Underapproximation in VMCAI

David Schmidt

Kansas State University

www.cis.ksu.edu/~schmidt

Belated 60th birthday greetings to Ed Clarke and a big thank you for your leadership in the field!

90% of this talk originates from...

1. *Clarke, Grumberg, Long [POPL92, TOPLAS 94]:* state-space abstractions via semi-homorphisms to validate ∀CTL*-properties.

2. *Cousot-Cousot [POPL77, POPL79, JLC 92]:* abstract-interpretation frameworks that synthesize abstract functions and ensure soundness

3. *Dams [thesis 96, and TOPLAS 97 with Gerth and Grumberg]:* formalization of underapproximation functions on abstract models

Introduction

Overapproximation *“states the possibilities” or “covers the behaviors” of an entity or process*

Examples:

- *a program’s control-flow graph* — its paths represent a superset of the program’s actual executions
- *an entity-relation model* — it states all legal relationships between objects in a knowledge base
- *data-type information* of a program’s variables states the variables’ range of values
- *logical program assertions* that have not been refuted state ”what the program might possibly do”

Underapproximation ***“lists the necessities” or gives concrete examples of an entity or process***

Examples:

- *logical program assertions* that have been proved true or are required of a program define a subset of the program’s theory
- *test-execution traces* assert guaranteed program behaviors
- *execution monitoring* remembers values that have been assigned to program variables at some point
- *an object subdiagram* displays relationships the program must construct during its execution

Testing is a commonly used underapproximation of program behavior; so is *Bounded Model Checking* (trace generation to some fixed $k \geq 0$).

Such *concrete underapproximations* construct *witnesses* to an *existential property* (e.g., a trace that ends in an error state).

Sometimes, concrete underapproximations can be clevely applied to other ends:

- **Păsăreanu, et al. CAV05:** each concrete trace refines a predicate-abstraction model of the program's control-flow graph; the limit of the refinements is a model that is bisimular to the program's concrete control-flow graph (modulo the predicates selected for the analysis)

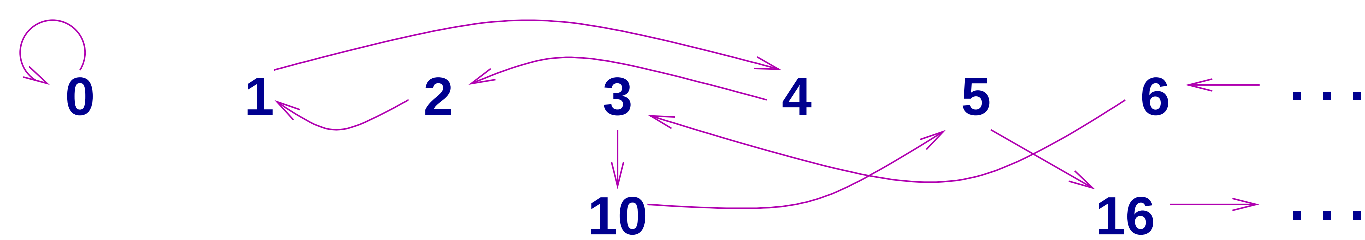
- **Grumberg, et al. POPL05**: SAT-generated *BMC* traces attempt to refute a property via a counterexample trace. If no counterexample found, the SAT-trace-proof is disassembled to see if the bounding on the traces appears in the trace-proof. If no, then the trace-proof is *rebuilt* into a proof that *proves* the property.

We will study over- and underapproximations calculated on **abstractions** *of states/data*

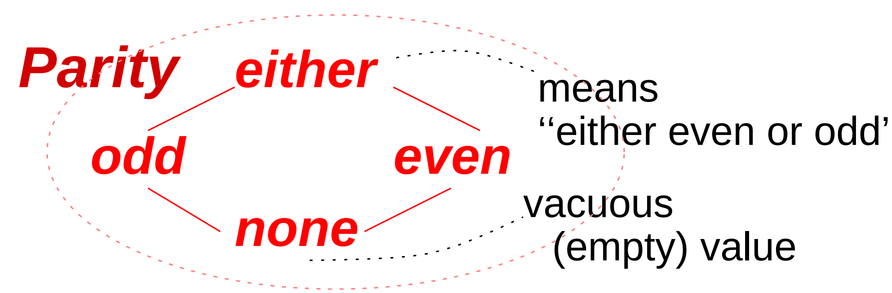
Example:
the Collatz function:

```
loop
  n div2 == 0 ?  n := n div2;
  n div2 == 1 ?  n := 3n + 1;
endloop
```

The function's graph — what really happens:



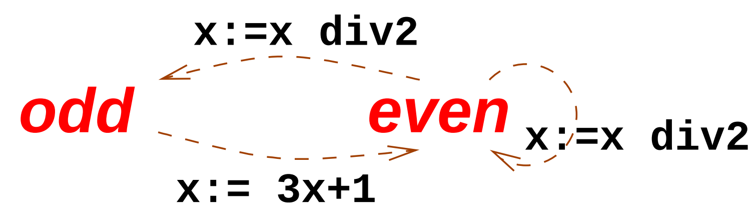
The Parity abstraction of the function's domain:



Example: $2 \in \gamma(\mathrm{even}) = \{0, 2, 4, \cdots\}$, $3 \in \gamma(\mathrm{odd}) = \{1, 3, 5, \cdots\}$, for the modelling (*concretization*) function, $\gamma : \mathrm{Parity} \to \mathcal{P}(\mathrm{Nat})$.

An overapproximation (“may”) graph based on Parity

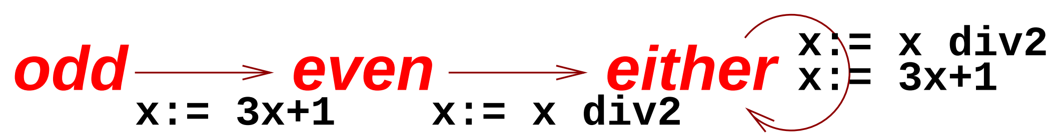
($\exists\exists$-*approximation*): covers all concrete paths but contains “false paths”:



Example: $8 \rightarrow 4$ and $10 \rightarrow 5$, hence $even \rightarrow even$ and $even \rightarrow odd$. **You may add transitions to this model and it remains an overapproximation.**

An underapproximation (”must”) graph based on Parity

($\forall\exists$-*approximation*): all paths are guaranteed to exist as concrete executions:



Example: for every odd number, $2n + 1 \geq 0$, there is some $2m$ such that $2n + 1 \rightarrow 2m$, hence $odd \rightarrow even$. **You may remove transitions from this model and it remains an underapproximation.**

We validate logical properties on abstract models

Let C be the set of concrete values (states) and A be the set of abstract ones.

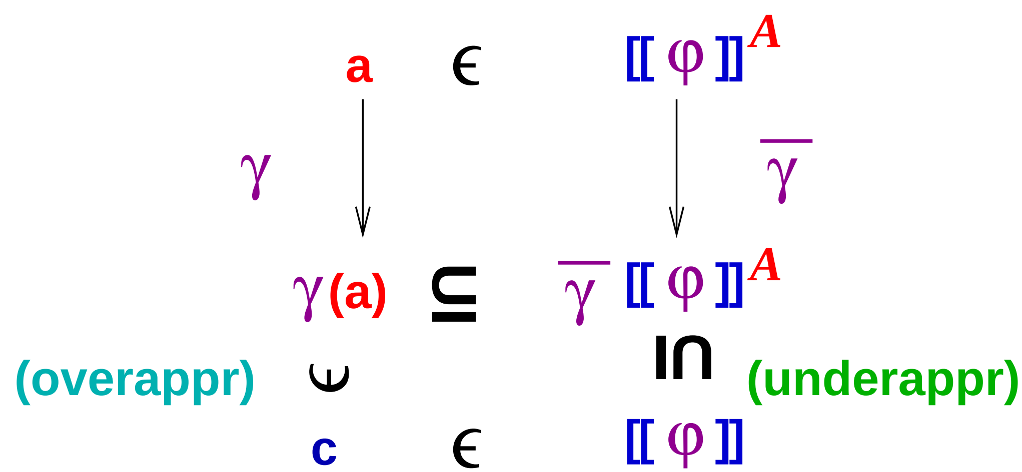
Say that $\phi \in \mathcal{L}$*, a logic, and* $[\![\phi]\!] \subseteq C$*.*

For $c \in C$*, write* $c \models \phi$ *when* $c \in [\![\phi]\!]$*.*

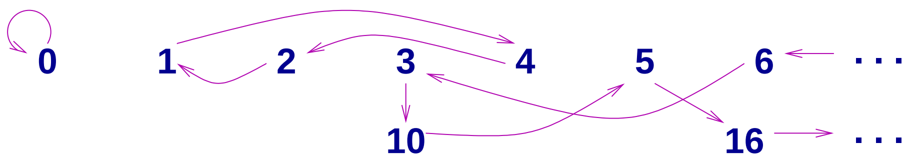
For $a \in A$, we wish to check $a \models^A \phi$ and *infer* $c \models \phi$ for those $c \in \gamma(a)$. Recall that $\gamma : A \to \mathcal{P}(C)$ is the modelling/concretization function.

Slogan: Overapproximate *the computation and* **underapproximate** *the logic*:

if $a \models^A \phi$
(i.e., $a \in [\![\phi]\!]^A$),
then, for all $c \in \gamma(a)$
$c \models \phi$ (i.e, $c \in [\![\phi]\!]$)



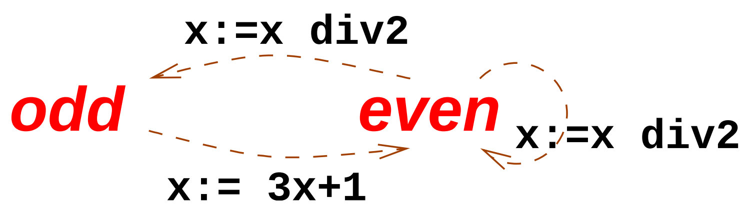
But we also know that an *over* approximating model is used to validate *universal* properties and an *under* approximating model is used to validate *existential* properties:




```
loop
  n div2 == 0?  n:= n div2;
  n div2 == 1?  n:= 3n+1;
endloop
```

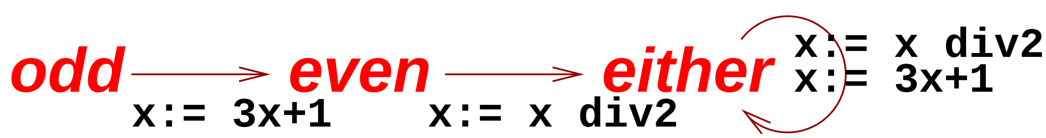
$\mathrm{odd} \models^{A} \Box\mathit{even}$

$\mathit{even} \models^{A} \forall(\mathrm{GF}\mathit{even})$



$\mathrm{odd} \models^{A} \Diamond\mathit{even}$

$\mathrm{odd} \models^{A} \exists(\mathrm{F}\mathit{even})$



We might even *mix* the two: $\mathit{even} \models^{A} \Box(\mathit{even} \vee \Diamond\mathit{even})$

How do we make sense of this?

