UPDR + Theories + (Implicit) Index predicates

1 Introduction

We introduce a class of symbolic transition systems for modelling parameterized systems, similar to MCMT/Ivy. We consider two theories; a theory $\mathcal{T}_{\mathcal{I}} = (\Sigma_I, \mathcal{C}_I)$, called the *index* theory, whose class of models includes all possible finite (but unbounded) structures. This is the theory we use for quantified variables.

In addition, we consider a theory of elements $\mathcal{T}_E = (\Sigma_E, \mathcal{C}_E)$, used to model the data of the system. Relevant examples consider as $\mathcal{T}_E$ the theory of an enumerated datatype, or linear arithmetic (integer or real). Note that this theories can have infinite models.

Then, with **arrays** we denote variables of the theory whose models are set of total functions from a model of $\mathcal{T}_I$ to a model of $\mathcal{T}_E$.[1]

Definition 1 *In the following, we consider symbolic transition systems* $S = (X, I(X), T(X, X'))$ *where:*

- X *are arrays;*
- $I(X), T(X, X')$ *are first-order formulae possibly containing quantifiers over variables of sorts* $\mathcal{T}_I$*.*

The *Invariant Problem* we consider is the problem of proving (or disproving) that a given formula ϕ, possibly containing quantified variables of sort $\mathcal{T}_I$, is an invariant for S.

Example 1. Let's model a simplified version of the Bakery algorithm.
Let T_I be the theory of finite sets with equality. As for T_E, we need two theories: an enumerated datatype with values $\{idle, wait, critical\}$, and Linear Integer arithmetic.
We have four state variables: one array *state* with values in $\{idle, wait, critical\}$, one array *token* with values in $\mathbb{N}$, and integer variables *next_token* and *to_serve* with values in $\mathbb{N}$.
The initial formula of the system is:

$$\forall i. state[i] = idle \land token[i] = 0 \land next\_token = 1 \land to\_serve = 1$$

The transition formula is the disjunction of the three formula; the first one models a process that goes from idle to wait and takes a ticket:

$$\exists i. \big(state[i] = idle \land state'[i] = wait \land t'[i] = next\_token \land next\_token' = next\_token + 1 \\ \land to\_serve' = to\_serve \land \forall j. (j \neq i \rightarrow s'[j] = s[j] \land t'[j] = t[j])\big).$$

[1] The equality relation among arrays should be replaced by the corresponding universal axioms; i.e. if x, y are arrays, instead of $x = y$ one should use $\forall j. x[j] = y[j]$.

Then, a process enters the critical section if its ticket is selected:

$$\exists i.\big(state[i] = wait \land state'[i] = crit \land t[i] = to\_serve \land next\_token' = next\_token \\ \land\, to\_serve' = to\_serve \land \forall j.(j \neq i \rightarrow s'[i] = s[i]) \land (\forall j.t'[j] = t[j])\big).$$

Finally, a process exits from the critical state and reset its ticket:

$$\exists i.\big(state[i] = critical \land state'[i] = idle \land t'[i] = 0 \land next\_token' = next\_token \\ \land\, to\_serve' = to\_serve + 1 \land \forall j.(j \neq i \rightarrow s'[j] = s[j] \land t'[j] = t[j])\big).$$

The property we want to prove is mutual exclusion:

$$\forall i, j.(i \neq j \rightarrow \neg(state[i] = crit \land state[j] = crit)).$$

2 Implicit Indexed Abstraction

Given a system S and a candidate invariant ϕ, we denote as $\mathcal{P}(\underline{i})$ the set of atoms/index predicates (possibly containing free variables of sort T_I) **of sort** $\mathcal{T}_E$ which appears in the initial formula and in ϕ.
The idea for the abstraction is to replace the system with an abstract version which contains only predicates over (quantified) variables of sort $\mathcal{T}_I$, which by hypothesis has the finite model property.

