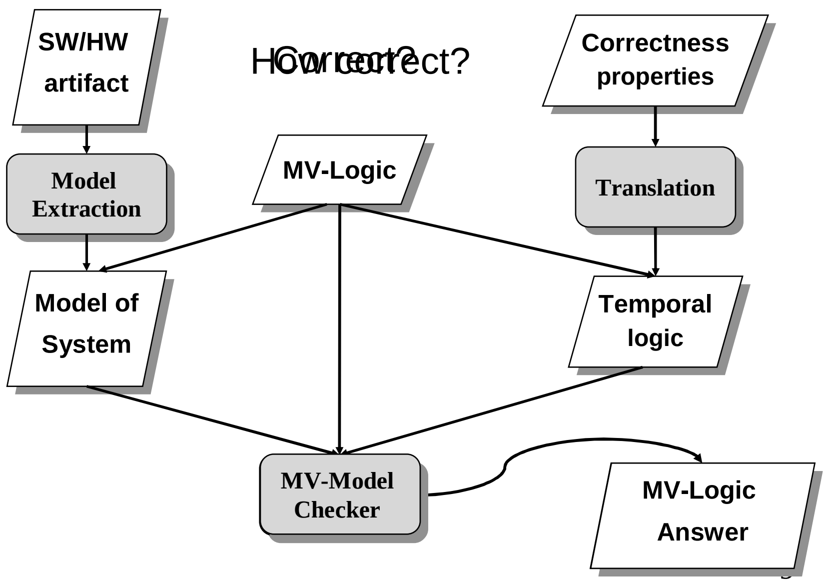


  - *M* contains least amount of information
  - *T*, *F* – maximum amount of information
  - If one refines *M* – it can change to *T* or *F* or stay at *M*

•8

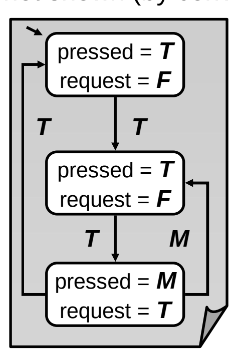
Overview of MV Model Checking

Overview of Model Checking



Multi-valued state machines: Xkripke structures

- **Extension of conventional state machines (Kripke structures)**
  - variables take any value from the logic (*T*, *F*, *M*)
  - transitions between states take any value from the logic
    - False transitions are not shown (by convention)
- **Example:**



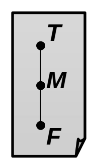
•10

Formally,

- **Kripke structures extended for MV case**
  - $M = \langle L, S, A, s_0, I, R \rangle$
  - L is a quasi-boolean algebra $(\mathscr{L}, \sqsubseteq, \sqcap, \sqcup, \neg)$, where $(\mathscr{L}, \sqsubseteq)$ is a lattice
  - S is a (finite) set of states, each with a unique name
  - A is a set of atomic propositions
  - s_0 is a unique initial state ($s_0 \in S$)
  - $I: S \times A \rightarrow \mathscr{L}$ is the interpretation function that assigns a logic value to each atomic proposition
  - $R: S \times S \rightarrow \mathscr{L}$ is the function that assigns a logic value to each transition between states

•11

3-valued CTL



- **multi-valued extension of CTL**
  - same syntax as CTL
  - plus constants from the logic (*T*, *M*, *F*)
- **semantics:**
  - replace existential quantification by disjunction, universal quantification by conjunction, so
    $(EX\ \phi)(s) = \bigvee_{t \in S} (R(s,t) \wedge \phi(t))$
  - other operators are defined as in CTL:

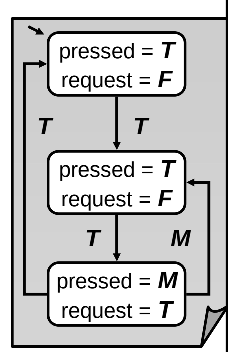
For all states s,

$$(AX\ \phi)(s) = (\neg EX(\neg \phi))\ (s)$$
$$(EG\ \phi)(s) = \phi(s) \wedge (EX\ EG\ \phi)(s)$$
$$(AG\ \phi)(s) = (\neg EF(\neg \phi))(s)$$