How we make sense of over-underapproximation

1. Kripke structures, simulations, Galois connections
2. Logics
3. How to underapproximate a logic
4. How to over- and underapproximate a program's control structure
5. How to underapproximate a data structure
6. Over- and underapproximation in specification

Kripke structures, simulations, Galois connections

A program's semantics is coded as a State Transition System

Definition: An *STS* is (C, f), where C is the state set and $f \subseteq C \times C$ (or $f : C \to \mathcal{P}(C)$) is the transition relation (function).

Collatz example: C is Nat, and

$$f = \{(n, n\ div2) \mid n\ div2 == 0\} \cup \{(n, 3n+1) \mid n\ div2 == 1\}$$

A *run* (*trace*) is $c_0 \to c_1 \to \cdots c_i \to c_{i+1} \to \cdots$ such that for all $i \geq 0$, $(c_i, c_{i+1}) \in f$.

Let $Prop$ be a set of primitive properties and let $\gamma : Prop \to \mathcal{P}(C)$ interpret it.

Definition: A *Kripke structure* is an STS + γ.

Collatz example: $Prop = \{even, odd\}$, and

$$\gamma(even) = \{2n \mid n \geq 0\}, \quad \gamma(odd) = \{2n+1 \mid n \geq 0\}$$

When (C, f) ***is "too large," we must* abstract *it***

Say that $\gamma : \mathrm{Prop} \to C$ *partitions* C ($\forall c \in C, \exists! p \in \mathrm{Prop}, c \in \gamma(p)$).

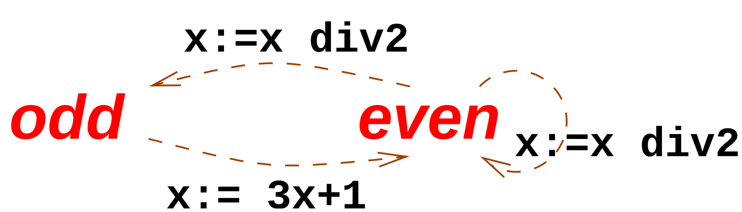
Then we can define the abstract STS, $(\mathrm{Prop}, f^\sharp)$, where

$$(p, q) \in f^\sharp \text{ iff } \exists c \in \gamma(p), \exists c' \in \gamma(q), (c, c') \in f$$

$f^\sharp$ is an $\exists\exists$-relation.

Collatz example:

$$\mathrm{Prop} = \{even, odd\},$$
$$\gamma(even) = \{2n \mid n \geq 0\}, \quad \gamma(odd) = \{2n + 1 \mid n \geq 0\}, \text{ and}$$
$$f^\sharp = \{(odd, even), (even, even), (even, odd)\}$$



Soundness of the abstraction is asserted by a **simulation**

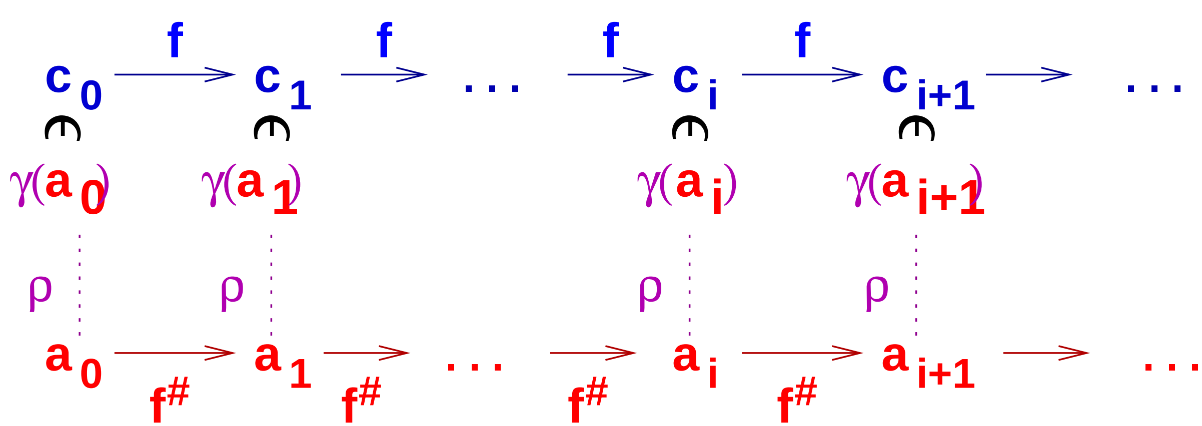
Define the *modelling relation*, $\rho \subseteq C \times A$, from γ as

$$c \,\rho\, p \textit{ iff } c \in \gamma(p).$$

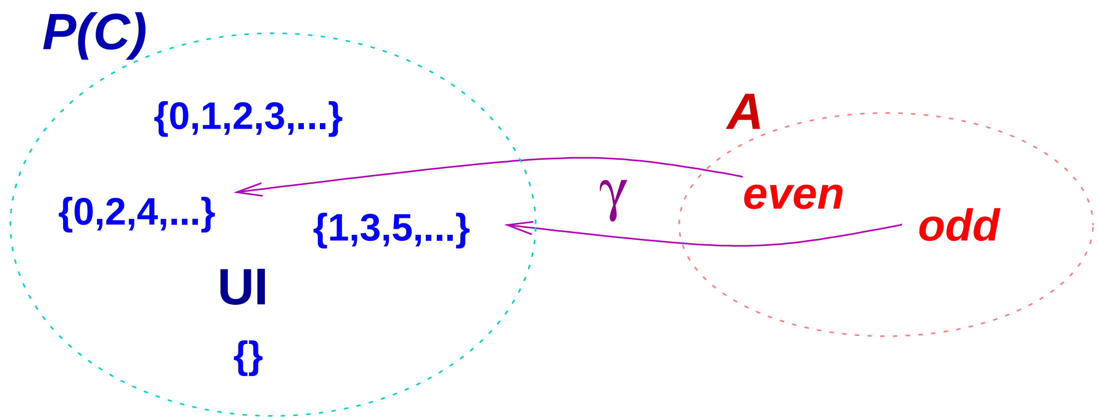
Definition: For STSs (C, f) and $(A, f^\sharp)$, ρ is a *simulation* iff for all $c \in C$, $a \in A$,

$$c \,\rho\, a \text{ and } (c, c') \in f \text{ imply there exists } a' \in A \text{ such that } (a, a') \in f^\sharp \text{ and } c' \,\rho\, a'.$$

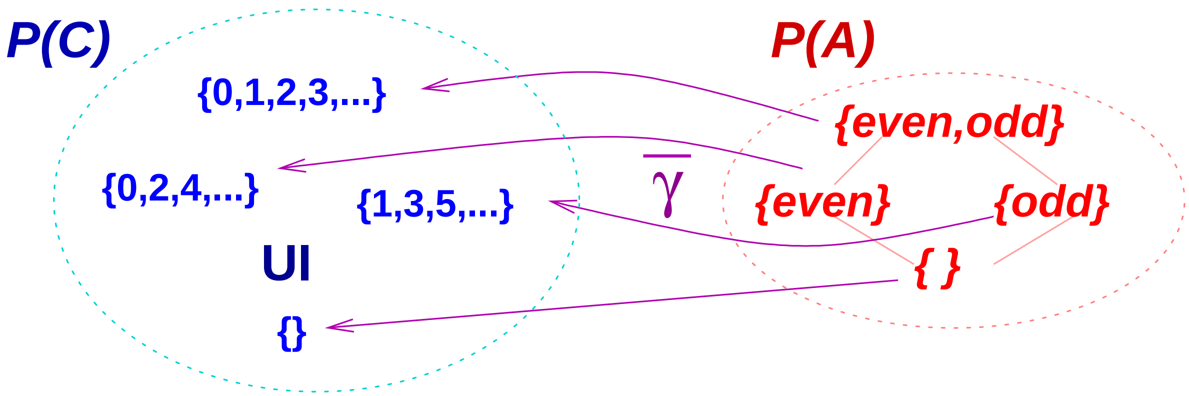
$f^\sharp$ mimicks f, modulo ρ. We write $f \lhd f^\sharp$:



Given γ**:**



we lift it to $\overline{\gamma} : \mathcal{P}(A) \to \mathcal{P}(C)$**:**



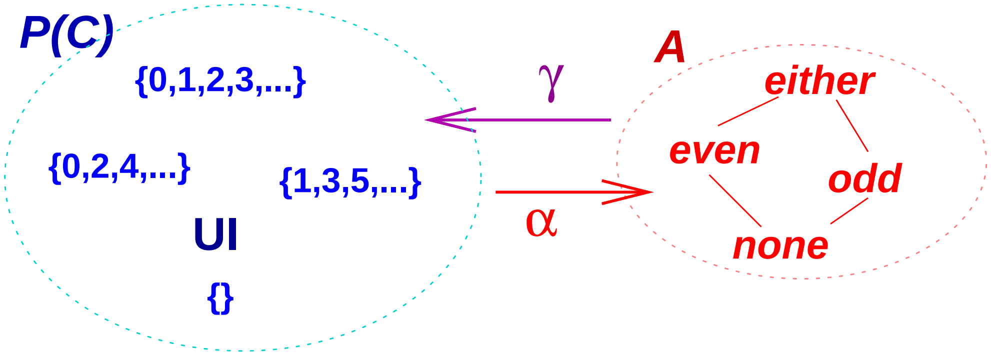
$\overline{\gamma}(T) = \cup_{a \in T}\, \gamma(a)$

This lets us define an abstract transition *function*, $f^\sharp : A \to \mathcal{P}(A)$.

Collatz example: $f^\sharp(even) = \{even, odd\}$, $f^\sharp(odd) = \{even\}$.