Example 2. In the bakery example, we have:

$$\mathcal{P}(i_1) = \{state[i_1] = 0, t[i_1] = 0, next\_token = 1, to\_serve = 1, state[i_1] = crit\}$$

Note that the predicate '$i_1 \neq i_2$', which appears as an atom in the formula ϕ, is not considered in $\mathcal{P}$ since it is a predicate over the signature of $\mathcal{T}_I$. $\square$

The idea of implicit abstraction is to represent the simulation between abstract states and concrete state by means of logical formulae. To do that, define

Definition 2 *Given $\mathcal{P}(\underline{i})$ a set of index predicates, we define*

- $X_{\mathcal{P}(\underline{i})}$ *is a set of fresh predicates, one for each element $p(\underline{i}, X[\underline{i}]) \in \mathcal{P}(\underline{i})$, and with the same ariety:*

  $$X_{\mathcal{P}(\underline{i})} = \{x_{p(\underline{i},X[\underline{i}])}(\underline{i}) \mid p(\underline{i}, X[\underline{i}]) \in \mathcal{P}(\underline{i}), x_{p(\underline{i},X[\underline{i}])} \textit{ fresh predicate of ariety } \#\underline{i}\};$$

- *the formula*

  $$H_{\mathcal{P}}(X_{\mathcal{P}}, X) = \forall \underline{i}.(\bigwedge_{p(\underline{i},X[\underline{i}])\in\mathcal{P}} x(\underline{i})_{p(\underline{i},X[\underline{i}])} \leftrightarrow p(\underline{i}, X[\underline{i}]))$$

- *the formula*

  $$EQ_{\mathcal{P}}(X, X') = \forall \underline{i}.(\bigwedge_{p(\underline{i},X[\underline{i}])\in\mathcal{P}} p(\underline{i}, X[\underline{i}]) \leftrightarrow p(\underline{i}, X'[\underline{i}]))$$

Also, if ψ is a formula, we denote with $\hat{\psi}$ the formula obtained by the substituing each of its atoms occurring in $\mathcal{P}(\underline{i})$ with the ones in $X_{\mathcal{P}(\underline{i})}$, i.e.

$$\hat{\psi} = \psi[\mathcal{P}(\underline{i})/X_{\mathcal{P}(\underline{i})}]$$

.

Example 3. In the bakery example, given

$\mathcal{P}(i_1) = \{state[i_1] = idle, t[i_1] = 0, next\_token = 1, to\_serve = 1, state[i_1] = crit\}$,

we have a set of predicates

$X_{\mathcal{P}(i_1)} = \{x_{state[i_1]=idle}(i_1), x_{t[i_1]=0}(i_1), x_{next\_token=1}, x_{to\_serve=1}, x_{state[i_1]=crit}(i_1)\}$

a set of variables

$$X = \{state, token, next\_token, to\_serve\}.$$

$H_{\mathcal{P}}(X_{\mathcal{P}}, X)$ is equal to:

$$\begin{aligned}\forall i_1.(x_{state[i_1]=idle}(i_1) \leftrightarrow state[i_1] = idle) \wedge \forall i_1.(x_{token[i_1]=0}(i_1) \leftrightarrow token[i_1] = 0)\\ \wedge\, x_{next\_token=1} \leftrightarrow next\_token = 1 \wedge x_{to\_serve=1} \leftrightarrow to\_serve = 1 \wedge\\ \forall i_1.(x_{state[i_1]=critical}(i_1) \leftrightarrow state[i_1] = critical)\end{aligned}$$

□

In the following, when working in the abstract space of S, the critical step for UPDR is repeatedly checking whether a candidate countermodel ψ is inductive relative to the frame F. Frames will be defined as a set of universal formulae (except for the initial frame F_0 shich is $\hat{I}$). The inductivness check, if encoded naively, would require the explicit construction of the abstract version of T. The key insight underlying implicit abstraction is to perform the check without actually computing the abstract version of T, but by encoding the simulation relation with a logical formula. This is done by checking the (possibly quantified over T_I) formula:

$$\begin{aligned}AbsRelInd(F, T, \psi, \mathcal{P}) = F(X_{\mathcal{P}}) \wedge \psi(X_{\mathcal{P}}) \wedge H_{\mathcal{P}}(X_{\mathcal{P}}, X) \wedge H_{\mathcal{P}}(X'_{\mathcal{P}}, X')\\ \wedge EQ_{\mathcal{P}}(X, \bar{X}) \wedge T(\bar{X}, \bar{X}') \wedge EQ_{\mathcal{P}}(\bar{X}', X') \wedge \neg\psi(X'_{\mathcal{P}})\end{aligned}$$

3 Diagrams

The last notion we need to adapt UPDR is the one of diagrams. Diagrams are a method (actually the '*best*' method) to represent a **finite** first-order model with an existentially quantified formula.

