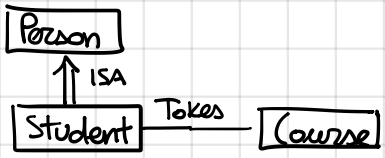

Edoardo Puglisi
Formal Methods

FIRST ORDER LOGIC

FOL is the logic to speak about objects which are the domain of universe, concerned about properties and relationships. It also has functions including constants that denote objects.



Student(x) = "x is a student"
Person(x)
Course(x)
Takes(x,y) = "x takes y"
Names Arities

} This is our kind of alphabet!

$\forall x \; St(x) \rightarrow Pe(x)$ [ISA]

$\forall x (\underbrace{\exists y \; Takes(x,y)}_{\alpha(x)}) \rightarrow St(x)$ [First argument of "Takes" is a student]

$\forall y (\underbrace{\exists y \; Takes(x,y)}_{\alpha(y)}) \rightarrow Co(y)$ [Second argument of "Takes" is a Course]

$\Rightarrow \forall x \forall y \; Takes(x,y) \rightarrow \underbrace{St(x) \wedge Co(y)}_{\beta(x,y)}$

Note: first 3 formulas are talking in general, not about any particular object (Student, Person or Course). Just Metadata → INTENSIONAL KNOWLEDGE

Student(John)
Takes(John, Math)
Person(Mary)
} Γ (= gamma)

Constants → Data → EXTENSIONAL KNOWLEDGE

↳ $\Gamma \models Pe(John)$ ✓ [Is John a person?]
$\Gamma \models Co(Math)$ ✓
$\Gamma \models St(Mary)$ ×
$\Gamma \models \neg St(Mary)$ ×
} Because general rules can't lead to these two conclusions

Vars = $\{x_1 \dots x_n\}$ set of individual variables. Vars $\subseteq$ Terms eg constants
FOL interpretation: $I = (\Delta^I, P_1^I, P_2^I \dots f_1^I, f_2^I \dots)$ P_x^I = predicate f_x^I = formula
- Δ^I = domain, set of objects

Given I, an assignment is a function $\alpha : Vars \rightarrow \Delta^I$ that assigns to each variable $x \in Vars$ an object $\alpha(x) \in \Delta^I$ → $\hat{\alpha} : Terms \rightarrow \Delta^I$
- $\hat{\alpha}(x) = \alpha(x)$ if $x \in Vars$
- $\hat{\alpha}(f(t \dots t_k)) = f^I(\hat{\alpha}(t_1) \dots \hat{\alpha}(t_k))$

$I, \alpha \models \varphi$: φ is true in interpretation I wrt an assignment α

Variable x in formula φ is free if x doesn't occur in scope of any quantifier, bounded otherwise. φ is closed if has no free variable (sentence), open otherwise

A FOL query is an open formula. When φ is a query with free variables $(x_1 \dots x_k)$ then we write it as $\varphi(x_1 \dots x_k)$ and say that φ has arity k.

Boolean query is a FOL query without free variables

LOGICAL TASKS

Validity: φ is valid iff $\forall I, \alpha$ we have $I, \alpha \models \varphi$

Satisfiability: φ is satisfiable iff $\exists I, \alpha$ s.t. $I, \alpha \models \varphi$, unsatisfiable otherwise

Logical Implication: φ logically implies ψ $(\varphi \models \psi)$ iff $\forall I, \alpha$ if $I, \alpha \models \varphi$ then $I, \alpha \models \psi$

Logical Equivalence: φ is logical equivalent to ψ iff $\forall I, \alpha$ we have that $I, \alpha \models \varphi$ iff $I, \alpha \models \psi$ i.e $\varphi \models \psi$ and $\psi \models \varphi$

QUERY EVALUATION

Checking if a formula is true given interpretation and assignement. To do so we have to consider finite alphabet (finite predicates, functions) and finite interpretation (Δ^I finite)

$\Rightarrow$ Given query $\varphi(x_1 \dots x_k)$ compute $\varphi^I = \{(a_1 \dots a_k) \mid I, \langle a_1 \dots a_k \rangle \models \varphi(x_1 \dots x_k)\}$

Recognition problem: given finite I, a query and a tuple, check is the tuple is a solution.

Combined Complexity: complexity of $\{\langle I, \alpha, \varphi \rangle \mid I, \alpha \models \varphi\}$ i.e I, α, φ all part of input

Data Complexity: All but φ (fixed) is part of input

Query Complexity: All but I (fixed) is part of input

Theorem: Complexity of $\{\langle I, \alpha \rangle \mid I, \alpha \models \varphi\}$ is

- TIME: polynomial
- SPACE: LogSpace

This is the reason why FOL queries are used in databases!

Example

We consider FOL interpretations exactly as used in relational databases. This requires to drop functions except for constants. Moreover we assume that the interpretation of constants is the identity function, that is constants are interpreted as themselves. This allows us to drop also the interpretation of constants from our interpretations, which now have the form:

$$\mathcal{I} = (\Delta^{\mathcal{I}}, P_1^{\mathcal{I}}, P_2^{\mathcal{I}}, \dots, P_n^{\mathcal{I}}).$$

Interpretation: $\mathcal{I}$ is as follows (also given in relational notation):

- $\Delta^{\mathcal{I}} = \{john, paul, george, mick, ny, london, 0, 1, \dots, 100\}$
- $Person^{\mathcal{I}} = \{(john, 30), (paul, 60), (george, 35), (mick, 35)\}$
- $Lives^{\mathcal{I}} = \{(john, ny), (paul, ny), (george, london), (mick, london)\}$
- $Manages^{\mathcal{I}} = \{(paul, john), (george, mick), (paul, mick)\}$

FOL

$Person^{\mathcal{I}}$

name	age
john	30
paul	60
george	35
mick	35

$Lives^{\mathcal{I}}$

name	city
john	ny
paul	ny
george	london
mick	london

$Manages^{\mathcal{I}}$

boss	emp. name
paul	john
george	mick
paul	mick

DATABASE

Query: find name and age of persons who live in the same city as their boss.

$$\exists z, w.\text{Person}(x, y) \wedge \text{Manages}(z, x) \wedge \text{Lives}(x, w) \wedge \text{Lives}(z, w)$$

"Find" or "Return" mean assign a value to open variable! → Query = open formula (as we said)

EXAMPLES

S(s_id, name) B(b_id, color) R(s_id, b_id, date)

① Find names of sailors who have reserved boat 103

$\exists x,z,w,t\ S(x,y) \wedge R(x,z,w) \wedge B(z,t) \wedge z=103$

↳ this means we ONLY CARE of y

B(z,t) → i.e "whatever x,z,w,t"; needed to "force" z to be a boat id and not just a number

② find names of sailors who have reserved a red boat

$\exists sid,bid,d\ S(sid,n) \wedge R(sid,bid,d) \wedge B(bid,\text{"red"})$

③ Find colors of boat reserved by BOB

$\exists bid,sid,d\ B(bid,c) \wedge R(sid,bid,d) \wedge S(sid,\text{"BOB"})$

④ Find names of sailors who have reserved at least one boat

$\exists sid,bid,d,c\ S(sid,n) \wedge R(sid,bid,d) \wedge B(bid,c)$

⑤ Find names of sailors who reserved red and green

$\exists sid,bid,d,bid_2,d_2\ S(sid,n) \wedge R(sid,bid,d) \wedge B(bid,\text{"RED"}) \wedge R(sid,bid_2,d_2) \wedge B(bid_2,\text{"GREEN"})$

⑥ ... red OR green

$\exists sid,bid,d\ S(sid,n) \wedge R(sid,bid,d) \wedge (B(bid,\text{"RED"}) \text{ OR } B(bid,\text{"GREEN"}))$

⑦ ... reserved AT LEAST 2 boats

$\exists sid,bid,bid_2 \ldots\ S(sid,n) \wedge R(sid,bid,d) \wedge B(bid,c) \wedge R(sid,bid_2,d_2) \wedge B(bid_2,c_2) \wedge \neg(bid=bid_2)$

⑧ ... not reserved red boats

$\exists x\ S(x,n) \wedge \neg\exists y,d\ R(x,y,d) \wedge B(y,\text{"RED"})$

⑨ ... reserved ALL boat

$\exists x\ S(x,n) \wedge \forall y\,(\exists c\ B(y,c)) \rightarrow (\exists d\ R(x,y,d))$

⑩ ... all red boats

$\exists x\ S(x,n) \wedge \forall y\ B(y,\text{"red"}) \rightarrow \exists d\ R(x,y,d,$

Imply: $\rightarrow = \supset$

⑪ ... only red boats

$\exists x\ S(x,n) \wedge \forall y\ \exists d\ R(x,y,d) \rightarrow B(y,\text{"red"})$

⑫ ... exactly one boat

$\exists x\ S(x,n) \wedge (\exists y,d,c\ R(x,y,d) \wedge B(y,c)) \wedge (\forall yy'(\exists d\ R(x,y,d)) \wedge (\exists d\ R(x,y',d) \supset y=y'))$

TABLEAU METHOD

Proving in mechanical manner that a given set of formulas is not satisfiable

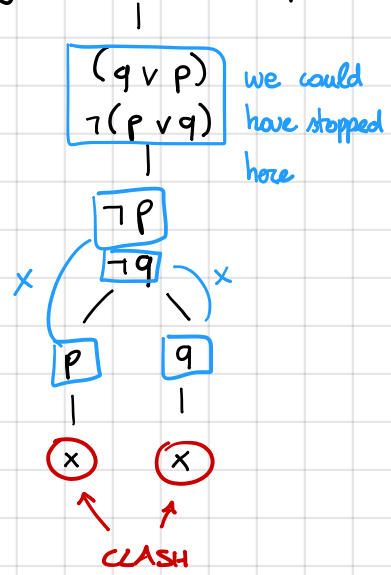
given : set of premises Γ and conclusion ϕ

task : prove $\Gamma \models \phi$

how : show $\Gamma \cup \neg \phi$ is not satisfiable i.e. add the complement of the conclusion to the premises and derive a contradiction

Th. $\Gamma \models \phi$ iff $\Gamma \cup \{\neg \phi\}$ is unsatisfiable

eg. $\neg(q \vee p \supset p \vee q)$



α rules

$$\frac{\phi \wedge \psi}{\phi \quad \psi} \qquad \frac{\neg(\phi \vee \psi)}{\neg\phi \quad \neg\psi} \qquad \frac{\neg(\phi \supset \psi)}{\phi \quad \neg\psi}$$

¬¬-Elimination

$$\frac{\neg\neg\phi}{\phi}$$

β rules

$$\frac{\phi \vee \psi}{\phi \mid \psi} \qquad \frac{\neg(\phi \wedge \psi)}{\neg\phi \mid \neg\psi} \qquad \frac{\phi \supset \psi}{\neg\phi \mid \psi}$$

Branch Closure

$$\frac{\phi \quad \neg\phi}{X}$$

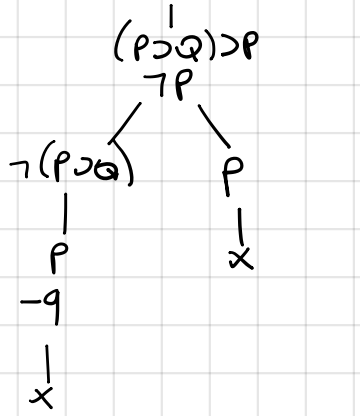
Do α before β always

Note: These are the standard ("Smullyan-style") tableau rules.

We omit the rules for $\equiv$. We rewrite $\phi \equiv \psi$ as $(\phi \supset \psi) \wedge (\psi \supset \phi)$

Eg $\models ((p \supset q) \supset p) \supset p$

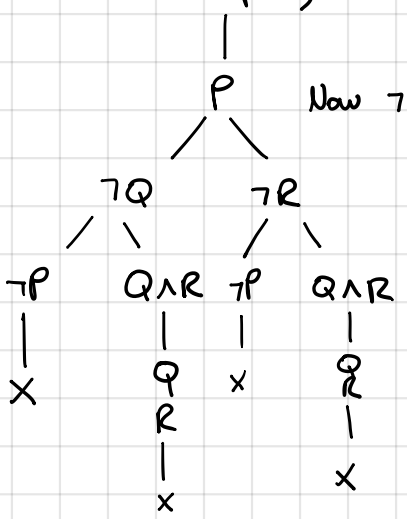
$\neg(((p \supset q) \supset p) \supset p)$



Eg $p \supset (q \wedge r), \neg q \vee \neg r \models \neg p$

$p \supset q \wedge r$
$\neg q \vee \neg r$
$\neg\neg p$

} In a "node" with multiple formulas we compute one formula at time (now ¬¬p)



Now ¬q ∨ ¬r

Now $p \supset q \wedge r$ for each node

Note : order is ininfluent eg $q \vee p$ = $p \mid q$ or $q \mid p$

Note : $\models \ldots$ becomes $\neg\neg \ldots$

$\neg(P \vee Q \supset P \wedge Q)$ is satisfiable?

$P \vee Q$

$\neg(P \wedge Q)$

P | Q

$\neg P$ $\neg Q$ $\neg P$ $\neg Q$

x 0 0 x

yes, it is

The Tableau shows us all possible interpretations ($\{P\}, \{Q\}$) that satisfy the formula.
eg $I = \{P\}$ means only P is true

Th. Soundness
If $\Gamma \vdash$ (proves) ϕ then $\Gamma \models \phi$

Th. Completeness
If $\Gamma \models \phi$ then $\Gamma \vdash \phi$

Other rules for propositional logic (β rules)

① $\phi \equiv \psi$ → $\phi \mid \neg\phi$; $\psi \mid \neg\psi$

② $\neg(\phi \equiv \psi)$ → $\phi \mid \neg\phi$; $\neg\psi \mid \psi$

δ rules:

① $\dfrac{\exists x\ \phi(x)}{\phi(c)}$ ② $\dfrac{\neg\forall x\ \phi(x)}{\neg\phi(c)}$

c = fresh constant = new constant not previously appearing in tableaux

γ rules:

① $\dfrac{\neg\exists x\ \phi(x)}{\neg\phi(t)}$ ② $\boxed{\dfrac{\forall x\ \phi(x)}{\phi(t)}}$

t = any term (not fresh)

→ This means that for every object of the domain the property $\phi(x)$ should be true. A term t that occurs in the tableaux denotes an object of the domain.
Therefore ϕ must be true for all terms t in the tableaux i.e. $\forall$ can be applied as many times as one want to any term.

Notation: $\phi[x/t]$ = formula we get by substitute all x with t

eg $\forall x P(x,y)[x/b] = \forall x P(x,y)$

$P(x,y,f(x))[x/a] = P(a,y,f(a))$

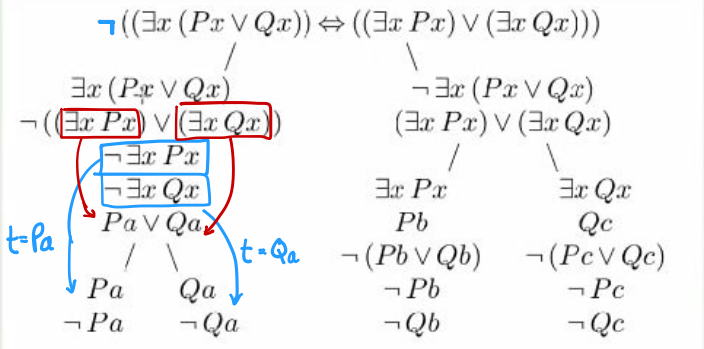
$\exists x P(x,x) \wedge Q(x)[x/c] = \exists x P(x,x) \wedge Q(c)$

$\forall x P(x,y)[y/f(x)]$ = not allowed, x bounded to "$\forall$"

Example

To check if the formula $(\exists x(P(x) \vee Q(x))) \equiv ((\exists x P(x)) \vee (\exists x Q(x)))$ is ~~satisfiable~~ valid, we start with a tableaux with this formula:

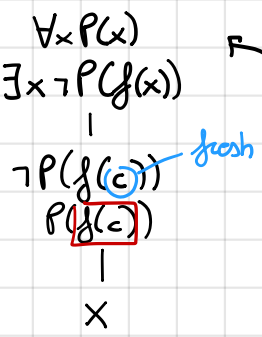
Tableaux only do UNSAT but...



We have closhes → UNSAT
⇒ original formula is valid

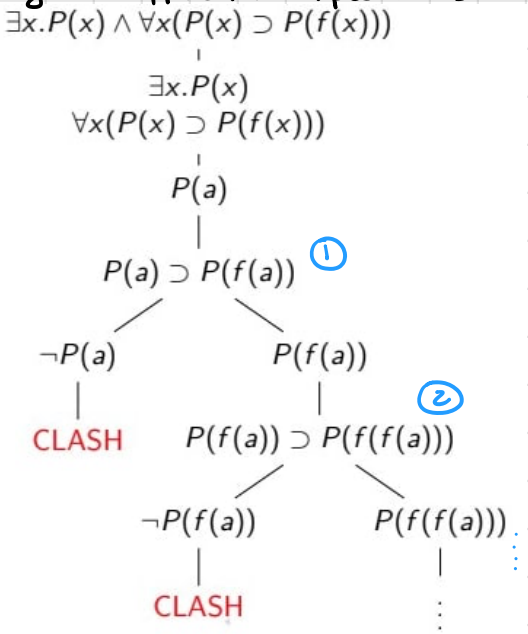
GENERAL RULE
- is valid? → negate and check if all branches are closhed
- is satisfiable? → check if there is at least one open branch.

Ex. check if $\forall x P(x) \wedge \exists x \neg P(f(x))$ is SAT

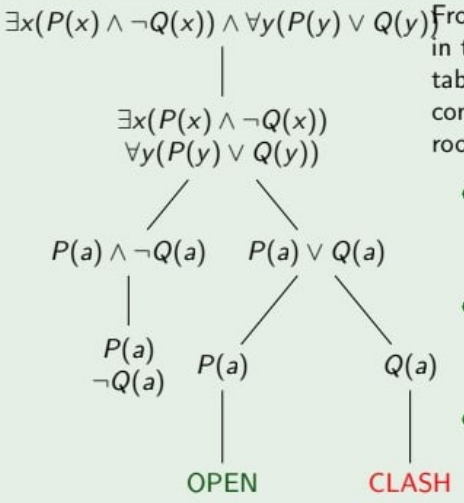


Note: in cases like this, start with ∃ and then use ∀ to find clashes.

Eg. ∀ applied multiple times



Understand the model:



From the formulas appearing in the OPEN branch of the tableaux it is possible to construct a model for the root formula.

- $\Delta = \{a\}$, the constants appearing in the formulas
- $I(P) = \{a\}$, since the formula $P(a)$ appears in the open branch
- $I(Q) = \{\}$ since the formula $\neg Q(a)$ appears in the open branch

CONJUNCTIVE QUERIES

A query of form $\exists \vec{y}\, conj(\vec{x}, \vec{y})$ where "conj" is a conjunction ("and") of atoms and EQUALITIES over free variable $\vec{x}$, the existentially quantified variable $\vec{y}$ and possibly constants.

Relational alphabet:
Person(name, age), Lives(person, city), Manages(boss, employee)

Query: find the name and the age of the persons who live in the same city as their boss.

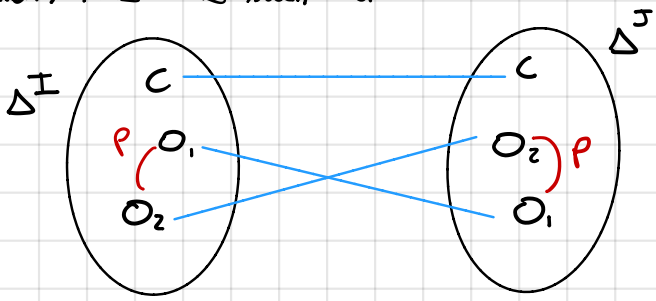
$\Rightarrow$ $\exists b, e, p_1, c_1, p_2, c_2.\text{Person}(n, a) \wedge \text{Manages}(b, e) \wedge \text{Lives}(p1, c1) \wedge \text{Lives}(p2, c2) \wedge n = p1 \wedge n = e \wedge b = p2 \wedge c1 = c2$

Or simpler: $\exists b, c.\text{Person}(n, a) \wedge \text{Manages}(b, n) \wedge \text{Lives}(n, c) \wedge \text{Lives}(b, c)$

HOMOMORPHISM

Suppose two interpretations I and J on some alphabet.
A homomorphism from I to J is a mapping (function) $h: \Delta^I \to \Delta^J$ such that:

- $h(c^I) = c^J$
- $(o_1 \dots o_k) \in P^I$ implies $(h(o_1) \dots h(o_k)) \in P^J$



If the map cover the entire domain then it is called isomorphism. FOL is unable to distinguish between interpretations that are isomorphic.

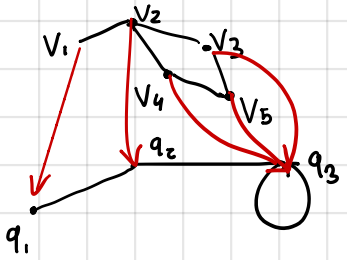
Eg $\Delta^I = \{v_1, v_2, v_3, v_4, v_5\}$
$e = \{(v_1, v_2), (v_2, v_3), (v_2, v_4), (v_4, v_3), (v_4, v_5)\}$
$a^I = v_1$
$b^I = v_3$

$\Delta^J = \{q_1, q_2, q_3\}$
$e^J = \{(q_1, q_2), (q_2, q_3), (q_3, q_3)\}$
$a^J = q_1$
$b^J = q_3$

We want $h(v_1)$, $h(v_2)$ etc

$h(v_1) = q_1$ — because $h(c^I) = c^J \to h(a^I) = a^J \to h(v_1) = q_1$ (same for b^I / b^J)
$h(v_2) = q_2$ — because $(v_1, v_2) \in e^I \Rightarrow (h(v_1), h(v_2)) \in e^J$ i.e we check e^J and search q_1 in first position
$h(v_3) = q_3$
$h(v_4) = q_3$ again q_3 like above
$h(v_5) = q_3$ "

IT IS HOMO



CANONICAL INTERPRETATION

Given a (boolean) CQ eg $q(c) \leftarrow E(c,y).E(y,z).E(z,c)$
equal to $\exists y z \; E(c,y) \wedge E(y,z) \wedge E(z,c)$ can be transformed into a canonical interpretation

$$I_q = \begin{cases} \Delta^{I_q} = \{c, y, z\} \\ c^{I_q} = c \\ E^{I_q} = \{(c,y), (y,z), (z,c)\} \end{cases}$$

Def.: The canonical interpretation $\mathcal{I}_q$ associated with q is the interpretation $\mathcal{I}_q = (\Delta^{\mathcal{I}_q}, P^{\mathcal{I}_q}, \dots, c^{\mathcal{I}_q}, \dots)$, where

- $\Delta^{\mathcal{I}_q} = \{x_1, \dots, x_n\} \cup \{c \mid c \text{ constant occurring in } q\}$, i.e., all the variables and constants in q;
- $c^{\mathcal{I}_q} = c$, for each constant c in q;
- $(t_1, \dots, t_k) \in P^{\mathcal{I}_q}$ iff the atom $P(t_1, \dots, t_k)$ occurs in q.

Th. For boolean CQs, $I \models$ iff there exists a homomorphism from I_q to I

Observations:

1. $I_q \models q$ always true.
2. If h - homomorphism from h I to I_2 and h' from I_2 to I_3 then $h \circ h'$ is an omomorphism from I to I_3

Two interpretations I and J are homomorphically equivalent if exists homomorphism $h_{I,J}$ from I to J and h_{JI} from J to I.

CQs are unable to distinguish between interpretations that are homomorphic equivalent.

QUERY CONTAINMENT: given two FOL queries φ on ψ of same arity, φ is contained in ψ, denoted $\varphi \subseteq \psi$ if $\forall$ interpretations I and assignements α we have that: $I, \alpha \models \varphi$ implies $I, \alpha \models \psi$
For FOL queries, query containment is undecible
For CQs, $q_1(\vec{x}) \subseteq q_2(\vec{x})$ can be reduced to query evaluation (NP complete)
how:

$$\forall \vec{x}\ q_1(\vec{x}) \rightarrow q_2(\vec{x}) \quad \text{VALID}$$

① Freeze the free variables i.e consider them as constants
$\Rightarrow I_{\alpha,\vec{c}} \models q_1(\vec{c})$ implies $I_{\alpha,\vec{c}} \models q_2(\vec{c})$ $\forall I_{\alpha,\vec{c}}$ where $\vec{c}$ are new constants and $I_{\alpha,\vec{c}}$ extends I to new constants with $c^{I_{\alpha,\vec{c}}} = \alpha(x)$
② Construct canonical interpretation $I_{q_1(\vec{c})}$ of CQ $q_1(\vec{c})$
③ Evaluate on $I_{q_1(\vec{c})}$ the CQ $q_2(\vec{c})$ i.e check if $I_{q_1(\vec{c})} \models q_2(\vec{c})$

Th. For CQs $q_1(\vec{x}) \subseteq q_2(\vec{x})$ iff $I_{q_1(\vec{c})} \models q_2(\vec{c})$ where $\vec{c}$ are new constants
Th. For CQs $I \models q$ iff $q_I \subseteq q$
Th. CQs containment is NP-complete

UNION OF CQ (UCQs)

UCQ has the form $\bigvee_{i=1\dots n} \exists \vec{y_i}.\, conj_i(\vec{x}, \vec{y_i})$ i.e an OR of conjunctive queries.
As a normal "$\vee$", $I, \alpha \models \vee(\dots)$ iff $I, \alpha \models \exists \vec{y_i} conj_i(\vec{x}, \vec{y_i})$ for some i i.e at least one y_i satisfy
Also for UCQs we can have containment: $\{q_1 \dots q_k\} \subseteq \{q'_1 \dots q'_n\}$ iff for each q_i there is a q'_j s.t. $q_i \subseteq q'_j$

QUERY ANSWERING WITH INCOMPLETE INFORMATION

If we dont have complete information we cant do $D \models Q$ (evaluation)
$\Rightarrow \forall I, I \models D$ implies $I \models Q$ (logical implication) i.e for each model we must check if Q is true in it too

A common form in which D can be expressed is INCOMPLETE DATABASE / NAIVE TABLES in which we can also find labelled nulls (unknown values)

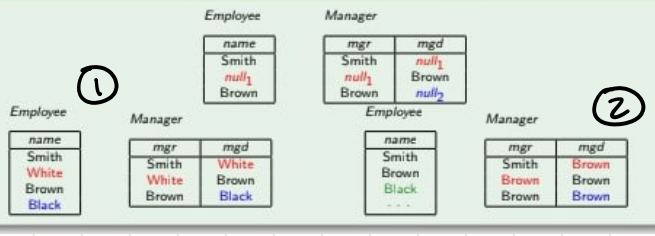
i.e different values of null
SQL cant use them!

Semantics of incomplete databases:

- A valuation function for nulls is a assignment function $\sigma : Nulls \rightarrow Const$ (essentially nulls are considered as individual variables in logic).
- We denote by $\mathcal{I}, \sigma \models D$ the fact that for every tuple $(t_1, \dots, t_n) \in P$ for each table P we have $\mathcal{I}, \sigma \models P(t_1, \dots, t_n)$.
- We define in logic the set of databases completing D as

$$Models(D) = \{\mathcal{I} \mid \text{there exists a } \sigma \text{ such that } \mathcal{I}, \sigma \models D\}$$

Example



Answering a query in incomplete databases means computing the certain answer denoted as $cert(q, \mathcal{D})$, the set of tuples $\vec{c}$ of constants of Const s.t. $\vec{c} \in q^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{D}$

- if q is boolean and $\mathcal{D}$ incomplete : $\mathcal{D} \models q$ iff q evaluates to true in every model $\mathcal{I}$ of $\mathcal{D}$ ($\mathcal{D} \not\models q$ otherwise)
- $\mathcal{D} \not\models q$ is totally different from $\mathcal{D} \models \neg q$

How to use CQs in incomplete DB?

- Each tuple in a table of $\mathcal{D}$ becomes an atom in conjunctive query $q_{\mathcal{D}}$
- Each labelled null occuring in $\mathcal{D}$ become an existentially quantified variable in $q_{\mathcal{D}}$

Th. $\mathcal{D} \models q$ iff $q_{\mathcal{D}} \subseteq q$ with q = boolean(union) of conjunctive queries.

Th. $\mathcal{D} \models q$ iff $I_{q_{\mathcal{D}}} \models q$ → nulls becomes constants (x_1, x_2 ...)