Theorem: *Simulation equals abstract-interpretation soundness:*
For $f : C \to \mathcal{P}(C)$, $f^\sharp : A \to \mathcal{P}(A)$, $f \lhd f^\sharp$ iff $f^* \circ \gamma \sqsubseteq \gamma \circ f^\sharp$, where $f^*(S) = \cup_{c \in S} f(c)$.

The previous slide hints that γ and $\overline{\gamma}$ have inverse maps — *they do* — and this situation is called a *Galois connection*:

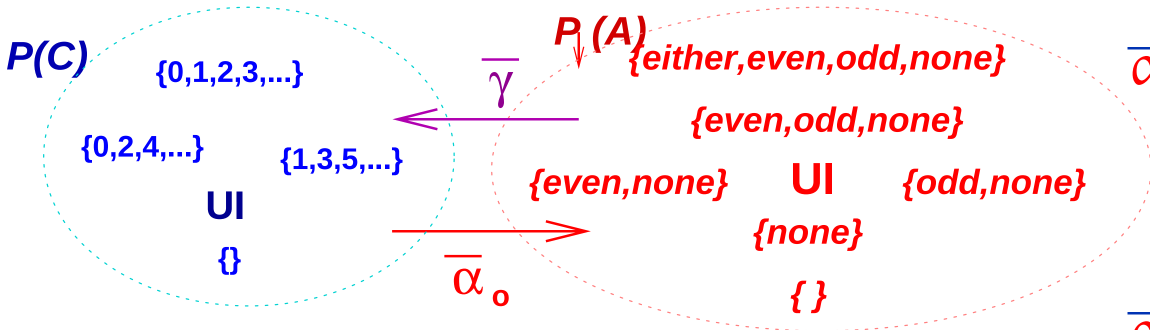


$\alpha(S) = \sqcap\{a \mid \gamma(a) \subseteq S\}$

e.g., $\alpha\{2,6\} = even$

$\alpha\{0,1,2,3\} = either$

$either$ and $none$ help $Parity$ become a complete lattice.



$\overline{\alpha}_o(S) = \cap\{T \mid S \subseteq \overline{\gamma}(T)\}$

e.g.,

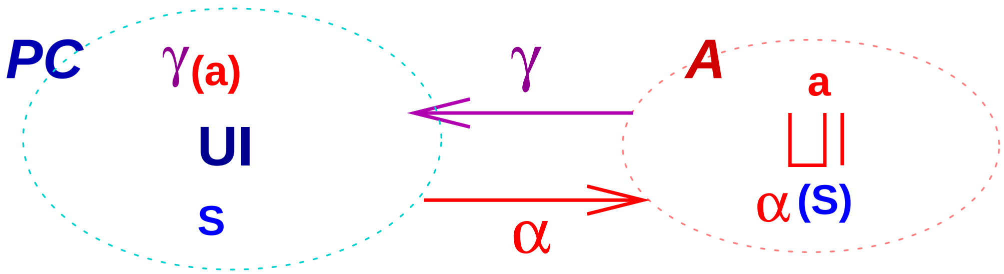
$\overline{\alpha}_o\{2,6\} = \{even, none\}$

$\overline{\alpha}_o\{0,2,3\} = \{even, odd, none\}$

Down-closed sets are needed to make $\overline{\alpha}_o$ monotone.

Note that $\overline{\gamma}\{either, even, odd, none\} = \overline{\gamma}\{even, odd, none\}$; the former is superfluous and can be deleted. The same is true for $\{\}$ and $\{none\}$.

Why are Galois connections important?



$$\alpha(\gamma(a)) \sqsubseteq a$$

$$S \subseteq \gamma(\alpha(S))$$

1. for $S \in PC$, $\alpha(S)$ gives the most precise approximant in A
2. it formalizes (over)approximation ($\subseteq$ in PC above!): $S \subseteq \gamma(\alpha(S))$, and for all $S \in PC$, $f(S) \subseteq \gamma(f^{\sharp}(\alpha(S)))$
3. it ensures that $\sqcap_A$ is *conjunction* — $a \sqcap a'$ is read as $a \wedge a'$ — because $\gamma(\sqcap_i a_i) = \cap_i \gamma(a_i)$
4. for $f : PC \rightarrow PC$, we can synthesize the most precise approximation, $f^{\sharp}_{best} = \alpha \circ f \circ \gamma$

   Dams: for $f : C \rightarrow \mathcal{P}(C)$ *in an STS,* $f^{\sharp}_{best} : A \rightarrow \mathcal{P}_{\downarrow}(A)$ *is*

   $f^{\sharp}_{best}(a) = \downarrow\{\alpha\{c'\} \mid c \in \gamma(a), c' \in f(c)\}$ *— the minimal* $\exists\exists$*-relation*

Logics

We use a logic to state properties of (C, f)

$$p \in \mathrm{Prop} \qquad \phi \in \mathcal{L} \qquad\qquad [\![\cdot]\!] \subseteq C$$

$$\phi ::= p \mid op_g^k(\phi_i)_{i<k} \qquad\qquad [\![p]\!] = \gamma(p)$$

$$[\![op_g^k(\phi_i)_{i<k}]\!] = g[\![\phi_i]\!]_{i<k}$$

Collatz example:

$$\phi ::= even \mid odd \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \Box\phi$$

$$[\![even]\!] = \gamma(even) \qquad [\![odd]\!] = \gamma(odd)$$

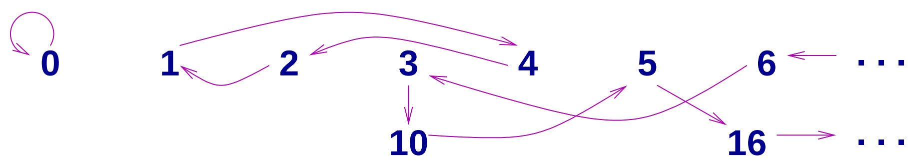
$$[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$$

$$[\![\phi_1 \vee \phi_2]\!] = [\![\phi_1]\!] \cup [\![\phi_2]\!]$$

$$[\![\Box\phi]\!] = \widetilde{pre}_f[\![\phi]\!], \text{ where } \widetilde{pre}_f(S) = \{c \mid f(c) \subseteq S\}$$

$c \in [\![\Box\phi]\!]$ *means "$\forall f.\phi$" — all next* $f(c)$*-states belong to* ϕ *— for transition function,* $f : C \to \mathcal{P}(C)$*.*

Collatz examples:



$$\gamma\{\text{even}\} = \{2n \mid n \geq 0\}$$

$$\gamma\{\text{odd}\} = \{2n+1 \mid n \geq 0\}$$

$3 \in [\![\text{odd}]\!]$

$3 \in [\![\text{odd} \wedge \Box\text{even}]\!]$

$10 \in [\![\Box\text{odd}]\!]$

$12 \notin [\![\Box\text{odd}]\!]$

We'll examine path logics, where $[\![\psi]\!] \subseteq C^*$, later.

Approximating the logic

We will apply the logic to the abstract STS, $(A, f^\sharp)$

For $(A, f^\sharp)$, $\gamma : A \to \mathcal{P}(C)$, we must define a $[\![\cdot]\!]^A \subseteq A$ so that it is

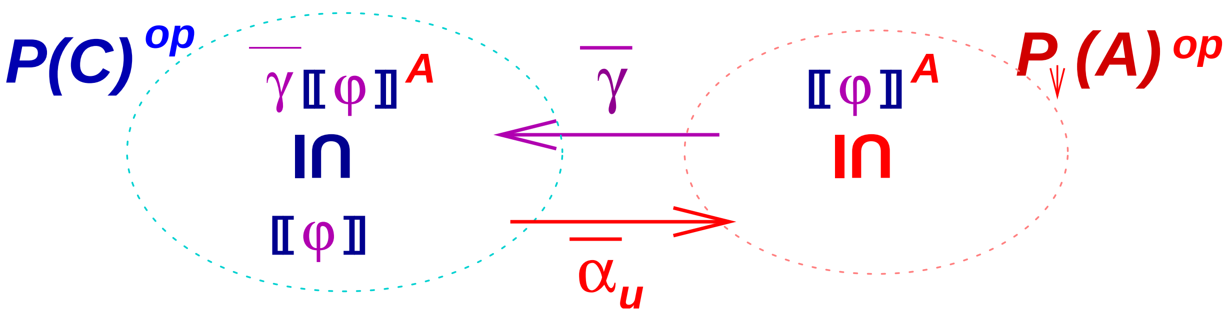
1. ***weakly preserving (sound)***:

   for all $a \in A$, $a \in [\![\phi]\!]^A$ implies $c \in [\![\phi]\!]$, for all $c \in \gamma(a)$

2. ***best preserving***:

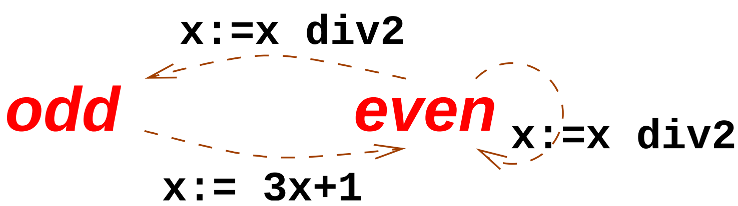
   for all $a \in A$, $a \in [\![\phi]\!]^A$ iff $\gamma(a) \subseteq [\![\phi]\!]$

For soundness, $[\![\cdot]\!]^A$ must *underapproximate* $[\![\cdot]\!]$:



that is, $\overline{\gamma}[\![\phi]\!]^A \subseteq [\![\phi]\!]$.