Note that a formula over the predicates X_P has only finite models, since the theory T_I has only finite models. Indeed, a model for a formula $\phi(X_P)$ consists of a finite structure M and a valuation I of the predicates.

Definition 3 *(Diagrams) Given a finite model (M, I) over X_P, the diagram of the model is a formula over X_P constructed as follows:*

- *For every element $m \in M$, a fresh variable i is introduced.*
- *$\phi_{distinct}$ is a conjunction of inequalities of the form $i_a \neq i_b$ for every pair of distinct elements (m_a, m_b)in the model.*
- *$\phi_{constant}$ is a conjunction of equalities of the form $i = c$ for every constant symbol c such that $(M, I) \models c = i$*
- *ϕ_{atomic} is a conjunction of atomic formulae that include for every predicate $x_p \in X_P$ the atomic formula $x_p(i_1, \ldots, i_n)$ if $(M, I) \models x_p(m_1, \ldots, m_n)$ and $\neg x_p(i_1, \ldots, i_n)$ otherwise.*

Thus, the diagram of the model is the formula

$$\exists i_1, \ldots, i_n, \phi_{distinct} \wedge \phi_{constat} \wedge \phi_{atomic}.$$

4 Pseudocode

Algorithm 1: UPDR + IA

```
Input: $S = (X, I(X), T(X, X')), \phi(X)$
$\mathcal{P}(i) = \{$set of atoms occuring in $I, \phi\}$
if not $\hat{I} \wedge H_P \models \hat{\phi}$:
    return cex  # cex in initial state
$F_0 = \hat{I}$
$k = 1, F_k = \top$
while True:
    while $F_k \wedge H_P \wedge \neg\hat{\phi}$ is sat:
        extract $\psi$ a diagram from the model over $X_P$
        if not $RecBlock(\psi, k)$:
            # proof of no universal invariant over $X_P$
            # an abstract counterexample $\pi$ is provided
            if not $Concretize(\pi)$:
                $\mathcal{P} = \mathcal{P} \cup Refine(\pi)$
            else:
                Return Cex
    # if no models, add new frame:
    $k = k + 1, F_k = \top$
    # propagation phase
    for $i = 1, \ldots, k - 1$:
        for each diagram $\psi \in F$:
            if $AbsRelInd(F_i, T, \psi, \mathcal{P})$ is unsat:
                add $\psi$ to $F_{i+1}$
            if $F_i = F_{i+1}$:
                Return property proved!
```

Algorithm 2: *RecBlock*(ψ, N)

if $N = 0$
    Return cex
While $AbsRelInd(F_{i-1}, T, \neg\psi, \mathcal{P})$ is sat:
    extract a diagram ψ' from the model over X_P
    if not $RecBlock(\psi', N-1)$:
        Return cex
$g = Generalize(\neg\psi, N)$ # *e.g. dropping conjuncts of the diagram as long as the query is unsat;*
add g to $F_1, \ldots, F_N$
Return True

Example 4. **STILL INCOMPLETE** We give an example of the procedure over the bakery algorithm. We have:

$$\mathcal{P}(i_1) = \{state[i_1] = 0, t[i_1] = 0, next\_token = 1, to\_serve = 1, state[i_1] = crit\}$$

$$X_P = \{x_{state[i_1]=idle}(i_1), x_{t[i_1]=0}(i_1), x_{next\_token=1}, x_{to\_serve=1}, x_{state[i_1]=crit}(i_1)\}$$

$$\hat{I} = \forall i.\big(x_{state[i_1]=idle}(i_1) \wedge x_{t[i_1]=0}(i_1) \wedge x_{next\_token=1} \wedge x_{to\_serve=1}\big)$$

$$\hat{\phi} = \forall i, j.\big(i \neq j \to \neg(x_{state[i_1]=crit}(i) \wedge x_{state[i_1]=crit}(j)\big)$$