<u>Examples:</u>

AG (request -> AX pressed) $\stackrel{?}{=}$

AG (pressed $\vee$ request) $\stackrel{?}{=}$



•12

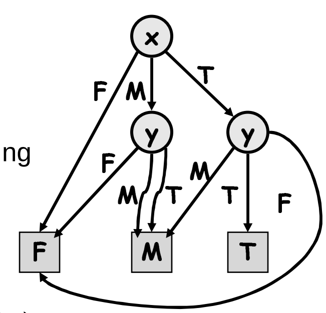
Model-Checking Cont'd

- **Can a *True* property evaluate to *M*?**
- **Answer:**
  - Yes
  - AG (pressed ∨ ¬ pressed) = *M*
  - Comes from law of excluded middle
- **Some terminology:**
  - Compositional semantics
    - Evaluate each CTL operator, compose according to lattice rules
  - Thorough semantics [Bruns&Godefroid 00]
    - Property evaluates to *M* iff exists a refinement where it evaluates to T and a refinement where it evaluates to F.
    - $$ to evaluate

•13

Symbolic mv model-checking

- **Similar idea to classical model-checking**
  - recursively go through the structure of XCTL property
  - encode sets of states symbolically
  - encode transition relation symbolically
- **Data structures**
  - direct approach: MDDs
    - the number of terminal nodes and branching factor equal to number of values in logic
    - Example: x∧y in 3-valued logic



  - can use BDD vector …
  - or mixed approaches (MBTDDs, MTBDDs)

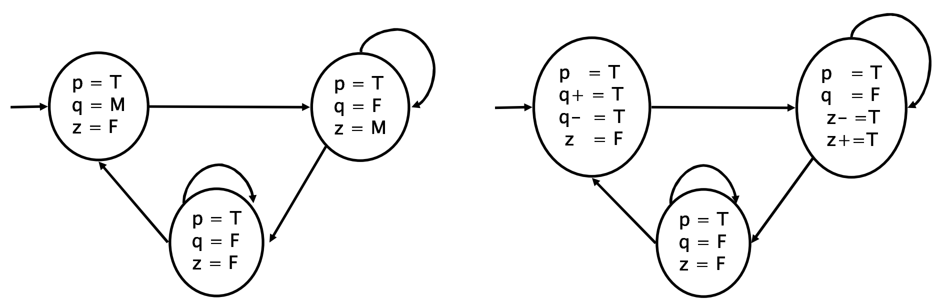
•14

Reduction to Classical

- **[Bruns&Godefroid'99]. Assumption: transition relation is classical**
  - Move negation to level of atomic propositions
  - Create a positive and negative version of every atomic proposition
  - Let x = M.
  - Positive cut:
    - Set x and and $\neg$ x to True
    - PosAnswer = check property
  - Negative cut:
    - Set x and and $\neg$ x to False
    - NegAnswer = check property
- **If NegAnswer = PosAnswer (True or False)**
  - Return this as answer
- **Else**
  - Return Maybe

