

Metoda ověřování modelu (MC) pro konečně stavové systémy a lineární temporální logiku. Princip překladu formulí LTL na automaty nad nekonečnými slovy. Základní symbolické a explicitní algoritmy pro MC a jejich teoretická složitost.

1 LTL MODEL CHECKING

- TODO: Uplnost Hoareho logiky? Unknown?
- Nedeterminizmus typicky vzniká ako dôsledok paralelizmu.
- Formálny model systému: Kripkeho štruktúra — stavy S , prechodová relácia $T \subseteq S \times S$ (väčšinou sa predpokladá totálna, ak nie, opakujem deadlock stav), labelling $I \subseteq S \times \mathcal{P}(AP)$, iniciálny stav s_0 . Prípadne môžem mať ešte prechody označené akciami — Kripkeho prechodový systém — (typicky jedna akcia je vykonávaná viacerými prechodmi, podľa toho v akom stave sú napr. ostatné procesy).
- Buchi automaton — $\mathcal{B} = (\Sigma, S, s, \delta, F)$, Σ je abeceda, S sú stavy, s je iniciálny stav, $\delta \subseteq S \times \Sigma \times S$ je prechodová relácia a F sú akceptujúce stavy. Beh akceptuje ak je niečo z F nekonečne krát navštívené.
- LTL formula je preložiteľná na Buchi automaton. Kripkeho štruktúra je preložiteľná na Buchi automaton (abeceda je powerset propozícií, akceptujem všetkými stavmi - všetky behy akceptujú).
- Synchronna kompozícia týchto dvoch automatov je triviálna, keďže jeden automaton akceptuje všetkými stavmi, teda ako akceptujúce stavy mi stačí kartézsky súčin (inak by som musel riešiť preskakovanie z tieru do tieru aby som zabezpečil že sa budú opakovať stavy z oboch automatov)
- Teda $M \models \varphi \iff L(M) \subseteq L(\varphi) \iff L(M) \cap co - L(\varphi) = \emptyset \iff L(M) \cap L(\neg\varphi) = \emptyset \iff L(M \times \neg\varphi) = \emptyset$ (φ a M preťažujem, takže je to buď pôvodný systém/formula alebo už konkrétny Buchi automaton)
- Buchi reprezentuje neprázdny jazyk práve vtedy keď existuje dosiahnuteľný akceptujúci cyklus (laso).
- Algoritmus na hľadanie akceptujúceho cyklu: Nested DFS - zložitosť je lineárna k veľkosti systému, lebo keď sa vynorím z "nested" časti, určite nevedie žiadna nepreskúmaná hrana von, a teda tam nemôže byť cyklus do ktorého by som vedel vstúpiť z iného miesta.
- NestedDFS: Dve prehľadávania — Prvé vypočíta post-order akceptujúcich stavov z daného iniciálneho stavu. Druhé potom kontroluje prítomnosť cyklu pre jednotlivé akceptujúce stavy, ale kvôli nested argumentu nemusí ísť do už preskúmaných častí grafu. — nepotrebujem MC, stačí reachability

- Terminálny Buchi automat - všetky akceptujúce cykly sú selfloopy - safety vlastnosti.
- Slabý Buchi automat - všetky komponenty obsahujú buď len akceptujúce cykly, alebo len neakceptujúce (nemám zmiešané komponenty) - weak vlastnosti (typicky nejake $G(x \implies Fy)$) — nepotrebujem NestedDFS, stačí DFS.

2 LTL TO BA

- Formula v normálnom tvare — negácia na literáloch a všetko je prevedené na X, U a R.
- Použije sa zobecnený automat — mám viacero množín pričom beh musí nekonečne krát prejsť každou z nich.
- Preklad zobecneného automatu na štandardný: Zreplikujem stavový priestor pre každú množinu + jeden extra nultý tier. Do ďalšieho tieru môžem postúpiť vždy len cez akceptujúci stav (teda do tieru k sa dostanem ak som v tieru $k - 1$ a VSTUPUJEM do akceptujúceho stavu množiny F_k). Akceptujúce stavy sú všetky stavy posledného (alebo vlastne hociktorého) tieru. Teda pokiaľ mám nekonečný akceptujúci beh, tak to znamená že prejde akceptujúcim stavom každého z tierov.
- Konštrukcia grafu k formuli: Rekurzívne po štruktúre formule. Pre každý stav grafu si pamätám čo tam už mám *Now*, čo tam chcem *New* a čo má platiť v ďalšom stave *Next*. Začínam s jedným stavom ktorý má v *New* celú formulu a postupne ju rozbaľujem.
- Ak v *New* propozíciu, pozriem sa či je splniteľná s aktuálnym *Now*, ak nie, mažem tento uzol, ak ano, pridávam následníka ako presnú kópiu, iba s pridanou propozíciou. Ak mám and, vytváram nového následníka v ktorom do *New* dám obidve podformule, ak mám or, pridávam dvoch následníkov kde každý má v *New* jednu podformulu. Ak mám until, vytvorím dvoch následníkov — jeden má v *New* path formulu a v *Next* má zase until a druhý má v *New* reach formulu. Pre release mám jedného čo má "released" formulu v new a release v next a druhého čo má "released" aj "releaser" v new a v next nemá nič nové.
- Vo všetkých následníkoch samozrejme platí expandovaná formula v *Now*. Plus pokiaľ už nie je čo expandovať, tak skontrolujem či som náhodou nevyrobil niečo čo už mám, a ak ano, tak len incoming hrany do duplicitného stavu presmerujem tam. Ak som skutočne vyrobil nový stav, tak ho pridám a vytvorím nový stav, ktorý bude mať v *New* veci z môjho aktuálneho *Next*.