Example

E(mployee)

name
Smith
$null_1$
Brown

M(anager)

mgr	*mgd*
Smith	$null_1$
$null_1$	Brown
Brown	$null_2$

- Queries:
  $q_1(x, y) \leftarrow M(x, y)$
  $q_2(x) \leftarrow \exists y.M(x, y)$
  $q_3(x) \leftarrow \exists y_1, y_2, y_3.M(x, y_1) \wedge M(y_1, y_2) \wedge M(y_2, y_3)$
  $q_4(x, y_3) \leftarrow \exists y_1, y_2.M(x, y_1) \wedge M(y_1, y_2) \wedge M(y_2, y_3)$
- Answers:
  q_1: { }
  q_2: { Smith, Brown }
  q_3: { Smith }
  q_4: { }

For non boolean (U)CQs :

- evaluate q in $\mathcal{D}$ as it was a complete database
- filter all answers where nulls appears and remove them

UML CLASS DIAGRAM IN FOL

Lets stat with an exercise : → The specification of this domain can be done in UML

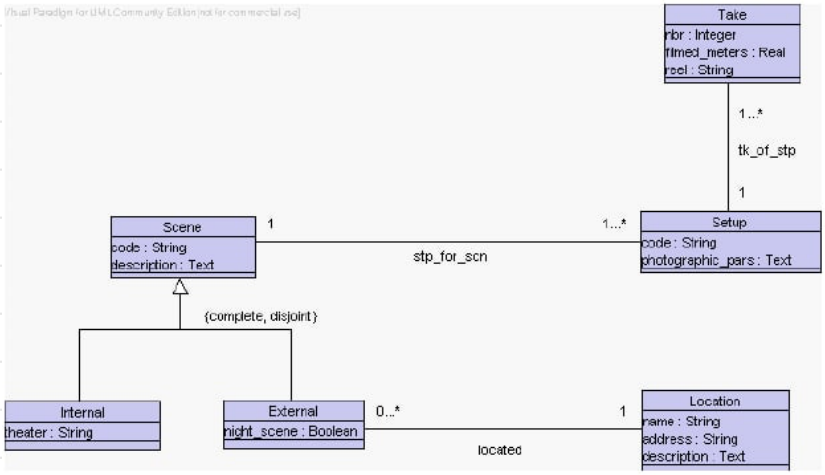
Requirements: We are interested in building a software application to manage filmed scenes for realizing a movie, by following the so-called "Hollywood Approach".

Every **scene** is identified by a code (a string) and it is described by a text in natural language.

Every scene is filmed from different positions (at least one), each of this is called a **setup**. Every setup is characterized by a code (a string) and a text in natural language where the photographic parameters are noted (e.g., aperture, exposure, focal length, filters, etc.). Note that a setup is related to a single scene.

For every setup, several **takes** may be filmed (at least one). Every take is characterized by a (positive) natural number, a real number representing the number of meters of film that have been used for shooting the take, and the code (a string) of the reel where the film is stored. Note that a take is associated to a single setup.

Scenes are divided into **internals** that are filmed in a theater, and **externals** that are filmed in a **location** and can either be "day scene" or "night scene". Locations are characterized by a code (a string) and the address of the location, and a text describing them in natural language.



All good but : not precise, verification done by humans, machine incomprehensible etc.

⇒ Use logic

Alphabet: $Scene(x)$, $Setup(x)$, $Take(x)$, $Internal(x)$, $External(x)$, $Location(x)$, $stp\_for\_scn(x,y)$, $ck\_of\_stp(x,y)$, $located(x,y)$, ...

Axioms:

$\forall x, y.\ code_{Scene}(x,y) \supset Scene(x) \wedge String(y)$
$\forall x, y.\ description(x,y) \supset Scene(x) \wedge Text(y)$

$\forall x, y.\ code_{Setup}(x,y) \supset Setup(x) \wedge String(y)$
$\forall x, y.\ photographic\_pars(x,y) \supset Setup(x) \wedge Text(y)$

$\forall x, y.\ nbr(x,y) \supset Take(x) \wedge Integer(y)$
$\forall x, y.\ filmed\_meters(x,y) \supset Take(x) \wedge Real(y)$
$\forall x, y.\ reel(x,y) \supset Take(x) \wedge String(y)$

$\forall x, y.\ theater(x,y) \supset Internal(x) \wedge String(y)$

$\forall x, y.\ night\_scene(x,y) \supset External(x) \wedge Boolean(y)$

$\forall x, y.\ name(x,y) \supset Location(x) \wedge String(y)$
$\forall x, y.\ address(x,y) \supset Location(x) \wedge String(y)$
$\forall x, y.\ description(x,y) \supset Location(x) \wedge Text(y)$

$\forall x.\ Scene(x) \supset (1 \leq \sharp\{y \mid code_{Scene}(x,y)\} \leq 1)$
. . .

$\forall x, y.\ stp\_for\_scn(x,y) \supset Setup(x) \wedge Scene(y)$
$\forall x, y.\ tk\_of\_stp(x,y) \supset Take(x) \wedge Setup(y)$
$\forall x, y.\ located(x,y) \supset External(x) \wedge Location(y)$

$\forall x.\ Setup(x) \supset 1 \leq \sharp\{y \mid stp\_for\_scn(x,y)\} \leq 1$
$\forall y.\ Scene(y) \supset 1 \leq \sharp\{x \mid stp\_for\_scn(x,y)\}$
$\forall x.\ Take(x) \supset 1 \leq \sharp\{y \mid tk\_of\_stp(x,y)\} \leq 1$
$\forall x.\ Setup(y) \supset 1 \leq \sharp\{x \mid tk\_of\_stp(x,y)\}$
$\forall x.\ External(x) \supset 1 \leq \sharp\{y \mid located(x,y)\} \leq 1$

$\forall x.\ Internal(x) \supset Scene(x)$
$\forall x.\ External(x) \supset Scene(x)$
$\forall x.\ Internal(x) \supset \neg External(x)$
$\forall x.\ Scene(x) \supset Internal(x) \vee External(x)$

On the contrary : hard for humans, difficult to generate

⇒ 

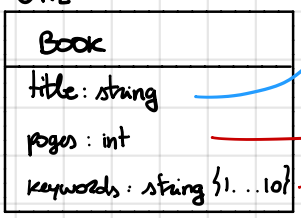
⇐

- Assign formal semantics to constructs of the conceptual design diagram
- Use it as usual
- Read diagrams as logical theories

LOOK AT SLIDES FOR UML RECAP

ATTRIBUTES formalization

UML



FOL

$\forall x,y.\ title(x,y) \supset Book(x) \wedge String(y)$

$\forall x.\ Book(x) \supset (1 \leq \#\{y \mid title(x,y)\} \leq 1)$ (→ only one title)

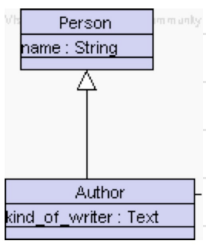
. . .

. . .

Each attributes is described using two FOL formulas
Second one (multiplicity) is a short hand for:
- At least 1 : $\exists y, a(x,y)$
- At most
  - 1 : $\forall y_1, y_2, a(x,y_1) \wedge a(x,y_2) \supset y_1 = y_2$
  - x : $\exists y_1 \dots y_m \quad a(x,y_1) \wedge \dots \wedge a(x,y_m) \wedge y_1 \neq y_2 \wedge y_1 \neq y_2 \wedge \dots \wedge y_{m-1} \neq y_m$

Some thing can be done with ASSOCIATIONS
eg $\forall x,y$ written_by $(x,y) \supset Book(x) \wedge Author(y)$
$\forall x$ Book $(x) \supset (1 \leq \#\{y \mid$ written_by $(x,y)\})$

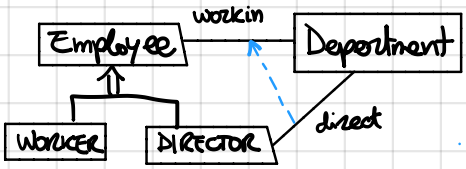
ISA



$\forall x$ Author$(x) \supset$ Person(x)
$\forall x$ name$(x,y) \supset$ Person$(x) \wedge$ String(y)

$\forall x$ Author$(x) \supset 1 \leq \#\{y \mid$ kow$(x,y)\} \leq 1$
$\forall xy$ kow$(x,y) \supset$ Author$(x) \wedge$ Text(y)

This works also for "subsets" of associations

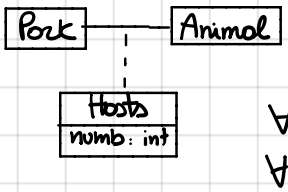


$\forall x,y$ Directs$(x,y) \supset$ worksin(x,y)

Note: direct inherits max multiplicity from workin
workin inherits min multiplicity from direct
This is logically implied anyway.

complete: $\forall x$ $E(x) \supset Dir(x) \vee W(x)$
disjoint: $\forall x$ $W(x) \supset \neg Dir(x)$ Note: viceversa not needed

Association Class = association with attributes
eg

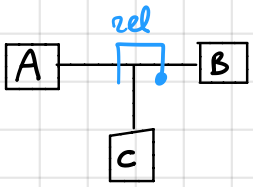


$\forall xyz$ numbers $(x,y,z) \supset$ hosts $(x,y) \wedge int(z)$
$\forall xy$ hosts $(x,y) \supset 1 \leq \#\{z \mid$ numbers $(x,y,z)\} \leq 1$
+ normal stuff of classic associations

KEY

$\forall x y \; Pd(x,y) \supset P(x) \wedge D(y)$
$\forall x y \; Pp(x,y) \supset P(x) \wedge P_2(y)$
} + multiplicity

$\forall x x' y z \quad pd(x,y) \wedge pd(x',y) \wedge pp(x,z) \wedge pp(x',z) \supset x = x'$



$\forall x y z \; rel(x,y,z) \supset A(x) \wedge B(y) \wedge C(z)$ as always
plus $\forall x y z z' \; rel(x,y,z) \wedge rel(x,y,z') \supset z = z'$

FORMS OF REASONING

Let be Γ the set of FOL assertions corresponding to UML diagram and $C(x)$ the predicate of class C.

C is consistent iff: $\Gamma \not\models \forall x \; C(x) \supset false$ i.e there must exist a model of Γ where the extension of $C(x)$ is not the empty set

The diagram is consistent iff Γ is satisfiable i.e Γ admits at least one model

C_1 subsumes C_2 if the class diagram implies that C_1 is a generalization of C_2 : $\Gamma \models \forall x \; C_2 \supset C_1(x)$

Two classes are equivalent if they denote the same set of instances whenever the conditions imposed by diagram are satisfied

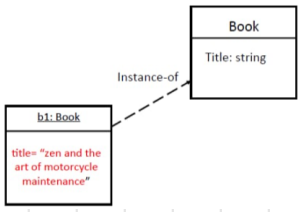
CHECKING UML DIAGRAMS INSTANTIATIOS

An instantiation (object diagram) is an extension to all classes, associations and attributes, describing properties of single objects or relations between them ⟶ Describe actual data = database.

Two cases:
① NO ISA : diagram acts as constraints
② ISA : instantiation is partial, not explicit instantiate superclasses. Start from "complete" form.

① In this case the instantiation of the diagram is a first order interpretation of FOL :
$I = (Obj^I, C_1^I \dots, C_n^I, A^I \dots, A_m^I, T^I \dots T_e^I, a^I \dots a_k^I$

eg.



$B(x), t(x,y), \text{String}(x)$

UML Class Diagram Axioms

$\forall x,y.t(x,y) \rightarrow B(x) \wedge \text{String}(y)$
$\forall x.B(x) \rightarrow 1 \leq \sharp\{y \mid t(x,y)\} \leq 1$
Γ

⟶

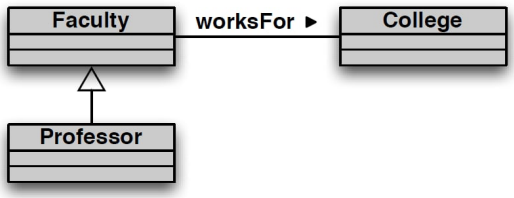
Instantiation

$\mathcal{I} = (Obj^{\mathcal{I}}, B^{\mathcal{I}}, String^{\mathcal{I}}, t^{\mathcal{I}})$

$B^{\mathcal{I}} = \{b1\}$
$t^{\mathcal{I}} = \{(b1, "zen")\}$
$String^{\mathcal{I}} = all\ strings$
$Obj^{\mathcal{I}} = \{b1\}$

If the instantiation is correct we can query it Just computing query q over $\mathcal{I}$ ignoring constraints Γ

② ISA - complete : each object in the subclass is also in the superclass



$\mathcal{I}$: $\text{Faculty}^{\mathcal{I}} = \{ \text{john}, \text{mary}, \text{paul} \}$
$\text{Professor}^{\mathcal{I}} = \{ \text{john}, \text{paul} \}$
$\text{College}^{\mathcal{I}} = \{ \text{collA}, \text{collB} \}$
$\text{worksFor}^{\mathcal{I}} = \{ (\text{john},\text{collA}), (\text{mary},\text{collB}) \}$

ISA - NOT complete : for every object it is assumed that the instantiation state explicitly which are the most specific classes it is instance of

$\mathcal{I}$: $\text{Faculty}^{\mathcal{I}} : \{\text{mary}\}$ – *incomplete!*
$\text{Professor}^{\mathcal{I}} : \{ \text{john}, \text{paul} \}$
$\text{College}^{\mathcal{I}} : \{ \text{collA}, \text{collB} \}$
$\text{worksFor}^{\mathcal{I}} : \{ (\text{john},\text{collA}), (\text{mary},\text{collB}) \}$

In this case, when answering query, we have to use complete instantiation. eg. John and Paul are still "Faculty" too

STRUCTURAL OPERATIONAL SEMANTICS OF PROGRAM

Evolution Semantic: $(\delta, s) \longrightarrow s'$ where δ is a program, s is memory state before evaluation and s' the state after the evaluation.

How do we define this relation?

STRUCTURAL RULES

$$\frac{\text{CONSEQUENT}}{\text{ANTECEDENT}} \quad \text{IF} \quad \text{SIDE-CONDITION} \qquad \Rightarrow \forall(\text{ANT} \wedge \text{SIDE} \supset \text{CONS})$$

$$Act: \quad \frac{(a, s) \longrightarrow s'}{true} \quad \text{if } s \models Pre(a) \text{ and } s' = Post(a, s)$$

special case: assignment $\quad \frac{(x := v, s) \longrightarrow s'}{true} \quad \text{if } s' = s[x = v]$

$$Skip: \quad \frac{(skip, s) \longrightarrow s}{true}$$

$$Seq: \quad \frac{(\delta_1; \delta_2, s) \longrightarrow s'}{(\delta_1, s) \longrightarrow s'' \ \wedge \ (\delta_2, s'') \longrightarrow s'}$$

$$if: \quad \frac{(\text{if } \phi \text{ then } \delta_1 \text{ else } \delta_2, s) \longrightarrow s'}{(\delta_1, s) \longrightarrow s'} \quad \text{if } s \models \phi \qquad \frac{(\text{if } \phi \text{ then } \delta_1 \text{ else } \delta_2, s) \longrightarrow s'}{(\delta_2, s) \longrightarrow s'} \quad \text{if } s \models \neg\phi$$

$$while: \quad \frac{(\text{while } \phi \text{ do } \delta, s) \longrightarrow s}{true} \quad \text{if } s \models \neg\phi \qquad \frac{(\text{while } \phi \text{ do } \delta, s) \longrightarrow s'}{(\delta, s) \longrightarrow s'' \ \wedge \ (\text{while } \phi \text{ do } \delta, s'') \longrightarrow s'} \quad \text{if } s \models \phi$$

ex Compute S_f in following cases assuming that in S_0 we have $x=10$ and $y=0$

- $(X := x+1;\ x := x*2, S_0) \longrightarrow S_f$

↳ $\dfrac{(X := x+1;\ x := x*2, S_0) \longrightarrow S_f}{(x := x+1, S_0) \longrightarrow S_1 \ \wedge \ (x := x*2, S_1) \longrightarrow S_f}$ (true, true)

S_1: $x = 11$, $y = 0$ $\qquad$ S_f: $x = 22$, $y = 0$

- $(x := x+1;\ \text{if } (x<10) \text{ then } x := 0 \text{ else } x := 1;\ x := x+1, S_0) \longrightarrow S_f$

↳

$$\frac{(x := x+1;\ \text{if } (x<10) \text{ then } x := 0 \text{ else } x := 1;\ x := x+1, S_0) \longrightarrow S_f}{x := x+1, S_0 \longrightarrow S_1 \ \wedge \ \dfrac{(\text{if } \ldots, x := x+1, S_1) \longrightarrow S_f}{\dfrac{(\text{if } \ldots, S_1) \longrightarrow S_2 \ \wedge \ (x+1, S_2) \longrightarrow S_f}{x := 1, S_1 \longrightarrow S_2}}}$$

(true, true, true)

S_1: $x = 11$, $y = 0$

S_2: $x = 1$, $y = 0$

S_f: $x = 12$, $y = 0$

TRANSITION SEMANTICS

This kind of semantic just compute a single step: $\delta, s \rightarrow \delta', s'$

Given δ and s, compute the state s' and the program δ' that remains to be executed obtained by executing a single step of δ in s

Note: long arrow $\longrightarrow$ for normal semantic, short arrow $\rightarrow$ for transition semantics.

Act : $\dfrac{(a, s) \longrightarrow (\epsilon, s')}{true}$ if $s \models Pre(a)$ and $s' = Post(a, s)$

special case: assignment $\dfrac{(x := v, s) \longrightarrow (\epsilon, s')}{true}$ if $s' = s[x = v]$

$Skip$: $\dfrac{(skip, s) \longrightarrow (\epsilon, s)}{true}$

Seq : $\dfrac{(\delta_1; \delta_2, s) \longrightarrow (\delta_1'; \delta_2, s')}{(\delta_1, s) \longrightarrow (\delta_1', s')}$ $\dfrac{(\delta_1; \delta_2, s) \longrightarrow (\delta_2', s')}{(\delta_2, s) \longrightarrow (\delta_2', s')}$ if $(\delta_1, s)\checkmark$ → if δ_1 is terminated $= \epsilon$

if : $\dfrac{(\textbf{if } \phi \textbf{ then } \delta_1 \textbf{ else } \delta_2, s) \longrightarrow (\delta_1', s')}{(\delta_1, s) \longrightarrow (\delta_1', s')}$ if $s \models \phi$ $\dfrac{(\textbf{if } \phi \textbf{ then } \delta_1 \textbf{ else } \delta_2, s) \longrightarrow (\delta_2', s')}{(\delta_2, s) \longrightarrow (\delta_2', s')}$ if $s \models \neg\phi$

$while$: $\dfrac{(\textbf{while } \phi \textbf{ do } \delta, s) \longrightarrow (\delta'; \textbf{while } \phi \textbf{ do } \delta, s')}{(\delta, s) \longrightarrow (\delta', s')}$ if $s \models \phi$

「ϵ is the empty program」

TERMINATION RULES

ϵ : $\dfrac{(\epsilon, s)\checkmark}{true}$

Seq : $\dfrac{(\delta_1; \delta_2, s)\checkmark}{(\delta_1, s)\checkmark \ \wedge \ (\delta_2; s)\checkmark}$

if : $\dfrac{(\textbf{if } \phi \textbf{ then } \delta_1 \textbf{ else } \delta_2, s)\checkmark}{(\delta_1, s)\checkmark}$ if $s \models \phi$ $\dfrac{(\textbf{if } \phi \textbf{ then } \delta_1 \textbf{ else } \delta_2, s)\checkmark}{(\delta_2, s)\checkmark}$ if $s \models \neg\phi$

$while$: $\dfrac{(\textbf{while } \phi \textbf{ do } \delta, s)\checkmark}{true}$ if $s \models \neg\phi$ $\dfrac{(\textbf{while } \phi \textbf{ do } \delta, s)\checkmark}{(\delta, s)\checkmark}$ if $s \models \phi$

Eg Compute δ', s' assuming in S_0 $x=10$ and $y=0$

- $(\text{while } (y<2) \text{ do } \{x := x*2, y := y+1\}, S_0) \longrightarrow S_f$ long arrow

$S_2 = \{x=20, y=0\}$ $S_1 = \{x=20, y=1\}$ $S_4 = \{x=40, y=1\}$ $S_3 = \{x=40, y=2\}$

$(x := x*2; y := y+1; S_0) \longrightarrow S_1 \ \wedge \ (\text{while } \_, S_1 \longrightarrow S_f)$

$(x := x*2, S_0) \longrightarrow S_2 \ \wedge \ (y := y+1, S_2) \longrightarrow S_1$ ↳ $(x := x*2; y := y+1; S_1) \longrightarrow S_3 \wedge (\text{while } \_ S_3) \longrightarrow S$

true true

$(x := x*2, S_1) \longrightarrow S_4 \ \wedge \ (y := y+1; S_4) \longrightarrow S_3$

true true

Some thing but with transition semantics

$(\text{while } (y<2) \text{ do } \{x := x*2, y := y+1\}, S_0) \longrightarrow (\delta_1; \text{while } \_; S_1)$

$\delta_1 = \delta_2; y = y+1 = \epsilon; y = y+1$

$\delta_2 = \epsilon$

$S_1 = \{x=20, y=0\}$

$\dfrac{(x = x*2; y = y+1, S_0) \longrightarrow (\delta_1, S)}{(x = x*2, S_0) \xrightarrow{\delta_2} (\epsilon, S_1)}$

$\dfrac{\epsilon; y := y+1; \text{while } \_, S_1 \longrightarrow \delta_3, \text{while } \_, S_2}{y := y+1; \text{while } \_, S_1 \longrightarrow \delta_3, S_2}$

$y = y+1, S_1 \longrightarrow (\epsilon, S_2)$

$\delta_3 = \epsilon, \text{while}$

$S_2 = \{x=20, y=1\}$

Continue in my nightmares...

To characterize a whole computation using single steps: $\longrightarrow^*$

$$0 \text{ step}: \quad \frac{(\delta, s) \longrightarrow^* (\delta, s)}{true}$$

$$n \text{ step}: \quad \frac{(\delta, s) \longrightarrow^* (\delta'', s'')}{(\delta, s) \longrightarrow (\delta', s') \ \wedge \ (\delta', s') \longrightarrow^* (\delta'', s'')} \quad \text{(for some } \delta', s')$$

Notice that such relation is the **reflexive-transitive closure** of (single step) $\longrightarrow$.

Th. For every while-program δ and states s and s_f:
$(\delta, s_0) \longrightarrow s_f \equiv (\delta, s_0) \rightarrow^* (\delta_f, s_f) \wedge (\delta_f, s_f)^{\checkmark}$ for some δ_f

Why we do this transaction semantic? Because it can handle concurrency.

- $(\delta_1 \parallel \delta_2)$ concurrent execution
- if ϕ then δ_1 else δ_2 synchronized conditional
- while ϕ do δ synchronized loop

] test is "free", no transition used for it.

$$\text{transition}: \quad \frac{(\delta_1 \parallel \delta_2, s) \longrightarrow (\delta_1' \parallel \delta_2, s')}{(\delta_1, s) \longrightarrow (\delta_1', s')} \qquad \frac{(\delta_1 \parallel \delta_2, s) \longrightarrow (\delta_1 \parallel \delta_2', s')}{(\delta_2, s) \longrightarrow (\delta_2', s')}$$

$$\text{termination}: \quad \frac{(\delta_1 \parallel \delta_2, s)^{\checkmark}}{(\delta_1, s)^{\checkmark} \ \wedge \ (\delta_2, s)^{\checkmark}}$$

Presence of $\delta_1 \parallel \delta_2$ makes the transition relation NONDETERMINISTIC

HOARE LOGIC

Used to reason about the correctness of programs.
We start with some language we already saw (eg skip, assignment etc)

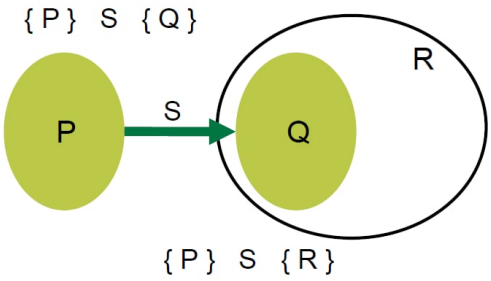
Given state $s \models P$ and $s' \models Q$, with program δ we want to check that
$s \models P \wedge \delta, s \longrightarrow s' \supset s' \models Q$.

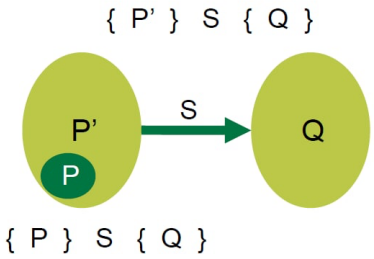
$\chi_P = \{s \mid s \models P\}$ = set of all states satisfying P
$P(s) = s \in \chi_P$

P is stronger than Q, and is weaker than P if $P \Rightarrow Q$

$\{P\}$ Pr(ogram) $\{Q\} = \forall s \; s \models P \Rightarrow (\forall s' \; \underbrace{s \; Pr \; s'}_{\delta, s \longrightarrow s'} \Rightarrow s' \models Q)$

Since the model does not express non-termination, we assume Pr terminates.
With this assumption we talk about partial correctness, total otherwise.





Post-condition weakening Rule:

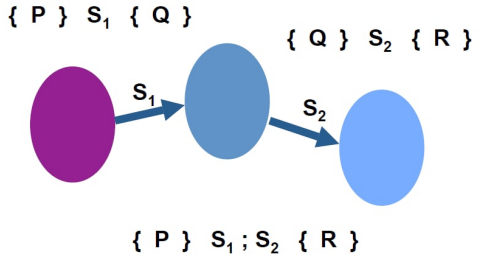
$$\frac{\{P\}\ S\ \{Q\} \quad , Q \Rightarrow R}{\{P\}\ S\ \{R\}}$$

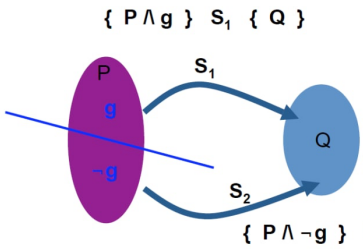
Pre-condition strengthening Rule:

$$\frac{P \Rightarrow P' \ , \ \{P'\}\ S\ \{Q\}}{\{P\}\ S\ \{Q\}}$$

Premise 1, 2

Conclusion





{ P } if g then S_1 else S_2 { Q }

$$\frac{\{P\}\ S_1\ \{Q\} \ , \ \{Q\}\ S_2\ \{R\}}{\{P\}\ S_1 ; S_2\ \{R\}}$$

$$\frac{\{P \wedge g\}\ S_1\ \{Q\} \ , \ \{P \wedge \neg g\}\ S_2\ \{Q\}}{\{P\}\ \text{if } g \text{ then } S_1 \text{ else } S_2\ \{Q\}}$$

WEAKEST PRECONDITION

wp: Statement × Pred ⟶ Pred
Let $W = wp(S, Q)$:
- Reachability: from any $s \models W$ if $s \llbracket S \rrbracket s'$ then $s' \models Q$
- Maximality: $s \llbracket S \rrbracket s'$ and $s' \models Q$ implies $s \models W$

$\Rightarrow \{P\}\ S\ \{Q\} = P \Rightarrow wp(S, Q)$
- wp skip Q = Q
- wp (x := e) Q = Q[e/x]
- wp ((S₁ ; S₂), Q) = wp(S₁, (wp(S₂, Q)))
- wp((if g then S₁ else S₂), Q) = g ∧ wp(S₁, Q) ∨ ¬g ∧ wp(S₂, Q)

Note: No while!