•15

Example



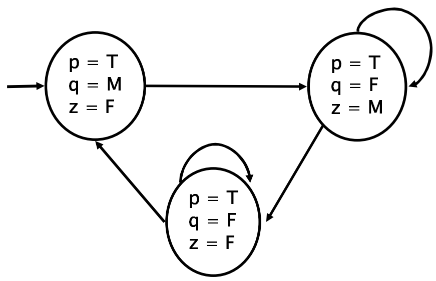
Model

Positive Cut

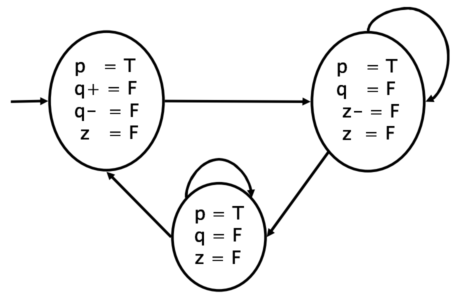
$[[p \wedge \neg q \vee z]](s_0) = T$

•16

Example



Model



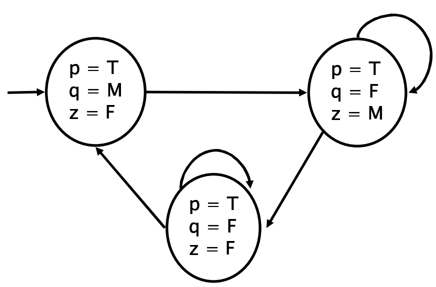
Negative Cut

$[[p \wedge \neg q \vee z]](s_0) = F$

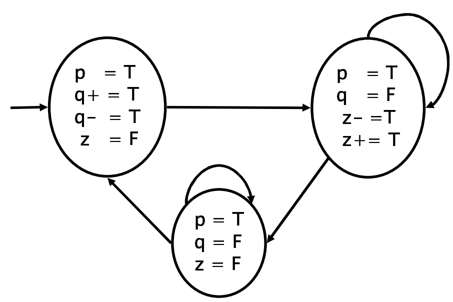
Therefore, the answer is M

•17

Example



Model

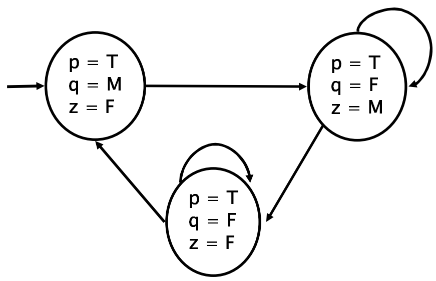


Positive Cut

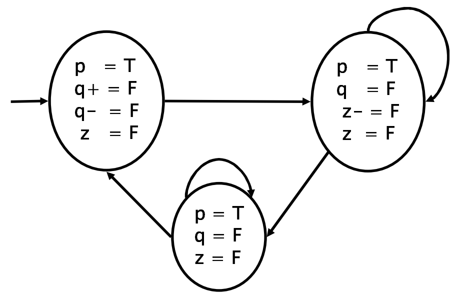
$[[\text{EX } (p \wedge \neg q \vee z)]](s_0) = T$

•18

Example



Model



Negative Cut

$[[\text{EX } (p \wedge \neg q \vee z)]](s_0) = T$
Therefore, the answer is T

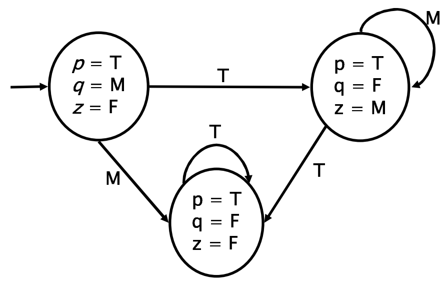
•19

Reduction to Classical (Take Two)

- **[Gurfinkel&Chechik 2003]**
- **Assumptions:**
  - States can be 3-valued, transition relation can be three-valued
- **Reduction steps**
  - for True and Maybe, construct a cut formula equivalent to $[[\varphi]](s) \sqsupseteq j$
    - logic: from mv CTL to restricted mv-logic with two-valued answers
    - model: unchanged
  - transform each cut to a classical model-checking problem
    - logic: from restricted mv-logic to classical CTL
    - model: from XKripke structure to classical Kripke structure

•20

Propositional Logic



$[[p \wedge \neg q \vee z]](s_0) = M$

$[[T \wedge \neg M \vee F]](s_0)$

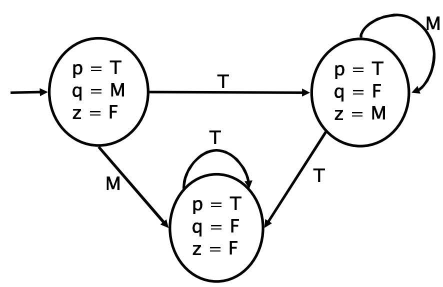
$[[T \wedge M \vee F]](s_0)$

$[[M \vee F]](s_0)$

$[[M]](s_0)$

•21

Propositional Logic – the cut



$[[p \wedge \neg q \vee z]](s_0) \sqsupseteq T$

$[[(p \sqsupseteq T) \wedge (\neg q \sqsupseteq T) \vee (z \sqsupseteq T)]](s_0)$

$[[T \wedge F \vee F]](s_0)$

$[[F \vee F]](s_0)$

F

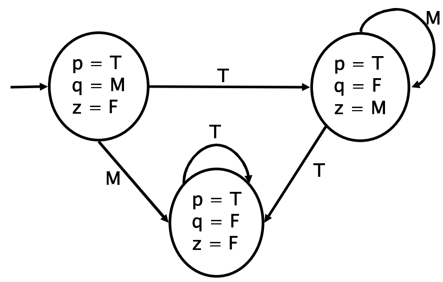
$\| p \wedge \neg q \vee z \|(s_0) \sqsupseteq M$