Collatz examples: abstract transition function, $f^\sharp : \mathcal{A} \to \mathcal{P}(\mathcal{A})$:



$$f^\sharp(\mathrm{odd}) = \{\mathit{even}\}$$
$$f^\sharp(\mathit{even}) = \{\mathit{even}, \mathrm{odd}\}$$

We anticipate that

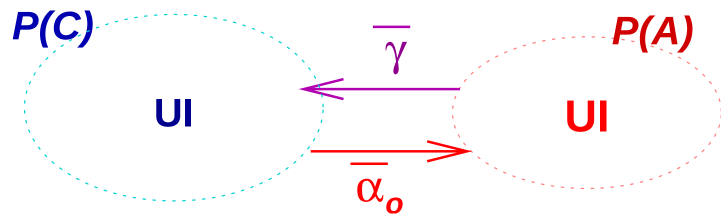
$\mathrm{odd} \in [\![\mathrm{odd}]\!]^{\mathcal{A}}$

$\mathrm{odd} \in [\![\mathrm{odd} \wedge \Box \mathit{even}]\!]^{\mathcal{A}}$

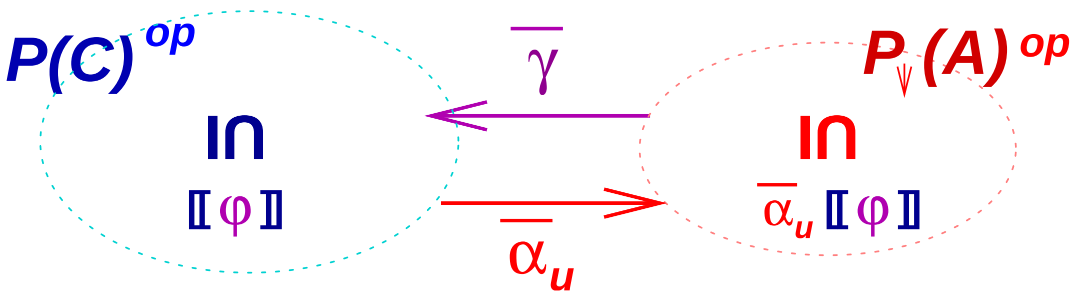
but $\mathit{even} \notin [\![\Box \mathrm{odd}]\!]^{\mathcal{A}}$ due to loss in precision.

How do we define $[\![\cdot]\!]^{\mathcal{A}}$ **?**

How to compute $[\![\cdot]\!]^A$ *from* $[\![\cdot]\!]$

For this Galois connection, 

$\overline{\gamma} : \mathcal{P}_{\downarrow}(A) \rightarrow \mathcal{P}(C)$ preserves $\sqcup$ *as well as* $\sqcap$ —it can be *inverted*:



$\overline{\alpha}_u : \mathcal{P}(C)^{op} \rightarrow \mathcal{P}_{\downarrow}(A)^{op}$ is new:

$\overline{\alpha}_u(S) = \{a \mid \gamma(a) \subseteq S\}$

$\overline{\alpha}_u$ tells us how to abstract $[\![\phi]\!]$ —use $\overline{\alpha}_u[\![\phi]\!]$!

Theorem: $\overline{\alpha}_u[\![\cdot]\!]$ is best preserving. But it's not defined inductively....

A sound inductive definition requires a $g^\flat$

$$[\![p]\!]^A = \overline{\alpha}_u[\![p]\!] = \{a \mid \gamma(a) \subseteq \gamma(p)\}$$

$$[\![op_g^k(\phi_i)_{i<k}]\!]^A = g^\flat[\![\phi_i]\!]^A{}_{i<k}$$

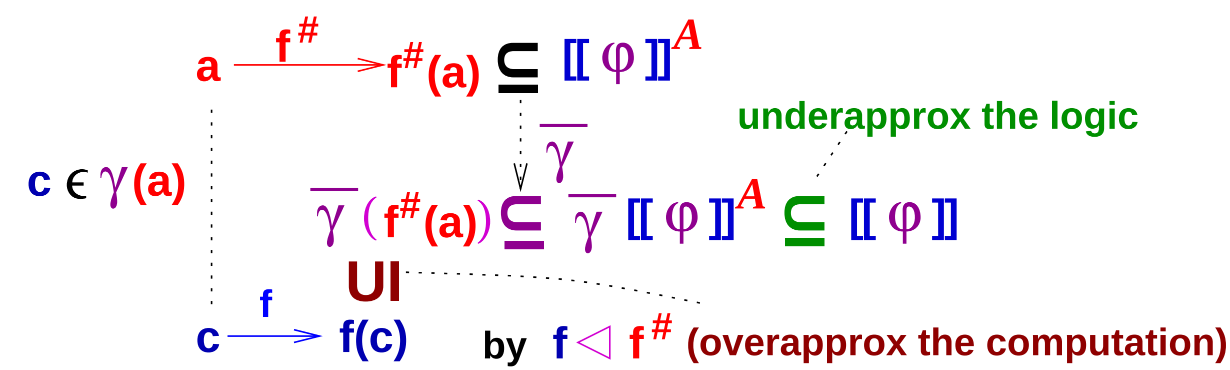
where $g^\flat$ *underapproximates* g: $\overline{\gamma}(g^\flat(T)) \subseteq g(\overline{\gamma}(T))$, for $T \in \mathcal{P}_\downarrow(A)$.

Sometimes, obvious selections for $g^\flat$ work well:

$[\![\phi_1 \wedge \phi_2]\!]^A = [\![\phi_1]\!]^A \cap [\![\phi_2]\!]^A$ because $\overline{\gamma}$ preserves both

$[\![\phi_1 \vee \phi_2]\!]^A = [\![\phi_1]\!]^A \cup [\![\phi_2]\!]^A$ meets and joins in $\mathcal{P}_\downarrow(A)$.

$[\![\Box\phi]\!]^A = \widetilde{pre}_{f^\sharp}[\![\phi]\!]^A$, where $f \lhd f^\sharp$, because



if all $f^\sharp(a)$'s answers have ϕ, so must $f(c)$'s, because $f^\sharp$ "covers" f

The most precise (largest-set) inductive definition

$$[\![p]\!]^{A} = \overline{\alpha}_u [\![p]\!]$$

$$[\![op^k_g(\phi_i)_{i<k}]\!]^{A} = g^{\flat}_{best} [\![\phi_i]\!]^{A}{}_{i<k},$$

where $g^{\flat}_{best} = \overline{\alpha}_u \circ g \circ \overline{\gamma}^k$, for $g : \mathcal{P}(C)^k \to \mathcal{P}(C)$

Example: $(\tilde{pre}_f)^{\flat}_{best} = \overline{\alpha}_u \circ \tilde{pre}_f \circ \overline{\gamma} = \{a \mid f^*(\gamma(a)) \subseteq \overline{\gamma}(T)\}$.

Proposition: (weak preservation) $\overline{\alpha}_u [\![\phi]\!] \supseteq [\![\phi]\!]^{A}$

Theorem: (best preservation) When $\overline{\gamma}$ is *complete with respect to* g, then $\overline{\alpha}_u [\![\phi]\!] = [\![\phi]\!]^{A}$.

$\overline{\gamma}$ is *complete w.r.t* g iff soundness is exact: $g \circ \overline{\gamma} = \overline{\gamma} \circ g^{\flat}_{best}$.

Theorem: $(\tilde{pre}_f)^{\flat}_{best} = \tilde{pre}_{(f^*)^{\sharp}_{best}} \sqsubseteq \tilde{pre}_{f^{\sharp}}$

The preimage of $(f^*)^{\sharp}_{best} = \overline{\alpha}_o \circ f^* \circ \gamma$, the most precise overapproximation of f, equals the most precise underapproximation of $\tilde{pre}_f$.

An existential assertion: $\Diamond\phi$

$\Box\phi$ is a “universal assertion.” In contrast, $\Diamond\phi$ *asserts there exists a next state with* ϕ*:*

$$[\![\Diamond\phi]\!] = pre_f[\![\phi]\!],$$

where $f : C \to \mathcal{P}(C)$ and $pre_f(S) = \{c \mid f(c) \cap S \neq \emptyset\}$.

Here is its most precise (largest) underapproximation:

$$[\![\Diamond\phi]\!]^A = (\overline{\alpha}_u \circ pre_f \circ \overline{\gamma})[\![\phi]\!]^A$$
$$= \{a \mid \text{for all } c \in \gamma(a), f(c) \cap \overline{\gamma}(T) \neq \emptyset\}$$

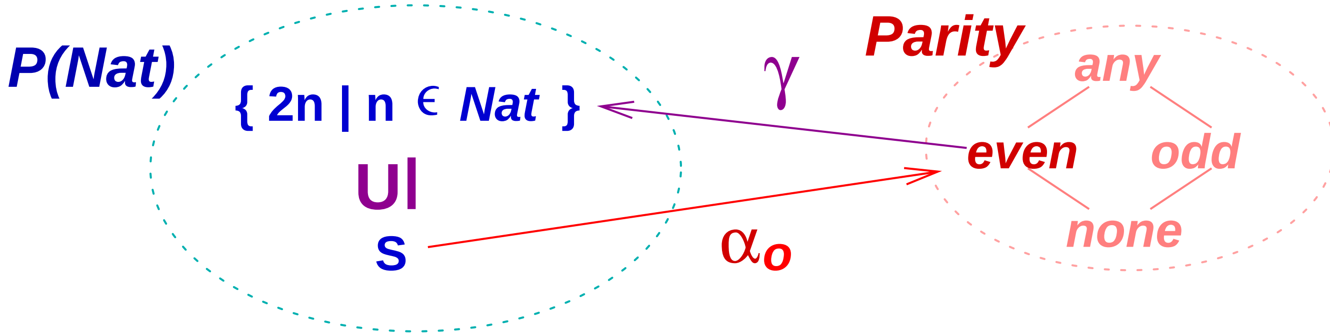
This is a $\forall\exists$-set.

Can we use $[\![\Diamond\phi]\!]^A = pre_{f^\sharp}[\![\phi]\!]^A$ **? NO.** **Collatz example:** $even \to even$, $10 \in \gamma(even)$, yet $10 \to 5$ only — *an even is not guaranteed to transit to an even.*

Can we use $[\![\Diamond\phi]\!]^A = pre_{f_0^\flat}[\![\phi]\!]^A$**, where** $f_0^\flat = \overline{\alpha}_u \circ f^* \circ \gamma$ **? NO.** All sets, $[\![\phi]\!]^A$, are *downwards closed* in A. But $pre_{f_0^\flat}(T)$, for downclosed T, is an *upwards-closed set* !

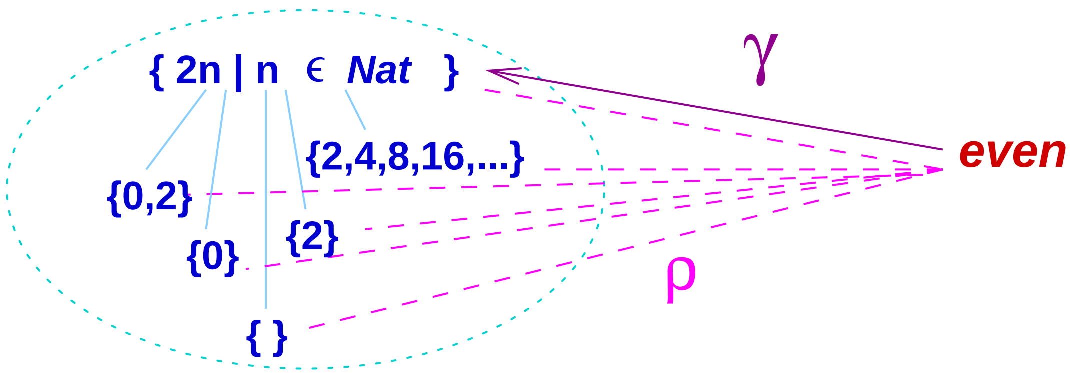
Underapproximating control

Some perspective: Over-approximation states a property of a program's outputs



$even \in Parity$ asserts "$\forall even$" — all concrete outputs in set S are even-valued. (*We might write* $S \; \rho \; even$ *or* $S \models even$.)