Ex. Compute the weakest precondition of getting $x = y$ from following program

```
S  x := y + 1
   if (y > 0) then
       x := x + y
   else x := y + 100
   x := x + y
```

wp(S, x = y)

⟶

$\{(y > 0 \wedge y + 1 = -y) \vee (y = -100)\} = \{y = -100\}$ (y = -1/2 False)

x := y + 1

$\{(y > 0 \wedge x = -y) \vee (y \leq 0 \wedge y = -100)\}$

if (y > 0) then

$\{x + y = 0\} = \{x = -y\}$

x := x + y

some $\{x = 0\}$

$\{y + 100 = 0\} = \{y = -100\}$

else x := y + 100

$\{x = 0\}$

$\{x + y = y\} = \{x = 0\}$

x := x + y

$\{x = y\}$

WHILE

$P \Rightarrow I$
$\{g \wedge I\}\ S\ \{I\}$ ← setting up invariance
$I \wedge \neg g \Rightarrow Q$ ← exit condition

$\{P\}$ while g do S $\{Q\}$

Ex Check if following Hoare triple is correct using as invariant $(0 \le i \wedge 0 \le J \wedge i+J \le 5)$

$\{i=0 \text{ AND } J=5\}$ while $(i<5)$ do $\underbrace{(J=J-1;\ i:=i+1)}_{\delta}$ $\{J=0\}$

Solution: $P = \{i=0 \text{ AND } J=5\}$

$Q = \{J=0\}$

$I = (0 \le i \wedge 0 \le J \wedge i+J \le 5)$

1. Check $P \supset I$

$i=0 \wedge J=5 \supset 0 \le i \wedge 0 \le J \wedge i+J \le 5$ ✓

3. Check $\neg g \wedge I \supset Q$

$i \ge 5 \wedge 0 \le i \wedge 0 \le J \wedge i+J \le 5 \supset J=0$ ✓

$\llcorner\ i=5 \wedge J=0$

2. Check $\{g \wedge I\}\ \delta\ \{I\} \quad = \quad g \wedge I \supset wp(\delta, I)$

$i<5 \wedge 0 \le i \wedge 0 \le J \wedge i+J \le 5 \supset wp(\delta, I)$

$\{0 \le i+1 \wedge 0 \le J-1 \wedge i+1+J-1 \le 5\}$

$J := J-1$

$\{0 \le i+1 \wedge 0 \le J \wedge i+1+J \le 5\}$

$i := i+1$

$\{0 \le i \wedge 0 \le J \wedge i+J \le 5\}$

checking i and J we can say that the Hoare triple is correct using I as invariant

Ex Compute weakest precondition for getting $\{x=0\}$ for following program:

```
x := 50+y
if (x>50) then {
    if (y>0) then {
        x := x-y
    else y := y
    }
else x := x+y
y := y+50
```

$\{y=-25\}$

$50=0 \qquad \{y \le 50 \wedge y=-25\}$

$\{50+y>50 \wedge y=50+y \vee 50+y \le 50 \wedge 50+y+y=0\}$

$x := 50+y$

$\{x>50 \wedge y=x \vee x \le 50 \wedge x+y=0\}$

$x>50 \wedge (y>0 \wedge x=y \vee y \le 0 \wedge x=0) \vee (x \le 50 \wedge x+y=0)$

$x>50$

if $(x>50)$ then $\{$

$y>0 \wedge x=y \vee y \le 0 \wedge x=0$

if $(y>0)$ then

$\{x-y=0\}$

$x := x-y$

$\{x=0\}$

else $y := y$

$\{x=0\}$

$\{x+y=0\}$

else $x := x+y$

$\{x=0\}$

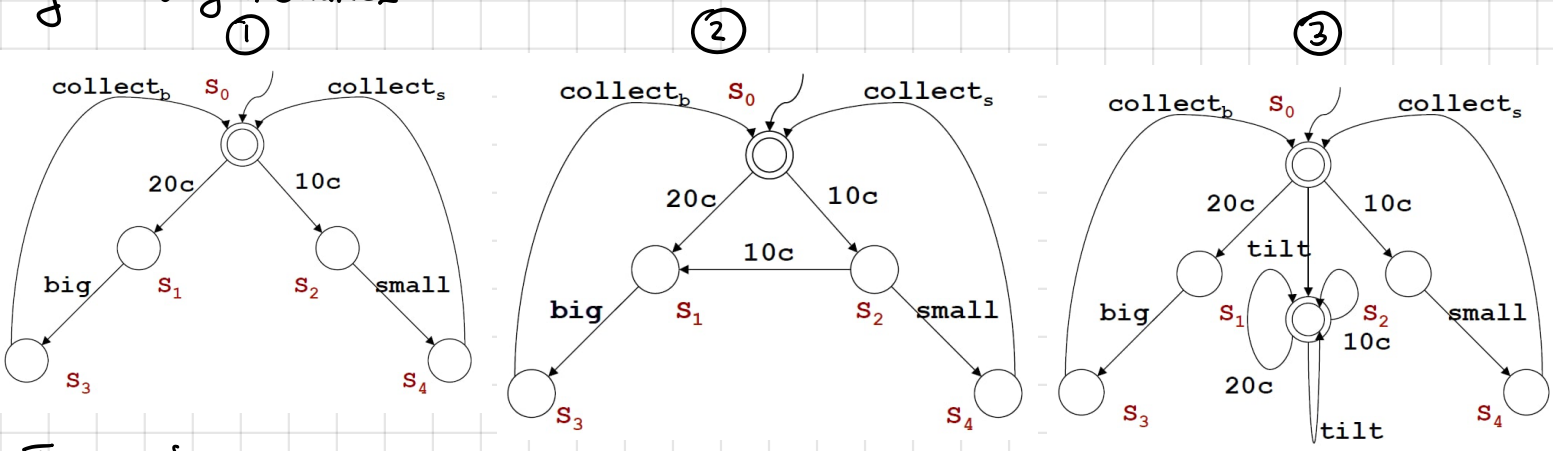
$y := y+50$

$\{x=0\}$

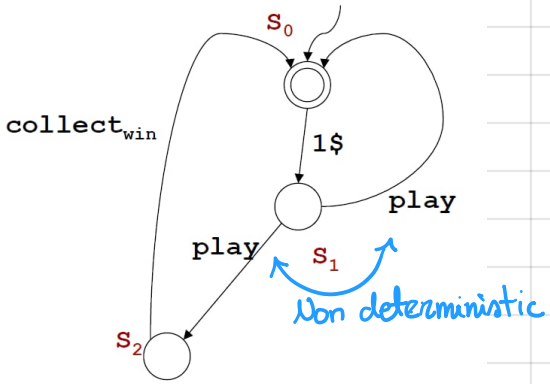
TRANSITION SYSTEMS AND BISIMULATION

Graph of transitions. Termination and reachability (of final state) let us query the system (fixpoint theory, model checking)

Eg vending machines



Eg Slot machine NON DETERMINISTIC



A transition system TS is a tuple $T = \langle A, S, S^0, \delta, F \rangle$ where:

- A = set of actions
- S = set of states
- $S^0 \subseteq S$ set of initial state
- $\delta \subseteq S \times A \times S$ = transition relation
- $F \subseteq S$ = set of final states

+ equivalent relations!

REACHABILITY

A binary relation R is a reachability like relation iff:

- $(s, s) \in R$
- if $\exists\, a, s'$ s.t. $s \rightarrow s_a \wedge (s', s'') \in R$ then $(s, s'') \in R$

A state S_0 reaches a state S_f iff $\forall$ reachability-like relations R we have $(S_0, S_f) \in R$

Algorithm ComputingReachability

Input: transition system TS
Output: the **reachable-from** relation (the smallest reachability-like relation)

Body

```
R = ∅
R' = {(s,s) | s ∈ S}
while (R ≠ R') {
        R := R'
        R' := R' ∪ {(s,s'') | ∃ s',a. s →a s' ∧ (s',s'')∈ R }
}
return R'
```

YdoB

BISIMULATION

Two transition systems are bisimilar if they have the same behaviour:
- locally they look indistinguishable
- every action that can be done on one of them can also be done on the other remaining indistinguishable.

R is a bisimulation iff $(s,t) \in R$ implies that:
- s is final iff t is final
- $\forall$ actions a:
  - if $s \rightarrow_a s'$ then $\exists t'$ s.t. $t \rightarrow_a t'$ and $(s',t') \in R$
  - if $t \rightarrow_a t'$ then $\exists s'$ s.t. $s \rightarrow_a s'$ and $(s',t') \in R$

A state s_0 is bisimilar (or equivalent) to state t_0 iff there exists a bisimulation between the initial states s_0 and t_0

LEAST AND GREATEST FIXPOINTS

Consider $X = f(X)$ where f is an operator from 2^S to 2^S (2^S = set of all subsets of S)
Every solution $\mathcal{E}$ of this equation is called fixpoint of f. Smallest and greatest solution are called least and greatest fixpoints.

Every $\mathcal{E}$ s.t. $f(\mathcal{E}) \subseteq \mathcal{E}$ is called pre-fixpoint
Every $\mathcal{E}$ s.t. $\mathcal{E} \subseteq f(\mathcal{E})$ is called post-fixpoint

f is monotonic if $\mathcal{E}_1 \subseteq \mathcal{E}_2$ implies $f(\mathcal{E}_1) \subseteq f(\mathcal{E}_2)$. In this case:
Th. • there exists a unique least fixpoints of f which is given by: $\bigcap \{\mathcal{E} \subseteq S \mid f(\mathcal{E}) \subseteq \mathcal{E}\}$ (pre-fixpoints)
• there exists a unique greatest fixpoint of f which is given by: $\bigcup \{\mathcal{E} \subseteq S \mid \mathcal{E} \subseteq f(\mathcal{E})\}$ (post-fixpoints)

APPROXIMATES

The approximates for a least fixpoint $\mathcal{L} = \bigcap \{\mathcal{E} \subseteq S \mid f(\mathcal{E}) \subseteq \mathcal{E}\}$ are as follow:

$$\left.\begin{array}{l} Z_0 = \emptyset \\ Z_1 = f(Z_0) \\ Z_2 = f(Z_1) \end{array}\right\} \quad \underline{\forall i,\ Z_i \subseteq Z_{i+1}} \quad , \quad \underline{\forall i\ \ Z_i \subseteq \mathcal{L}}$$

...

Th. If for some n $Z_{n+1} = Z_n$ then $Z_n = \mathcal{L}$

Least fixpoint algorithm

$Z_{old} := \emptyset;$
$Z := f(Z_{old});$
while $(Z \neq Z_{old})\{$
  $Z_{old} := Z;$
  $Z := f(Z);$
$\}$

Similar for greatest fixpoint $\mathcal{G} = \bigcup \{\mathcal{E} \subseteq S \mid \mathcal{E} \subseteq f(\mathcal{E})\}$.

$$\left.\begin{array}{l} Z_0 = S \\ Z_1 = f(Z_0) \\ Z_2 = f(Z_1) \end{array}\right\} \quad \underline{\forall i\ \ Z_{i+1} \subseteq Z_i} \quad , \quad \underline{\forall i\ \ \mathcal{G} \subseteq Z_i}$$

...

Th. If for some n, $Z_{n+1} = Z_n$ then $Z_n = \mathcal{G}$

Greatest fixpoint algorithm

$Z_{old} := S;$
$Z := f(Z_{old});$
while $(Z \neq Z_{old})\{$
  $Z_{old} := Z;$
  $Z := f(Z);$
$\}$

μ - CALCULUS

Logic based on fixpoints. Composed by:
- Propositions: to denote properties of the global store in a given configuration
- Modalities: to denote the capability of performing certain actions in a given configuration
- Least/Greatest Fixpoints: to denote "temporal" properties of the system (induction / coinduction)

μ-calculus syntax

modal operator — least — greatest

$$\Phi ::= A \mid true \mid false \mid \neg\Phi \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid \langle a \rangle \Phi \mid [a]\Phi \mid \mu X.\Phi \mid \nu X.\Phi \mid X$$

FOL — diamond — box — fixpoint theory

Diamond = exist a state where Φ holds
Box = every states imply Φ

For formulae of the form $\mu X.\Phi$ and $\nu X.\Phi$ we require syntactic monotonicity of Φ.
Every occurrence of variable X in Φ must be within the scope of an even number of negation signs.
In μ-calculus, given the requirement of syntactic monotonicity, the least fixpoint and the greatest one ALWAYS exist.

Transition System: $T = (S, \{R_a \mid a \in A\}, \Pi)$
P = given proposition set → S = set of states
A = given set of atomic actions → R_a = family of relations
Π = mapping from P to a subset of S

Valuation: given transition system T, a valuation V on T is a mapping from variables in Var to subsets of states in T.

μ-calculus semantics

$$
\begin{aligned}
(A)^T_V &= \Pi(A) \subseteq S \\
(X)^T_V &= V(X) \subseteq S \\
(true)^T_V &= S \\
(false)^T_V &= \emptyset \\
(\neg\Phi)^T_V &= S - (\Phi)^T_V \\
(\Phi_1 \wedge \Phi_2)^T_V &= (\Phi_1)^T_V \cap (\Phi_2)^T_V \\
(\Phi_1 \vee \Phi_2)^T_V &= (\Phi_1)^T_V \cup (\Phi_2)^T_V \\
(\langle a \rangle \Phi)^T_V &= \{s \in S \mid \exists s'.\ (s,s') \in R_a \text{ and } s' \in (\Phi)^T_V\} \\
([a]\Phi)^T_V &= \{s \in S \mid \forall s'.\ (s,s') \in R_a \text{ implies } s' \in (\Phi)^T_V\} \\
(\mu X.\Phi)^T_V &= \bigcap\{\mathcal{E} \subseteq S \mid (\Phi)^T_{V[X \leftarrow \mathcal{E}]} \subseteq \mathcal{E}\} \quad \text{least fp. of } \Phi \\
(\nu X.\Phi)^T_V &= \bigcup\{\mathcal{E} \subseteq S \mid \mathcal{E} \subseteq (\Phi)^T_{V[X \leftarrow \mathcal{E}]}\} \quad \text{greatest fp. of } \Phi
\end{aligned}
$$

Given $T = (S, \{R_a \mid a \in 2^A\}, \Pi)$

- $(\Phi)^T_V$ = set of states that satisfy Φ

Examples:
1) ⟨next⟩true = capability of making a next-transition
2) [next]false = inability of making any next-transition
3) ⟨next⟩true ∧ [next]P = next-transition are allowed and all reach states where P holds
4) μX. P ∨ ⟨next⟩X = exists an evolution of the system s.t. P eventually holds.
5) νX.P ∧ [next]X = ¬μX.¬P∨⟨next⟩X = invariance of P under all of the evolutions of the system.
6) μX.P ∨(⟨next⟩true ∧ [next]X) = for all evolutions of the system P eventually holds
7) νX.μY.(P∧⟨next⟩X) ∨ (⟨next⟩Y) = strong fairness, there exists a run where P is true infinitely often

We are interested in model checking: querying the transition system.
Let $T = (S, \{R_a \mid a \in A\}, \Pi)$ be a transition system, let $s \in S$ be one of its states and Φ be a closed (no free variables) μ-calculus formula. The related model checking problem is to verify whether $s \in (\Phi)^T_v$ where v is any valuation since Φ is closed. Also written $T, s \models \Phi$ or just $s \models \Phi$

μ-calculus model checking algorithm

$$
\begin{array}{rcl}
[\![A]\!]^T_v & = & \Pi(A) \\
[\![X]\!]^T_v & = & v(X) \\
[\![true]\!]^T_v & = & S \\
[\![false]\!]^T_v & = & \emptyset \\
[\![\neg\Phi]\!]^T_v & = & S - [\![\Phi]\!]^T_v \\
[\![\Phi_1 \wedge \Phi_2]\!]^T_v & = & [\![\Phi_1]\!]^T_v \cap [\![\Phi_2]\!]^T_v \\
[\![\Phi_1 \vee \Phi_2]\!]^T_v & = & [\![\Phi_1]\!]^T_v \cup [\![\Phi_2]\!]^T_v
\end{array}
\qquad
\begin{array}{rcl}
[\![\langle a \rangle \Phi]\!]^T_v & = & \mathrm{PREE}(a, [\![\Phi]\!]^T_v) \\
[\![[a]\Phi]\!]^T_v & = & \mathrm{PREA}(a, [\![\Phi]\!]^T_v) \\
 & & \\
[\![\mu X.\Phi]\!]^T_v & = & \mathrm{LFP}X.[\![\Phi]\!]^T_v \\
[\![\nu X.\Phi]\!]^T_v & = & \mathrm{GFP}X.[\![\Phi]\!]^T_v
\end{array}
$$

$\mathrm{PreE}(a, \mathcal{E})$ = existential a-preimage of $\mathcal{E}$ $= \{s \in S \mid \exists s'.\ (s, s') \in R_a$ and $s' \in \mathcal{E}\}$
$\mathrm{PreA}(a, \mathcal{E})$ = universal a-preimage of $\mathcal{E}$ $= \{s \in S \mid \forall s'.\ (s, s') \in R_a$ implies $s' \in \mathcal{E}\}$

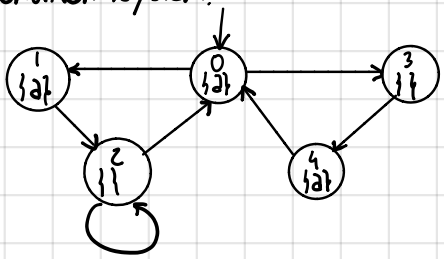
Procedure $\mathrm{LFP}X.[\![\Phi]\!]^T_v$

$\mathcal{X}_{old} := [\![False]\!]^T_v$;
$\mathcal{X} := [\![\Phi]\!]^T_{v[X \leftarrow \mathcal{X}_{old}]}$;
while $(\mathcal{X} \neq \mathcal{X}_{old})$ {
  $\mathcal{X}_{old} := \mathcal{X}$;
  $\mathcal{X} := [\![\Phi]\!]^T_{v[X \leftarrow \mathcal{X}_{old}]}$;
}
return $\mathcal{X}$;

Procedure $\mathrm{GFP}X.[\![\Phi]\!]^T_v$

$\mathcal{X}_{old} := [\![True]\!]^T_v$;
$\mathcal{X} := [\![\Phi]\!]^T_{v[X \leftarrow \mathcal{X}_{old}]}$;
while $(\mathcal{X} \neq \mathcal{X}_{old})$ {
  $\mathcal{X}_{old} := \mathcal{X}$;
  $\mathcal{X} := [\![\Phi]\!]^T_{v[X \leftarrow \mathcal{X}_{old}]}$;
}
return $\mathcal{X}$;

Ex Model check the μ-calculus formula $\mu X.\ \neg a \vee \langle next \rangle X$ against following transition system.



Lets define X_i the i-th approximation of X
$[X_i]$ = states that make formula true
$[X_0] = [false] = \emptyset$
$[X_1] = [\neg a \vee \langle next \rangle X_0] = [\neg a] \cup [\langle next \rangle X_0]$
$= [\neg a] \cup \mathrm{PreE}(\langle next \rangle, [X_0])$ ($[X_0] = \emptyset$)
$= \{2,3\} \cup \emptyset = \{2,3\}$

This set MUST get bigger!

$[X_2] = [\neg a \vee \langle next \rangle X_1] = [\neg a] \cup \mathrm{PreE}(next, [X_1]) = \{2,3\} \cup \{0,1,2\} = \{0,1,2,3\}$ — Again! ok
$[X_3] = [\neg a \vee \langle next \rangle X_2] = [\neg a] \cup \mathrm{PreE}(next, [X_2]) = \{2,3\} \cup \{0,1,2,4\} = \{0,1,2,3,4\}$ — Equal!
$[X_4] = [\neg a \vee \langle next \rangle X_3] = [\neg a] \cup \mathrm{PreE}(next, [X_3]) = \{2,3\} \cup \{0,1,2,3,4\} = \{0,1,2,3,4\}$ — Least Fix Point

Since our initial state is in (the "solution"), we can say that is true that from initial state we can reach a state where $\neg a$ is true.

What if we change formula to $\nu X.\ \neg a \vee \langle next \rangle X$?
$[X_0]$ = entire set $= \{0,1,2,3,4\}$
$[X_1] = [\neg a \vee \langle next \rangle X] = [\neg a] \cup \mathrm{PreE}(next, [X_0]) = \{2,3\} \cup \{0,1,2,3,4\} = \{0,1,2,3,4\}$ — Greatest Fix Point

$[\nu X.\ \neg a \wedge \langle next \rangle X] =$ $\neg a$ always true

$[X_0] = \{1,2,3,4\}$

In this case the set MUST shrink!

$[X_1] = [\neg a \wedge \langle next \rangle X_0] = [\neg a] \cap Pre\exists(next, [X_0]) = \{2,3\} \cap \{0,2,3,4\} = \{2,3\}$

$[X_2] = [\neg a \wedge \langle next \rangle X_1] = [\neg a] \cap Pre\exists(next, [X_1]) = \{2,3\} \cap \{0,1,2\} = \{2\}$

$[X_3] = [\neg a \wedge \langle next \rangle X_2] = [\neg a] \cap Pre\exists(next, [X_2]) = \{2,3\} \cap \{1,2\} = \{2\}$ Equal! Greatest FixPoint

Since our initial state is not in the fixpoint the formula is false

$[\mu X.\ \neg a \vee [next] X] =$ all executions reach $\neg a$ eventually

$[X_0] = \emptyset$

$[X_1] = [\neg a \vee [next] X_0] = [\neg a] \cup Pre\forall(next, [X_0]) = \{2,3\} \cup \{\emptyset\} = \{2,3\}$

$[X_2] = [\neg a \vee [next] X_1] = [\neg a] \cup Pre\forall(next, [X_1]) = \{2,3\} \cup \{1\} = \{1,2,3\}$

$[X_3] = [\neg a \vee [next] X_2] = [\neg a] \cup Pre\forall(next, [X_2]) = \{2,3\} \cup \{0,1\} = \{0,1,2,3\}$

$[X_4] = [\neg a \vee [next] X_3] = [\neg a] \cup Pre\forall(next, [X_3]) = \{2,3\} \cup \{0,1,2,4\} = \{0,1,2,3,4\}$

$[X_5] = [\neg a \vee [next] X_4] = [\neg a] \cup Pre\forall(next, [X_4]) = \{2,3\} \cup \{0,1,2,3,4\} = \{0,1,2,3,4\}$ Least Fix Point

Initial state in solution so formula is true

$[\nu X\ \mu Y.\ (a \wedge \langle next \rangle X \vee [next] Y)]$

$[X_0] = \{0,1,2,3,4\}$

$[X_1] = [\mu Y (a \wedge \langle next \rangle X_0) \vee [next] Y] = \{0,1,3,4\}$

$[Y_{10}] = \emptyset$

$[Y_{11}] = [a \wedge \langle next \rangle X_0 \vee [next] Y_{10}] = [a] \cap Pre\exists(next, [X_0]) \cup Pre\forall[next, Y_{10}] = \{0,1,4\} \cap \{0,1,2,3,4\} \cup \emptyset = \{0,1,4\}$

$[Y_{12}] = [a \wedge \langle next \rangle X_0 \vee [next] Y_{11}] = [a] \cap Pre\exists(next, [X_0]) \cup Pre\forall[next, Y_{11}] = \{0,1,4\} \cup \{4,3\} = \{0,1,3,4\}$

$[Y_{13}] = [a \wedge \langle next \rangle X_0 \vee [next] Y_{12}] = [\ldots] = \{0,1,4\} \cup \{0,3,4\} = \{0,1,3,4\}$ Least Fix Point "of X_1"

$[X_2] = [\mu Y (a \wedge \langle next \rangle X_1 \vee [next] Y] = \{0,3,4\}$

$[Y_{20}] = \emptyset$

$[Y_{21}] = [a \wedge \langle next \rangle X_1 \vee [next] Y_{20}] = \{0,1,4\} \cap \{0,2,3,4\} \cup \emptyset = \{0,4\}$

$[Y_{22}] = [a \wedge \langle next \rangle X_1 \vee [next] Y_{21}] = \{0,4\} \cup \{3,4\} = \{0,3,4\}$

$[Y_{23}] = [a \wedge \langle next \rangle X_1 \vee [next] Y_{22}] = \{0,4\} \cup \{3,4\} = \{0,3,4\}$ Least Fix Point "of X_2"

$[X_3] = [\mu Y (a \wedge \langle next \rangle X_2 \vee [next] Y] = \{0,3,4\}$

$[Y_{30}] = \emptyset$

$[Y_{31}] = [a \wedge \langle next \rangle X_2 \vee [next] Y_{30}] = \{0,1,4\} \cap \{0,2,3,4\} \cup \emptyset = \{0,4\}$

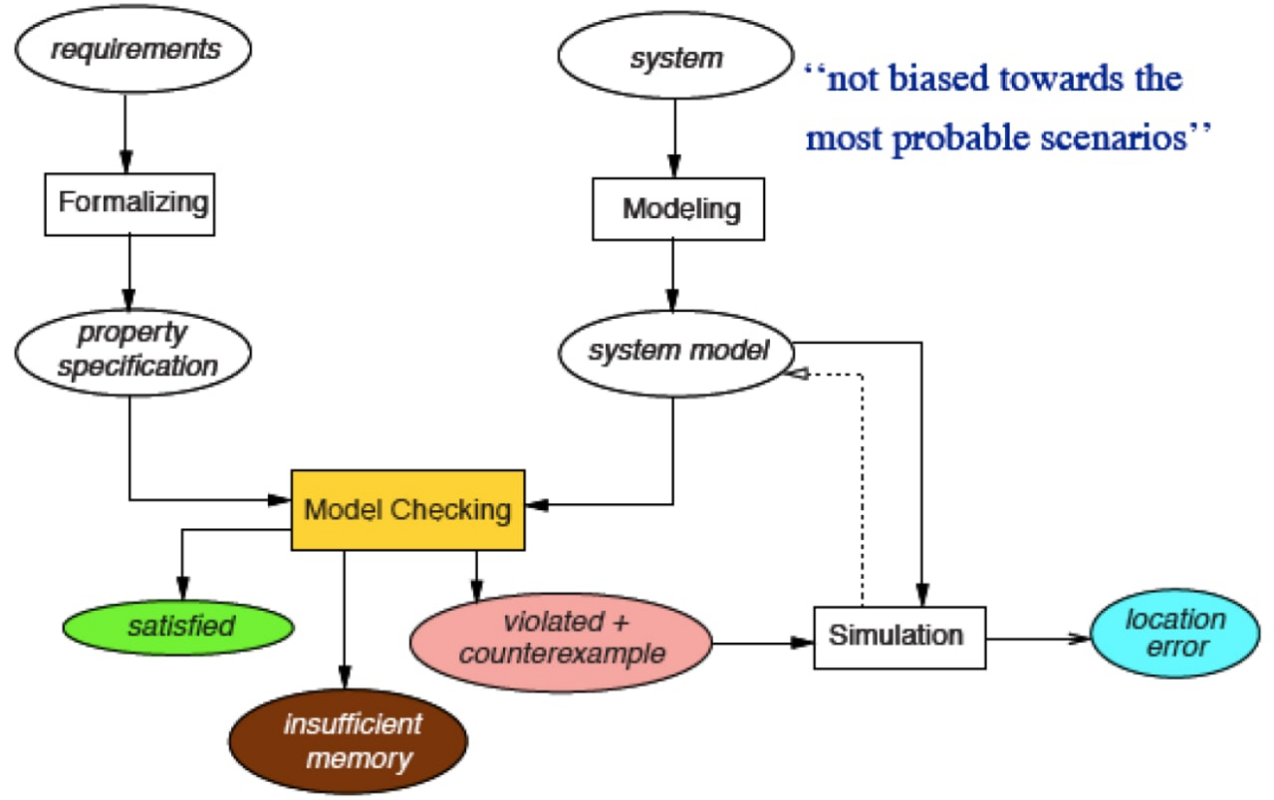
$[Y_{32}] = [a \wedge \langle next \rangle X_2 \vee [next] Y_{31}] = \{0,4\} \cup \{3,4\} = \{0,3,4\}$

$[Y_{33}] = [a \wedge \langle next \rangle X_2 \vee [next] Y_{32}] = \{0,4\} \cup \{3,4\} = \{0,3,4\}$ Least Fix Point "of X_3"

$X_2 = X_3 \rightarrow$ Greatest Fix Point

Initial state in solution $\rightarrow$ formula satisfied!

System Verification: check whether a system fulfills the qualitative requirements that have been identified. Checking we are building something in the right way.