●
●
●
●
●

T

•22

Combining Results



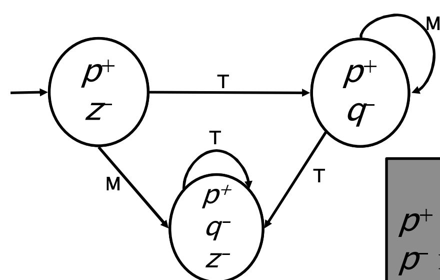
$[[p \wedge \neg q \vee z]](s_0) \not\sqsupseteq \mathsf{T}$

$[[p \wedge \neg q \vee z]](s_0) \sqsupseteq \mathsf{M}$

Therefore, $[[p \wedge \neg q \vee z]](s_0) = \mathsf{M}$

T
|
M
|
F

Propositional Logic – final step



Legend

p^+ represents $p \sqsupseteq$ j

p^- represents $\neg p \sqsupseteq$ j

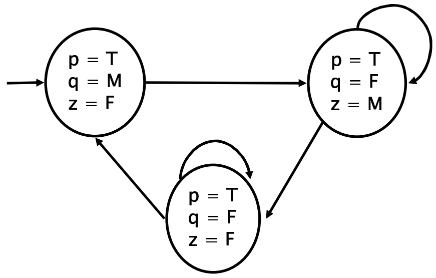
$[[(p \sqsupseteq \mathsf{T}) \wedge (\neg q \sqsupseteq \mathsf{T}) \vee (z \sqsupseteq \mathsf{T})]](s_0)$

$[[p^+ \wedge q^- \vee z^+]](s_0)$

$[[\mathsf{T} \wedge \mathsf{F} \vee \mathsf{F}]](s_0)$

$[[\mathsf{F}]](s_0)$

Existential Temporal Logic – the cut



$[[EX\ (p \wedge \neg q \vee z)]](s_0) \sqsupseteq T$

$\bigvee_{t \in S} R(s_0,t) \wedge [[p \wedge \neg q \vee z]](t) \quad \sqsupseteq T$

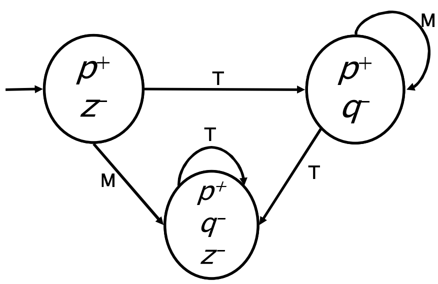
$\bigvee_{t \in S} (R(s_0,t) \sqsupseteq T) \wedge ([[p \wedge \neg q \vee z]](t) \sqsupseteq T)$

$\bigvee_{t \in S} (R(s_0,t) \sqsupseteq T) \wedge [[(p \sqsupseteq T) \wedge (\neg q \sqsupseteq T) \vee (z \sqsupseteq T)]](t)$

$[[EX_{\sqsupseteq T}\ ((p \sqsupseteq T) \wedge (\neg q \sqsupseteq T) \vee (z \sqsupseteq T))]](s_0)$

•25

Existential Temporal Logic – final step



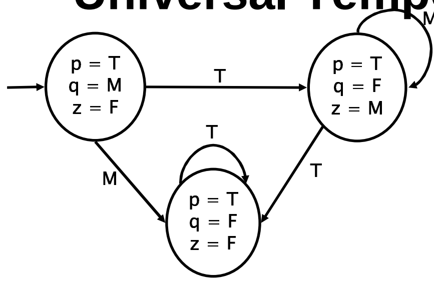
$[[EX_{\sqsupseteq T}\ ((p \sqsupseteq T) \wedge (\neg q \sqsupseteq T) \vee (z \sqsupseteq T))]](s_0)$

$[[EX_{\sqsupseteq T}\ (p^+ \wedge q^- \vee z^+)]](s_0)$

$[[EX(p^+ \wedge q^- \vee z^+)]](s_0)$

•26

Universal Temporal Logic – the cut



Dealing with negation

- In 3-valued logic
  - $\neg b \sqsupseteq \mathrm{T}$ iff $\neg(b \sqsupseteq \mathrm{M})$
    - since $\neg b \sqsupseteq \mathrm{T}$ iff $b = \mathrm{F}$

$[[AX(p \wedge \neg q \vee z)]](s_0) \sqsupseteq \mathrm{T}$

$\wedge_{t \in S} R(s_0,t) \Rightarrow [[p \wedge \neg q \vee z]](t) \quad \sqsupseteq \mathrm{T}$

