MASARYK UNIVERSITY
FACULTY OF INFORMATICS

«title»

BACHELOR'S THESIS

«author»

Brno, Fall 2020

Replace this page with a copy of the official signed thesis assignment and a copy of the Statement of an Author.

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

«author»

Advisor: «advisor»

Abstract

«abstract»

Keywords

«keywords»

Contents

List of Tables

List of Figures

1 Introduction

2 Preliminaries

2.1 Büchi Automaton

Büchi automaton is a theoretical finite state machine, that decides which infinitely long words are recognized by (given?) language. Let us define transition-based Buchi Automaton (TBA).

definition 1 TBA is a quintuple $A = (\Sigma, Q, q_0, \Delta, \Gamma)$. An alphabet is a set of letters. Σ is a finite alphabet recognized by A. Q is a finite set of states of A, $q_0 \in Q$ is called the initial state of A. Transitions of A directionally connect 2 states inside Q with a letter from alphabet Σ. We write the set of transitions as $\Delta \subseteq Q \times \Sigma \times Q$. Subset of the transitions $\Gamma \subseteq \Delta$ are accepting transitions.

2.2 TGBA

3 To Delete Chapter

3.1 Büchi Automaton

A nondeterministic Büchi automaton (BA) is a tuple $A = (\Sigma, Q, q_0, \Delta, \Gamma)$, where

- Σ is a finite alphabet
- Q is finite set of states
- $q_0 \in Q$ is the initial state
- $\Delta \subseteq Q \times \Sigma \times Q$ are transitions
- $\Gamma \subseteq \Delta$ are accepting transitions

run A run r of A on $w \in \Sigma^\omega$ is an ω-word $r_0, w_0, r_1, w_1, ...$ in $(Q \times \Sigma)^\omega$ such that $r_0 = q_0 \wedge \forall i > 0, (r_{i-1}, w_{i-1}, r_i) \in \Delta$

inf(r) We write $inf(r) \subseteq \Delta$ for the set of transitions that appear infinitely often in the run r.

accepting run A run r is accepting if $inf(r) \cap \Gamma \neq \emptyset$

language The language $L_A \subseteq \Sigma^\omega$ is recognized by A.
$\forall w \in L_A \exists r$ on w such that r is accepting.

ω-regular language A language is ω-regular if it is accepted by BA.

deterministic automaton $A = (\Sigma, Q, q_0, \Delta, \Gamma)$ is deterministic if $(q, \rho, q^{'}), (q, \rho, q^{''}) \in \Delta \implies q^{'} = q^{''}$

complete automaton A is complete if, $\forall w \in \Sigma, \forall q \in Q, \exists (q, w, q^{'}) \in \Delta$. A word in Σ^ω has exactly one run in a deterministic, complete automaton.

nepouzivat ρ , kombinace ρ a q je spatna
zminit v intro

3.2 Markov Decision Processes

A Markov decision process (MDP) M is a tuple (S, A, T, Σ, L), where

- S is a finite set of states
- A is a finite set of actions
- $T : S \times A \to D(S)$, where $D(S)$ is set of probability distributions over S, is the probabilistic transition (partial) function
- Σ is an alphabet
- $L : S \times A \times S \to \Sigma$ is the labeling function of the set of transitions. For a state $s \in S$, $A(s)$ denotes the set of actions available in s.

run A run of M is an ω-word $s_0, a_1, ... \in A = S \times (A \times S)^\omega$ such that $Pr(s_{i+1}|s_i, a_{i+1}) > 0$ for all $i >= 0$. A finite run is a finite such sequence.

labeled run We define labeled run as $L(r) = L(s_0, a_1, s_1), L(s_1, a_2, s_2), ... \in \Sigma^\omega$.

paths We write $\Omega(M)(Paths(M))$ for the set of runs (finite runs) of M and $\Omega_s(M)(Paths_s(M))$ for the set of runs (finite runs) of M starting from state s. When the MDP is clear from the context we drop the argument M.

strategy A strategy in M is a function $\mu : Paths \to D(A)$ such that $supp(\mu(r)) \subseteq A(last(r))$, where $supp(d)$ is the support of d and $last(r)$ is the last state of r. Let Ω_μ^M denote the subset of runs Ω^M that correspond to strategy μ and initial state s. Let Π_M be the set of all strategies.

pure strategy We say that a strategy μ is pure if $\mu(r)$ is a point distribution for all runs $r \in Paths$.

behavior The behavior of an MDP M under a strategy μ with starting state s is defined on a probability space $(\Omega_s^\mu, F_s^\mu, Pr_s^\mu)$ over the set of infinite runs of μ from s.

3.3 [WIP]Good-for-MDP (GFM) Automata

Given an MDP M and an automaton $A = (\Sigma, Q, q_0, \Delta, \Gamma)$, we want to compute an optimal strategy satisfying the objective that the run of M is in the language of A.

semantic satisfaction probability for given automaton and strategy We define the semantic satisfaction probability for A and strategy μ from state s as:
$PSem_A^M(s, \mu) = Pr_s^\mu\{r \in \Omega_s^\mu : L(r) \in L_A\}$

semantic satisfaction probability for given automaton .
$PSem_A^M(s) = \sup_{\mu \in \Pi_M} PSem_A^M(s, \mu)$

syntactic variant of the acceptance condition When using automata for given analysis of MDPs, we need a syntactic variant of the acceptance condition.

product of MDP and automaton Given an MDP $M = (S, A, T, \Sigma, L)$ with initial state $s_0 \in S$ and automaton $A = (\Sigma, Q, q_0, \Delta, \Gamma)$, the product $M \times A = (S \times Q, (s_0, q_0), A \times Q, T^\times, \Gamma^\times)$ is an MDP augmented with an inital state $s_0 \in S$ and accepting transitions $\Gamma^\times$. The (partial) function $T^\times : (S \times Q) \times (A \times Q) \to D(S \times Q)$ is defined by
$T^\times((s,q),(a,q^{'}))((s^{'},q^{'})) = \begin{cases} T(s,a)(s^{'}) & if(q, L(s,a,s^{'}), q^1) \in \Delta \\ undefined & otherwise \end{cases}$

GFM Automata An automaton A is good for MDPs if, for all MDPs M, $PSYN_A^M(s_0) = PSEM_A^M(s_0)$ holds, where s_0 is the initial state of M.

3.4 to be defined

ω-word?, point distribution?, what is F_s^μ in 'pure strategy' paragraph?, TGBA, describe Semi-determistic as I am going to compare them with SBA

3.4.1 text

GF MDP, model checking

3.5 Algorithms

BP + both slim

4 Implementation

4.1 Technologies

4.2 Implementation inside Seminator

5 Evaluation

5.1 Alternative Algorithm

5.2 Different Implementation - ePMC

5.3 Semi-deterministic Automata

6 Conclusion