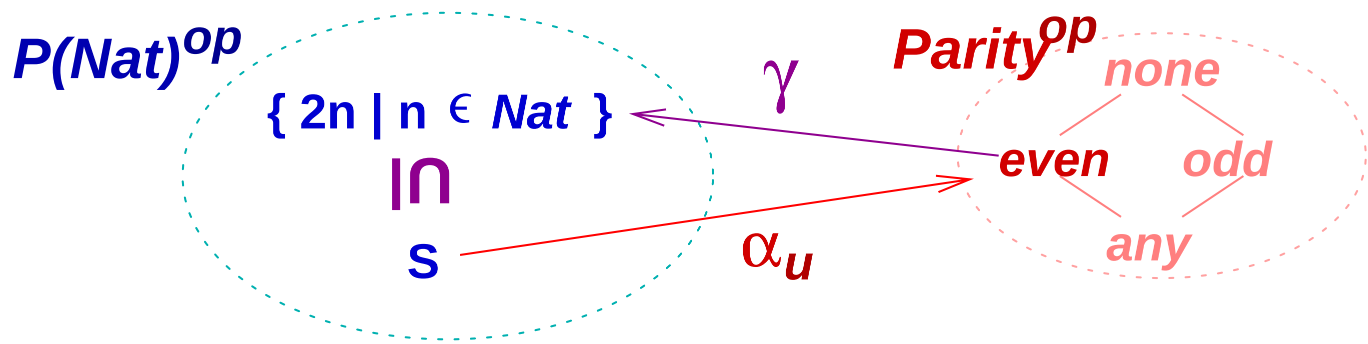
The upper adjoint, γ, selects the largest set approximated by $even$:



$$\gamma(a) = \cup\{S \mid S \; \rho \; a\}$$

Under-approximation *might be stated as the dual*

Here, *even* asserts that all evens are *included* in the concrete outputs:



But as we saw earlier, this subset-underapproximation is not well suited to underapproximating computation:

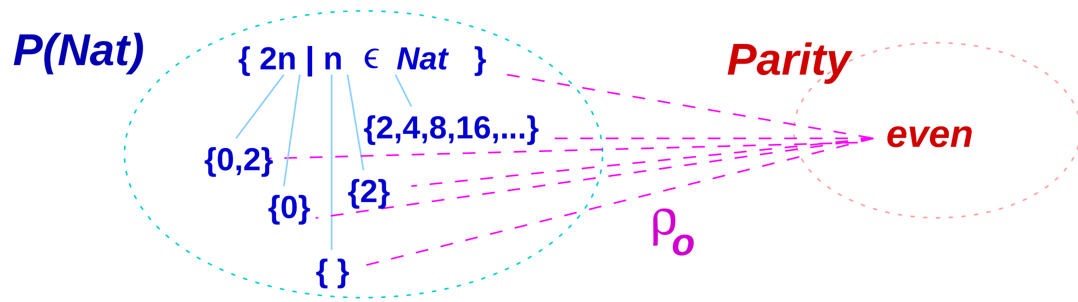
Collatz example: $f = \{0 \to 0, 1 \to 4, 2 \to 1, 3 \to 10, \cdots\}$

For $f^\flat_0 = \overline{\alpha}_u \circ f^* \circ \gamma$,

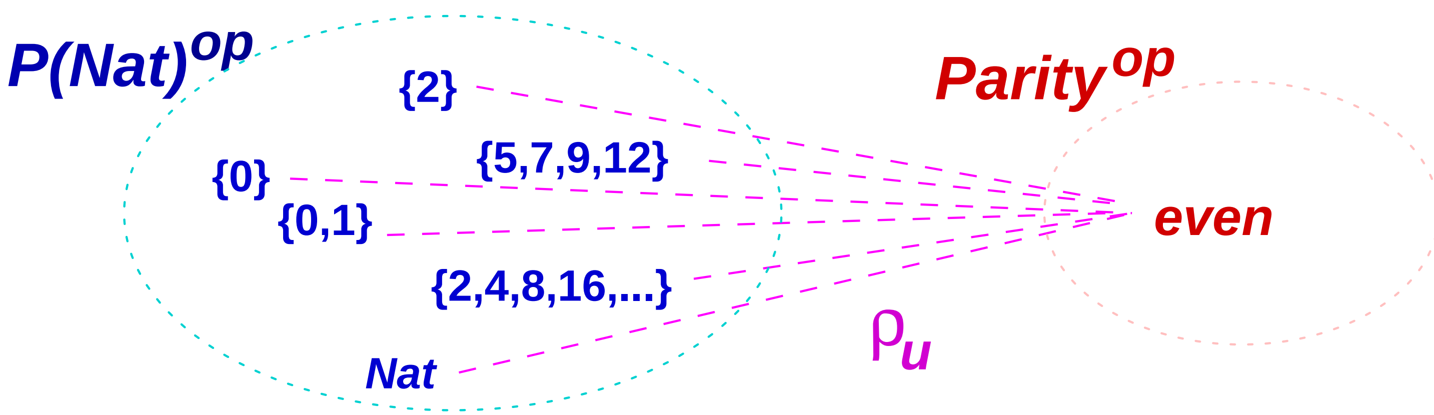
$f^\flat_0(\mathrm{odd}) = \overline{\alpha}_u(f^*\{1, 3, 5, \cdots\} = \overline{\alpha}_u\{4, 10, 16, 22, \cdots\} = \{\mathrm{none}\}$ (**!**)

Under-approximation as existential quantification

If the over-approximating $even \in Parity$ asserts "$\forall even$,"

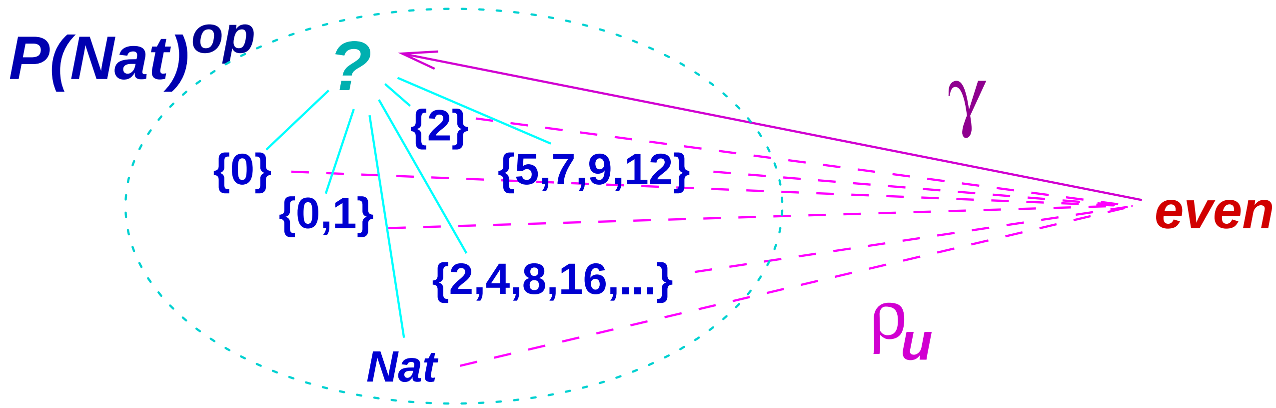


then the under-approximating $even \in Parity^{op}$ *should assert* *"*$\exists even$*"* —there exists an even number in the program's outputs:



For the Collatz example, this lets us define $f^{\flat}(odd) = even$ —for every odd-valued argument, there exists an even-valued answer.

But we cannot define $\gamma : \mathit{Parity}^{op} \rightarrow \mathcal{P}(\mathit{Nat})^{op}$ in the usual way:



There is no best, minimal set that contains an even number.

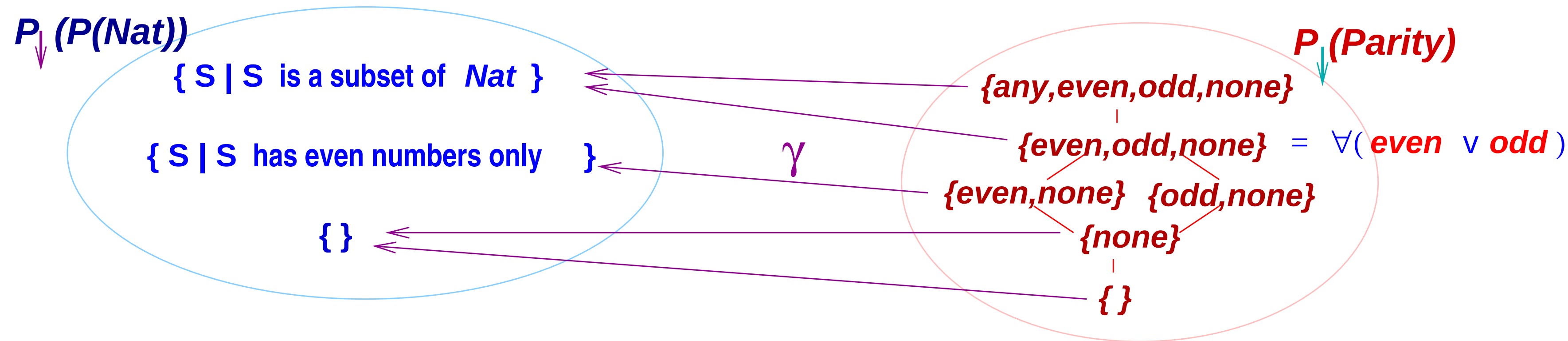
Indeed, *even*'s concretization is not a single set — it is a *set of sets*:

$$\gamma(\mathit{even}) = \{S \in \mathcal{P}(\mathit{Nat})^{op} \mid S \; \rho_u \; \mathit{even}\}$$

This suggests that we work with *power-domains* in both the concrete and abstract domains.