$H_{\mathcal{P}}(X_{\mathcal{P}}, X)$ is equal to:

$$\begin{aligned}\forall i_1.(x_{state[i_1]=idle}(i_1) \leftrightarrow state[i_1] = idle) \wedge \forall i_1.(x_{token[i_1]=0}(i_1) \leftrightarrow token[i_1] = 0)\\ \wedge\, x_{next\_token=1} \leftrightarrow next\_token = 1 \wedge x_{to\_serve=1} \leftrightarrow to\_serve = 1 \wedge\\ \forall i_1.(x_{state[i_1]=critical}(i_1) \leftrightarrow state[i_1] = critical)\end{aligned}$$

It is easy to see that $\hat{I} \wedge H_P \wedge \neg\hat{\phi}$ is unsat. Therefore, we have $F_0 = \hat{I}$ and $F_1 = \top$. We enter the main loop, and consider models of $\top \wedge H_P \wedge \neg\hat{\phi}$; this formula is satisfiable, and we consider the projection of a model over X_P. Suppose that it is given by:

- A finite structure $M = \{a, b\}$ with $a \neq b$
- $x_{next\_token=1}$ is true in the model, $x_{to\_serve=1}$ is false;
- predicate $x_{state[i_1]=idle}$ do not hold for a, b but $x_{state[i_1]=crit}$ does;

so the corresponding diagram is

$$\begin{aligned}\psi = \exists i_1, i_2.\big(&i_1 \neq i_2 \wedge x_{next\_token=1} \wedge \neg x_{to\_serve=1} \wedge \neg x_{state[i_1]=idle}(i_1)\\ &cube \wedge \neg x_{state[i_1]=idle}(i_2) \wedge x_{state[i_1]=crit}(i_1) \wedge x_{state[i_1]=crit}(i_2)\big).\end{aligned}$$

We now call $RecBlock(\psi, 1)$, and therefore consider $AbsRelInd(F_0, T, \neg\psi, \mathcal{P})$. This in unsat, so we can add (a generalization of) $\neg\psi$ to F_1. Now, $F_1 \wedge H_P \wedge \neg\hat{\phi}$

is no longer sat, so we introduce a new frame F_2. (Let's skip propagation for the moment). At this point, $F_2 \wedge H_P \wedge \neg\hat{\phi}$ is satisfiable, and it is possible to have a diagrm ψ such that $RecBlock(\psi, 2)$ recusevely calls $Rec(\psi', 1)$ and again it calls $Rec(\psi'', 0)$. This is an abstract counterexample, and therefore this is a proof that no universal invariant exists over the set of predicates $\mathcal{P}(i_1)$. Suppose however that the corresponding BMC query is unsat: the counterexample is spurious and we need to refine our abstraction.

Suppose now that some oracle tell us that adding e.g. the predicate $t[i_1] = \mathit{to\_serve}$ will block the counterexample; with this new set of predicate we can obtain a new diagram and make progress in the algorithm.

5 Combination with invisible invariants

5.1 Ground instances

Copy and paste from previous work

Traditionally, the exploration of finite instances has always been recognized as a source of helpful heuristics, especially in the verification of parameterized systems.
In the following, we denote with n an integer, and with $\underline{c} = c_1, \ldots, c_n$ a set of fresh constants of index sort. These will be frozen variables of the ground instance, i.e. we will implicitly consider a constraint $\underline{c}' = \underline{c}$ as a conjunction of the transition formula; moreover, they will be also considered all implicitly different.

Definition 4 *In the following, if* $\phi = Q_1 i_1, ..., Q_m i_m.\phi'(\underline{i}, \underline{x}[\underline{i}])$, *with* $Q_j \in \{\forall, \exists\}$ *is a formula with quantifiers of only sort* $\mathcal{T}_I$, *we denote* $\phi^n(\underline{c}, \underline{x}[\underline{c}])$ *the ground formula obtained by obtained by recursively applying the rules*

$$\forall i.\phi'(i, \underline{x}[i]) \to \bigwedge_{i=1}^{n} \phi'(c_i, \underline{x}[c_i])$$

$$\exists i.\phi'(i, \underline{x}[i]) \to \bigvee_{i=1}^{n} \phi'(c_i, \underline{x}[c_i])$$