$\wedge_{t \in S} \neg R(s_0,t) \vee [[p \wedge \neg q \vee z]](t) \quad \sqsupseteq \mathrm{T}$

$\wedge_{t \in S} \neg R(s_0,t) \sqsupseteq \mathrm{T} \vee [[p \wedge \neg q \vee z]](t) \sqsupseteq \mathrm{T}$

$\wedge_{t \in S} \neg(R(s_0,t) \sqsupseteq \mathrm{M}) \vee [[p \wedge \neg q \vee z]](t) \sqsupseteq \mathrm{T}$

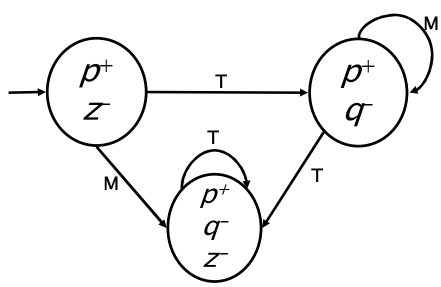
$\wedge_{t \in S}(R(s_0,t) \sqsupseteq \mathrm{M}) \Rightarrow [[p \wedge \neg q \vee z]](t) \sqsupseteq \mathrm{T}$

$[[AX_{\sqsupseteq M}((p \sqsupseteq \mathrm{T}) \wedge (\neg q \sqsupseteq \mathrm{T}) \vee (z \sqsupseteq \mathrm{T}))]](s_0)$

T
|
M
|
F

•27

Universal Temporal Logic – final step



$[[AX_{\sqsupseteq M}((p \sqsupseteq \mathrm{T}) \wedge (\neg q \sqsupseteq \mathrm{T}) \vee (z \sqsupseteq \mathrm{T}))]](s_0)$

$[[AX_{\sqsupseteq M}(p^{+} \wedge q^{-} \vee z^{+})]](s_0)$

$[[AX(p^{+} \wedge q^{-} \vee z^{+})]](s_0)$

•28

Handling Mixed Modalities

- **The first reduction step does not change**
  - $[[AX\ EX p]](s_0) \sqsupseteq T$ is transformed into $[[AX_{\sqsupseteq M}\ EX_{\sqsupseteq T}(p \sqsupseteq T)]](s_0)$
- **Problem with the second step**
  - need a Kripke structure with two types of transitions
    - $\sqsupseteq M$ for universal modality
    - $\sqsupseteq T$ for existential modality
- **Solution**
  - treat transitions labels as actions
  - convert the resulting Labeled Transition System into a Kripke structure
- **Disadvantage**
  - introduces a new variable
  - size of the statespace doubles

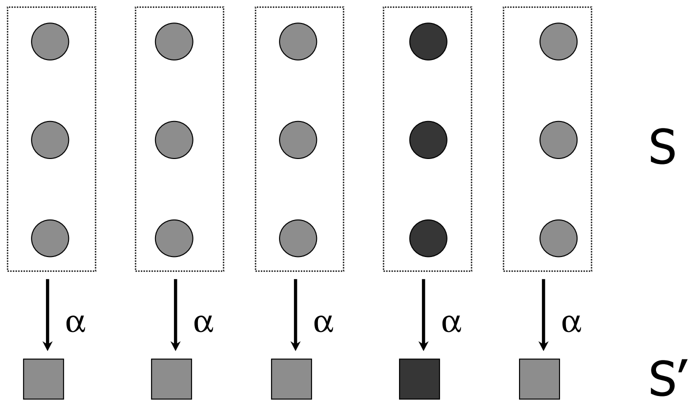
•29

Summary of the Reduction

- **Multi-valued model-checking problem is reduced to several classical problems**
  - one classical problem for True and one for Maybe
  - size of the formula does not change
    - atomic literals are changed to “plus” and “minus” versions
    - other parts remain unchanged
  - for universal and existential fragments
    - statespace of resulting Kripke structure is similar to the original
  - for formulas with both universal and existential modalities
    - statespace of the resulting Kripke structure is double of the original
  - formulas with fixpoint operators are handled similarly
    - (see Gurfinkel, Chechik, CONCUR’03)

