A4M33AU

report pro semestrální práci

Filip Bouška

Úvod

Nejdříve jsem zpracoval vstupní soubor (umístěný v složce input). Z informací v tomto souboru jsem si vytvořil strukturu *TrainStation*, která obsahuje informace o vstupních, výstupních a vnitřních (výhybky) uzlech daného nádraží, dále pak také všechny možné kombinace cest z jednotlivých vstupních uzlů do uzlů výstupních. Pro každou dvojici vstupního a výstupního uzlu jsem si uložil jen tu nejkratší cestu. Všechny tyto informace následně využívám při tvorbě axiomů.

Tvorba axiomů

Zde popíši všechny funkce, které využívám pro generaci axiomů.

1. **printLinearOrder()**
    - Axiomy lineárního uspořádání

2. **printPhysicalRestriction()** - fyzikální omezení nádraží
    - at_uniq - vlak je v určitý čas jen na jednom místě
    - at_nondup - vlak, který je v nádraží do něj nemůže vjet znovu
    - input_nocol - vlak nikdy nevjede na již obsazený uzel
    - enter_uniq - dva vlaky nevjedou současně do stejného vstupního uzlu
    - node_empty - jestliže je uzel prázdný, není v něm žádný vlak
    - node_safe - v daném čase není v daném uzlu kolize
    - not_blocked - v daném vstupní uzlu nikdy nezůstane zablokovaný vlak
    - train_will_move - vlak v budoucnu opustí uzel
    - train_entered - vlak vjel v minulosti do nádraží
    - train_moves - vlak se pohne, jakmile je to možné

3. **printDomainRestriction()** - omezení domény symbolů
    - at_restr - omezní pro hodnoty predikátu *at* (vlak může být pouze v uzlech, které jsou součástí daného nádraží)
    - open_restr - pouze vstupní uzly mohou být otevřené
    - input_restr - pravdivé pouze pro vstupní uzly
    - gate_restr - vlak může opustit nádraží pouze výstupními uzly
    - distinct_nodes - jednotlivé uzly nemohou být stejné
    - enter_values - vlak může vstoupit do nádraží pouze vstupními uzly

4. **printSwitchesRestriction()** - omezení pro výhybky
    - switch_X_values - omezí hodnot, kam může směřovat daná výhybka X

- switch_X_with_gate_Y - nastavení hodnot pro daný uzel X, když uvažujeme cestu s výstupním uzlem Y

5. **printMoveRestriction()** - omezení pro pohyby vlaku
   - Pro výhybky (vnitřní uzly)
     - moves_X - vlak je v uzlu X v čase T + 1 právě tehdy, když byl v čase T v uzlu, který je v dané trase před uzlem X
   - Pro výstupní uzly
     - moves_X - vlak je v uzlu X v čase T + 1 právě tehdy, když byl v čase T v uzlu, který je v dané trase před uzlem X

6. **printPathRestriction()** - axiomy pro cesty
   - path_from_to_values - omezení hodnot pro cesty, každá cesta začíná ve vstupním uzlu a končí ve výstupním
   - open_X - pro každý vstupní uzel X platí, že je otevřený pokud se na cestě do cílového uzlu nevyskytuje jiný vlak
   - path_free_from_X_to_Y - cesta ze vstupního uzlu X do výstupního uzlu Y je volná, pokud se na žádném z uzlů dané cesty nevyskytuje vlak

7. **printSwitchCritical()**
   - switch_critical - vlak stojí na výhybce a ta se přepne

8. **printCollisionCritical()**
   - collision_critical - dva nebo více vlaků přijede do stejného uzlu

9. **printBlockCritical()**
   - block_critical - vstupní uzel zůstane permanentně zavřený

Výsledky

Dokazování jsem prováděl na grafech *nadr_small, nadr1, nadr2* a *nad*. Jako dokazovač jsem použil ***prover9*** a jako hledač modelů ***mace4****.*

Pro všechny grafy se mi podařilo nalézt model.

Kritický stav *block* - "*Vstupní návěstidlo zůstane trvale uzavřené.*" se mi podařilo dokázat pro všechny grafy.

Kritický stav *collision* - *"Dva nebo více vlaků přijede do stejného uzlu."* se mi podařilo dokázat pro se mi podařilo dokázat pro graf *nadr_small* a *nadr1*. Důkaz grafu *nadr2* jsem po 30 minutách vypnul.

Kritický stav *switch* - *"Vlak stojí v uzlu (který je zároveň výhybkou, viz výše), a dojde k přepnutí výhybky."* se mi podařilo dokázat pro graf *nadr_small* a *nadr1*. Důkaz grafu *nadr2* jsem po 30 minutách vypnul.