LINEAR TEMPORAL LOGIC (LTL)

Only one timeline

$\bigcirc\varphi$	φ is true in the *next* moment in time
$\Box\varphi$	φ is true in *all* future moments
$\Diamond\varphi$	φ is true in *some* future moment
$\varphi \,\mathcal{U}\, \psi$	φ is true *until* ψ is true

SEMANTIC

$\langle M, i\rangle \models p$ iff $p \in I(i)$ for $p \in \Sigma$ = set of atomic propositions

$$\langle \mathcal{M}, i\rangle \models \neg\varphi \quad \textbf{iff} \quad \langle \mathcal{M}, i\rangle \not\models \varphi$$

$$\langle \mathcal{M}, i\rangle \models \varphi \wedge \psi \quad \textbf{iff} \quad \langle \mathcal{M}, i\rangle \models \varphi \text{ and } \langle \mathcal{M}, i\rangle \models \psi$$

$$\langle \mathcal{M}, i\rangle \models \varphi \vee \psi \quad \textbf{iff} \quad \langle \mathcal{M}, i\rangle \models \varphi \text{ or } \langle \mathcal{M}, i\rangle \models \psi$$

$$\langle \mathcal{M}, i\rangle \models \varphi \Rightarrow \psi \quad \textbf{iff} \quad \text{if } \langle \mathcal{M}, i\rangle \models \varphi \text{ then } \langle \mathcal{M}, i\rangle \models \psi$$

NEXT

Provides a constraint on the next moment in time. $\langle M, i\rangle \models \bigcirc\varphi$ iff $\langle M, i+1\rangle \models \varphi$

eg $(sad \wedge \neg rich) \Rightarrow \bigcirc sad$

EVENTUALLY / SOMETIME

φ will be true but we dont know when. $\langle M, i\rangle \models \Diamond\varphi$ iff $\exists\, j\ (j \geq i) \wedge \langle M, j\rangle \models \varphi$

eg. $(\neg resigned \wedge sad) \Rightarrow \Diamond famous$

ALWAYS

Invariant properties. $\langle M, i\rangle \models \Box\varphi$ iff $\forall\, j$ if $(j \geq i)$ then $\langle M, j\rangle \models \varphi$

eg lottery-win $\Rightarrow \Box$ rich

UNTIL

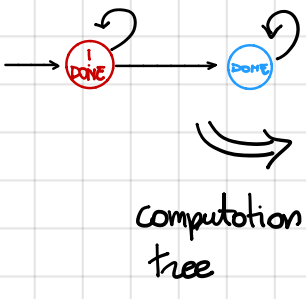
$\langle M, i\rangle \models \varphi \,\mathcal{U}\, \psi$ iff $\exists\, j\ (j \geq i) \wedge \langle M, j\rangle \models \psi \wedge \forall\, k\ \ (i \leq k \leq j) \Rightarrow \langle M, k\rangle \models \varphi$

eg. start lecture $\Rightarrow$ talk $\mathcal{U}$ end-lecture

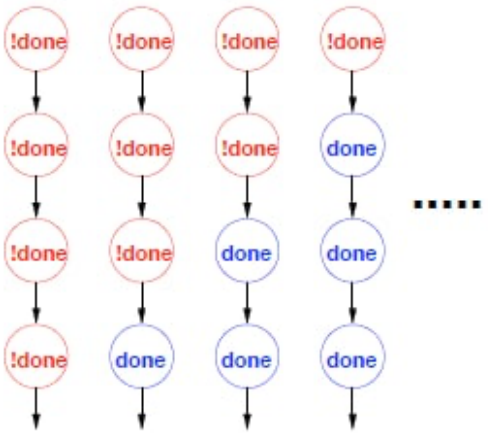
born $\Rightarrow$ alive $\mathcal{U}$ dead (φ cont lost forever and ψ sooner or later will happen)

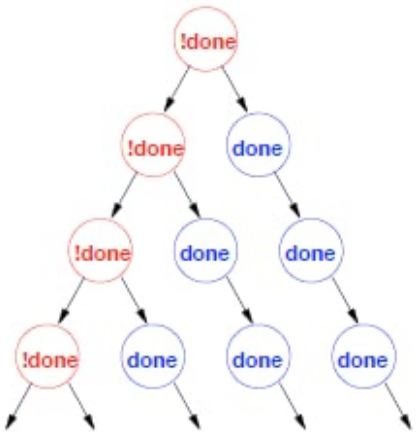
Note:

1) $\neg\Box\varphi \equiv \Diamond\neg\varphi$
2) $\Diamond\varphi \equiv \top \,\mathcal{U}\, \varphi$ ($\top$ = true)



computation tree





Sometimes symbols are translated with letters:

◇ → Future

□ → Globally in the future

O → neXtime

COMPUTATION TREE LOGIC (CTL)

Based on the possibility of several "branches" (as we saw before).
CTL explicitly introduces path quantifiers:
- all paths = A
- exists a path = E

Use some temporal operators as LTL that can be combined with quantifiers:
Universal modalities AF, AG, AX, AU
Existential modalities EF, EG, EX, EU

Syntax for single state:

$\mathcal{KM}, s_i \models \neg\varphi$ **iff** $\mathcal{KM}, s_i \not\models \varphi$

$\mathcal{KM}, s_i \models \varphi \wedge \psi$ **iff** $\mathcal{KM}, s_i \models \varphi$ and $\mathcal{KM}, s_i \models \psi$

$\mathcal{KM}, s_i \models \varphi \vee \psi$ **iff** $\mathcal{KM}, s_i \models \varphi$ or $\mathcal{KM}, s_i \models \psi$

$\mathcal{KM}, s_i \models \varphi \Rightarrow \psi$ **iff** if $\mathcal{KM}, s_i \models \varphi$ then $\mathcal{KM}, s_i \models \psi$

$\mathcal{KM}, s_i \models \top$

$\mathcal{KM}, s_i \not\models \bot$

For a path $\pi = (s_i, s_{i+1}, \ldots)$ outgoing from state s_i

$\mathcal{KM}, s_i \models \mathbf{AX}\varphi$ iff $\forall\pi = (s_i, s_{i+1}, \ldots)$ $\mathcal{KM}, s_{i+1} \models \varphi$

$\mathcal{KM}, s_i \models \mathbf{EX}\varphi$ iff $\exists\pi = (s_i, s_{i+1}, \ldots)$ $\mathcal{KM}, s_{i+1} \models \varphi$

$\mathcal{KM}, s_i \models \mathbf{AG}\varphi$ iff $\forall\pi = (s_i, s_{i+1}, \ldots)$ $\forall j \geq i. \mathcal{KM}, s_j \models \varphi$

$\mathcal{KM}, s_i \models \mathbf{EG}\varphi$ iff $\exists\pi = (s_i, s_{i+1}, \ldots)$ $\forall j \geq i. \mathcal{KM}, s_j \models \varphi$

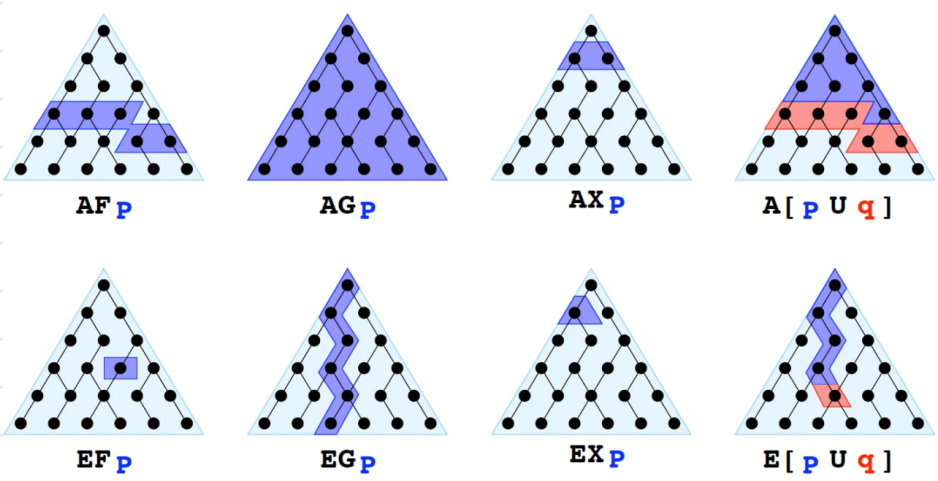
$\mathcal{KM}, s_i \models \mathbf{AF}\varphi$ iff $\forall\pi = (s_i, s_{i+1}, \ldots)$ $\exists j \geq i. \mathcal{KM}, s_j \models \varphi$

$\mathcal{KM}, s_i \models \mathbf{EF}\varphi$ iff $\exists\pi = (s_i, s_{i+1}, \ldots)$ $\exists j \geq i. \mathcal{KM}, s_j \models \varphi$

$\mathcal{KM}, s_i \models (\varphi \mathbf{AU} \psi)$ iff $\forall\pi = (s_i, s_{i+1}, \ldots)$ $\exists j \geq i. \mathcal{KM}, s_j \models \psi$ and $\forall i \leq k < j : M, s_k \models \varphi$

$\mathcal{KM}, s_i \models \varphi \mathbf{EU} \psi)$ iff $\exists\pi = (s_i, s_{i+1}, \ldots)$ $\exists j \geq i. \mathcal{KM}, s_j \models \psi$ and $\forall i \leq k < j : \mathcal{KM}, s_k \models \varphi$

finally P | globally P | next P | P until q



Logic of CTL is based on combining these 8 operators!

MODEL CHECKING

Construct denotation of φ = set of states where formula holds and then compare it with set of initial states. Denotation of φ = $[\varphi]$

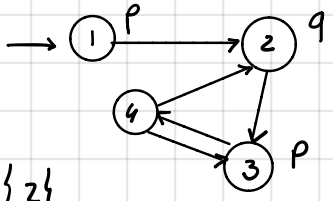
$EX p \leftrightarrow \langle next \rangle p$
$AX p \leftrightarrow [next] p$
} ⟶ we can use μ-calculus, we must "translate" to μ-calculus.

CTL	μ-calculus
p	p
$\vee, \wedge, \rightarrow$	$\vee, \wedge, \rightarrow$
$EX\varphi$	$\langle next \rangle t(\varphi)$
$AX\varphi$	$[next] t(\varphi)$
$EF\varphi$	$\mu Z\ t(\varphi) \vee \langle next \rangle Z$
$AF\varphi$	$\mu Z\ t(\varphi) \vee [next] Z$
$EG\varphi$	$\nu Z\ t(\varphi) \wedge \langle next \rangle Z$
$AG\varphi$	$\nu Z\ t(\varphi) \wedge [next] Z$
$\varphi EU \psi$	$\mu Z.\ t(\psi) \wedge (t(\varphi) \wedge \langle next \rangle Z)$
$\varphi AU \psi$	$\mu Z\ t(\psi) \vee (t(\varphi) \wedge [next] Z)$

Eg Model checking

$AG(p \Rightarrow AFq)$

$\alpha \quad \beta, \gamma$



$\alpha = AFq \rightarrow \mu Z\ q \vee [next] Z = \{1,2\}$

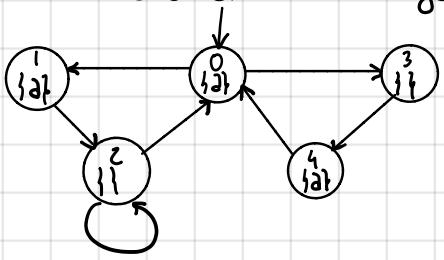
$[Z_0] = \emptyset$
$[Z_1] = [q \vee [next] Z_0] = [q] \cup Pre_A(next, [Z_0]) = \{2\}$
$[Z_2] = [q \vee [next] Z_1] = [q] \cup Pre_A(next, [Z_1]) = \{2\} \cup \{1\} = \{1,2\}$
$[Z_3] = [q \vee [next] Z_2] = [q] \cup Pre_A(next, [Z_2]) = \{2\} \cup \{1\} = \{1,2\}$

$\beta = \neg p \vee \alpha \longrightarrow [\neg p] \cup [\alpha] = \{1,2,4\}$
$\gamma = AG\beta \longrightarrow \nu Z\ \beta \wedge [next] Z = \emptyset$

$[Z_0] = \{1,2,3,4\}$
$[Z_1] = [\beta \wedge [next] Z_0] = [\beta] \cap Pre_A(next, [Z_0]) = \{1,4\} \cap \{1,2,3,4\} = \{1,2,4\}$
$[Z_2] = [\beta \wedge [next] Z_1] = [\beta] \cap Pre_A(next, [Z_1]) = \{1,2,4\} \cap \{1,3\} = \{1\}$
$[Z_3] = [\beta \wedge [next] Z_2] = [\beta] \cap Pre_A(next, [Z_2]) = \{1,2,4\} \cap \emptyset = \emptyset$
$[Z_4] = [\beta \wedge [next] Z_3] = [\beta] \cap Pre_A(next, [Z_3]) = \{1,2,4\} \cap \emptyset = \emptyset$

$T \models \gamma$? No because initial state not in γ

Ex Model check CTL formula $EF(AG(a \supset AXEX\,\neg a))$ showing its translation in μ-calculus



$[\alpha] = [EX\,\neg a]$
$[\beta] = [AX\,\alpha]$
$[\gamma] = [a \supset \beta]$
$[\delta] = [AG(\gamma)]$
$[\eta] = [EF(\delta)]$

$\alpha = EX\,\neg a \longrightarrow \langle next \rangle \neg a$
$[\alpha] = [\langle next \rangle \neg a] = pre_E(next, [-a]) = \{0,1,2\}$ ($[-a] = \{2,3\}$)
$\beta = AE\alpha \longrightarrow [-]\alpha$ (next)
$[\beta] = [[-]\alpha] = pre_A(-, [\alpha]) = \{1,2,4\}$
$[\gamma] = [a \supset \beta] = [\neg a \vee \beta] = [\neg a] \cup [\beta] = \{2,3\} \cup \{1,2,4\} = \{1,2,3,4\}$
$\delta = \underbrace{AG\,\gamma}_{Z} = \gamma \wedge AX\underbrace{AG\,\gamma}_{Z} \longrightarrow Z = \gamma \wedge AX\,Z \longrightarrow \nu Z.\, \gamma \wedge [-]Z$ gfp
$[\delta] = [AG\gamma] = [\nu Z\, \gamma \wedge [-]Z] = \emptyset$
$[Z_0] = \{0,1,2,3,4\}$
$[Z_1] = [\gamma \wedge [-]Z_0] = [\gamma \cap pre_A(-, [Z_0])] = \{1,2,3,4\} \cap \{0,1,2,3,4\} = \{1,2,3,4\}$
$[Z_2] = [\gamma \wedge [-]Z_1] = [\gamma \cap pre_A(-, [Z_1])] = \{1,2,3,4\} \cap \{0,1,3\} = \{1,3\}$
$[Z_3] = [\gamma \wedge [-]Z_2] = [\gamma \cap pre_A(-, [Z_2])] = \{1,2,3,4\} \cap \{0\} = \emptyset$
$[Z_4] = [\gamma \wedge [-]Z_3] = [\gamma \cap pre_A(-, [Z_3])] = \{1,2,3,4\} \cap \emptyset = \emptyset$ gfp
$\eta = EF\,\delta = \delta \vee EXEF\,\delta \longrightarrow Z = \gamma \vee EX\,Z \longrightarrow \mu Z\, \delta \vee \langle - \rangle Z$ lfp
$[\eta] = [EF\,\delta] = [\mu Z\, \delta \vee \langle - \rangle Z] = \emptyset$
$[Z_0] = \emptyset$
$[Z_1] = [\delta \vee \langle - \rangle Z_0] = [\delta] \cup [pre_E(-, [Z_0])] = \emptyset \cup \emptyset = \emptyset$ lfp

$\alpha = \langle - \rangle \neg a$
$\beta = [-]\alpha$
$\gamma = a \supset \beta$
$\delta = \nu Z\, \gamma \wedge [-]Z$
$\eta = \mu Z\, \delta \vee \langle - \rangle Z$

$T \models \phi$? = Is formula true in transition system?
More precisely in state 0: $T_0 \models \phi$?
$0 \in [\phi] = [\eta]$? No $\longrightarrow$ formula not true in the transition system

AUTOMATA THEORETIC LTL MODEL CHECKING

Σ^ω = set of infinite words, infinite sequences of letters from alphabet Σ
Σ^* = set of finite words.
$L \subseteq \Sigma^*$ = finite language eg words ending with a
$L \subseteq \Sigma^\omega$ = infinite language eg words containing a finite number of a

Language recognition problem: determine whether a word w belong to language L → Use automata
We say $w \in L(D) \subseteq \Sigma^*$ if the corresponding run ρ in the automaton D ends in a final state

Nondeterministic Finite-state automaton N accepts $w \in L(N)$ if at least one run is accepting since more than one run is possible on some word.

$N = \{Q, \Sigma, I, \delta, F\}$ with:
Q = finite set of states
Σ = finite alphabet
$I \subseteq Q$ = set of initial states
$F \subseteq Q$ = set of final states
$\delta : Q \times \Sigma \to 2^Q$ = non deterministic transition function

UNION

Given two NFA $N_1 = \{Q_1, \Sigma, I_1, \delta_1, F_1\}$ and $N_2 = \{Q_2, \Sigma, I_2, \delta_2, F_2\}$, the union automaton $N_1 \cup N_2 = \{Q, \Sigma, I, \delta, F\}$ is defined as

$Q = Q_1 \cup Q_2$
$I = I_1 \cup I_2$
$F = F_1 \cup F_2$
$\delta(q, \sigma) = \begin{cases} \delta_1(q, \sigma) & q \in Q_1 \\ \delta_2(q, \sigma) & q \in Q_2 \end{cases}$

Intuitively, $N_1 \cup N_2$ chooses nondeterministically to execute either N_1 or N_2
Note: Q_1 and Q_2 must be disjoint

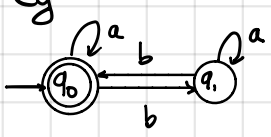
PRODUCT

In some way, $N_1 \times N_2$ is defined as

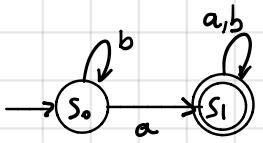
$Q = Q_1 \times Q_2$
$I = I_1 \times I_2$
$F = F_1 \times F_2$
$\delta((q_1, q_2), \sigma) = (\delta(q_1, \sigma), \delta_2(q_2, \sigma))$

$N_1 \times N_2$ executes N_1 and N_2 in parallel

Eg

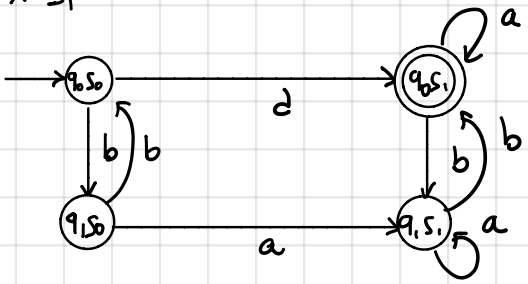


D_1 = recognizes even number of b



S_1 = at least one a

$D_1 \times S_1$



Th. For every regular expression α there exists a NFA N_α s.t. $L(\alpha) = L(N_\alpha)$
For every NFA N there exists a regular expression α_N s.t. $L(\alpha_N) = L(N)$

BÜCHI AUTOMATA

Deterministic (DBA) and Nondeterministic (NBA) Büchi automata are like DFA and NFA but they read infinite words $w \in \Sigma^\omega$
As there is no lost state in the corresponding runs ρ, the acceptance condition is to visit a final state in F infinitely many times

Th. The language $L = \{w \in \Sigma^\omega : w \text{ contains finitely many } a\text{s}\}$ can be recognized by a NBA but not by any DBA

Th. For a given NBA N, there exists a NBA $\overline{N}$ s.t. $L(\overline{N}) = \Sigma^\omega \setminus L(N)$

Union of two NBAs works exactly as for NFA.
Product $N_1 \otimes N_2$ called synchronous product is defined as:
$Q = Q_1 \times Q_2 \times \{1, 2\}$ ← bit to flag which automaton im focusing on
$I = I_1 \times I_2 \times \{1\}$
$F = F_1 \times Q_2 \times \{1\}$

$$\delta((q_1, q_2, 1), \sigma) = \begin{cases} (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma), 1) & \text{if } q_1 \notin F_1 \\ (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma), 2) & \text{if } q_1 \in F_1 \end{cases}$$

$$\delta((q_1, q_2, 2), \sigma) = \begin{cases} (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma), 2) & \text{if } q_2 \notin F_2 \\ (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma), 1) & \text{if } q_2 \in F_2 \end{cases}$$

$N_1 \otimes N_2$ switches the index every time a final state is hit

A language is called ω-regular if it is the union of expressions of the form $\alpha \cdot \beta^\omega$ with α and β being regular languages

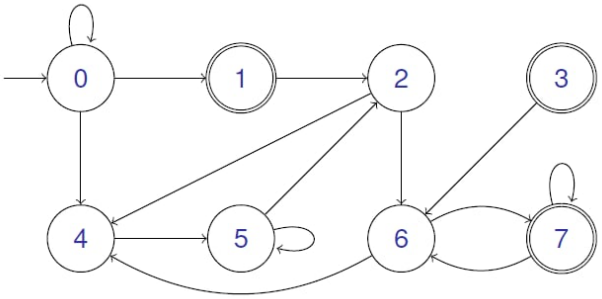
NONEMPTINESS

Given an NFA N decide whether $L(N) \overset{?}{\neq} \emptyset$ ⇒ Does a word w accepted by N exist?

↳ w is accepted by N iff exists a run whose path start from 0 and ends in a final state

⇒ Nonemptines ⇒ Reachability

REACHABILITY WITH FIX-POINT THEORY



$F = \{1, 7\} \subseteq Q$

$Z = F \vee \langle next \rangle Z$ (= reachability)

$\Downarrow$

$\mu Z.(F \vee \langle next \rangle Z) = Reach(F)$

Given a NBA instead we not only have to find a path from 0 to final state but must visit the final state infinitely many times. ⟶ Recurrent reachability.

Using fixpoint theory:

$Y = Reach(F \wedge \langle next \rangle Y)$

$\nu Y (Reach(F \wedge \langle next \rangle Y)) \longrightarrow \nu Y.(\mu Z((F \wedge \langle next \rangle Y) \vee \langle next \rangle Z))$

[A Generalized NBA (GNBA) is like a normal NBA except that $F = (F_1, F_2 \dots F_n)$
A run ρ in N is accepting iff it visits every F_i infinitely often]

Th. For an LTL formula φ we can construct a GNBA N_φ s.t $\mathcal{L}(N_\varphi) = \mathcal{L}(\varphi)$

To do this we need:

Fischer-Ladner Closure: for a given LTL formula φ, the FS-closure of φ denoted $cl(\varphi)$ is the set of subformulas of φ and their negations. Defined as follow:

- $\varphi \in cl(\varphi)$
- if $\psi \in cl(\varphi)$ then $\neg\psi \in cl(\varphi)$
- if $\psi_1 \wedge \psi_2 \in cl(\varphi)$ then $\psi_1, \psi_2 \in cl(\varphi)$
- if $X\psi \in cl(\varphi)$ then $\psi \in cl(\varphi)$
- if $\psi_1 U \psi_2 \in cl(\varphi)$ then $\psi_1, \psi_2 \in cl(\varphi)$

eg $\varphi = p \wedge ((Xp) U q)$

$cl(\varphi) = \{p \wedge ((Xp) U q), \neg(p \wedge ((Xp) U q)), p, \neg p, (Xp) U q, \neg((Xp) U q), Xp, \neg Xp, q, \neg q\}$

A set $\alpha \subseteq cl(\varphi)$ is called atom if it is maximally consistent, that is:

- $\forall \psi \in cl(\varphi)$ either $\psi \in \alpha \vee \neg\psi \in \alpha$
- $\psi_1 \wedge \psi_2 \in \alpha$ iff $\psi_1, \psi_2 \in \alpha$

eg $\varphi = p \mathcal{U} q$ $cl(\varphi) = \{p\mathcal{U}q, \neg(p\mathcal{U}q), p, \neg p, q, \neg q\}$

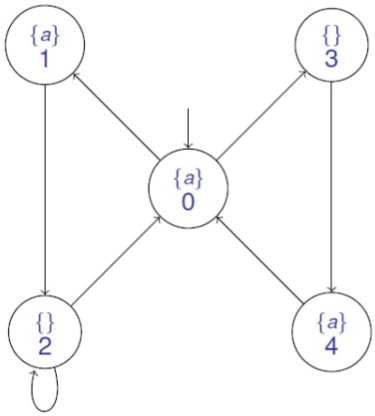
→ Atoms: $\alpha_1 = \{p\mathcal{U}q, p, q\}$ $\alpha_2 = \{p\mathcal{U}q, p, \neg q\}$ $\alpha_3 = \{p\mathcal{U}q, \neg p, q\}$ $\alpha_4 = \{p\mathcal{U}q, \neg p, \neg q\}$
$\alpha_5 = \{\neg(p\mathcal{U}q), p, q\}$ $\alpha_6 = \{\neg(p\mathcal{U}q), p, \neg q\}$ $\alpha_7 = \{\neg(p\mathcal{U}q), \neg p, q\}$ $\alpha_8 = \{\neg(p\mathcal{U}q), \neg p, \neg q\}$

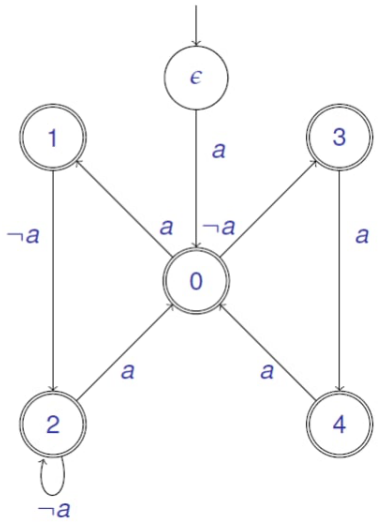
A state α (eg α_1) gives information on which subformulas of φ need to be satisfied when computation start from α itself (for α_1 $p\mathcal{U}q \wedge p \wedge q$ must be true)

Every atom containing φ is an initial state ($\alpha_1, \alpha_2, \alpha_3, \alpha_4$)
To move from atom α to α' ($\sigma \in \Sigma = 2^{Prop}$) i.e $\alpha' \in \delta(\alpha, \sigma)$ we have to check:
① $\sigma = \alpha \cap Prop$ → move only if you read something consistent
② $X\psi \in \alpha$ iff $\psi \in \alpha'$ → from state with $X\psi$ we can reach only states with ψ
③ $\psi_1 \mathcal{U} \psi_2 \in \alpha$ iff either $\psi_2 \in \alpha$ or both $\psi_1 \in \alpha$ and $\psi_1 \mathcal{U} \psi_2 \in \alpha'$
↳ in this case we can move wherever we want because we "verified" α

Final States: every subformulas $\psi_1 \mathcal{U} \psi_2$ in α holds on acceptance and thus contributes to F with the set $F_{\psi_1 \mathcal{U} \psi_2} = \{\alpha \in Q : \psi_2 \in \alpha \text{ or } \neg(\psi_1 \mathcal{U} \psi_2) \in \alpha\}$

FROM LABECED TRANSITION SYSTEM TO NBA





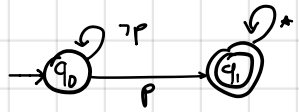
For a given LTS $\mathcal{T}$ and an LTL formula φ, Model Checking is the problem of verifying that all executions of $\mathcal{T}$ satisfy φ. ⟹ $\mathcal{T} \models \varphi \iff \mathcal{L}(\mathcal{T}) \subseteq \mathcal{L}(\varphi)$

- $\mathcal{T} \rightarrow N_{\mathcal{T}}$ $\mathcal{L}(\mathcal{T}) = \mathcal{L}(N_{\mathcal{T}})$
- $\varphi \rightarrow N_{\varphi}$ $\mathcal{L}(\varphi) = \mathcal{L}(N_{\varphi})$
- $\mathcal{L}(\mathcal{T}) \subseteq \mathcal{L}(\varphi) \iff \mathcal{L}(N_{\mathcal{T}}) \subseteq \mathcal{L}(N_{\varphi}) \iff \mathcal{L}(N_{\mathcal{T}}) \cap \mathcal{L}(\overline{N}_{\varphi}) = \emptyset$! Can we avoid the need of complement of a NBA?

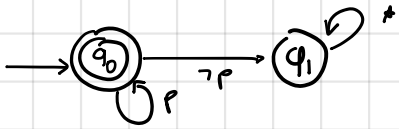
→ $\mathcal{L}(\overline{N}_{\varphi}) = \mathcal{L}(N_{\neg\varphi})$

Ex From LTL to (G)NBA

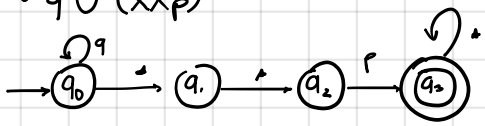
- Fp



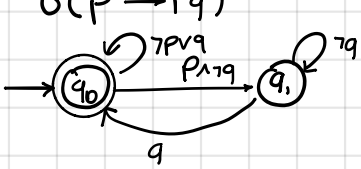
- Gp

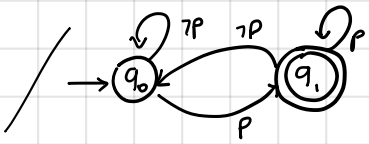


- $q \, U \, (XXp)$

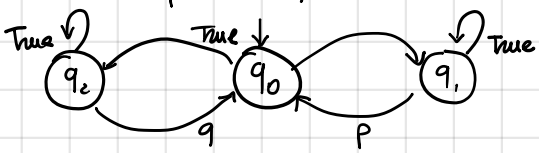


- $G(p \rightarrow Fq)$





- $GFp \wedge GFq$



$F_1 = \{q_1\}$
$F_2 = \{q_2\}$

LTL model checking algorithms takes:
- a model T
- a formula φ

and returns
- YES if $T \models \varphi$
- NO and a counter example if $T \not\models \varphi$

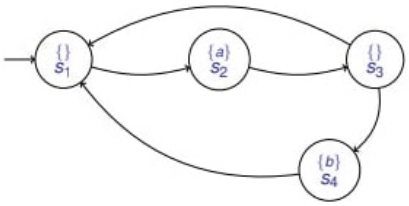
Consider a model T and an LTL property φ
$T \models \varphi$ if $\forall$ paths π of T it holds that $\pi \models \varphi$ namely if $\pi \in \mathcal{L}(\varphi)$

$\rightarrow T \models \varphi \iff \mathcal{L}(T) \subseteq \mathcal{L}(\varphi)$
$\iff \mathcal{L}(T) \cap \overline{\mathcal{L}(\varphi)} = \emptyset$
$\iff \mathcal{L}(T) \cap \mathcal{L}(\neg\varphi) = \emptyset$

1. construct N_T automaton from LTS
2. construct $N_{\neg\varphi}$ automaton from LTL formula
3. construct product automaton $N_{T,\neg\varphi} = N_T \otimes N_{\neg\varphi}$
4. Solve nonempiteness problem $\mathcal{L}(N_{T,\neg\varphi}) \overset{?}{\neq} \emptyset$

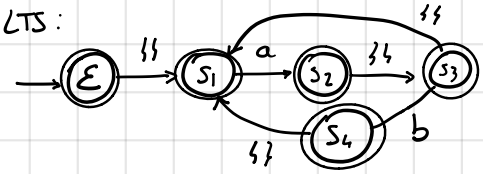
} YES if $\mathcal{L}(N_{T,\neg\varphi}) = \emptyset$, NO otherwise

Ex.