Definition 5 *Given* $S = (\underline{x}, \iota(\underline{x}), \tau(\underline{x}, \underline{x'}))$ *a transition system and* n *an integer, the* ground instance *of* S *of size* n, *denoted with* S^n, *is obtained in the following way:*

- *for each function symbol* a *in* Σ *whose codomain type is* $\mathcal{T}_I$, *consider the formula*
$$\forall i_1, \ldots, i_m \exists j.a(i_1, \ldots, i_m) = j^{2},$$
where m *is the arity of* a, *and* $i_1, \ldots, i_m, j$ *are fresh variables of appropriate sort;*

[2] These are 'cardinality axioms', used to restrict the values of functions in appropriate models.

– *add the formulae generated in this way in conjuction to the initial formula ι and the transition formula τ;*
– *Instantiate all the quantifiers in the modified formulae with $\underline{c}$, thus obtaining a* ***quantifier-free*** *transition system*

$$S^n = \big(\underline{c} \cup \underline{x}, \iota^n(\underline{c}, \underline{x}[\underline{c}]), \tau^n(\underline{c}, \underline{x}[\underline{c}], \underline{x}'[\underline{c}])\big).$$

We observe that a state of S^n is given by: (i) an assignment of $\underline{c}$ to a finite model of cardinality n of $\mathcal{T}_I$, and (ii) an interpretation of the state variables as functions from that model to a model of $\mathcal{T}_E$. Note that even if the models of T_I have finte cardinality, the set of states of S^n can be infinite, since $\mathcal{T}_E$ could have an infinite model, e.g. if integer or real variables are in the system. Nevertheless, the system can be model checked efficiently by modern symbolic SMT techniques like IC3IA.

5.2 Exploring ground instances and UPDR

We want to use the exploration of ground instances before using **Algorithm 1**, in the same spirit of the work based on the invisible invariant.

So, before starting the UPDR algorithm, we choose an n, and we compute S^n. We then check with a model checker (e.g. IC3IA) whether $S^n \models \phi^n$. If this does not hold, we have already a counterexample, and we do not need to run UPDR. Otherwise, we obtain a set of quantified lemmas $\{\eta_j(X)\}_{j\in J}$ from a generalization of the inductive invariant obtained in size n.

Now, we can start the algorithm, but we try to prove the stronger property $\phi(X) \wedge \bigwedge_{j\in J} \eta_j(X)$. If, during the blocking phase, we find some counterexamples which do not model $\neg\phi(X)$, but only some η_i, we simply remove the latter and continue the algorithm.

Note also that, at any point, we could decide to explore a new instance S^m to find a new set of formulae$\{\mu_k\}_{k\in K}$ to be added to the current set of candidate lemmas.

The advantage of having lemmas is two-fold:

1. by strengthening the property, we hope to have more chances of achieving convergence sooner;
2. lemmas could have new predicates,that we hope can improve the abstraction.

Of course, the possible disadvantages are the ‘duals’:

1. with too many wrong lemmas, UPDR will takes lot of times in discovering counterexamples to them;
2. a large number of useless predicates could make the abstraction not scalable.

5.3 Concretizing counterexamples and refinement

Something than is still missing is a procedure for concretizing abstract counterexample, and a procedure for refine the set of predicates.

UPDR finds abstract counterexamples as sequences of diagrams $\psi_0(X_{\mathcal{P}}), \ldots, \psi_k(X_{\mathcal{P}})$. In order to check whether such a sequence correspond to a concrete counterexample, we could check the satisfiability of the concrete unrolling – called in the following $Unroll(X^0, \ldots, X^n)$ – by replacing atoms in the diagram with their non-abstracted version :

$$\psi_0[X^0_{\mathcal{P}}/\mathcal{P}^0] \wedge \bigwedge_{i=1,\ldots,k} T(X^{i-1}, X^i) \wedge \psi_i[X^i_{\mathcal{P}}/\mathcal{P}^i]. \tag{1}$$

However, a naive call to an SMT solver may result in an inefficient procedure, since the formula latter is possibly 'heavily' quantified. What we could do instead is considering a ground instance of Equation (1) in a size n', with $n' > n$. That is to say, instead of try to unroll the counterexample in all the instances of the paramterezied system, we try to block it only up to a size n'. However, how should we choose such n'?