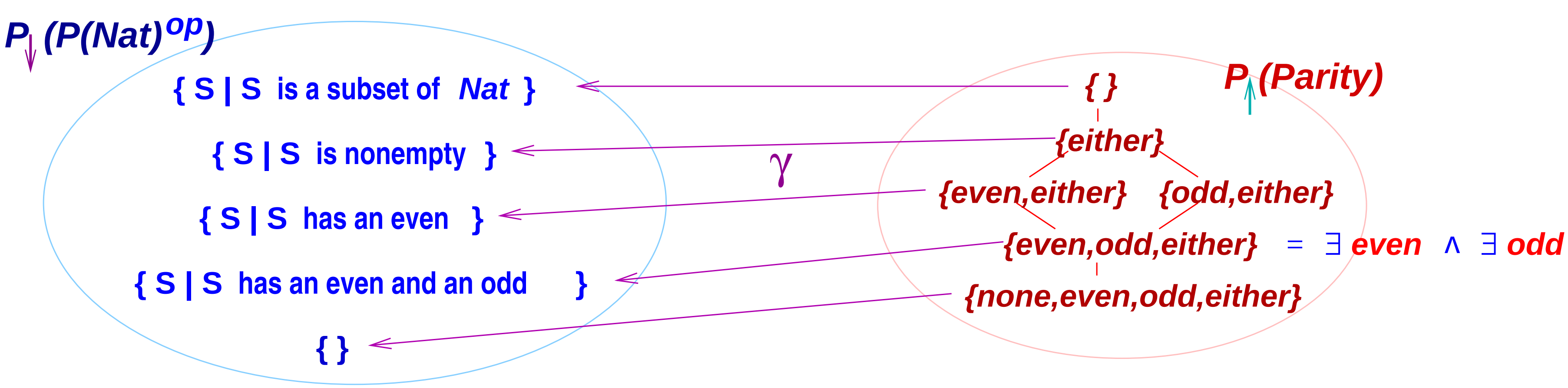
Universal (over-approximating) interpretation: $\{even, odd\}$

asserts $\forall\{even, odd\} \equiv \forall(even \vee odd)$ —all outputs are even- or odd-valued: Use a lower power-domain (lower sets) for the abstract domain.



Existential (under-approximating) interpretation: $\{even, odd\}$

asserts $\exists\{even, odd\} \equiv \exists even \wedge \exists odd$ —there exists an even-valued and an odd-valued output: Use an upper power-domain (upper sets).



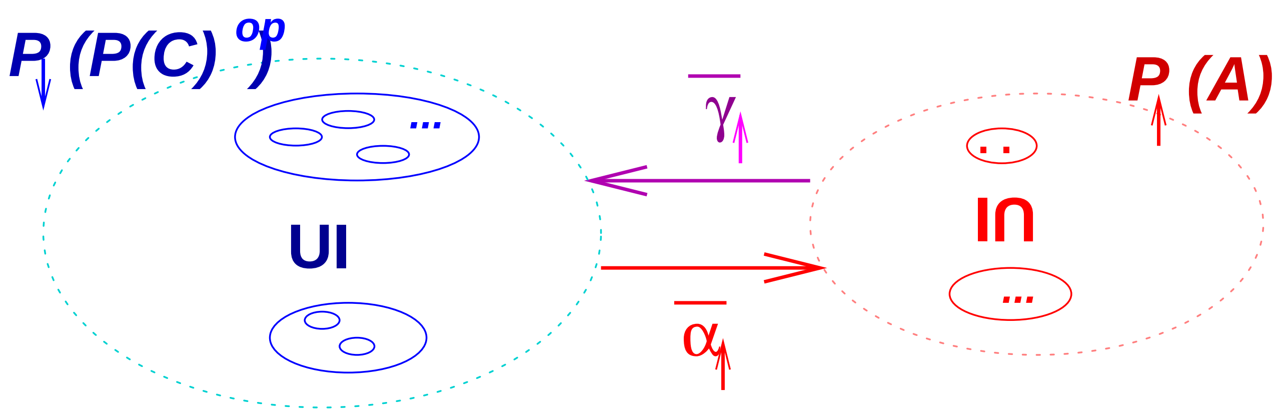
Existential approximation uses the Smyth-powerdomain ordering

For concrete values, $S \subseteq C$, and abstract values, $T \subseteq A$,

$$S \; \rho_{\uparrow} \; T \text{ iff for all } a \in T\text{, there exists } c \in S \text{ such that } c \in \gamma(a)$$

Every $a \in T$ is a witness to some $c \in S$. (Smyth-powerdomain ordering)

$\overline{\gamma}_{\uparrow}(T) = \{S \mid S \; \rho_{\uparrow} \; T\}$ concretizes T to all sets that T "witnesses" — *It is an overapproximation of an underapproximation:*



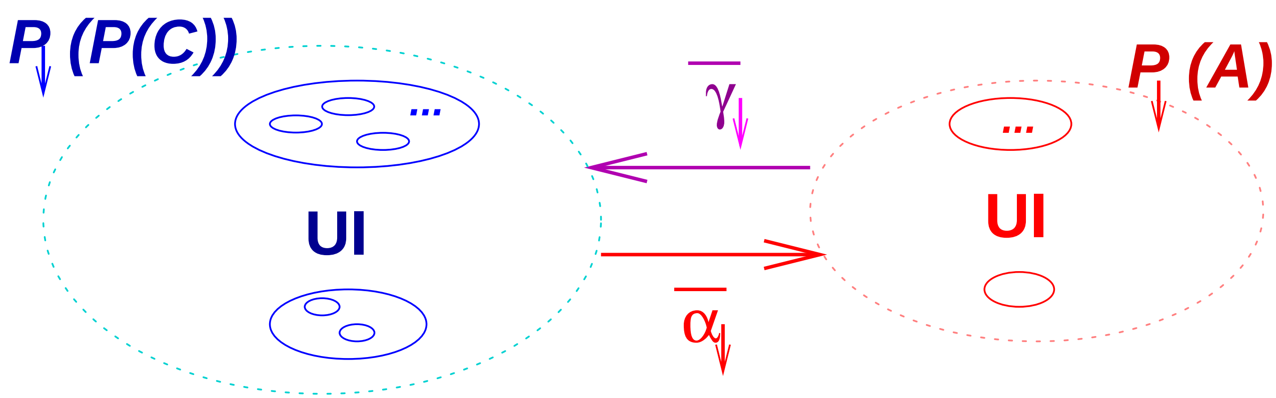
$$\overline{\alpha}_{\uparrow}(\overline{S}) = \cup\{T \mid \text{ for all } S \in \overline{S}, S \; \rho_{\uparrow} \; T\}$$
$$= \{a \mid \text{ for all } S \in \overline{S}, \text{ exists } c \in S, c \in \gamma(a)\}$$

Universal approximation uses the lower-powerdomain ordering

$S \ \rho_{\downarrow} \ T$ iff for all $c \in C$, there exists $a \in A$ such that $c \in \gamma(a)$

Every $c \in S$ is modelled by some $a \in T$ — (lower ("Hoare") powerdomain ordering)

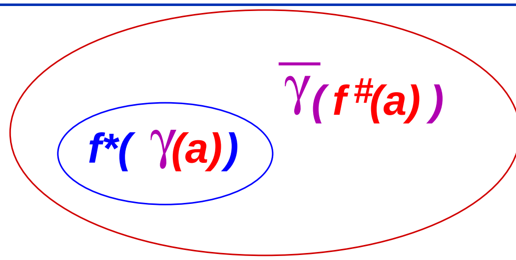
$\overline{\gamma}_{\downarrow}(T) = \{S \mid S \ \rho_{\downarrow} \ T\}$ concretizes T to all the sets covered by T — *It is an overapproximation of an overapproximation:*

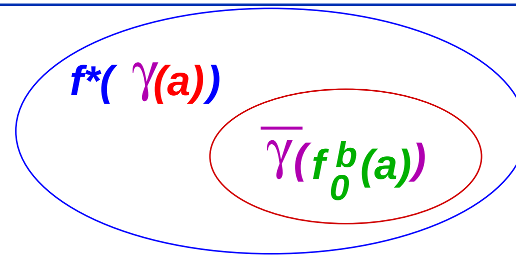


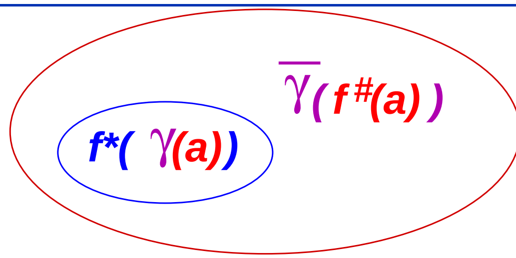
$$\overline{\alpha}_{\downarrow}(S) = \cap\{T \mid \text{ for all } S \in \overline{S}, S \ \rho_{\downarrow} \ T\}$$

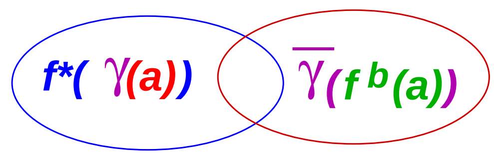
Summary: *Forms of approximation*

For $f : C \to \mathcal{P}(C)$ and its lift, $f^* : \mathcal{P}(C) \to \mathcal{P}(C)$,
for $f^\sharp : A \to \mathcal{P}_\downarrow(A)$, $f_0^\flat : A \to \mathcal{P}_\downarrow(A)$, and $f^\flat : A \to \mathcal{P}_\uparrow(A)$,

<table>
<tr><th></th><th>overapproximation</th><th>underapproximation</th></tr>
<tr><td>set inclusion</td><td>

$f^*(\gamma(a)) \subseteq \overline{\gamma}(f^\sharp(a))$</td><td>

$f^*(\gamma(a)) \supseteq \overline{\gamma}(f_0^\flat(a))$</td></tr>
<tr><td>quantifi cation</td><td>

$f^*(\gamma(a)) \models \forall(f^\sharp(a))$</td><td>

$f^*(\gamma(a)) \models \exists(f^\flat(a))$</td></tr>
</table>

- Read $\forall(f^\sharp(a)) \equiv \forall\{a_0, a_1, \cdots, a_i, \cdots\} \equiv \forall(a_0 \vee a_1 \vee \cdots \vee a_i \vee \cdots)$.
- Read $\exists(f^\flat(a)) \equiv \exists\{a_0, a_1, \cdots, a_i, \cdots\} \equiv \exists a_0 \wedge \exists a_1 \wedge \cdots \wedge \exists a_i \wedge \cdots$.