•30

Abstraction

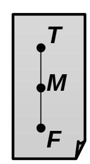


Abstraction Function $\alpha : S \rightarrow S'$

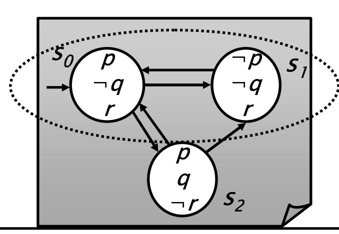
•31

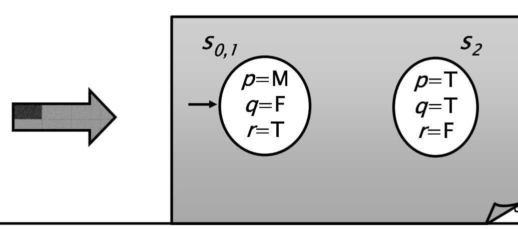
Abstraction

- **Using 3-valued logic**
  - introduce new special value *Maybe* to stand for "unknown"
  - Formally:



    - $[[v]]$ (a) = T iff $\forall$ s $\in$ γ(a) $[[v]]$(s) = T
    - $[[v]]$ (a) = F iff $\forall$ s $\in$ γ(a) $[[v]]$(s) = F
    - $[[v]]$ (a) = M iff $\exists$ s $\in$ γ(a) $[[v]]$(s) = T and $\exists$ t $\in$ γ(a) $[[v]]$(t) = F
  - Examples:
    - $r(s_{0,1})=T \Rightarrow r(s_0)=T$ $\qquad$ $p(s_{0,1})=M$





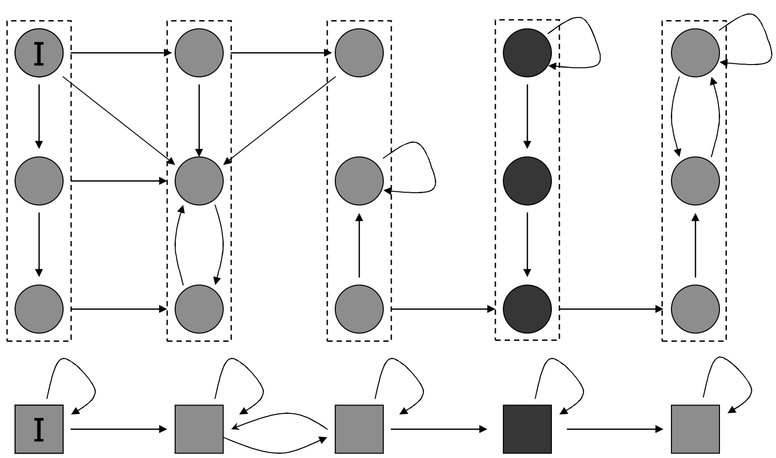
•32

Refresher: Over- and Under-approximations

- ➲ ***M'*** **is an over-approximation of *M*, or *M'* simulates *M* if**
  - ✎ $R^{\exists\exists}$ [Dams'97]: $(t, t_1) \in R'$ iff $\exists\, s \in \gamma(t)$ s.t. $\exists\, s_1 \in \gamma(t_1)$ and $(s, s_1) \in R$
- ➲ ***M'*** **is an under-approximation of *M*, or *M* simulates *M'* if**
  - ✎ $R^{\forall\exists}$ [Dams'97]: $(t, t_1) \in R'$ iff $\forall\, s \in \gamma(t)$ s.t. $\exists\, s_1 \in \gamma(t')$ and $(s, s_1) \in R$

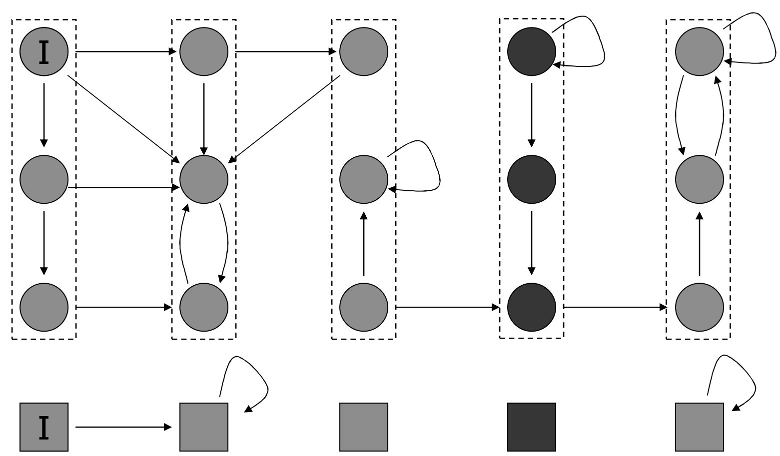
•33

Existential Abstraction (Over-Approximation)