$Xa \wedge (G(b \rightarrow Xa)) \wedge Fa$

LTS:

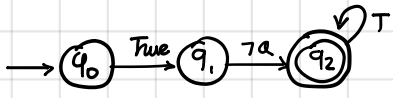


LTL: $\varphi = Xa \wedge (G(b \rightarrow Xa)) \wedge Fa$

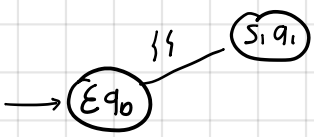
$\rightarrow \neg\varphi = \varphi_1 \vee \varphi_2 \vee \varphi_3 = \neg Xa \vee \neg G(b \rightarrow Xa) \vee \neg Fa$
$= X\neg a \vee (b \wedge \neg Xa) \vee G\neg a$

We just need one of φ_1, φ_2 and φ_3 to be "true"

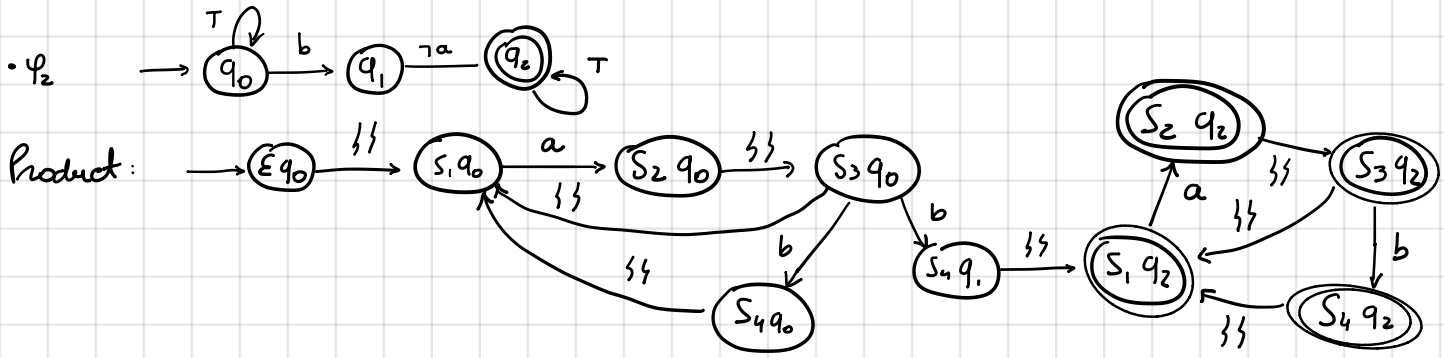
- φ_1:



Product (LTS always accepting!):



From s_1 we move only with "a"
From q_1 we move only with "$\neg a$"
$\rightarrow$ STUCK $\rightarrow$ empty



$B = \{(s_1 q_2), (s_2 q_2), (s_3 q_2), (s_4 q_2)\}$

Recurrent Reachability:

$\text{Buchi}(B) = \nu Y (\text{Reach}(B \wedge \langle next \rangle Y))$ $\quad [\text{Reach}(F) = \mu Z (F \vee \langle next \rangle Z)]$

$= \nu Y \mu Z ((B \wedge \langle next \rangle Y) \vee \langle next \rangle Z)$

Y_0 = all states

$Z_0 = \emptyset$

$Z_1 = (B \cap \langle next \rangle Y_0) \cup \langle next \rangle Z_0 = \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2\} \cup \{\text{all states} \setminus \varepsilon q_0\} = \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2\}$

$Z_2 = (B \cap \langle next \rangle Y_0) \cup \langle next \rangle Z_1 = \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2\} \cup \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2, s_4 q_1\} = \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2, s_4 q_1\}$

$Z_3 = (B \cap \langle next \rangle Y_0) \cup \langle next \rangle Z_2 = \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2\} \cup \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2, s_4 q_1, s_3 q_0\}$

$= \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2, s_4 q_1, s_3 q_0\}$

$Z_4 = (B \cap \langle next \rangle Y_0) \cup \langle next \rangle Z_3 = \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2, s_4 q_1, s_3 q_0, s_2 q_0\}$

$Z_5 = (B \cap \langle next \rangle Y_0) \cup \langle next \rangle Z_4 = \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2, s_4 q_1, s_3 q_0, s_2 q_0, s_1 q_0\}$

$Z_6 = (B \cap \langle next \rangle Y_0) \cup \langle next \rangle Z_5 = \{s_1 q_2, s_2 q_2, s_3 q_2, s_4 q_2, s_4 q_1, s_3 q_0, s_2 q_0, s_1 q_0, \varepsilon q_0, s_4 q_0\}$

$Z_7 = (B \cap \langle next \rangle Y_0) \cup \langle next \rangle Z_6 = Z_6$ l.f.p.

$Y_1 = Y_0$ g.f.p.

$\Rightarrow G(b \rightarrow Xa)$ is false in LTS $\longrightarrow$ φ not satisfied in LTS

Ex Check whether CQ q_1 is contained in CQ q_2 reporting canonical DBs and homomorphism

$q_1 \leftarrow edge(r,g), edge(g,b), edge(b,r)$

$q_2 \leftarrow edge(x,y), edge(y,z), edge(z,v), edge(v,w), edge(w,z)$

There are no free variable $\longrightarrow$ no need to freeze them!

I_{q_1} = canonical DB of q_1

e	
r	g
g	b
b	r

For second query we need to guess assignments for existencial variables

α

We start from $\alpha(x)$ and we assign to it (at random) r

$\alpha(x) = r$

$\longrightarrow edge(x,y) = edge(r,y)$

we check on canonical DB of q_1 a predicate starting with r and "guess" $\alpha(y)$: only row starting with r has a g at second place

$\alpha(y) = g$

We do the same for $edge(y,z)$. $\alpha(y) = g$, there is only [g b] $\longrightarrow g(z) = b$

$edge(z,x) \longrightarrow edg(b,r)$ that is in canonical DB ✓

$edge(z,v)$, $\alpha(z) = b \longrightarrow g(v) = r$

$edge(v,w)$, $\alpha(v) = r \longrightarrow g(w) = g$

$edge(w,z) \longrightarrow edge(g,b)$ is in canonical DB ✓

Satisfy assignment of existencial variables.

I_{q_2}

e	
x	y
y	z
z	x
z	v
v	w
w	z

Finding homomorphism means extend assignment to constants. But in our case there aren't.

So? Nothing to do

$h = \hat{\alpha} = \alpha$

$\hat{\alpha}$ = α on constants

To check homomorphism we need to check that

$h(c^{I_{q_1}}) = h(c^{I_{q_2}})$ $\forall$ constants

but since we don't have ... is true

$(x,y) \in e^{I_{q_2}} \Rightarrow h(x), h(y) \in e^{I_{q_1}}$ how to check? Simply use assignments: $h(x)\,h(y) \rightarrow [r\ g]$

[r g] is $e^{I_{q_1}}$ ✓

Do so every entry in $e^{I_{q_2}}$. They should be all true, mistakes otherwise.

Ex Check whether the following FOL formula is valid using tableaux:
$(\forall x\, P(x) \supset Q(x)) \supset (\exists x\, P(x) \supset \exists x\, Q(x))$

Tableaux can only solve satisfiability problems, not validity
→ negate the formula and check unsatisfiability

$\neg((\forall x\, P(x) \supset Q(x)) \supset (\exists x\, P(x) \supset \exists x\, Q(x))$

$\forall x\, P(x) \supset Q(x)$ ~~$\forall x\, P(x) \supset Q(x)$~~ $[\neg(\alpha \supset \beta)]$

$\neg(\exists x\, P(x) \supset \exists x\, Q(x))$ ~~$\neg(\exists x\, P(x) \supset \exists x\, Q(x))$~~

↓ $[\neg(\alpha \supset \beta)]$

~~$\exists x\, P(x)$~~

~~$\neg \exists x\, Q(x)$~~

↓

$P(\alpha)$ ×

we need to generate a clash so we need a $\neg P(\alpha)$
→ useless to expand $\neg \exists x Q(x)$!

↓

$P(\alpha) \supset Q(\alpha)$

/ \

$\neg P(\alpha)$ × $Q(\alpha)$ ×

↓

$\neg Q(\alpha)$ ×

Ex Check if valid using tableaux and if not exhibit an interpretation that is a counter example

$\neg((\forall x\, P(x) \vee Q(x)) \equiv (\forall x\, P(x) \vee \forall x\, Q(x))$ $[\neg \alpha \equiv \beta]$

/ \

~~$\forall x P(x) \vee Q(x)$~~ ~~$\forall x P(x) \vee \forall x Q(x)$~~

~~$\neg(\forall x P(x) \vee \forall x Q(x))$~~ ~~$\neg(\forall x P(x) \vee Q(x))$~~

↓ ↓

~~$\neg \forall x P(x)$~~ $\neg(P(c) \vee Q(c))$

~~$\neg \forall x Q(x)$~~ ↓

↓ $\neg P(c)$ ×

$\neg P(\alpha)$ × $\neg Q(c)$ ×

↓ / \

$\neg Q(\beta)$ $\forall x P(x)$ $\forall x Q(x)$

↓ ↓ ↓

$P(\alpha) \vee Q(\alpha)$ $P(c)$ × $Q(c)$ ×

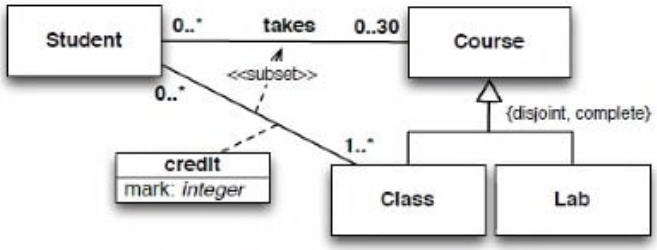
/ \

$P(\alpha)$ × $Q(\alpha)$ $\neq Q(\beta)$

no clash!

Δ^I = all constants = $\{\alpha, \beta\}$ [only of first branch]
$P^I = \{\beta\}$
$Q^I = \{\alpha\}$

Exercise 1. Express the following UML class diagram in FOL.



Exercise 2. Consider the above UML class diagram and the following (partial) instantiation.

Student

peter
paul
mary
jane

Class

calculus
AI
FM
algorithms

Lab

IoT lab
db lab
hacking lab

credit/mark

peter	algorithm	30
paul	calculus	27
mary	algorithms	28
mary	AI	30
jane	FM	30
jane	algorithms	30

takes

peter	IoT lab
paul	IoT lab
mary	FM
jane	db lab
jane	hacking lab
jane	IoT lab

1. Check whether the instantiation (once completed) is correct (and explain why it is or it is not).
2. Express in FOL and evaluate the following queries:
   (a) Return students that have taken at least 3 courses.
   (b) Return students that have taken only classes.
   (c) Check if there exists a student that has taken all labs.
   (d) Check if there is a student that has taken all classes, but not for credit.

①

$St(x)$, $Course(x)$, $Clan(x)$, $Lab(x)$, $takes(x,y)$, $credit(x,y)$, $mark(x,y,z)$, $int(x)$

takes:
- $\forall x,y\ takes(x,y) \supset St(x) \wedge Course(y)$
- $\forall x\ St(x) \supset \#\{y \mid takes(x,y)\} \leq 30$ → $1 \leq \ldots \leq 30$ to be more precise, but logic will do for us.

credit:
- $\forall x,y\ credit(x,y) \supset St(x) \wedge clan(y)$
- $\forall x\ St(x) \supset 1 \leq \#\{y \mid credit(x,y)\}$
- $\forall x,y\ credit(x,y) \supset takes(x,y)$

mark:
- $\forall x,y,z\ mark(x,y,z) \supset credit(x,y) \wedge int(z)$
- $\forall x,y\ credit(x,y) \supset \exists z\ mark(x,y,z)$
- $\forall x,y\ credit(x,y) \supset \forall z,z'\ mark(x,y,z) \wedge mark(x,y,z') \supset z = z'$

ISA:
- $\forall x\ Clan(x) \supset Course(x)$
- $\forall x\ Lab(x) \supset Course(x)$

disjoint:
- $\forall x\ Clan(x) \supset \neg Lab(x)$ [OR viceversa]

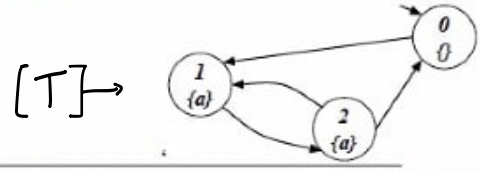
complete:
- $\forall x\ Course(x) \supset Clan(x) \vee Lab(x)$

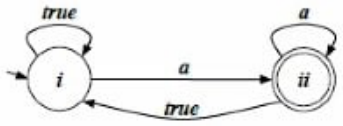
②

1. Use CHASE algorithm (apply/check ISA on instantiation 1 time more than the max lenght of ISA chain in diagram (1 in our case)
   Instantiation is correct (every clan and lab is also a course, create the table)

2.
- $\exists y_1 y_2 y_3\ St(x) \wedge takes(x,y_1) \wedge takes(x,y_2) \wedge takes(x,y_3) \wedge y_1 \neq y_2 \wedge y_1 \neq y_3 \wedge y_2 \neq y_3$
- $St(x) \wedge \forall takes(x,y) \supset clan(y)$
- $\exists x.\ St(x) \wedge \forall y\ Lab(y) \supset takes(x,y)$
- $\exists x.\ St(x) \wedge \forall y\ Clan(y) \supset (takes(x,y) \wedge \neg credit(x,y))$

Exercise 6 (optional). [1] Model check the LTL formula $\Diamond\Box\neg a$ against the following transition system, by considering that the Büchi automaton for $\neg(\Diamond\Box\neg a)$ is the one below:





←[$A_{\neg\Diamond\Box\neg a}$]

[1] *The student can get the maximum grade even without doing Exercise 6.*

$\forall$ traces of $T \supset$ traces satisfy $\Diamond\Box\neg a$

$\forall t \in \mathcal{L}(A_T) \Rightarrow t \in \mathcal{L}(A_{\Diamond\Box\neg a})$

↳ $\exists t \in \mathcal{L}(A_T) \wedge t \notin \mathcal{L}(A_{\Diamond\Box\neg a})$

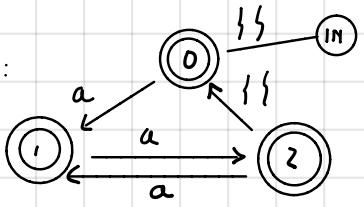
↳ find complementary $\overline{A_{\Diamond\Box\neg a}}$

We must check emptyness of $A_T \wedge \overline{A_{\Diamond\Box\neg a}}$

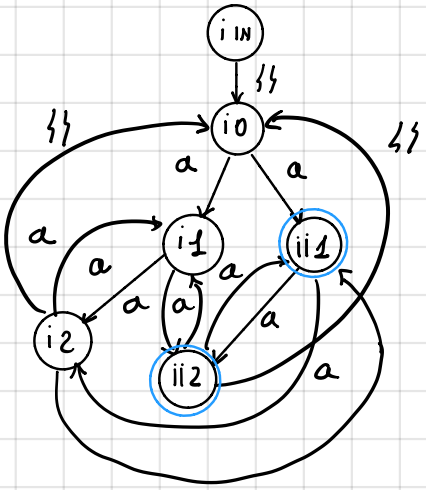
$\exists t \in \mathcal{L}(A_T) \wedge t \in \mathcal{L}(A_{\neg(\Diamond\Box\neg a)})$

check emptyness of $A_T \wedge A_{\neg(\Diamond\Box\neg a)}$

A_T :



$A_T \wedge A_{\neg\phi}$



Non emptyness:

$\nu X\ \mu Y\ (Acc \wedge \langle\!-\!\rangle X) \vee \langle\!-\!\rangle Y$

X_0 = All states

$X_1 = \mu Y(Acc \wedge \langle\!-\!\rangle X_0) \vee \langle\!-\!\rangle Y$ = All states ↳ greatest fixpoint

$[Y_{10}] = \emptyset$

$[Y_{11}] = ([Acc] \cap Pre\exists(-, [X_0])) \cup Pre\exists(-, [Y_{10}])$

$= \{ii_1, ii_2\} \cup \emptyset = \{ii_2, ii_2\}$

$[Y_{12}] = \{ii_1, ii_2\} \cup Pre\exists(-, [Y_{11}])$

$= \{ii_2, ii_2\} \cup \{i_0, i_2, i_2, ii_1, ii_2\} = \{i_0, i_1, i_2, ii_1, ii_2\}$

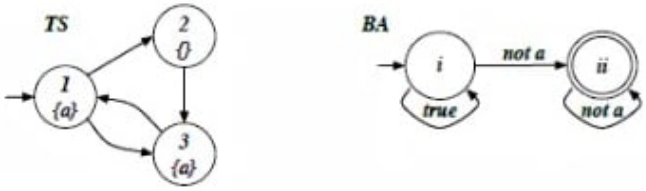
$[Y_{13}] = \{ii_2, ii_2\} \cup Pre\exists(-, [Y_{12}])$

= All states least fixpoint

Not empty! $\Rightarrow \exists t$. t is generated by $T \wedge t \models \neg(\Box\Diamond\neg a)$

$\Rightarrow T \not\models \phi$

Exercise 5. Consider the transition system TS below. Model check the LTL formula $\Box\Diamond a$, by considering that the Büchi automaton BA for $\neg\Box\Diamond a$ (i.e., $\Diamond\Box\neg a$) is the one below:



$\forall t.\ t \in T \supset t \models \phi$

$\mathcal{L}(T) \subseteq \mathcal{L}(\phi)$

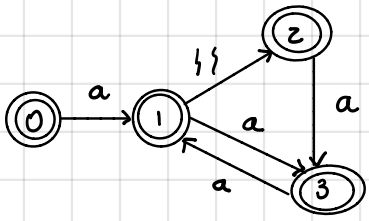
$\mathcal{L}(T) \cap \mathcal{L}(\neg\phi) = \emptyset$

$\mathcal{L}(A_T) \cap \mathcal{L}(A_{\neg\phi}) = \emptyset$

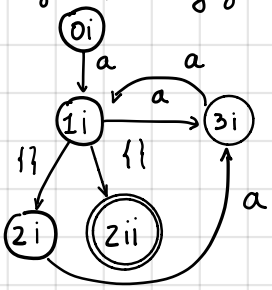
$\mathcal{L}(A_T \wedge A_{\neg\phi}) = \emptyset$

① Build A_T:
- all states are final
- add "input node" (0)
- move "actions" on edges (a and {})

$\Rightarrow$



② Build $A_T \wedge A_{\neg\phi}$ → given:
- run both automatons from "input" and "i" (0i)
- final state if final in both automatons



3) Check non-emptyness using "magic formula"

→ final state

$\nu X \mu Y (Acc \wedge \langle\!-\!\rangle X) \vee \langle\!-\!\rangle Y$

$[X_0] = S$

$[X_1] = \mu Y (Acc \wedge \langle\!-\!\rangle X) \vee \langle\!-\!\rangle Y = \emptyset$ greatest fixpoint

$[Y_{10}] = \emptyset$

$[Y_{11}] = ([Acc] \cap PreE(-, [X_0])) \cup PreE(-, [Y_{10}])$

$= (\{2ii\} \cap \{0i, 1i, 2i, 3i\}) \cup \emptyset = \emptyset$ least fixpoint

Empty! $\Rightarrow \neg\exists\, t \in T \wedge t \not\models \neg\Box\Diamond a \Rightarrow T \models \phi$

General rule: if initial state (0i) • is NOT in fixpoint → $T \models \phi$ $(\mathcal{L}(N_{T,\neg\phi}) = \emptyset)$

• is in fixpoint → $T \not\models \phi$ $(\mathcal{L}(N_{T,\neg\phi}) \neq \emptyset)$

RECAP RULES & μ-calculus CONVERSIONS

α rules

$$\frac{\phi \wedge \psi}{\phi \quad \psi} \qquad \frac{\neg(\phi \vee \psi)}{\neg\phi \quad \neg\psi} \qquad \frac{\neg(\phi \supset \psi)}{\phi \quad \neg\psi}$$

¬¬-Elimination

$$\frac{\neg\neg\phi}{\phi}$$

β rules

$$\frac{\phi \vee \psi}{\phi \mid \psi} \qquad \frac{\neg(\phi \wedge \psi)}{\neg\phi \mid \neg\psi} \qquad \frac{\phi \supset \psi}{\neg\phi \mid \psi}$$

Branch Closure

$$\frac{\phi \quad \neg\phi}{X}$$

$$\frac{\Phi \equiv \Psi}{\Phi \mid \neg\Phi \quad \Psi \mid \neg\Psi} \qquad \frac{\neg(\Phi \equiv \Psi)}{\Phi \mid \neg\Phi \quad \neg\Psi \mid \Psi}$$

$$\frac{\exists x\ \phi(x)}{\phi(c)} \qquad \frac{\neg\forall x\ \phi(x)}{\neg\phi(c)}$$

c = fresh constant = new constant not previously appearing in tableaux

$$\frac{\neg\exists x\ \phi(x)}{\neg\phi(t)} \qquad \frac{\forall x\ \phi(x)}{\phi(t)}$$

t - any term = not fresh

CTL → μ-calculus

CTL	μ-calculus
p	p
∨, ∧, →	∨, ∧, →
EXφ	⟨next⟩ t(φ)
AXφ	[next] t(φ)
EFφ	μZ t(φ) ∨ ⟨next⟩ Z
AFφ	μZ t(φ) ∨ [next] Z
EGφ	νZ t(φ) ∧ ⟨next⟩ Z
AGφ	νZ t(φ) ∧ [next] Z
φEUψ	μZ. t(ψ) ∧ (t(φ) ∧ ⟨next⟩Z)
φAUψ	μZ t(ψ) ∨ (t(φ) ∧ [next]Z)

E → has always ⟨−⟩
A → has always [−]

F → μ ... ∨ ...
G → ν ... ∧ ...

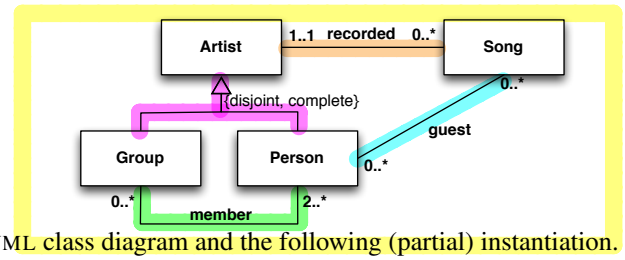
Note: A ⊃ B → ¬A ∨ B

SAPIENZA Università di Roma – MSc. in Engineering in Computer Science

Formal Methods – July 25, 2016

(Time to complete the test: 2 hours)

Exercise 1. Express the following UML class diagram in FOL.



Exercise 2. Consider the above UML class diagram and the following (partial) instantiation.

Person

John
Paul
George
Ringo
Mick
Keith

member

John	Be
Paul	Be
George	Be
Ringo	Be
Mick	RS
Keith	RS

Group

Be
RS

Song

I wanna be your man (original)
I wanna be your man (cover)

recorded

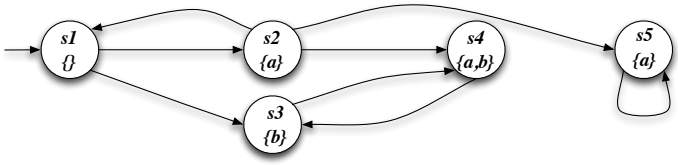
RS	I wanna be your man (original)
Be	I wanna be your man (cover)

guest

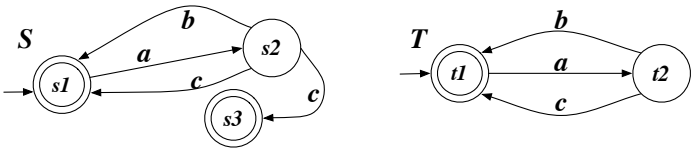
John	I wanna be your man (original)
Paul	I wanna be your man (original)

1. Check whether the above instantiation, once completed, is correct, and explain why it is or it is not.
2. Express in FOL and evaluate the following queries:
   (a) Return groups to with more that 3 members.
   (b) Return person that are guest of all songs that they, or group they are member of, did not recorded.
   (c) Check whether there are no songs recoded by groups whose members also participated as guests to the song.

Exercise 3. Model check the Mu-Calculus formula $\nu X.\mu Y.((b \wedge \langle next \rangle X) \vee \langle next \rangle Y)$ and the CTL formula $EG(AX(\neg a \vee AFb))$ (showing its translation in Mu-Calculus) against the following transition system:



Exercise 4. Consider the following two transition systems:



Write the definition of bisimilarity and compute the bisimilarity relation for the two transition systems.

Exercise 5. Check whether the following Hoare triple is correct, using as *invariant* $i \leq 10$.

```
{i=0} while(i<10) do i:= i+1 {i=10}
```

1) ALPHABET: $A(x), G(x), P(x), S(x)$, $member(x,y)$, $guest(x,y)$, $recorded(x,y)$

ISA: $\forall x \; A(x) \supset G(x) \vee P(x)$

$\forall x \; G(x) \supset \neg P(x)$

$\forall x \; G(x) \supset A(x)$

$\forall x \; P(x) \supset A(x)$

member: $\forall xy \; member(x,y) \supset G(x) \wedge P(y)$

$\forall x \; G(x) \supset 2 \leq \#\{y \mid member(x,y)\}$

guest: $\forall xy \; guest(x,y) \supset P(x) \wedge S(y)$

recorded: $\forall xy \; recorded(x,y) \supset A(x) \wedge S(y)$

$\forall y \; S(y) \supset 1 \leq \#\{x \mid recorded(x,y)\} \leq 1$

2) 1. To check if instantiation is correct, we apply chose algorithm untill we reach a fixpoint (no more modifications possible)

$I_0 = \emptyset$

$I_1 = I$

$I_2 = I_1 \cup \{group \cup person\}$ = create Artist table

$I_3 = I_2 \cup \emptyset = I_2$ fixpoint (complete instantiation)

The complete instantiation is correct because all axioms are made true.

2. $-\exists yy'y'' \; member(x,y) \wedge member(x,y') \wedge member(x,y'') \wedge \neg(y=y') \wedge \neg(y'=y'') \wedge \neg(y=y'')$

$\Rightarrow \{Be\}$

$- \quad P(x) \wedge \forall y \; guest(x,y) \supset \neg recorded(x,y) \vee (\forall z \; member(z,x) \supset recorded(z,y))$

$\Rightarrow \{Paul, John\}$

$- \quad \neg \exists xyz \; recorded(x,y) \wedge G(x) \supset \exists z \; member(x,z) \wedge guest(z,y)$

$\Rightarrow \{\emptyset\}$

4) Two transition systems are bisimilar if:

- locally they look equal
- each action that can be done on one can also be done on the other.

R_0 = cartesian product = $\{(s_1,t_1)(s_1,t_2)(s_2,t_1)(s_2,t_2)(s_3,t_1)(s_3,t_2)\}$

R_1 = remove pairs that violate local conditions on final states

$= \{(s_1,t_1)(s_2,t_2)(s_3,t_1)\}$

R_2 = remove pairs that allow action only on one of the two states

$= \{(s_1,t_1)(s_2,t_2)\}$

R_3 = remove pairs that lead to pairs no more in the list

$= \{(s_1,t_1)(s_2t_2)\}$

$(s_1 t_1)$ belongs to gfp $\longrightarrow$ S and T are bisimilar

5) $P = \{i = 0\}$
$Q = \{i = 10\}$
$\delta = \{i = i+1\}$
$G = \{i < 10\}$
$I = \{i \leq 10\}$

- Check if $P \supset I$
  $i = 0 \supset i \leq 10$ ✓
- Check $\neg G \wedge I \supset Q$
  $i \leq 10 \wedge i \leq 10 \supset i = 10$ ✓ $(i = 10)$
- Check $G \wedge I \supset Wp(\delta, I)$
  $i < 10 \wedge i \leq 10 \supset$ $i \leq 9$ ✓ $(i = 9)$
  $i = i + 1$
  $i \leq 10$

I is an invariant → the Hoare triple is correct!