Since a diagram ψ_i is an existential closed formula which represents an index model with cardinality n_i, where n_i is the number of existentially quantified variables in ψ_i, we need to chose a size able to represent at least all the diagrams in the abstract counterexample. Therefore, a possible choice is $n' = max\{n_i | i = 1, \ldots, k\}$. In this way, we can consider $Unroll^{n'}(X^0, \ldots, X^n)$ which is equivalent to

$$\psi_0^{n'}[X^0_{\mathcal{P}}/\mathcal{P}^0] \wedge \bigwedge_{i=1,\ldots,k} T^{n'}(X^{i-1}, X^i) \wedge \psi_i^{n'}[X^i_{\mathcal{P}}/\mathcal{P}^i].$$

which is a **quantifier-free** formula (all quantifiers are instantiated in $c_1, \ldots, c_{n'}$) represending a BMC of the systems with up to n' elements in the index models. If the latter is sat, we have a real counterexample. Othewise, we can compute an interploant sequence in the usual way. Such a sequence will contains predicates of the form $p(c_1, \ldots, c_{n'}, X[c_1, \ldots, c_{n'}])$. We then can add to $\mathcal{P}(\underline{i})$ the predicates $p(i_1, \ldots, i_{n'}, X[i_1, \ldots, i_{n'}])$.

Note however that such a refinement is enough to rule out the concrete counterexample represented by $Unroll^{n'}(X^0, \ldots, X^n)$, but it is not guaranteed to rule out (1): that is, the counterexample can occur again, and we report unkown: no universal invariant exists over $\mathcal{P}(\underline{i})$, and we fail in increasing the set of predicates.

5.4 Pseudocode/2

Algorithm 3: UPDR + IA + Invisible Invariants

```
Input: $S = (X, I(X), T(X, X')), \phi(X), n$
**if** $S^n \models \phi^n$:
    Compute $\{\eta_j\}_{j \in J}$ a set of lemmas
**else**:
    **return** cex
$\mathcal{P}(i) = \{$*set of atoms occuring in* $I, \phi, \{\eta_j\}_{j \in J}\}$
**while** $\hat{I} \wedge H_P \wedge \neg(\hat{\phi} \wedge \bigwedge_j \hat{\eta}_j)$ is sat:
    let $\sigma$ be the model over $X_P$
    **if** $\sigma \models \neg\hat{\phi}$:
        **return** cex  # *cex in initial state*
    **else**:
        remove all lemmas $\eta_i$ such that $\sigma \models \neg\eta_i$
# *start main loop*
$F_0 = \hat{I}$
$k = 1, F_k = \top$
**while** True:
    **while** $F_k \wedge H_P \wedge \neg\hat{\phi}$ is sat:
        extract $\psi$ a diagram from the model over $X_P$
        **if not** $RecBlock(\psi, k)$:
            # *proof of no universal invariant over* $X_P$
            # *an abstract counterexample* $\pi$ *is provided*
            $n'$ = max num of exist. quantified vars in diagrams in $\pi$
            **if not** $Concretize(n', \pi)$:
                $Refine(\pi)$ = extract a set of predicates from interpolants
                **if** $Refine(\pi) \subset \mathcal{P}$:
                    **return** FAIL
                **else**:
                    $\mathcal{P} = \mathcal{P} \cup Refine(\pi)$
            **else:**
            # *a real cex is found*
                let $\sigma$ be the model over $X_P$
                **if** $\sigma \models \neg\hat{\phi}$:
                    **return** cex  # *cex in initial state*
                **else**:
                    remove all lemmas $\eta_i$ such that $\sigma \models \neg\eta_i$
    # *if no models, add new frame:*
    $k = k + 1, F_k = \top$
    # *propagation phase*
    **for** $i = 1, \ldots, k - 1$:
        **for each diagram** $\psi \in F$:
            **if** $AbsRelInd(F_i, T, \psi, \mathcal{P})$ is unsat:
                add $\psi$ to $F_{i+1}$
            **if** $F_i = F_{i+1}$:
                **Return** property proved!
```