•34

Universal Abstraction (Under-Approximation)



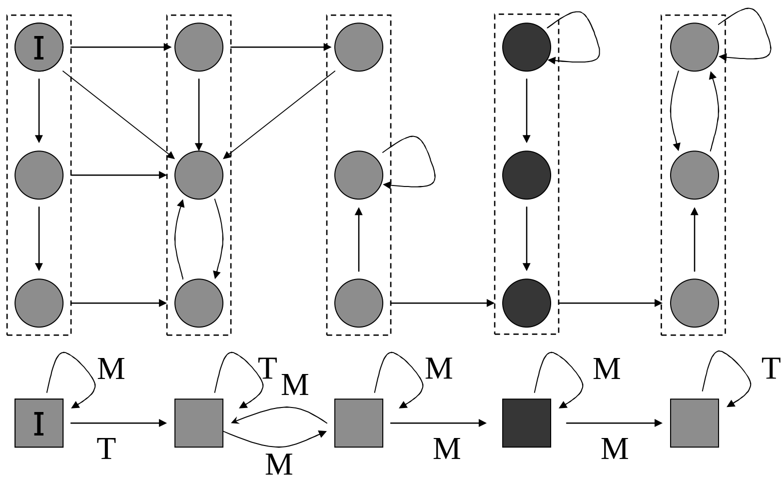
•35

3-Val Transition Relation

- **Let R(s,t) = T if R(s,t) $\in$ R$^{\forall\exists}$**
  - R$^{\forall\exists}$ [Dams'97]: $(t, t_1) \in R'$ iff $\forall\, s \in \gamma(t)$ s.t. $\exists\, s_1 \in \gamma(t')$ and $(s, s_1) \in R$
- **Let R(s,t) = F if R(s,t) $\notin$ R$^{\exists\exists}$**
  - R$^{\exists\exists}$ [Dams'97]: $(t, t_1) \in R'$ iff $\exists\, s \in \gamma(t)$ s.t. $\exists\, s_1 \in \gamma(t_1)$ and $(s, s_1) \in R$
- **Else R(s,t) = M**

•36

3-valued abstraction



•37

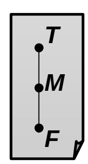
Abstraction

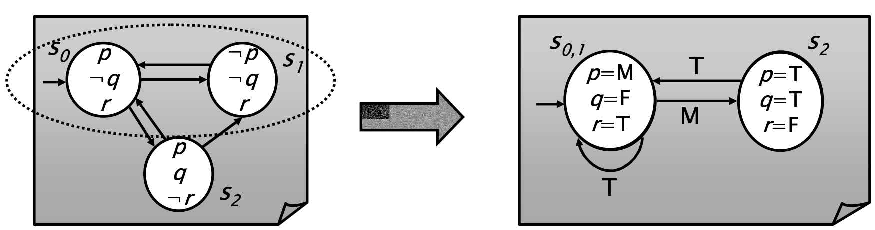
➲ **Using 3-valued logic**

↳introduce new special value *Maybe* to stand for "unknown"

AX $r(s_{0,1})$=M

EX $r(s_{0,1})$=T





•38

Model Checking 3-Val abstract Models

- Let φ be an arbitrary property (i.e., expressed in LTL, CTL, mu-calculus) and M' is a 3-val abstraction of M