3) $\nu X \mu Y((b \wedge \langle - \rangle X) \vee \langle - \rangle Y)$
$[X_0] = S$
$[X_1] = \mu Y((b \wedge \langle - \rangle X) \vee \langle - \rangle Y) = \{1,2,3,4\}$
$[Y_{10}] = \emptyset$
$[Y_{11}] = ([b] \cap Pre\exists(-, [X_0])) \cup Pre\exists(-, [Y_{10}])$
$= (\{3,4\} \cap S) \cup \{\emptyset\} = \{3,4\}$
$[Y_{12}] = ([b] \cap Pre\exists(-, [X_0])) \cup Pre\exists(-, [Y_{11}])$
$= \{3,4\} \cup \{1,2,3,4\} = \{1,2,3,4\}$
$[Y_{13}] = ([b] \cap Pre\exists(-, [X_0])) \cup Pre\exists(-, [Y_{12}])$ → least fixpoint
$= \{3,4\} \cup \{1,2,3,4\} = \{1,2,3,4\}$
$[X_2] = \mu Y((b \wedge \langle - \rangle X) \vee \langle - \rangle Y) = \{1,2,3,4\}$ greatest fixpoint
$[Y_{20}] = \emptyset$
$[Y_{21}] = ([b] \cap Pre\exists(-, [X_1])) \cup Pre\exists(-, [Y_{20}])$
$= (\{3,4\} \cap \{1,2,3,4\}) \cup \{\emptyset\} = \{3,4\}$
$[Y_{22}] = ([b] \cap Pre\exists(-, [X_1])) \cup Pre\exists(-, [Y_{21}])$
$= \{3,4\} \cup \{1,2,3,4\} = \{1,2,3,4\}$
$[Y_{23}] = ([b] \cap Pre\exists(-, [X_1])) \cup Pre\exists(-, [Y_{22}])$ → least fixpoint
$= \{3,4\} \cup \{1,2,3,4\} = \{1,2,3,4\}$

Initial state in solution (greatest fixpoint) → $\mathcal{T} \models \Phi$

$EG(AX(\underbrace{\neg a \vee \underbrace{AFb}_{\alpha}}_{\beta}))$ — γ = the whole formula

$\alpha = AFb = \mu X\ b \vee [-]X = \{3,4\}$

$[X_0] = \emptyset$

$[X_1] = [b] \cup pre_A(-, [X_0])$

$= \{3,4\} \cup \{\emptyset\} = \{3,4\}$

$[X_2] = [b] \cup pre_A(-, [X_1])$

$= \{3,4\} \cup \{3,4\} = \{3,4\}$

→ least fixpoint

$\beta = \neg a \vee \alpha = \{1,3,4\}$

$= \{1,3\} \cup \{3,4\} = \{1,3,4\}$

$\gamma = AX\beta = [-]\beta = pre_A(-, [\beta]) = \{3,4\}$

$EG\gamma = \nu X\ \gamma \wedge <->X$

$[X_0] = S$

$[X_1] = [\gamma] \cap pre_E(-, [X_0])$

$= \{3,4\} \cap S = \{3,4\}$

$[X_2] = [\gamma] \cap pre_E(-, [X_1])$

$= \{3,4\} \cap \{1,2,3,4\} = \{3,4\}$

→ greatest fixpoint

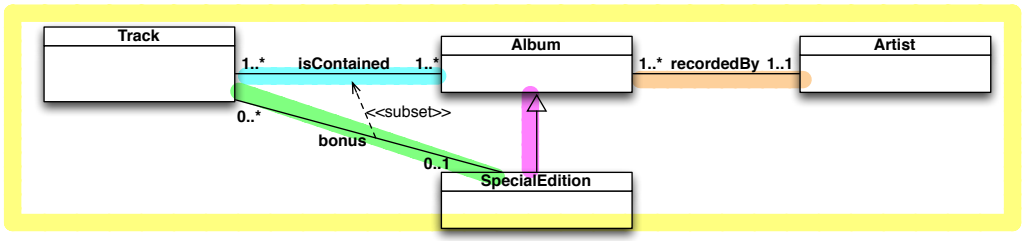
Initial states doesnt belong to solution (greatest fixpoint) $\longrightarrow T \not\models \phi$

SAPIENZA Università di Roma – MSc. in Engineering in Computer Science

Formal Methods - Final Test B – December 16, 2016

(Time to complete the test: 2 hours)

Exercise 1. Express the following UML class diagram in FOL.



Exercise 2. Consider the above UML class diagram and the following (partial) instantiation.

$Track$

t1
t2
t3
t4
t5
t6

$Album$

a1
a2
a3

$SpEd$

s1
s2

$Artist$

bt
rs

$isContained$

t1	a1
t2	a1
t3	a1
t1	a2
t4	a2
t5	a2
t5	a3

$bonus$

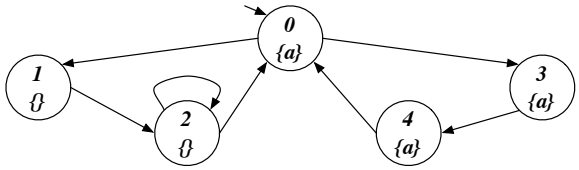
t5	s1
t6	s2

$recordedBy$

a1	bt
a2	bt
a3	rs
s1	rs
s2	bt

1. Check whether the above instantiation, once completed, is correct, and explain why it is or it is not.
2. Express in FOL the following queries and evaluate them over the completed instantiation:
   (a) Return the tracks that are contained in an album and a special edition of the same artist.
   (b) Return those artist that have recorded only albums that are not special editions.
   (c) Check if there is a track appearing in all special editions.

Exercise 3. Model check the Mu-Calculus formula $\nu X.\mu Y.((\neg a \wedge \langle next \rangle X) \vee ([next]Y))$ and the CTL formula $EF(AG(a \supset EXAX\neg a))$(showing its translation in Mu-Calculus) against the following transition system:



Exercise 4. Consider the following two transition systems:

Write the definition of bisimilarity and compute the bisimilarity relation for the two transition systems.

Exercise 5. Compute the certain answers to the following CQs over the following incomplete database (naive tables), and discuss how you obtained the result:

$$q() \leftarrow lives(x, y), incountry(y, z) \qquad q(x, z) \leftarrow lives(x, y), incountry(y, z)$$

$lives$

$person$	$city$
$null_0$	$null_1$
$null_2$	$null_3$
$null_4$	$null_5$
mary	$null_5$

$incountry$

$city$	$country$
$null_1$	IT
$null_3$	$null_6$
$null_5$	JP

1) ALPHABET: $T(x)$, $A(x)$, $Art(x)$, $S(x)$, $bonus(x,y)$, $contained(x,y)$, $recorded(x,y)$

ISA: $\forall x \; S(x) \supset A(x)$

BONUS: $\forall xy \; bonus(x,y) \supset T(x) \wedge S(y)$

$\forall x \; T(x) \supset \#\{y \mid bonus(x,y)\} \leq 1$

$\forall xy \; bonus(x,y) \supset contained(x,y)$

CONTAINED: $\forall xy \; contained(x,y) \supset T(x) \wedge A(y)$

$\forall x \; T(x) \supset 1 \leq \#\{y \mid contained(x,y)\}$

$\forall y \; A(y) \supset 1 \leq \#\{x \mid contained(x,y)\}$

RECORDED: $\forall xy \; recorded(x,y) \supset A(x) \wedge Art(y)$

$\forall x \; A(x) \supset 1 \leq \#\{y \mid recorded(x,y)\} \leq 1$

$\forall y \; Art(y) \supset 1 \leq \#\{x \mid recorded(x,y)\}$

2) Complete instantiation (merge Album with SE tables and Contained with Bonus) is correct because all axioms are made true

- $\exists yy'zz' \; bonus(x,y) \wedge contained(x,y') \wedge recorded(y,z) \wedge recorded(y',z) \wedge z = z'$
  $\Rightarrow \{t_5\}$
- $\forall x \; recored(x,y) \supset \neg S(x)$
  $\Rightarrow \{\emptyset\}$
- $\exists x \; \forall y \; S(y) \supset bonus(x,y)$
  $\Rightarrow \{\emptyset\}$

4) Two transition systems are bisimilar if:

- locally they look equal
- each action done on one of them can be done also on second one

R_0 = cartesian product $= \{(t_1,q_1)(t_1,q_2)(t_1,q_3)(t_2,q_1)(t_2,q_2)(t_2,q_3)\}$

R_1 = remove pairs that violate local condition on final states
$= \{(t_1,q_1)(t_2,q_2)(t_2,q_3)\}$

R_2 = remove pairs that can "accept" action only on one of the two states
$= \{(t_1,q_1)(t_2,q_2)\}$

R_3 = remove pairs that lead to pairs no more in the list → greatest fixpoint
$= \{(t_1,q_1)(t_2,q_2)\}$

(t_1,q_1) belongs to gfp → T and Q are bisimilar

3) $\nu X \mu Y((\neg a \wedge \langle - \rangle X) \vee ([-]Y))$

$[X_0] = S$

$[X_1] = \mu Y((\neg a \wedge \langle - \rangle X) \vee ([-]Y)) = \{1, 2\}$

$[Y_{10}] = \emptyset$

$[Y_{11}] = ([\neg a] \cap pre\exists(-, [X_0])) \cup pre\forall(-, [Y_{10}])$

$= (\{1, 2\} \cap \{5\}) \cup \emptyset = \{1, 2\}$

$[Y_{12}] = ([\neg a] \cap pre\exists(-, [X_0])) \cup pre\forall(-, [Y_{11}])$ → least fixpoint

$= \{1, 2\} \cup \{1\} = \{1, 2\}$

$[X_2] = \mu Y((\neg a \wedge \langle - \rangle X) \vee ([-]Y)) = \{1, 2\}$ → greatest fixpoint

$[Y_{20}] = \emptyset$

$[Y_{21}] = ([\neg a] \cap pre\exists(-, [X_1])) \cup pre\forall(-, [Y_{20}])$

$= (\{1, 2\} \cap \{0, 1, 2\}) \cup \{\emptyset\} = \{1, 2\}$

$[Y_{22}] = ([\neg a] \cap pre\exists(-, [X_1])) \cup pre\forall(-, [Y_{21}])$ → least fixpoint

$= \{1, 2\} \cup \{1\} = \{1, 2\}$

Initial state not in solution → $\mathcal{T} \not\models \phi$

CTL: $\mathcal{EF}(AG(a \supset \underbrace{\mathcal{EX}\underbrace{AX\ \neg a}_{\beta}}_{\alpha}))$ (δ = $AG(\cdots)$, γ = $a \supset \cdots$)

$\alpha = [-]a = pre\forall(-, [a]) = \{3, 4\}$

$\beta = \mathcal{EX}\alpha = \langle - \rangle \alpha = pre\exists(-, \{3, 4\}) = \{0, 3\}$

$\gamma = a \supset \beta = \neg a \vee \beta = \{1, 2\} \cup \{0, 3\} = \{0, 1, 2, 3\}$

$\delta = AG\,\gamma = \nu X\ \gamma \wedge [-]X = \emptyset$

$[X_0] = S$

$[X_1] = \{0, 1, 2, 3\} \cap pre\forall(-, [X_0]) = \{0, 1, 2, 3\}$

$[X_2] = \{0, 1, 2, 3\} \cap pre\forall(-, [X_1]) = \{0, 1, 2\}$

$[X_3] = \{0, 1, 2, 3\} \cap pre\forall(-, [X_2]) = \{1, 2\}$

$[X_4] = \{0, 1, 2, 3\} \cap pre\forall(-, [X_3]) = \{1\}$

$[X_5] = \{0, 1, 2, 3\} \cap pre\forall(-, [X_4]) = \emptyset$

$\mathcal{EF}\,\phi = \mu X\ \phi \vee \langle - \rangle X$

$[X_0] = \emptyset$

$[X_1] = \emptyset \cup \emptyset = \emptyset$

Initial state not in solution → $\mathcal{T} \not\models \phi$

⑤ $q() \leftarrow lives(x,y), incountry(y,z)$

- Evaluate query on DB as it was complete: True (eg $\{(null_0, null_1, IT)\}$)
- Remove null tuples: nothing to do since it is a boolean query

$q(x,z) \leftarrow lives(x,y), incountry(y,z)$

- Evaluate query on DB as it was complete: $\{(null_0, IT), (null_2, null_6), (null_4, JP), (mary, JP)\}$
- Remove null tuples: $\{(mary, JP)\}$

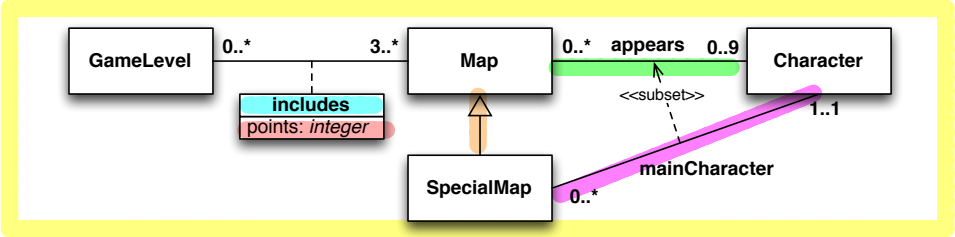
We remove tuples because the certain answer is constitued by tuples of constants.

SAPIENZA Università di Roma – MSc. in Engineering in Computer Science

Formal Methods – Jan. 25, 2017

(Time to complete the test: 2 hours)

Exercise 1. Express the following UML class diagram in FOL.



Exercise 2. Consider the above UML class diagram and the following (partial) instantiation.

Map

artica
bush

$SpecialMap$

city
desert

$Character$

adrian
bob
charline

$appaears$

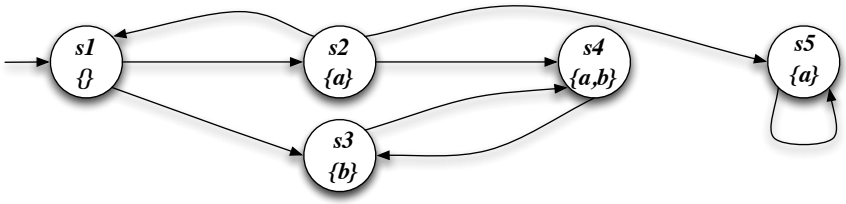
adrian	artica
adrian	bush
adrian	desert
bob	city
charline	artica

$mainCharacter$

adrian	city
charline	desert

1. Check whether the instantiation (once completed) is correct (and explain why it is or it is not).
2. Express in FOL and evaluate the following queries:
   (a) Return the maps with at least 3 distinct characters.
   (b) Return the characters that appear in maps only as main characters.
   (c) Check if there exists a map where all characters appears.

Exercise 3. Model check the Mu-Calculus formula $\nu X.\mu Y.((a \wedge \langle next \rangle X) \vee [next]Y)$ and the CTL formula $EF(\neg a \supset EXAGb)$ (showing its translation in Mu-Calculus) against the following transition system:



Exercise 4. Check whether the following Hoare triple is correct, using as *invariant* $(i + j = 9)$.

```
{i=0 AND j=9}  while(i<10) do (i:= i+1; j=j-1)  {j<0}
```

Exercise 5. Given the following conjunctive queries:

```
q1(x) :- edge(x,y), edge(y,y), edge(x,z), edge(y,z), edge(z,y).
q2(x) :- edge(x,y), edge(y,z), edge(x,v), edge(v,z), edge(v,y).
```

check whether $q1$ is contained into $q2$, explaining the technique used and, in case of containment, showing the homomorphism between the canonical databases.

1) ALPHABET: $G(X)$, $M(X)$, $SM(X)$, $C(X)$, $includes(X,Y)$ $Integer(X)$

ISA: $\forall x\ SM(x) \supset M(x)$

mainchoracter: $\forall xy\ mc(x,y) \supset C(x) \wedge SM(y)$

$\forall y\ SM(y) \supset 1 \leq \#\{x \mid mc(x)\} \leq 1$

$\forall xy\ mc(x,y) \supset opp(x,y)$

appears: $\forall xy\ opp(x,y) \supset C(x) \wedge M(y)$

$\forall y\ M(y) \supset \#\{x \mid opp(x,y)\} \leq 9$

includes: $\forall xy\ inc(x,y) \supset G(x) \wedge M(y)$

$\forall x\ G(x) \supset 3 \leq \#\{y \mid inc(x,y)\}$

$\forall xy\ inc(x,y) \supset 1 \leq \#\{z \mid points(x,y,z)\} \leq 1$

points: $\forall xyz\ points(x,y,z) \supset inc(x,y) \wedge Integer(z)$

2) The completed instantiation (merge mc with opp table) is correct because makes all axioms true.

- $\exists x x' x''\ opp(x,y) \wedge opp(x',y) \wedge opp(x'',y) \wedge x \neq x' \wedge x \neq x'' \wedge x' \neq x''$
  $\Rightarrow \{\emptyset\}$
- $\forall y\ opp(x,y) \supset mc(x,y)$
  $\Rightarrow \{\emptyset\}$
- $\exists y\ \forall x\ C(x) \supset opp(x,y)$
  $\Rightarrow \{\emptyset\}$

4) $I = \{i + j = 9\}$

$P = \{i = 0 \wedge j = 9\}$

$Q = \{j < 0\}$

$\delta = \{i = i+1 ;\ j = j-1\}$

$G = \{i < 10\}$

- Check $P \supset I$

  $i = 0 \wedge j = 9 \supset i + j = 9$ ✓
- Check $\neg G \wedge I \supset Q$

  $i \geq 10 \wedge i + j = 9 \supset j < 0$ ✓ $i = 10$ $j = -1$
- Check $G \wedge I \supset wp(\delta, I)$

  $i < 10 \wedge i + j = 9 \supset wp(\delta, I)$ $\longrightarrow$ $\{i+1 + j-1 = 9\}$

  $i = i+1$

  $\{i + j - 1 = 9\}$

  $j = j - 1$

  $i < 10 \wedge i + j = 9 \supset i + j = 9$ ✓ $\{i + j = 9\}$

I is an invariant so the Hoare triple is correct!

3) $\nu X \mu Y((a \wedge \langle -> \rangle X) \vee [-]Y)$

$[X_0] = S$

$[X_1] = \mu Y((a \wedge \langle -> \rangle X) \vee [-]Y) = S$ → greatest fixpoint

$[Y_{10}] = \emptyset$

$[Y_{11}] = ([a] \cap Pre_E(-, [X_0])) \cup Pre_A(-, [Y_{10}])$

$= (\{2,4,5\} \cap \{5\}) \cup \{\emptyset\} = \{2,4,5\}$

$[Y_{12}] = ([a] \cap Pre_E(-, [X_0])) \cup Pre_A(-, [Y_{11}])$

$= \{2,4,5\} \cap \{3,5\} = \{2,3,4,5\}$

$[Y_{13}] = ([a] \cap Pre_E(-, [X_0])) \cup Pre_A(-, [Y_{11}])$

$= \{2,4,5\} \cup \{1,2,3,4,5\} = S$ least fixpoint

Initial state in solution $\rightarrow T \models \phi$

CTL: $EF(\neg a \supset \overbrace{EX\underbrace{A G b}_{\alpha}}^{\beta})$

γ

$\alpha = \nu X\, b \wedge [-]X = \{3,4,5\}$

$[X_0] = S$

$[X_1] = [b] \cap Pre_A(-, [X_0])$

$= \{3,4,5\} \cap S = \{3,4,5\}$

$[X_1] = [b] \cap Pre_A(-, [X_1])$

$= \{3,4,5\} \cap \{2,3,4,5\} = \{3,4,5\}$ greatest fixpoint

$\beta = \langle -> \rangle \alpha = Pre_E(-, \{3,4,5\}) = S$

$\gamma = a \vee \beta = \{2,4,5\} \cup \{5\} = S$

$EF\,\gamma = \mu X (\gamma \vee \langle -> \rangle X) = \{5\}$

$[X_0] = \emptyset$

$[X_1] = S \cup Pre_E(-, [X_0]) = S$ least fixpoint

Initial state in solution $\rightarrow T \models \phi$

5)

1) Freeze free variables

$q_1(a) = e(a,y)\ e(y,y)\ e(a,z)\ e(y,z)\ e(z,y)$

$q_2(a) = e(a,y)\ e(y,z)\ e(a,v)\ e(v,z)\ e(v,y)$

2) Canonical interpretations

$$I_{q_1} = \begin{cases} \Delta^{I_{q_1}} = \{a,y,z\} \\ a^{I_{q_1}} = a \\ E^{I_{q_1}} = \{(a,y)(y,y)(a,z)(y,z)(z,y)\} \end{cases} \qquad I_{q_2} = \begin{cases} \Delta^{I_{q_2}} = \{a,y,z,v\} \\ a^{I_{q_2}} = a \\ E^{I_{q_2}} = \{(a,y)(y,z)(a,v)(v,z)(v,y)\} \end{cases}$$

3) Find homomorphism from I_{q_2} to I_{q_1} [CM77 theorem]

$h(a) = a$

$h(a,y) = (a,?) \rightarrow h(y) = y$

$h(y,z) = (y,?) \rightarrow h(z) = y$

$h(a,v) = (a,?) \rightarrow h(v) = z$

$h(v,z) = (z,y)$ ✓

$h(v,y) = (z,y)$ ✓

↓

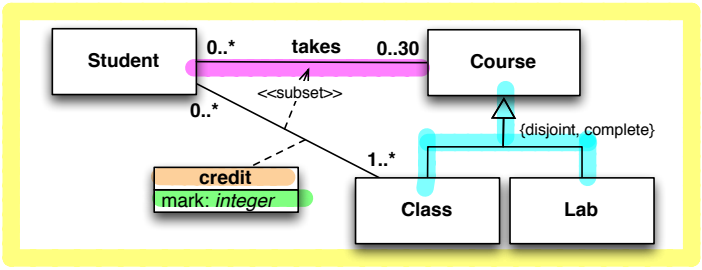
Homomorphism exists $\Rightarrow q_1 \subseteq q_2$

SAPIENZA Università di Roma – MSc. in Engineering in Computer Science

Formal Methods – Feb. 13, 2017

(Time to complete the test: 2 hours)

Exercise 1. Express the following UML class diagram in FOL.



Exercise 2. Consider the above UML class diagram and the following (partial) instantiation.

Student

peter
paul
mary
jane

Class

calculus
AI
FM
algorithms

Lab

IoT lab
db lab
hacking lab

credit/mark

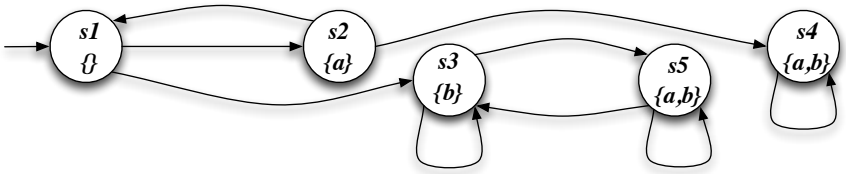
peter	algorithm	30
paul	calculus	27
mary	algorithms	28
mary	AI	30
jane	FM	30
jane	algorithms	30

takes

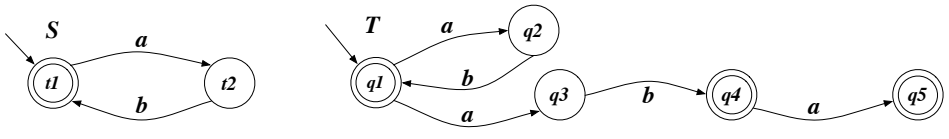
peter	IoT lab
paul	IoT lab
mary	FM
jane	db lab
jane	hacking lab
jane	IoT lab

1. Check whether the instantiation (once completed) is correct (and explain why it is or it is not).
2. Express in FOL and evaluate the following queries:
   (a) Return students that have taken at least 3 courses.
   (b) Return students that have taken only classes.
   (c) Check if there exists a student that has taken all labs.
   (d) Check if there is a student that has taken all classes, but not for credit.

Exercise 3. Model check the Mu-Calculus formula $\nu X.\mu Y.((a \wedge [next]X) \vee [next]Y)$ and the CTL formula $EF(\neg a \supset (EX\, a \wedge EX AG\, b))$ (showing its translation in Mu-Calculus) against the following transition system:



Exercise 4. Consider the following two transition systems:



Write the definition of bisimilarity and compute the bisimilarity relation for the two transition systems.

Exercise 5. Given the following conjunctive queries:

```
q1(x) :- edge(x,y), edge(y,z), edge(z,x).
q2(x) :- edge(x,y), edge(x,w), edge(y,z), edge(z,x), edge(z,v), edge(v,y), edge(v,w), edge(w,z).
```

check whether $q1$ is contained into $q2$, explaining the technique used and, in case of containment, showing the homomorphism between the canonical databases.

1) ALPHABET: $S(x)$, $C(x)$, $CL(x)$, $L(x)$, $credit(x,y)$, $mark(x,y,z)$, $Integer(x)$

ISA: $\forall x\ C(x) \supset CL(x) \vee L(x)$
$\forall x\ CL(x) \supset \neg L(x)$
$\forall x\ CL(x) \supset C(x)$
$\forall x\ L(x) \supset C(x)$

TAKES: $\forall xy\ takes(x,y) \supset S(x) \wedge C(y)$
$\forall x\ S(x) \supset \#\{y \mid takes(x,y)\} \leq 30$

CREDIT: $\forall x,y\ credit(x,y) \supset S(x) \wedge CL(y)$
$\forall x\ S(x) \supset 1 \leq \#\{y \mid credit\ y\}$
$\forall xy\ credit(x,y) \supset 1 \leq \#\{z \mid mark(x,y,z)\} \geq 1$
$\forall xy\ credit(x,y) \supset takes(x,y)$

MARK: $\forall xyz\ mark(x,y,z) \supset credit(x,y) \wedge Integer(z)$

2) Completed instantiation (add table for Course unifying class and lab and add missing line in "takes" from "credit" eg Peter - algorithm) is correct because all axioms are true.

- $\exists yy'y''\ takes(x,y) \wedge take(x,y') \wedge takes(x,y'') \wedge y \neq y' \wedge y \neq y'' \wedge y' = y''$
  $\Rightarrow \{Jone\}$
- $\forall y\ takes(x,y) \supset CL(y)$
  $\Rightarrow \{Mary\}$
- $\exists x \forall y\ L(y) \supset takes(x,y)$
  $\Rightarrow \{Jone\}$
- $\exists x \forall y\ CL(y) \supset takes(x,y) \wedge \neg credit(x,y)$
  $\Rightarrow \{\emptyset\}$

5) $q_1 \subseteq q_2$?