Summary: *Approximating the logic*

$$\phi ::= p \mid \cdots \mid \Box\phi \mid \Diamond\phi$$

$$[\![p]\!]^A = \overline{\alpha}_u[\![\phi]\!]$$

$$[\![\Box\phi]\!]^A = \widetilde{pre}_{f^\sharp}[\![\phi]\!]^A$$

$$[\![\Diamond\phi]\!]^A = pre_{f^\flat}[\![\phi]\!]^A$$

where $f^\sharp : A \to \mathcal{P}_\downarrow(A)$
and $f^\flat : A \to \mathcal{P}_\uparrow(A)$.

To overapproximate $f : C \to \mathcal{P}(C)$, **use** $f^\sharp_{best} : A \to \mathcal{P}_\downarrow(A)$,

$$f^\sharp_{best} = \overline{\alpha}_o \circ f^* \circ \gamma = \overline{\alpha}_\downarrow \circ (\{\!|\cdot|\!\} \circ f)^* \circ \gamma$$

In this thesis, Dams defined $f^\sharp_{best}(a) = \downarrow\{\alpha\{c'\} \mid c' \in f(c), c \in \gamma(a)\}$.

To underapproximate $f : C \to \mathcal{P}(C)$, **use** $f^\flat_{best} : A \to \mathcal{P}_\uparrow(A)$,

$$f^\flat_{best} = \overline{\alpha}_\uparrow \circ (\{\!|\cdot|\!\} \circ f)^* \circ \gamma$$

In his thesis, Dams defined $f^\flat_{best}(a) = \{a' \mid \gamma(a') \cap f(c) \neq \emptyset$, for all $c \in \gamma(a)\}$.

Dams, Gerth, and Grumberg proved that these definitions soundly validate the most logical properties of f.

We can also validate LTL: $\psi ::= p \mid F\psi \mid G\psi \mid \cdots$

$$[\![p]\!] = \{\pi \in C^\infty \mid \pi_0 \in \gamma(p)\}$$

$$[\![F\psi]\!] = \{\pi \in C^\infty \mid \text{ exists } i \geq 0, \pi_i \in [\![\psi]\!]\}$$

$$[\![G\psi]\!] = \{\pi \in C^\infty \mid \text{ for all } i \geq 0, \pi_i \in [\![\psi]\!]\}$$

Use $f^\sharp$ to generate paths of form, $\pi^\sharp = (a_i)_{i \geq 0}$, such that for all $i \geq 0$, ${\pi^\sharp}_{i+1} \in f^\sharp({\pi^\sharp}_i)$. **Validate**

$$a_0 \models \forall\psi$$ **"for all paths, $\pi^\sharp$ starting with a_0, $\pi^\sharp \models \psi$"**

to prove $c_0 \models \forall\psi$, for all $c \in \gamma(a_0)$, since $f \lhd f^\sharp$.

Use $f^\flat$ to generate paths $\pi^\flat = (a_i)_{i \geq 0}$, such that for all $i \geq 0$, ${\pi^\flat}_{i+1} \in f^\flat({\pi^\flat}_i)$. **Validate**

$$a_0 \models \exists\psi$$ **"there exists a path, $\pi^\flat$, starting with a_0, such that $\pi^\flat \models \psi$"**

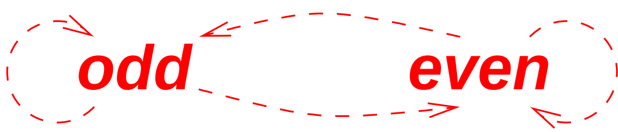
to prove $c_0 \models \exists\psi$, for all $c \in \gamma(a_0)$, since $f^\flat \lhd f$.

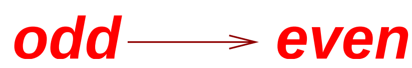
Mixed modalities and mixed models

To prove that transitions from even-valued states never “go far,”

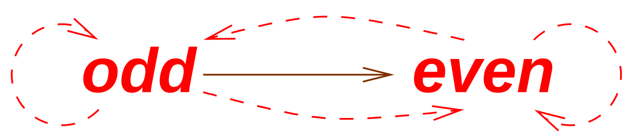
$$even \models^{A} \Box(even \vee \Diamond even)$$

we need more than just a sound overapproximation alone or a sound underapproximation alone:





Working with the two together —as a *mixed model* —we can validate the claim:



Definition: For Kripke structure, (C, f, γ), a *mixed model* is $(A, f^{\flat}, f^{\sharp})$ such that $f^{\flat} \lhd f$ and $f \lhd f^{\sharp}$.

Other extensions

The validation logic can contain repetition/recursion:

$$\phi ::= p \mid \Box\phi \mid \Diamond\phi \mid \cdots \mid \mu Z.\phi \mid \nu Z.\phi \mid Z$$

Sound checking is preserved on the mixed models.

We can define standard modalities, e.g.,

$$AG_{fin}\phi \equiv \nu Z.\phi \wedge \Box Z$$

$$AG\phi \equiv \nu Z.\phi \wedge \Diamond true \wedge \Box Z$$

and use them:

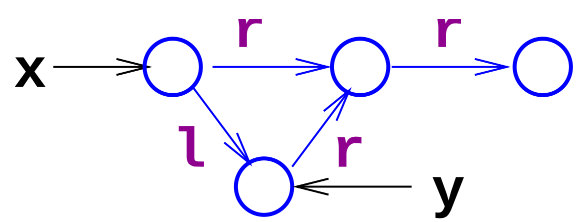
$$even \models^{A} AG(even \vee \Diamond even)$$

Finally, we can have more than one state-transition function, g, and the corresponding logical modalities, $[g]\phi$ and $\langle g\rangle\phi$, as in Hennessy-Milner logic and description logic.

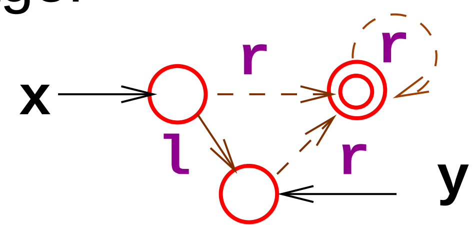
Underapproximating data structure

TVLA models of object networks *[Sagiv,Reps,Wilhelm]*

A collection of data objects,



can be modelled by partitioning the objects based on their properties and abstracting their linkage:



Say we have $\gamma : \mathrm{Prop} \to \mathcal{P}(\mathrm{Object})$, e.g.,
$\mathrm{Prop} = \{\mathrm{x\_pointsTo}, \mathrm{y\_pointsTo}\}$.

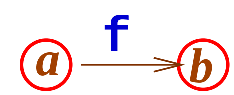
Partition Object by $c \equiv_\gamma c'$ iff (for all $p \in \mathrm{Prop}$, $c \in \gamma(p)$ iff $c' \in \gamma(p)$).

All partitions are nonempty **— distinguish between *singleton* partitions, ○, and *multiple-object* partitions, ◎.**

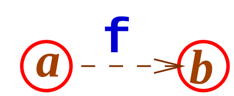
Each fieldname defines a "transition function"

Abstract a field function, $f : C \to C$**, as** $f^\sharp : A \times A \to \{0, 1, 1/2\}$**:**

$f^\sharp(a, b) = 1$: for all $c \in \gamma(a)$, for all $c' \in \gamma(b)$, $f(c) = c'$



$f^\sharp(a, b) = 1/2$: there exist $c \in \gamma(a)$, $c' \in \gamma(b)$, such that $f(c) = c'$



$f^\sharp(a, b) = 0$: otherwise (there are no $c \in \gamma(a)$, $c' \in \gamma(b)$, such that $f(c) = c'$)

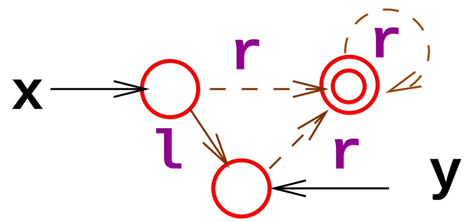
- TVLA is an *over*approximating model, where some nodes and links are *exact*.
- We might rework TVLA's approximations as $f^\flat : A \to \mathcal{P}_\uparrow(A)$ (for 1 values) and $f^\sharp : A \to \mathcal{P}_\downarrow(A)$ (for $1/2$ *and* 1 values).
- $f^\sharp(a, b) = 1$ is a $\forall\forall$-abstraction of f (the usual $f^\flat$ is a $\forall\exists$-abstraction), but $\forall\forall$ *coincides* with $\forall\exists$ in TVLA models, where concrete graphs are deterministic.

Logical properties of the **shape graph**

TVLA uses first-order predicate logic plus transitive closure:

$$\phi ::= p_k(e_i)_{i<k} \mid \cdots \mid \forall x.\phi \mid \exists x.\phi \mid p^+(e_1, e_2)$$

Example: for



$$\exists i.\ x\_pointsTo(i) \wedge \exists j.\ l(i,j) \wedge \neg(\exists k.\ r^+(j,k) \wedge x\_pointsTo(k))$$

holds true: *Starting from* x*'s* l*-field, repeated transitions of the* r*-field never lead back to* x*'s object.*

We could also use a Box-Diamond logic, like that used to state properties of control diagrams, to state properties of the shape graph:

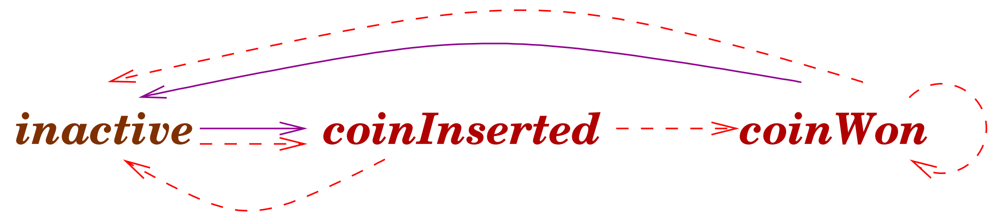
$$x \models \langle l \rangle AG_r \neg x\_pointsTo, \quad \text{where } AG_r\phi = \nu Z.\phi \wedge [r]Z$$

Underapproximation in specification

Modal transition systems *[Larsen,Thomson]*

A reactive system can be specified with an under-over-approximation.