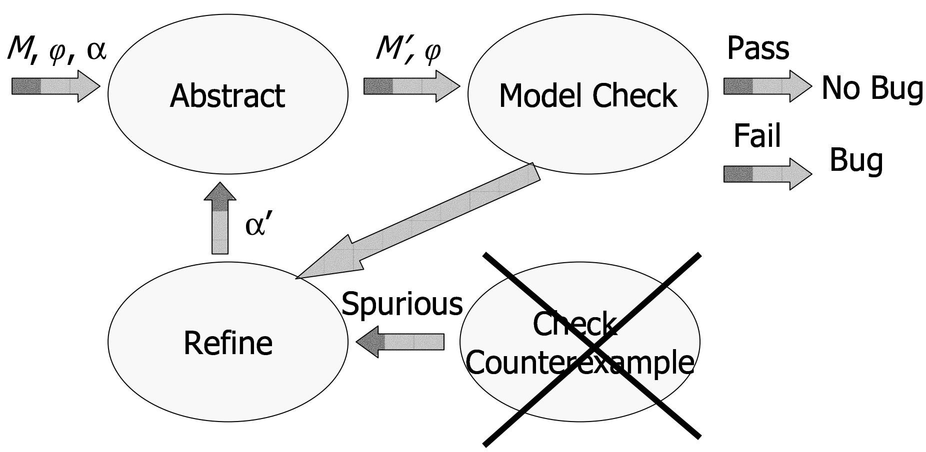
➲ **Preservation Theorem**

$$M' \models \varphi \;\Leftrightarrow\; M \models \varphi$$

- No guarantee is given about a "Maybe" answer
- False counterexample cannot be spurious
  - No need for simulation!
- Maybe counterexample requires refinement

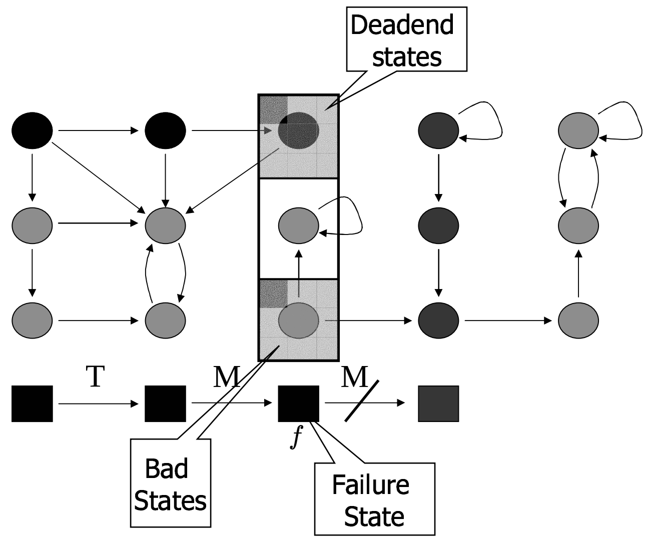
•39

3-Val Abstraction-Refinement Loop



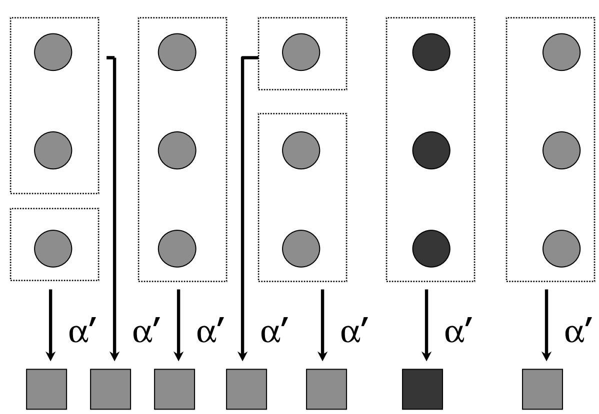
•40

No spurious counterexamples, but abstraction can be too coarse



•41

Refinement

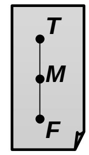


Refinement : α′

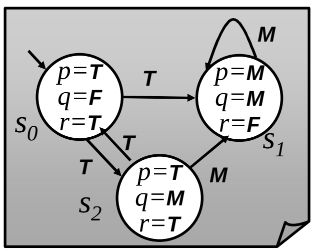
•42

Other use of 3-valued logic

➲ **Algebra:**



- ✧use three-valued algebra (Kleene)
- ✧intermediate value represents incomplete information or uncertainty
- ✧compact representation for all possible refinements of this model
- ✧if a property is *True/False* on the partial model, it is *True/False* on a refined one
- ✧initial theory developed by Bruns & Godefroid, CAV'99



Application:

- Most models are incomplete!
- Allows verification before specification is completed

•43

Summary

➲ **Abstraction**

- ✧Effective tool for combating state explosion
- ✧Over-approximation – sound for true universal properties, otherwise – check if counterexample is feasible and then refine
- ✧Under-approximation – same for existential properties

➲ **3-Valued Abstraction**

- ✧Specified in 3-val Kleene logic
- ✧Allows reasoning about mixed-quantifier properties
- ✧No need to check if counter-example is spurious
- ✧Counterexample used for refinement

➲ **3-Val Model-Checking**

- ✧Reduces to two runs of classical model-checker
- ✧Or can be done directly, say, using MDDs

•44

Next topic:

- **Software model-checking**
  - (and software model-checking with 3-valued logic)

•45