- Freeze free variables (x)
- Build canonical interpretation of q_1 and q_2

$$Iq_1 = \begin{cases} \Delta^{Iq_1} = \{x, y, z\} \\ x^{Iq_1} = x \\ E^{Iq_1} = \{(x,y), (y,z), (z,x)\} \end{cases}$$

$$Iq_2 = \begin{cases} \Delta^{Iq_2} = \{x, y, z, v, w\} \\ x^{Iq_2} = x \\ E^{Iq_2} = \{(x,y), (x,w), (y,z), (z,x), (z,v), (v,y), (v,w), (w,z)\} \end{cases}$$

- Check if $Iq_1 \models Iq_2$ $\longrightarrow$ find homomorphism from Iq_2 to Iq_1

- $h(x_2) = x_1$
- $h(x_2, y_2) = (x_1, ?) \rightarrow h(y_2) = y_1$
- $h(x_2, w_2) = (x_1, ?) \rightarrow h(w_2) = y_1$
- $h(y_2, z_2) = (y_1, ?) \rightarrow h(z_1) = z_1$
- $h(z_2, x_2) = (z_1, x_1)$ ✓
- $h(z_2, v_2) = (z_1, ?) \rightarrow h(v_2) = x_1$
- $h(v_2, y_2) = (x_1, y_1)$ ✓
- $h(v_2, w_2) = (x, y)$ ✓
- $h(w_2, z_2) = (y, z)$ ✓

Exists an homomorphism
$\Rightarrow q_1 \subseteq q_2$

3) $\nu X \mu Y ((a \wedge [-]X) \vee [-]Y)$

$[X_0] = S$

$[X_1] = \mu Y ((a \wedge [-]X) \vee [-]Y) = \{2,4,5\}$

$[Y_{10}] = \emptyset$

$[Y_{11}] = ([a] \cap PreA(-,[X_0])) \cup PreA(-,[Y_{10}])$

$= (\{2,5,4\} \cap \{5\}) \cup \{\emptyset\} = \{2,5,4\}$

$[Y_{12}] = ([a] \cap PreA(-,[X_0])) \cup PreA(-,[Y_{11}])$

$= \{2,5,4\} \cup \{4\} = \{2,5,4\}$ → least fixpoint

$[X_2] = \mu Y ((a \wedge [-]X) \vee [-]Y = \{4\}$

$[Y_{20}] = \emptyset$

$[Y_{21}] = ([a] \cap PreA(-,[X_1])) \cup PreA(-,[Y_{20}])$

$= (\{2,4,5\} \cap \{4\}) \cup \{\emptyset\} = \{4\}$

$[Y_{22}] = ([a] \cap PreA(-,[X_1])) \cup PreA(-,[Y_{21}])$ → least fixpoint

$= \{4\} \cup \{4\} = \{4\}$

$[X_3] = \mu Y ((a \wedge [-]X) \vee [-]Y = \{4\}$ → greatest fixpoint

$[Y_{30}] = \emptyset$

$[Y_{31}] = ([a] \cap PreA(-,[X_2])) \cup PreA(-,[Y_{30}])$

$= (\{2,4,5\} \cap \{4\}) \cup \{\emptyset\} = \{4\}$

$[Y_{32}] = ([a] \cap PreA(-,[X_2])) \cup PreA(-,[Y_{31}])$ → least fixpoint

$= (\{2,4,5\} \cap \{4\}) \cup \{4\} = \{4\}$

Initial state not in solution $\rightarrow$ $T \not\models \phi$

CTL: $EF(\underbrace{\neg a}_{\gamma} \supset (\underbrace{EX a}_{\alpha} \wedge \underbrace{EX AG b}_{\beta}))$ — γ: w, δ: β

$\alpha = \nu X\, b \wedge [-]X = \{3,4,5\}$

$[X_0] = S$

$[X_1] = [b] \cap PreA(-,[X_0])$

$= \{3,4,5\} \cap S = \{3,4,5\}$

$[X_2] = [b] \cap PreA(-,[X_1])$ → greatest fixpoint

$= \{3,4,5\} \cap \{3,4,5\} = \{3,4,5\}$

$\beta = <->\alpha = PreE(-,[\alpha]) = S$

$\gamma = <->a = PreE(-,[a]) = S$

$\delta = S \cap S = S$

$w = a \vee S = S$

$EF(w) = \mu X\, w \vee <->X = S$

$[X_0] = \emptyset$

$[X_1] = [w] \cup PreE(-,[X_0])$

$= S \cup \emptyset = S$ → least fixpoint

Initial state in solution $\rightarrow$ $T \models \phi$

4) Two transition system are bisimilar if:
- locally they look equal
- each action done on one of them can be done also on the second one

R_0 = cartesian product = $\{(t_1,q_1),(t_1,q_2),(t_1,q_3),(t_1,q_4),(t_1,q_5),(t_2,q_1),(t_2,q_2),(t_2,q_3)(t_2,q_4)(t_2,q_5)\}$

R_1 = remove pairs that violate local condition on final state

$= \{(t_1,q_1)(t_1,q_4)(t_1,q_5)(t_2,q_1)(t_2,q_2)(t_2,q_3)(t_2,q_4)(t_2,q_5)\}$

R_2 = remove actions that can be done only on one of the two states

$= \{(t_1,q_1)(t_1,q_4)(t_2,q_2)(t_2,q_3)\}$

R_3 = remove pairs that lead to pairs no more in the list

$= \{(t_1,q_1)(t_2,q_2)\}$ → greatest fixpoint

R_4 = same $= \{(t_1,q_1)(t_2,q_2)\}$

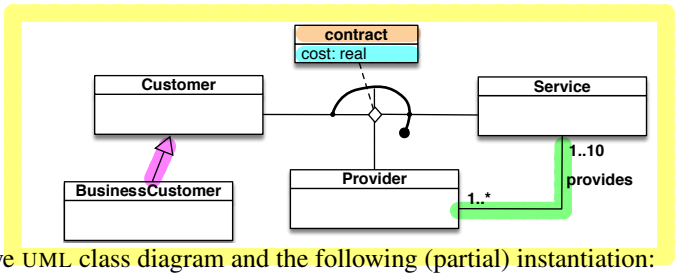
(t_1,q_1) belong to greatest fixpoint ⟶ S and T are bisimilar

SAPIENZA Università di Roma – MSc. in Engineering in Computer Science

Formal Methods - Final Test B – December 21, 2017

(Time to complete the test: 2 hours)

Exercise 1. Express the following UML class diagram in FOL:



Exercise 2. Consider the above UML class diagram and the following (partial) instantiation:

$Customer$

c1
c2
c3
c4

$BCustumers$

b1
b2
b3

$Services$

s1
s2
s3

$Provider$

p1
p2

$provides$

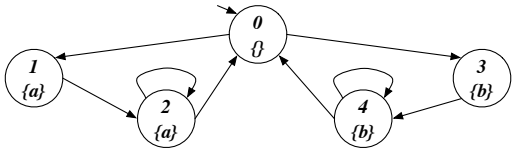
p1	s1
p1	s2
p1	s3
p2	s2

$contacts/cost$

c1	s1	p1	90.0
c1	s2	p1	80.0
c1	s3	p1	50.0
b2	s1	p2	170,0
b2	s2	p2	100,0

1. Check whether the above instantiation, once completed, is correct, and explain why it is or it is not.
2. Express in FOL the following queries and evaluate them over the completed instantiation:
   (a) Check that, for every provider x and service y involved in a contract, provider x does provide service y.
   (b) Return those customers that have contracts only for services provided by $p2$.
   (c) Return those customers that have a contract for with all providers.

Exercise 3. Model check the Mu-Calculus formula $\nu X.\mu Y.((b \wedge [next]X) \vee (a \wedge \langle next \rangle Y))$ and the CTL formula $EF(AG(a \supset EXAX\neg a))$(showing its translation in Mu-Calculus) against the following transition system:



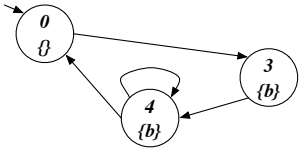
Exercise 4. Check whether the Hoare triple below is correct, by using $(x \geq 0 \wedge y \geq 0 \wedge x + y = 31)$ as invariant:

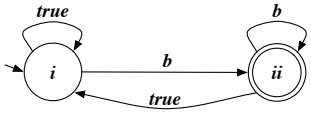
$$\{x = 31 \wedge y = 0\} \texttt{ while(x>0) do (x=x-1; y:= y+1) } \{y = 31\}$$

Exercise 5. Check whether the following FOL formula is valid, by using tableaux:

$$(\exists x.P(x) \vee \exists x.Q(x)) \equiv \exists x.(P(x) \vee Q(x))$$

Exercise 6 (optional). Model check the LTL formula $\Diamond\Box\neg b$ against the following transition system, by considering that the Büchi automaton for $\neg(\Diamond\Box\neg b)$ is the one below:





1) ALPHABET: $c(x)$, $B(x)$, $P(x)$, $S(x)$, $contr(x,y,z)$, $cost(x,y,z,w)$, $Real(x)$

ISA: $\forall x\ B(x) \supset c(x)$

PROVIDES: $\forall xy\ prov(x,y) \supset P(x) \wedge S(y)$

$\forall x\ P(x) \supset 1 \leq \#\{y \mid prov(x,y)\} \geq 10$

$\forall y\ S(y) \supset 1 \leq \#\{x \mid prov(x,y)\}$

CONTRACT: $\forall xyz\ contr(x,y,z) \supset c(x) \wedge S(y) \wedge P(z)$

$\forall x y z z'\ contr(x,y,z) \wedge contr(x,y,z') \supset z = z'$

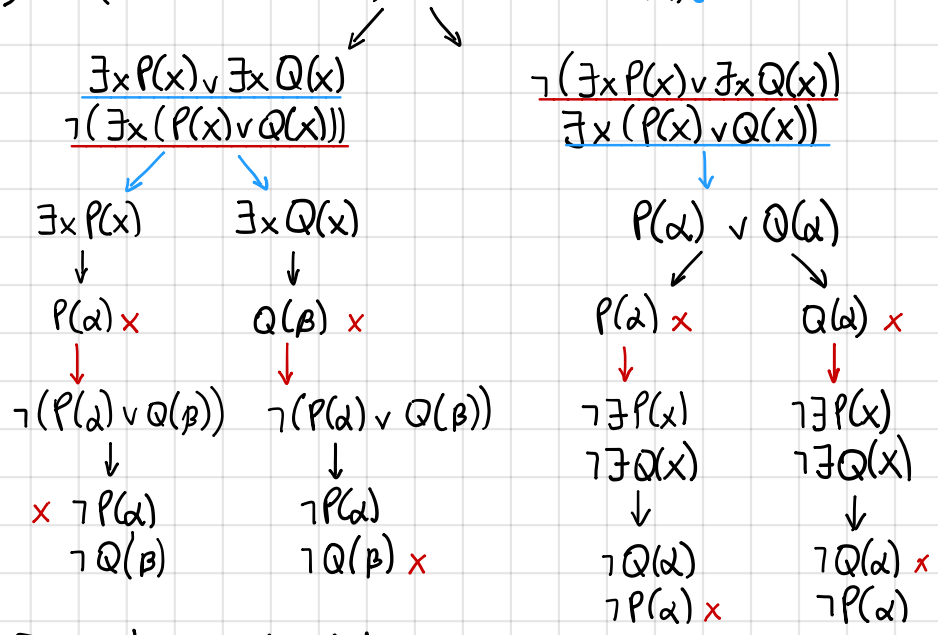
$\forall x\ y\ z\ contr(x,y,z) \supset 1 \leq \#\{w \mid cost(x,y,z,w)\} \geq 1$

COST: $\forall xyzw\ cost(x,y,z,w) \supset contr(x,y,z) \wedge Real(w)$

2) Completed instantiation is correct because all axioms are true

- $\forall xy\ contr(z,y,x) \supset prov(x,y)$
  $\Rightarrow$ false [$contr(b_2, s_1, p_2)$ but not $prov(p_2, s_1)$]
- $\exists yz\ contr(x,y,z) \supset prov(p_2, y)$
  $\Rightarrow \{\emptyset\}$
- $\exists z \forall y\ S(y) \supset contr(x,y,z)$
  $\Rightarrow \{\emptyset\}$

5) $\neg[(\exists x P(x) \vee \exists x Q(x)) \equiv \exists x (P(x) \vee Q(x))]$



$\neg\Gamma$ unsat $\Rightarrow$ Γ valid

3) $\nu X\, \mu Y ((b \wedge [-]X) \vee (a \wedge \langle - \rangle Y))$

$[X_0] = S$

$[X_1] = \mu Y ((b \wedge [-]X) \vee (a \wedge \langle - \rangle Y)) = \{3,4\}$

$[Y_{10}] = \emptyset$

$[Y_{11}] = ([b] \cap pre_A(-, [X_0])) \cup ([a] \cap pre_E(-, [Y_{10}]))$

$= (\{3,4\} \cap \{5\}) \cup (\{1,2\} \cap \emptyset) = \{3,4\}$

$[Y_{12}] = ([b] \cap pre_A(-, [X_0])) \cup ([a] \cap pre_E(-, [Y_{11}]))$ → least fixpoint

$= \{3,4\} \cup (\{1,2\} \cap \{0,3,4\}) = \{3,4\}$

$[X_2] = \mu Y ((b \wedge [-]X) \vee (a \wedge \langle - \rangle Y)) = \{3\}$

$[Y_{20}] = \emptyset$

$[Y_{21}] = ([b] \cap pre_A(-, [X_1])) \cup ([a] \cap pre_E(-, [Y_{20}]))$

$= (\{3,4\} \cap \{3\}) \cup (\{1,2\} \cap \emptyset) = \{3\}$

$[Y_{22}] = ([b] \cap pre_A(-, [X_1])) \cup ([a] \cap pre_E(-, [Y_{21}]))$ → least fixpoint

$= \{3\} \cup (\{1,2\} \cap \{0\}) = \{3\}$

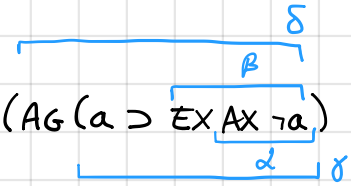
$[X_3] = \mu Y ((b \wedge [-]X) \vee (a \wedge \langle - \rangle Y)) = \emptyset$ greatest fixpoint

$[Y_{30}] = \emptyset$

$[Y_{31}] = ([b] \cap pre_A(-, [X_2])) \cup ([a] \cap pre_E(-, [Y_{30}]))$

$= (\{3,4\} \cap \emptyset) \cup (\{1,2\} \cap \emptyset) = \emptyset$ → least fixpoint

Initial state not in solution → $\mathcal{T} \not\models \Phi$



$\alpha = [-]\neg a = \{3,4\}$

$\beta = \langle - \rangle \alpha = \{0,3,4\}$

$\gamma = \neg a \vee \beta = \{0,3,4\} \cup \{0,3,4\} = \{0,3,4\}$

$\delta = \nu X\ \delta \wedge [-]X = \emptyset$

$[X_0] = \{5\}$

$[X_1] = [\delta] \cap pre_A(-, [X_0])$

$= \{0,3,4\} \cap \{5\} = \{0,3,4\}$

$[X_2] = [\delta] \cap pre_A(-, [X_1])$

$= \{0,3,4\} \cap \{3,4\} = \{3,4\}$

$[X_3] = [\delta] \cap pre_A(-, [X_2])$

$= \{3,4\} \cap \{3\} = \{3\}$

$[X_4] = [\delta] \cap pre_A(-, [X_3])$

$= \{3,4\} \cap \{\emptyset\} = \emptyset$ greatest fixpoint

$EF(\delta) = \mu X\ \delta \vee \langle - \rangle X = \emptyset$

$[X_0] = \emptyset$

$[X_1] = [\delta] \cup pre_E(-, [X_0])$

$= \emptyset \cup \emptyset$ least fixpoint

Initial state not in solution → $\mathcal{T} \not\models \phi$

4) $I = \{x \geq 0 \wedge y \geq 0 \wedge x+y = 31\}$
$P = \{x = 31 \wedge y = 0\}$
$G = \{x > 0\}$
$\delta = \{x = x-1 \; ; \; y = y+1\}$
$Q = \{y = 31\}$

- Check $P \supset I$

$x = 31 \wedge y = 0 \supset x \geq 0 \wedge y \geq 0 \wedge x+y = 31$ ✓

- Check $\neg G \wedge I \supset Q$

$x \leq 0 \wedge x \geq 0 \wedge y \geq 0 \wedge x+y = 31 \supset y = 31$ ✓

($x \leq 0 \wedge x \geq 0$: $x = 0$)

- Check $\{G \wedge I\}\, \delta\, \{I\} = G \wedge I \supset wp(\delta, I)$

$x > 0 \wedge x \geq 0 \wedge y \geq 0 \wedge x+y = 31 \supset wp(\delta, I)$ ⇝

$x > 0 \wedge x \geq 0 \wedge y \geq 0 \wedge x+y = 31 \supset x \geq 1 \wedge y \geq -1 \wedge x+y = 31$ ✓

$\{x \geq 1 \wedge y \geq -1 \wedge x+y = 31\}$
$x = x-1$
$\{x \geq 0 \wedge y \geq -1 \wedge x+y = 30\}$
$y = y+1$
$\{x \geq 0 \wedge y \geq 0 \wedge x+y = 31\}$

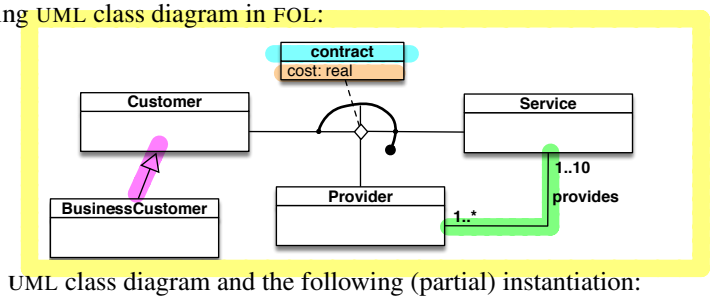
I is an invariant → the hoare triple is correct!

SAPIENZA Università di Roma – MSc. in Engineering in Computer Science

Formal Methods – January 18, 2018

(Time to complete the test: 2 hours)

Exercise 1. Express the following UML class diagram in FOL:



Exercise 2. Consider the above UML class diagram and the following (partial) instantiation:

$Customer$

c1
c2
c3
c4

$BCustumers$

b1
b2
b3

$Services$

s1
s2
s3

$Provider$

p1
p2

$provides$

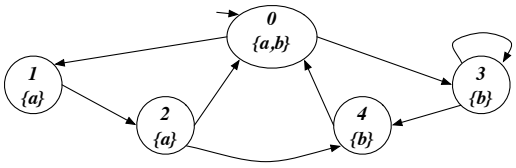
p1	s1
p1	s2
p1	s3
p2	s2

$contacts/cost$

c1	s1	p1	90.0
c1	s2	p1	80.0
c1	s3	p1	50.0
b2	s1	p2	170,0
b2	s2	p2	100,0

1. Check whether the above instantiation, once completed, is correct, and explain why it is or it is not.
2. Express in FOL the following queries and evaluate them over the completed instantiation:
   (a) Check whether there is a customer with contract with two providers for the same service.
   (b) Return those customers that have contracts only for one service.
   (c) Return those customers that have a contracts with the same provider for all their services.

Exercise 3. Model check the Mu-Calculus formula $\nu X.\mu Y.((a \wedge [next]X) \vee (b \wedge [next]Y))$ and the CTL formula $AF(EG(a \supset EXAXb))$ (showing its translation in Mu-Calculus) against the following transition system:



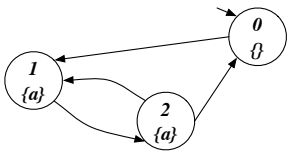
Exercise 4. Check whether CQ q_1 is contained in CQ q_2, reporting canonical DBs and homomorphism:

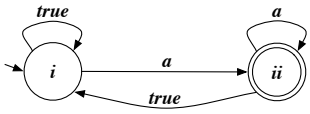
$$\begin{array}{lcl} q_1(x_r) & \leftarrow & e(x_r, x_g), e(x_g, x_b), e(x_b, x_r). \\ q_2(x) & \leftarrow & e(x, y), e(y, z), e(z, x), e(z, v)e(v, w), e(w, z). \end{array}$$

Exercise 5. Check whether the following FOL formula is valid, by using tableaux:

$$(\forall x.P(x) \supset Q(x)) \supset (\exists x.P(x) \supset \exists x.Q(x))$$

Exercise 6 (optional). [1] Model check the LTL formula $\Diamond\Box\neg a$ against the following transition system, by considering that the Büchi automaton for $\neg(\Diamond\Box\neg a)$ is the one below:





[1] *The student can get the maximum grade even without doing Exercise 6.*

1) Alphabet: $C(x)$, $B(x)$, $P(x)$, $S(x)$, $real(x)$
$cont_r(x,y,z)$
$cost(x,y,z,w)$

ISA: $\forall x\, B(x) \supset C(x)$

PROVIDES: $\forall\, x,y$ provides $(x,y) \supset P(x) \wedge S(y)$
$\forall\, x\ P(x) \supset (1 \leq \#\{y \mid \text{provides}(x,y)\} \leq 10$
$\forall\, y\ S(x) \supset (1 \leq \#\{x \mid \text{provides}(x,y)\})$

CONTRACT: $\forall\, x,y,z\ \ cont_r(x,y,z) \supset C(x) \wedge S(y) \wedge P(z)$
$\forall\, x,y,z,z'\ cont_r(x,y,z) \wedge cont_r(x,y,z') \supset z = z'$
$\forall\, x,y,z\ cont_r(x,y,z) \supset (1 \leq \#\{w \mid cost(x,y,z,w)\} \leq 1)$

COST: $\forall\, x,y,z,w\ \ cost(x,y,z,w) \supset cont(x,y,z) \wedge real(w)$

2) Interpretation is correct because satisfies all axioms in Γ

- $\exists\, y,z,z'\ cont_r(x,y,z) \wedge cont_r(x,y,z') \wedge z \neq z'$
No need to check $C(x)$. $S(y)$. $P(z)$ done because they are implied in $cont(x,y,z)$
$\Rightarrow \{\emptyset\}$
- $\exists\ y,z\ \ cont_r(x,y,z) \wedge [\forall y y'\ \exists z\ cont_r(x,y,z) \wedge \exists z'\ cont_r(x,y',z') \supset y = y']$
$\Rightarrow \{\emptyset\}$
- $\exists\, p\ \forall y\ \exists z\ cont_r(x,y,z) \supset cont_r(x,y,p)$
$\Rightarrow \{c_1, b_2\}$

5) $\neg((\forall x\, P(x) \supset Q(x)) \supset (\exists x\, P(x) \supset \exists x\, Q(x)))$

$\forall x\, P(x) \supset Q(x)$ $\qquad [\neg(\alpha \supset \beta)]$

$\neg(\exists x\, P(x) \supset \exists x\, Q(x))$

$\exists x\, P(x)$ $\qquad [\neg(\alpha \supset \beta)]$

$\neg \exists x\, Q(x)$

$P(\alpha)$ ×

we need to generate a clash so we need a $\neg P(\alpha)$
→ useless to expand $\neg \exists x Q(x)$!

$P(\alpha) \supset Q(\alpha)$

$\neg P(\alpha)$ × $\qquad$ $Q(\alpha)$ ×

$\neg Q(\alpha)$ ×

$\neg\Gamma$ is unsat. → Γ is valid!

3) $\nu X \mu Y ((a \wedge [-]X) \vee (b \wedge [-]Y))$

$[X_0] = S$

$[X_1] = \mu Y((a \wedge [-]X) \vee (b \wedge [-]Y)) = \{s_0, s_1, s_2, s_4\}$

$[Y_{10}] = \emptyset$

$[Y_{11}] = (a \wedge [-]X) \vee (b \wedge [-]Y)$

$= ([a] \cap pre_A(-, [X_0])) \cup ([b] \cap pre_A(-, [Y_{10}]))$

$= (\{s_0, s_1, s_2\} \cap \{s_0, s_1, s_2, s_3, s_4\}) \cup (\{s_0, s_3, s_4\} \cap \{\emptyset\}) = \{s_0, s_1, s_2\}$

$[Y_{12}] = (([a] \cap pre_A(-, [X_0])) \cup ([b] \cap pre_A(-, [Y_{11}])))$

$= \{s_0, s_1, s_2\} \cup \{s_4\} = \{s_0, s_1, s_2, s_4\}$

$[Y_{13}] = (([a] \cap pre_A(-, [X_0])) \cup ([b] \cap pre_A(-, [Y_{12}])))$ → least fixpoint

$= \{s_0, s_1, s_2\} \cup \{s_4\} = \{s_0, s_1, s_2, s_4\}$

$[X_2] = \mu Y((a \wedge [-]X) \vee (b \wedge [-]Y)) = \{s_1, s_2\}$

$[Y_{20}] = \emptyset$

$[Y_{21}] = (a \wedge [-]X_1) \vee (b \wedge [-]Y_{20})$

$= ([a] \cap pre_A(-, [X_1])) \cup ([b] \cap pre_A(-, [Y_{20}]))$

$= (\{s_0, s_1, s_2\} \cap \{s_1, s_2, s_4\}) \cup (\{s_0, s_3, s_4\} \cap \{\emptyset\}) = \{s_1, s_2\}$

$[Y_{22}] = ([a] \cap pre_A(-, [X_1])) \cup ([b] \cap pre_A(-, [Y_{21}]))$ → least fixpoint!

$= \{s_1, s_2\} \cup \{s_1\} = \{s_1, s_2\}$

$[X_3] = \mu Y((a \wedge [-]X) \vee (b \wedge [-]Y)) = \{s_1\}$

$[Y_{30}] = \emptyset$

$[Y_{31}] = (a \wedge [-]X_2) \vee (b \wedge [-]Y_{30})$

$= ([a] \cap pre_A(-, [X_2])) \cup ([b] \cap pre_A(-, [Y_{30}]))$

$= (\{s_0, s_1, s_2\} \cap \{s_1\}) \cup (\{s_0, s_3, s_4\} \cap \{\emptyset\}) = \{s_1\}$

$[Y_{32}] = ([a] \cap pre_A(-, [X_2])) \cup ([b] \cap pre_A(-, [Y_{31}]))$ — least fixpoint

$= \{s_1\} \cup \{\emptyset\} = \{s_1\}$

$[X_4] = \mu Y((a \wedge [-]X) \vee (b \wedge [-]Y)) = \{\emptyset\}$ greatest fixpoint

$[Y_{40}] = \emptyset$

$[Y_{41}] = (a \wedge [-]X_3) \vee (b \wedge [-]Y_{40})$

$= ([a] \cap pre_A(-, [X_3]) \cup ([b] \cap pre_A(-, [Y_{40}]))$

$= (\{s_0, s_1, s_2\} \cap \{\emptyset\}) \cup (\{s_0, s_2, s_4\} \cap \{\emptyset\}) = \{\emptyset\}$ least fixpoint

Initial stage not in solution $\Rightarrow \tau \neq \phi$

CTL: $AF(EG(a \supset EX\underbrace{AXb}_{\alpha}))$ (σ = $EG(\gamma)$, γ = $a \supset \beta$, β = $EX\alpha$, α = AXb)

$\alpha = AXb = [-]b = Pre A(-,[b]) = \{s_2, s_3, s_4\}$

$\beta = EX\alpha = \langle - \rangle \alpha = Pre E(-,[\alpha]) = \{s_0, s_1, s_2, s_3\}$

$\gamma = a \supset \beta = \neg a \vee \beta = \{s_3, s_4\} \cup \{s_0, s_1, s_2, s_3\} = S$

$\sigma = EG\gamma = \nu X\, \gamma \wedge \langle - \rangle X$

$[X_0] = S$

$[X_1] = [\gamma] \cap Pre E(-,[X_0]) = S \cap S = \{S\}$ fixpoint

$AF\sigma = \mu X\, \sigma \vee [-]X$

$[X_0] = \emptyset$

$[X_1] = [\sigma] \cup Pre A(-,[X_0]) = \{S\}$ fixpoint

Initial state in solution $\Rightarrow \mathcal{T} \models \phi$

4) $q_1 \subseteq q_2$?

- Freeze free variables (x_r and x)
- Build canonical interpretation of q_1 and q_2

$$I_{q_1} = \begin{cases} \Delta^{I_{q_1}} = \{x_r, x_g, x_b\} \\ x_r^{I_{q_1}} = x_r \\ E^{I_{q_1}} = \{(x_r, x_g), (x_g, x_b), (x_b, x_r)\} \end{cases}$$

$$I_{q_2} = \begin{cases} \Delta^{I_{q_2}} = \{x, y, z, v, w\} \\ x^{I_{q_2}} = x \\ E^{I_{q_2}} = \{(x,y), (y,z), (z,x), (z,v), (v,w), (w,z)\} \end{cases}$$

- Check if $I_{q_1} \models q_2$ $\longrightarrow$ find homomorphism from I_{q_2} to I_{q_1}

On order, check all $e \in E^{I_{q_2}}$

- $h(x) = x_r$ because constants
- $h(x,y) = (x_r, ?)$ $\longrightarrow$ find on e starting with x_r ✓ $\longrightarrow (x_r, x_g)$ $\longrightarrow h(y) = x_g$
- $h(y,z) = (x_g, ?)$ $\longrightarrow$ again ✓ $\longrightarrow (x_g, x_b)$ $\longrightarrow h(z) = x_b$
- $h(z,x) = (x_b, x_r)$ $\longrightarrow$ is present in $E^{I_{q_1}}$ ✓
- $h(z,v) = (x_b, ?)$ $\longrightarrow$ find e starting with x_b ✓ $\longrightarrow (x_b, x_r)$ $\longrightarrow h(v) = x_r$
- $h(v,w) = (x_r, ?)$ $\longrightarrow$ ... ✓ $\longrightarrow (x_r, x_g)$ $\longrightarrow h(w) = x_g$
- $h(w,z) = (x_g, x_b)$ $\longrightarrow$ is present in $E^{I_{q_1}}$ ✓

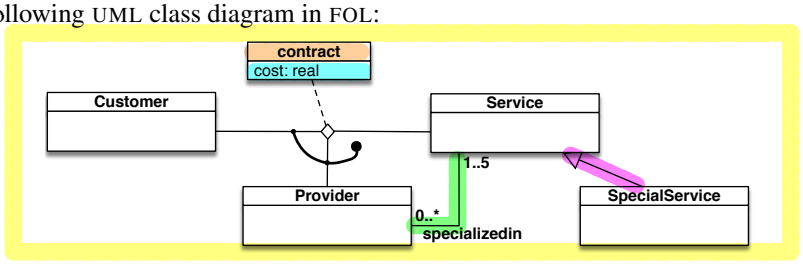
Exists an homomorphism $\Rightarrow q_1 \subseteq q_2$

SAPIENZA Università di Roma – MSc. in Engineering in Computer Science

Formal Methods – June 17, 2019

(Time to complete the test: 2 hours)

Exercise 1. Express the following UML class diagram in FOL:



Exercise 2. Consider the above UML class diagram and the following (partial) instantiation:

Customer

c1
c2
c3
c4

Service

s1
s2
s3

SpecialService

ss1
ss2

Provider

p1
p2

specializedin

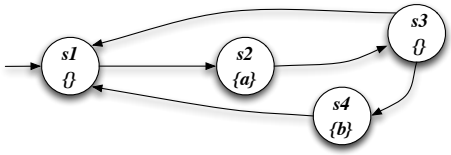
p1	s1
p1	s2
p1	s3
p2	ss1
p2	ss2

contacts/cost

c1	p1	s1	90.0
c1	p2	s2	80.0
c2	p1	s1	50.0
c3	p2	ss1	170,0
c2	p2	ss2	100,0

1. Check whether the above instantiation, once completed, is correct, and explain why it is or it is not.
2. Express in FOL the following queries and evaluate them over the completed instantiation:
   (a) Return those providers that have contracts with at least two customers.
   (b) Return those providers that have contracts only services they are specialized in.
   (c) Return those providers that have contracts all services they are specialized in.
   (d) Check whether there exists a customer with contracts for all services.

