Verification : Summary

Marius

07/01

LTL and FOL

Definition

TL(AP,SU,SS) is a temporal logic with all the past and future quantifiers.
FO(AP,¡) is the first-order logic.

Intuition

$w, i \models \phi \mathbf{SS} \psi \Leftrightarrow$ Il y a eu un instant passé auquel ψ était vrai et après lequel ϕ était vrai jusqu'à aujourd'hui.
$w, i \models \phi \mathbf{SU} \psi \Leftrightarrow$ Il y aura un instant futur auquel ψ sera vrai et jusqu'auquel ϕ sera vrai.
Les operateurs **SS** and **SU** permettent de recréer tout les quantificateurs usuels.

Theorems

TL $\subseteq$ FO [Using the model $\mathbb{N}$]

CTL and MSO

Definition

CTL***(AP,SU)** is a temporal logic enabling path quantifiers **E** and **A** .
CTL(AP,SU) is a temporal logic resticting CTL^* where all path quantifiers are followed by exactly one temporal operator.
MSO(AP,¡) is the monadic second-order logic [enabling quantifying over sets]

Intuition

$\mathcal{M}, \lambda \models \phi \Leftrightarrow \phi$ est vrai dans le chemin λ.

Theorems

CTL $\subseteq$ MSO [Using the model $\mathbb{N}$]
CTL* $\subseteq$ MSO [Using the model $\mathbb{N}$]

Buchi Automata

Definition

Büchi-automaton is NFA where a ω-word is recognized if it goes infinitely many times through F.
General Büchi-automaton is a BA where a ω-word is recognized if it goes infinitely many times through all the F_i.

Some properties

Büchi-automata are closed under boolean operations.
[intersection : go from a copy of the cartesian product to another when meeting a target state]
$LTL \subseteq L(BA)$
Büchi-automata and General Büchi-automata where are the same
Büchi-automata and Deterministic Büchi-automata are NOT the same

CTL* $\subseteq$ MSO [Using the model $\mathbb{N}$]

Partial Order Reduction : Definition

Definition

A **labelled Kripke structure** is extended with actions.
An **independence relation** over actions is irreflexive, symmetric and confluent [if both actions can be chosen, they can be done in both orders]
A set of actions is**invisible** if it never change the truth of a AP.
The **stuterring-equivalence** of sequences is "equality modulo multiplicity"

Some properties

Any LTL without **X** is invariant under stuterring.

Partial Order Reduction : General Idea

Ample set method

Idea : build a set of explored actions $red(s)$ following conditions :
1. Keep at least one action by state when possible [Avoid adding deadlocks]
2. Actions depending on $red(s)$ occurs after $red(s)$ in a concrete path starting with s.
2bis. In particular, all actions not from $red(s)$ are independent with all actions from $red(s)$
3. Keep all actions or only an invisible subset [Preserve stuttering equivalence].
4. An abstract action in a state s in a cycle is in $red(cycle)$ [No starving].

Fair Kripke structure and complexity

Definition

A **fair kripke structure** is extended states set visited infinitely often.
Can be simulated in CTL* with $\bigwedge GFF_i$
Cannot be simulated in CTL.

Complexity

LTL satisfiability is **PSPACE**-complete.
CTL* model checking is **PSPACE**-complete.
CTL model checking is decidable in time $O(|M| \cdot |\phi|)$ CTL model checking is decidable in time $O(|M| \cdot |\phi|)$

Binary Decision Diagrams

Definition

A **Binary Decision Graph** is a DAG labelling by variables and stricty ordering them. Leaves are labelled 0 and 1.
A **Binary Decision Diagram** is a BDG with no subgraphs isomorphic and no redundant nodes.
A **Binary Decision Diagram with complement arcs** is a BDD with potential filled circle on edges inverting the meaning

Simplification

One can merge nodes with the same successors and neglect the leaf labelled 0.

Theorem

Given a function $\mathcal{V} \to \mathbb{B}^n$, there is an unique BDD up to isomorphism.

E

quivalence problem can be solved through isomorphic test
Negation can be solved through exchange of leaves
Conjonction can be solved through the following formula :
$ite(x, F[x := 1] \wedge G[x := 1], F[x := 0] \wedge G[x := 0])$
Disjonction can be solved in the same way
CBDD are unique only if we prohibit negated 0-labelled edges

Pushdown Systems

Definition

A **Pushdown system** is basically the transition system of a nested stack automaton without any input.
The **P-Automaton** is the true notion of automaton with qw as inputs.

Intuition

Replacing the last item pushed $\sim$ classical instruction
Pushing a new item $\sim$ procedure call
Poping the last item $\sim$ return

Property

Reachability from a configuration to another is decidable

Petri Nets

Definition

A **Petri Net** is a tuple P, T, F, W, wm_0, where :

- P is the set of places
- T is the set of transitions
- $F \subseteq P \times T \cup T \times P$ is the flow relations
- T is the set of transitions

The **P-Automaton** is the true notion of automaton with qw as inputs.

Intuition

Replacing the last item pushed $\sim$ classical instruction
Pushing a new item $\sim$ procedure call
Poping the last item $\sim$ return

Property

Reachability from a configuration to another is decidable