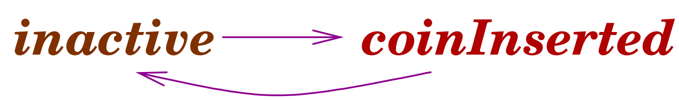
A slot-machine specification:



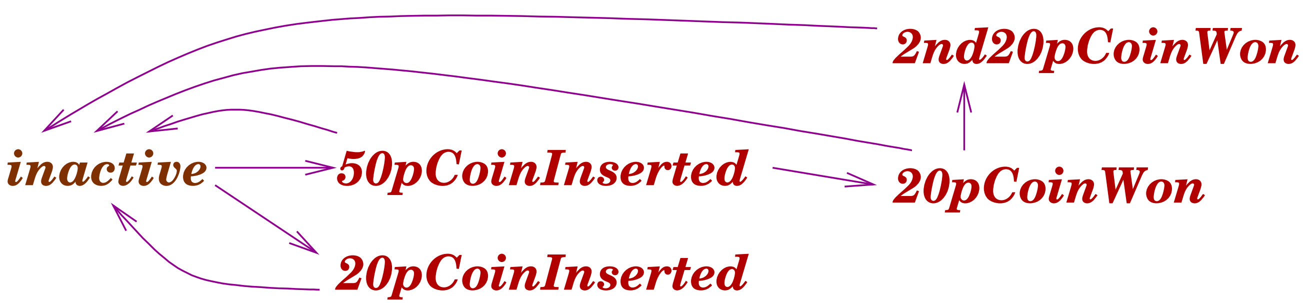
Must (“tight”)-transitions indicate behaviors that the completed machine must possess; *may (“loose”)*-transitions indicate the domain of acceptable (but not required) behaviors.

Two possible implementations:

(i) A machine that keeps all coins:



(ii) A machine that might pay one or two 20p coins for a 50p coin:



Definition: A modal transition system (MTS) is a mixed-transition system, (C, f_{must}, f_{may}), such that $f_{must}(c) \subseteq f_{may}(c)$, for all $c \in C$.

Every must-transition is also a may-transition ("what must be implemented, surely may be implemented").

An MTS can be stepwise refined into an implementation —an STS (this is an MTS where $f_{must} = f_{may}$) —via *refinement*:

Definition: For modelling relation, $\gamma : C' \rightarrow \mathcal{P}(C)$, $(C, f_{must}, f_{may}) \lhd (C', f'_{must}, f'_{may})$ iff $f_{may} \lhd f'_{may}$ and $f'_{must} \lhd f_{must}$.

May-transitions are preserved or deleted, and must-transitions are preserved/increased.

We can use the logics stated earlier to write properties of the MTS.

Description logic *[Baader, et al.]*

Description logic is used to specify knowledge bases:

$$\phi ::= p \mid \phi_1 \sqcup \phi_2 \mid \phi_1 \sqcap \phi_2 \mid \forall R.\phi \mid \exists R.\phi \mid \ \geq n.R \mid \ \leq n.R$$

Read $\forall R.\phi$ as $[R]\phi$ and $\exists R.\phi$ as $\langle R \rangle \phi$. Read $c \models \geq n.R$ iff $|\{c' \mid R(c, c')\}| \geq n$.

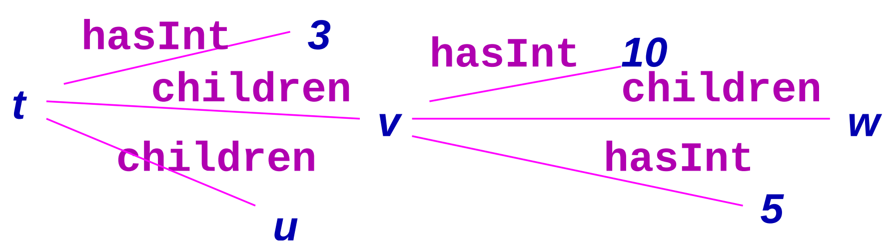
Example: $\text{Tbox}:$ $\text{IntTree} \equiv \ \geq 0.\text{hasInt} \sqcap \forall \text{children}.\text{IntTree}$

$\text{BinTree} \equiv \text{IntTree} \sqcap \ \leq 2.\text{children}$

$\text{Abox}:$ $\text{hasInt}(t, 3)$ $\text{children}(t, u)$ $\text{children}(t, v)$

$\text{hasInt}(v, 10)$ $\text{hasInt}(v, 5)$ $\text{children}(v, w)$

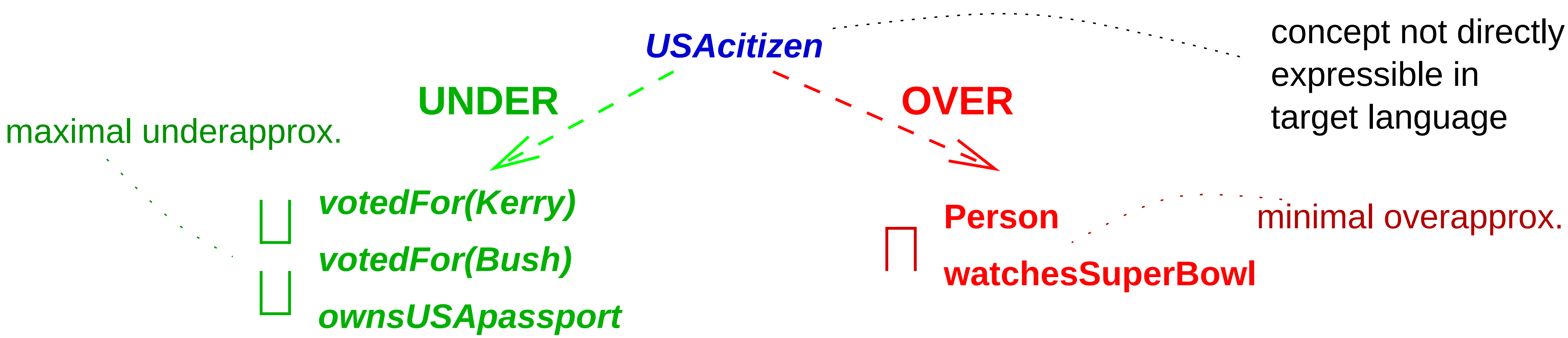
Description logic is a “super Prolog” whose inference engine is a model checker!



Is the example an *over-* or an *under-* specifi cation ?

In his thesis, Ciocoiu uses description logic as a meta-language for *inexact language translation*:

Both source and target languages are given description-logic semantics; translation of a source sentence into the target language produces an *overapproximation* translation and an *underapproximation* translation:

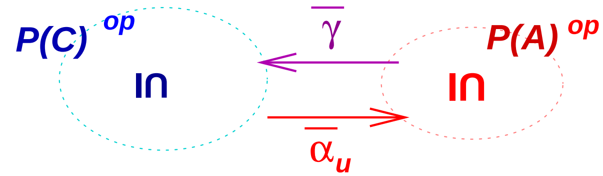


Taken together, the two inexact translations describe the source concept in the target language.

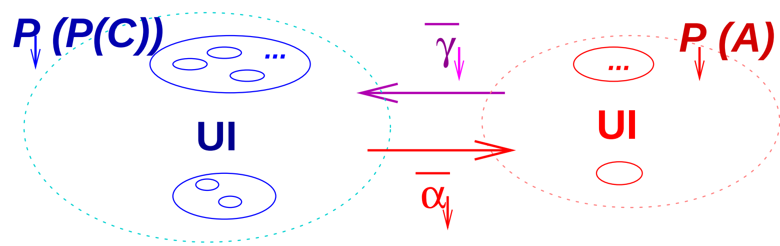
Conclusion

Explaining the slogan: “Overapproximate the computation and underapproximate the logic”

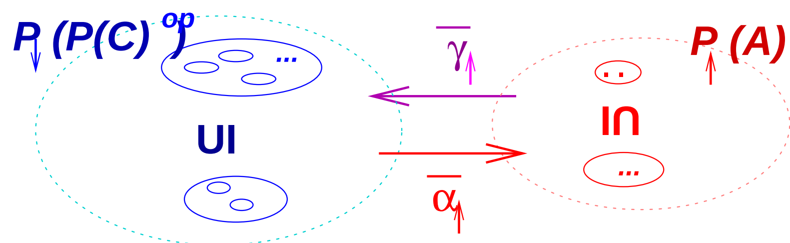
- the *logic* is underapproximated with $\mathcal{P}_{\downarrow}(A)^{op}$:



- a *computation* is approximated for universal properties with $\mathcal{P}_{\downarrow}(A)$:



- a *computation* is approximated for existential properties with $\mathcal{P}_{\uparrow}(A)$:



- *Both* of the previous approximations use concrete domains of form, $\mathcal{P}_{\downarrow}(\mathcal{P}(\cdot))$, making them *over* approximations of the concrete system.

References

Primary:

1. This talk: `www.cis.ksu.edu/~schmidt/papers`
2. E.M. Clarke, O. Grumberg, and D.E. Long. Model checking and abstraction. *ACM TOPLAS* 19-5, (1994).
3. P. Cousot and R.Cousot. Abstract interpretation frameworks. *Journal of Logic and Computation* 2 (1992).
4. P. Cousot and R.Cousot. Higher-order abstract interpretation. IEEE Conf. on Computer Languages, 1994.
5. D. Dams. Abstract interpretation and partition refinement for model checking. PhD thesis, Univ. Eindhoven, 1996.
6. D. Dams, R. Gerth, O. Grumberg. Abstract Interpretation of Reactive Systems. *ACM TOPLAS* 19 (1997).
7. M. Sagiv, T. Reps, R. Wilhelm. Parametric Shape Analysis via 3-Valued Logic. *ACM TOPLAS* 24-3 (2002).
8. D.A. Schmidt. A calculus of logical relations for over- and underapproximating static analyses. *Science of Comp. Prog.*, in press.

Secondary:

1. F. Baader, et al. *The Description Logic Handbook.* Cambridge Univ. Press 2003.
2. M. Ciocoiu, Ontology-based translation. Ph.D. thesis, Univ. North Carolina, 2001.
3. M. Huth, R. Jagadeesan, D. Schmidt. Modal transition systems: a foundation for three-valued program analysis, ESOP 2002. Also, A domain equation for refinement of partial systems, *J. MSCS*, in press.
4. O. Grumberg, F. Lerda, O. Strichman, M. Theobald. Proof-guided underapproximation-widening for multi-process systems. ACM POPL 2005.
5. K. Larsen and B. Thomsen. A modal process logic. 3d IEEE LICS Symp., 1988.
6. C. Păsăreanu, R. Pelánek, W. Visser. Concrete model checking with abstract matching and refinment. CAV 2005.
7. G. Plotkin. Domain theory. Lecture notes, Univ. Pisa 1982.