Exercise 3. Model check the Mu-Calculus formula $\nu X.\mu Y.((a \wedge \langle next \rangle X) \vee (\neg b \wedge \langle next \rangle Y)$ and the CTL formula $AG(AFa \wedge EFb \wedge EG\neg b)$ (showing its translation in Mu-Calculus) against the following transition system:



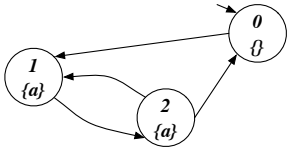
Exercise 4. Check whether CQ q_1 is contained in CQ q_2, reporting canonical DBs and homomorphism:

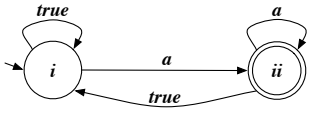
$$q_1() \leftarrow edge(r,g), edge(g,b), edge(b,r).$$
$$q_2() \leftarrow edge(x,y), edge(y,z), edge(z,x), edge(z,v), edge(v,w), edge(w,z).$$

Exercise 5. Check whether the following FOL formula is valid, by using tableaux:

$$(\forall x.\forall y.P(x,y) \supset Q(x)) \equiv (\forall x.(\exists y.P(x,y)) \supset Q(x))$$

Exercise 6 (optional). [1] Model check the LTL formula $\Diamond\Box\neg a$ against the following transition system, by considering that the Büchi automaton for $\neg(\Diamond\Box\neg a)$ is the one below:





[1] *The student can get the maximum grade even without doing Exercise 6.*

1) ALPHABET: $c(x)$, $s(x)$, $p(x)$, $ss(x)$, $contr(x,y,z)$, $cost(x,y,z,w)$, $Real(x)$

ISA: $\forall x\ ss(x) \supset s(x)$

SPECIALIZED IN: $\forall x,y\ spec(x,y) \supset p(x) \wedge s(x)$

$\forall x\ p(x) \supset 1 \leq \#\{y \mid spec(x,y)\} \leq 5$

CONTRACT: $\forall x,y,z\ contr(x,y,z) \supset c(x) \wedge p(y) \wedge s(z)$

$\forall x,y,z,z'\ contr(x,y,z) \wedge contr(x,y,z') \supset z = z'$

$\forall x,y,z\ contr(x,y,z) \supset 1 \leq \#\{w \mid cost(x,y,z,w)\} \leq 1$

COST: $\forall x,y,z,w\ cost(x,y,z,w) \supset contr(x,y,z) \wedge Real(w)$

2) The completed instantiation is correct because all axioms are made true.

- $\exists x,x',z\ contr(x,y,z) \wedge contr(x',y,z) \wedge \neg(x = x')$
  $\Rightarrow \{p_1, p_2\}$
- $\forall x\ \exists z\ contr(x,y,z) \supset spec(y,z)$
  $\Rightarrow \{p_1\}$
- $\forall y\ spec(x,y) \supset \exists z\ contr(x,y,z)$
  $\Rightarrow \{p_2\}$
- $\exists y\ \forall z\ s(z) \supset contr(x,y,z)$
  $\Rightarrow \{\emptyset\}$

5) $\neg\left((\forall x\ \forall y\ P(x,y) \supset Q(x)) \equiv (\forall x\ (\exists y\ P(x,y)) \supset Q(x))\right)$

$\forall x \forall y P(x,y) \supset Q(x)$

$\neg[\forall x (\exists y P(x,y)) \supset Q(x)]$

$\neg[(\exists y P(a,y)) \supset Q(a)]$

$\exists y P(a,y)$

$\neg Q(a)$ ×

$P(a,\beta)$ ×

$\forall y P(a,y) \supset Q(a)$

$P(a,\beta) \supset Q(a)$

$\neg P(a,\beta)$ × $\quad Q(a)$ ×

$\neg[\forall x \forall y P(x,y) \supset Q(x)]$

$\forall x (\exists y P(x,y)) \supset Q(x)$

$\neg[\forall y P(a,y) \supset Q(x)]$

$\neg[P(a,\beta) \supset Q(x)]$

$P(a,\beta)$ ×

$\neg Q(a)$ ×

$\exists y P(a,y) \supset Q(a)$

$\neg \exists y P(a,y)$ $\quad Q(a)$ ×

$\neg P(a,\beta)$ ×

All branches close $\Rightarrow$ Unsat $\neg\phi$ $\quad \models \phi$ valid

3) $\nu X \mu Y ((a \wedge \langle - \rangle X) \vee (\neg b \wedge \langle - \rangle Y))$

$[X_0] = S$

$[X_1] = \mu Y ((a \wedge \langle - \rangle X) \vee (\neg b \wedge \langle - \rangle Y)) = \{s_1, s_2, s_3\}$

$[Y_{10}] = \{\phi\}$

$[Y_{11}] = (a \wedge \langle - \rangle X_0) \vee (\neg b \wedge \langle - \rangle Y_{10})$

$= ([a] \wedge pre\exists(-, [X_0])) \vee ([\neg b] \wedge pre\exists(-, [Y_{10}]))$

$= (\{s_2\} \cap \{s\}) \cup (\{s_1, s_2, s_3\} \cap \{\phi\}) = \{s_2\}$

$[Y_{12}] = ([a] \wedge pre\exists(-, [X_0])) \vee ([\neg b \wedge pre\exists(-, [Y_{11}]))$

$= (\{s_2\} \cap \{s\}) \cup (\{s_1, s_2, s_3\} \cap \{s_1, s_2\}) = \{s_1, s_2\}$

$[Y_{13}] = ([a] \wedge pre\exists(-, [X_0])) \vee ([\neg b] \wedge pre\exists(-, [Y_{12}]))$

$= \{s_2\} \cup (\{s_1, s_2, s_3\} \cap \{s_1, s_3, s_4\}) = \{s_1, s_2, s_3\}$

$[Y_{14}] = ([a] \wedge pre\exists(-, [X_0])) \vee ([\neg b] \wedge pre\exists(-, [Y_{13}]))$

$= \{s_2\} \cup (\{s_1, s_2, s_3\} \cap \{s\}) = \{s_1, s_2, s_3\}$

→ greatest fixpoint

→ least fixpoint

$[X_2] = \mu Y ((a \wedge \langle - \rangle X) \vee (\neg b \wedge \langle - \rangle Y)) = \{s_1, s_2, s_3\}$

$[Y_{20}] = \{\phi\}$

$[Y_{21}] = (a \wedge \langle - \rangle X_1) \vee (\neg b \wedge \langle - \rangle Y_{20})$

$= ([a] \wedge pre\exists(-, [X_1])) \vee ([\neg b] \wedge pre\exists(-, [Y_{20}]))$

$= (\{s_2\} \cap \{s\}) \cup (\{s_1, s_2, s_3\} \cap \{\phi\}) = \{s_2\}$

$[Y_{22}] = ([a] \wedge pre\exists(-, [X_1])) \vee ([\neg b] \wedge pre\exists(-, [Y_{21}]))$

$= \{s_2\} \cup (\{s_1, s_2, s_3\} \cap \{s_1\}) = \{s_1, s_2\}$

$[Y_{23}] = ([a] \wedge pre\exists(-, [X_1])) \vee ([\neg b] \wedge pre\exists(-, [Y_{22}]))$

$= \{s_2\} \cup (\{s_1, s_2, s_3\} \cap \{s_1, s_3, s_4\}) = \{s_1, s_2, s_3\}$

$[Y_{24}] = ([a] \wedge pre\exists(-, [X_1])) \vee ([\neg b] \wedge pre\exists(-, [Y_{23}]))$

$= \{s_2\} \cup (\{s_1, s_2, s_3\} \cap \{s\}) = \{s_1, s_2, s_3\}$

→ least fixpoint

Initial state in solution $\rightarrow \mathcal{T} \models \phi$

$AG(\underbrace{AF\, a}_{\alpha} \wedge \underbrace{EF\, b}_{\beta} \wedge \underbrace{EG\, \neg b}_{\gamma})$

$\alpha = \mu X\ a \vee [-]X = S$

$[X_0] = \phi$

$[X_1] = [a] \cup pre\forall(-, [X_0])$

$= \{s_2\} \cup \phi = \{s_2\}$

$[X_2] = [a] \cup pre\forall(-, [X_1])$

$= \{s_2\} \cup \{s_1\} = \{s_1, s_2\}$

$[X_3] = [a] \cup pre\forall(-, [X_2])$

$= \{s_2\} \cup \{s_1, s_4\} = \{s_1, s_2, s_4\}$

$[X_4] = [a] \cup pre\forall(-, [X_3])$

$= \{s_2\} \cup \{s_1, s_3, s_4\} = S$ least fixpoint

$\beta = \mu X\ b \vee \langle - \rangle X = \{s_4\}$

$[X_0] = \emptyset$

$[X_1] = [b] \cup Pre\exists(-, [X_0])$

$= \{s_4\} \cup \emptyset = \{s_4\}$

$[X_2] = [b] \cup Pre\exists(-, [X_1])$

$= \{s_4\} \cup \{s_3\} = \{s_3, s_4\}$

$[X_3] = [b] \cup Pre(-, [X_2])$

$= \{s_4\} \cup \{s_2, s_3\} = \{s_2, s_3, s_4\}$

$[X_4] = [b] \cup Pre\exists(-, [X_3])$

$= \{s_4\} \cup \{s_1, s_2, s_3\} = S$ → least fixpoint

$\gamma = \nu X\ \neg b \wedge \langle - \rangle X$

$[X_0] = S$

$[X_1] = [\neg b] \cap Pre\exists(-, [X_0])$

$= \{s_1, s_2, s_3\} \cap \{S\} = \{s_1, s_2, s_3\}$

$[X_2] = [\neg b] \cap Pre\exists(-, [X_1])$

$= \{s_1, s_2, s_3\} \cap S = \{s_1, s_2, s_3\}$

} → greatest fixpoint

$AG(\alpha \wedge \beta \wedge \gamma) = AG(\gamma) = \nu X\ [\gamma] \wedge [-]X$

$[X_0] = S$

$[X_1] = [\gamma] \cap PreA(-, [X_0])$

$= \{s_1, s_2, s_3\} \cap \{S\} = \{s_1, s_2, s_3\}$

$[X_2] = [\gamma] \cap PreA(-, [X_1])$

$= \{s_1, s_2, s_3\} \cap \{s_1, s_2, s_4\} = \{s_1, s_2\}$

$[X_3] = [\gamma] \cap PreA(-, [X_2])$

$= \{s_1, s_2, s_3\} \cap \{s_1, s_4\} = \{s_1\}$

$[X_4] = [\gamma] \cap Pre(-, [X_3])$

$= \{s_1, s_2, s_3\} \cap \{\emptyset\} = \{s_4\}$ least fixpoint

Initial state not in solution → $T \not\models \phi$

4) $q_1 \subseteq q_2$?

- freeze free variables (none)
- Build canonical interpretation I_{q_1} and I_{q_2}

$$I_{q_1} = \begin{cases} \Delta^{I_{q_1}} = \{r, g, b\} \\ E^{I_{q_1}} = \{(r,g), (g,b), (b,r)\} \end{cases}$$

$$I_{q_2} = \begin{cases} \Delta^{I_{q_2}} = \{x, y, z, v, w\} \\ E^{I_{q_2}} = \{(x,y), (y,z), (z,x), (z,v), (v,w), (w,z)\} \end{cases}$$

- Check if $I_{q_1} \models I_{q_2}$ → find homomorphism from I_{q_2} to I_{q_1}
  - $h(x) = r$ at random
  - $h(x,y) = (r, ?) \rightarrow h(y) = g$
  - $h(y,z) = (g, ?) \rightarrow h(z) = b$
  - $h(z,x) = (b,r) \rightarrow$ ok
  - $h(z,v) = (b, ?) \rightarrow h(v) = r$
  - $h(v,w) = (r, ?) \rightarrow h(w) = g$
  - $h(w,z) = (g,b) \rightarrow$ ok

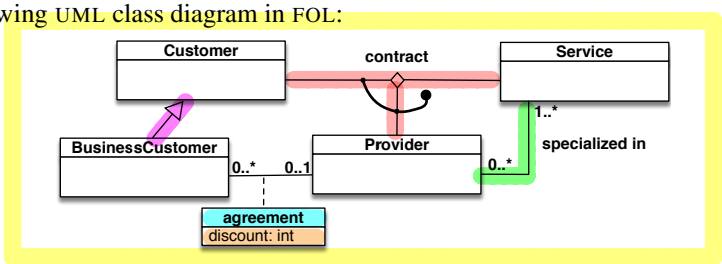
Homomorphism exists $\Rightarrow q_1 \subseteq q_2$

SAPIENZA Università di Roma – MSc. in Engineering in Computer Science

Formal Methods – July 12, 2019

(Time to complete the test: 2 hours)

Exercise 1. Express the following UML class diagram in FOL:



Exercise 2. Consider the above UML class diagram and the following (partial) instantiation:

$Customer$

c1
c2

$BusiCustumer$

b1
b2

$Provider$

p1
p2

$agreement/disc$

b1	p1	30

$Service$

s1
s2
s3
s4
s5

$specializedin$

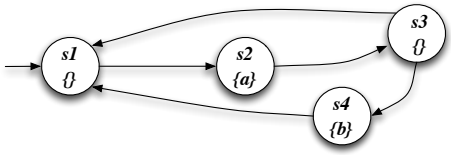
p1	s1
p1	s2
p1	s3
p2	s4
p2	s5

$contacts$

c1	p1	s1
c1	p2	s2
c2	p1	s1
b1	p1	s4
b2	p2	s5

1. Check whether the above instantiation, once completed, is correct, and explain why it is or it is not.
2. Express in FOL the following queries and evaluate them over the completed instantiation:
   (a) Return those providers that are specialized in at least two services.
   (b) Return those business customers that have contracts only with providers with whom they have an agreement.
   (c) Return those business customers that have contracts with all providers with whom have an agreement .
   (d) Check whether there exists a customer with contracts for all services.

Exercise 3. Model check the Mu-Calculus formula $\nu X.\mu Y.((a \wedge \langle next \rangle X) \vee ([next]\neg b \wedge \langle next \rangle Y)$ and the CTL formula $EG(AFa \wedge (EFb \vee AG\neg b))$ (showing its translation in Mu-Calculus) against the following transition system:



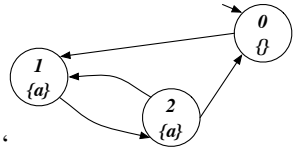
Exercise 4. Check whether the Hoare triple below is correct, by using $(x \geq 0 \wedge y \geq 0 \wedge x + y = 23)$ as invariant:

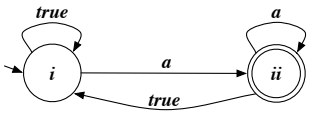
$$\{x = 23 \wedge y = 0\}\ \texttt{while(x>0) do (x=x-1; y:= y+1)}\ \{y = 23\}$$

Exercise 5. Check whether the following FOL formula is valid, by using tableaux:

$$(\,(\forall x.(A(x) \equiv B(x)))\,) \supset (\,((\forall y.A(y)) \equiv (\forall z.B(z)))\,)$$

Exercise 6 (optional). [1] Model check the LTL formula $\Diamond\Box\neg a$ against the following transition system, by considering that the Büchi automaton for $\neg(\Diamond\Box\neg a)$ is the one below:





[1] *The student can get the maximum grade even without doing Exercise 6.*

1) Alphabet: $C(x)$, $B(x)$, $P(x)$, $S(x)$, $contr(x,y,z)$, $agr(x,y)$, $discount(x,y,z)$

ISA: $\forall x\, B(x) \supset C(x)$

SPECIALIZED: $\forall x,y\ spec(x,y) \supset P(x) \wedge S(y)$

$\forall x\ P(x) \supset 1 \leq \#\{y \mid spec(x,y)\}$

CONTRACT: $\forall x,y,z\ \ contr(x,y,z) \supset C(x) \wedge P(y) \wedge S(z)$

$\forall x,y,z,z'\ contr(x,y,z) \wedge contr(x,y,z') \supset z = z'$

AGREEMENT: $\forall x,y\ \ agr(x,y) \supset B(x) \wedge P(y)$

$\forall x\ B(x) \supset \#\{y \mid agr(x,y)\} \leq 1$

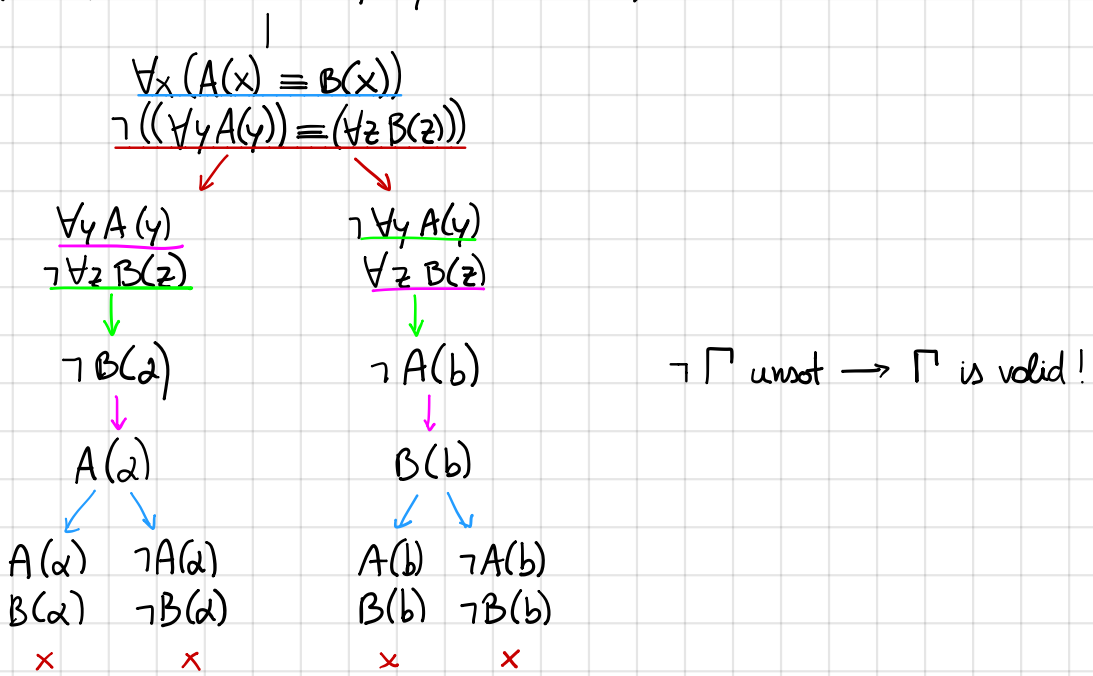
$\forall x,y\ agr(x,y) \supset 1 \leq \#\{z \mid disc(x,y,z)\} \leq 1$

DISCOUNT: $\forall x,y,z\ \ disc(x,y,z) \supset agr(x,y) \wedge Int(z)$

2) The completed instantiation (chose ISA) is correct because all axioms of Γ are made true

- $\exists y\, y'\ spec(x,y) \wedge spec(x,y') \wedge y \neq y'$
  $\Rightarrow \{p_1, p_2\}$
- $\forall y\ \exists z\ contr(x,y,z) \supset agr(x,y)$
  $\Rightarrow \{\emptyset\}$
- $\forall y\ agr(x,y) \supset \exists z\ contr(x,y,z)$
  $\Rightarrow \{b_1\}$
- $\exists y\ \forall z\ S(z) \supset contr(x,y,z)$ $\qquad \forall z\ S(z) \supset \exists y\ contr(x,y,z)$
  $\Rightarrow \{\emptyset\}$

5) $\neg((\forall x(A(x) \equiv B(x))) \supset ((\forall y\, A(y)) \equiv (\forall z\, B(z))))$



$\neg\Gamma$ unsat $\longrightarrow$ Γ is valid!

3) $\nu X \mu X ((a \wedge \langle\!-\!\rangle X) \vee ([-]\neg b \wedge \langle\!-\!\rangle Y))$

$[X_0] = \{s_1, s_2, s_3, s_4\}$

$[X_1] = \mu Y ((a \wedge \langle\!-\!\rangle X) \vee ([-]\neg b \wedge \langle\!-\!\rangle Y)) = \{s_1, s_2, s_4\}$

$[Y_{10}] = \{\emptyset\}$

$[Y_{11}] = ((a \wedge \langle\!-\!\rangle X_0) \vee ([-]\neg b \wedge \langle\!-\!\rangle Y_{10})) =$
$([a] \cap pre_E(-, [X_0])) \cup (pre_A(-, [\neg b]) \cap pre_E(-, [Y_{10}]))$
$= (\{s_2\} \cap \{s_1, s_2, s_3, s_4\}) \cup (\{s_1, s_2, s_4\} \cap \{\emptyset\}) = \{s_2\}$

$[Y_{12}] = ([a] \cap pre_E[-, [X_0]]) \cup (pre_A(-, [\neg b]) \cap pre_E(-, [Y_{11}])) =$
$= (\{s_2\} \cap \{s_1, s_2, s_3, s_4\}) \cup (\{s_1, s_2, s_4\} \cap \{s_2\}) = \{s_1, s_2\}$

$[Y_{13}] = ([a] \cap pre_E(-, [X_0])) \cup (pre_A(-, [\neg b]) \cap pre_E(-, [Y_{12}])) =$
$(\{s_2\} \cap \{s_1, s_2, s_3, s_4\}) \cup (\{s_1, s_2, s_4\} \cap \{s_1, s_3, s_4\}) = \{s_1, s_2, s_4\}$

$[Y_{14}] = ([a] \cap pre_E(-, [X_0])) \cup (pre_A(-, [\neg b]) \cap pre_E(-, [Y_{13}])) =$
$(\{s_2\} \cap \{s_1, s_2, s_3, s_4\}) \cup (\{s_1, s_2, s_4\} \cap \{s_1, s_3, s_4\}) = \{s_1, s_2, s_4\}$

→ least fixpoint!

$[X_2] = \mu Y ((a \wedge \langle\!-\!\rangle X) \vee ([-]\neg b \wedge \langle\!-\!\rangle Y)) = \{\emptyset\}$

$[Y_{20}] = \{\emptyset\}$

$[Y_{21}] = ((a \wedge \langle\!-\!\rangle X_1) \vee ([-]\neg b \wedge \langle\!-\!\rangle Y_{20})) =$
$([a] \cap pre_E(-, [X_1])) \cup (pre_A(-, [\neg b]) \cap pre_E(-, [Y_{20}])) =$
$(\{s_2\} \cap \{s_1, s_3, s_4\}) \cup (\{s_1, s_2, s_4\} \cap \{\emptyset\}) = \{\emptyset\}$

→ least fixpoint!

$[X_3] = \mu Y ((a \wedge \langle\!-\!\rangle X) \vee ([-]\neg b \wedge \langle\!-\!\rangle Y)) = \{\emptyset\}$

→ greatest fixpoint

$[Y_{30}] = \{\emptyset\}$

$[Y_{31}] = ((a \wedge \langle\!-\!\rangle X_2) \vee ([-]\neg b \wedge \langle\!-\!\rangle Y_{30})) =$
$([a] \cap pre_E(-, [X_2])) \cup (pre_A(-, [\neg b]) \cap pre_E(-, [Y_{30}])) =$
$(\{s_2\} \cap \emptyset) \cup (\{s_1, s_2, s_4\} \cap \emptyset) = \{\emptyset\}$

→ least fixpoint

Initial state not in solution ($\{\emptyset\}$) $\rightarrow T \not\models \Phi$ (Φ = μ-calculus formula)

CTL: $EG(AFa \wedge (\underbrace{EFb}_{\beta} \vee \underbrace{AG\neg b}_{\alpha}))$

$\alpha = \nu X \neg b \wedge [-]X = \{\emptyset\}$

$[X_0] = \{s_1, s_2, s_3, s_4\}$

$[X_1] = \neg b \wedge [-]X_0 = [\neg b] \cap pre_A(-, [X_0]) = \{s_1, s_2, s_3\} \cap \{s_1, s_2, s_3, s_4\} = \{s_1, s_2, s_3\}$

$[X_2] = [\neg b] \cap pre_A(-, [X_1]) = \{s_1, s_2, s_3\} \cap \{s_1, s_2, s_4\} = \{s_1, s_2\}$

$[X_3] = [\neg b] \cap pre_A(-, [X_2]) = \{s_1, s_2, s_3\} \cap \{s_1, s_2, s_4\} = \{s_1\}$

$[X_4] = [\neg b] \cap pre_A(-, [X_3]) = \{s_1, s_2, s_3\} \cap \{s_4\} = \emptyset$ fixpoint! (X_5 will be $\emptyset$)

$\beta = \mu X\, b \vee \langle\!-\!\rangle X = \{s_1, s_2, s_3, s_4\}$

$[X_0] = \emptyset$

$[X_1] = [b] \cup pre_E(-, X_0) = \{s_4\} \cup \{\emptyset\} = \{s_4\}$

$[X_2] = [b] \cup pre_E(-, X_1) = \{s_4\} \cup \{s_3\} = \{s_3, s_4\}$

$[X_3] = [b] \cup pre_E(-, X_2) = \{s_4\} \cup \{s_2, s_3\} = \{s_2, s_3, s_4\}$

$[X_4] = [b] \cup pre_E(-, X_3) = \{s_4\} \cup \{s_1, s_2, s_3\} = \{s_1, s_2, s_3, s_4\}$ fixpoint! (X_5 will be equal/S)

$EG(AFa \wedge (S \vee \emptyset)) = EG(AFa \wedge S) = EG(AFa)$ because $\wedge S \rightarrow \cap S \rightarrow$ intersection with all states is meaningless.

$AF_a = \mu X_a \, \nu [-]X = \{s\}$

$[X_0] = \emptyset$

$[X_1] = [a] \cup pre_A(-, X_0) = \{s_2\}$

$[X_2] = [a] \cup pre_A(-, X_1) = \{s_2\} \cup \{s_1, s_4\} = \{s_1, s_2\}$

$[X_3] = [a] \cup pre_A(-, X_2) = \{s_2\} \cup \{s_1, s_4\} = \{s_1, s_2, s_4\}$

$[X_4] = [a] \cup pre_A(-, X_3) = \{s_2\} \cup \{s_1, s_3, s_4\} = \{s\}$ fixpoint

$EG(AF_a) = EG(s) = \nu X \, s \wedge \langle - \rangle X = \{s\}$

$[X_0] = s$

$[X_1] = \{s\} \cup pre(-, [X_0]) = \{s\}$ fixpoint

Initial state in solution $\rightarrow I \not\models \phi$

4) $I = \{x \geq 0 \wedge y \geq 0 \wedge x + y = 23\}$

$P = \{x = 23 \wedge y = 0\}$

$Q = \{y = 23\}$

$\delta = \{x = x - 1 \; ; \; y = y + 1\}$

$G = \{x > 0\}$

- check $P \supset I$

$x = 23 \wedge y = 0 \supset x \geq 0 \wedge y \geq 0 \wedge x + y = 23$ ✓

- check $\neg g \wedge I \supset Q$

$x \leq 0 \wedge x \geq 0 \wedge y \geq 0 \wedge x + y = 23 \supset y = 23$ ✓

$x = 0$

- check $\{g \wedge I\} \, \delta \, \{I\} = g \wedge I \supset wp(\delta, I)$

$x > 0 \wedge x \geq 0 \wedge y \geq 0 \wedge x + y = 23 \supset wp(\delta, I)$

$\hookrightarrow x > 0 \wedge x \geq 0 \wedge y \geq 0 \wedge x + y = 23 \supset x - 1 \geq 0 \wedge y + 1 \geq 0 \wedge x + y = 23$

$\{x - 1 \geq 0 \wedge y + 1 \geq 0 \wedge x + y = 23\}$

$x = x - 1$

$\{x \geq 0 \wedge y + 1 \geq 0 \wedge x + y + 1 = 23\}$

$y = y + 1$

$I \rightsquigarrow \{x \geq 0 \wedge y \geq 0 \wedge x + y = 23\}$

I is an invariant so the Hoare triple is correct!