

POLITECHNIKA WARSZAWSKA
WYDZIAŁ MATEMATYKI I NAUK INFORMACYJNYCH



REPREZENTACJA WIEDZY

Raport z testów projektu grupy nr 1

Programy działań z efektami domyślnymi

Autorzy:

Dragan Łukasz
Flis Mateusz
Izert Piotr
Pielat Mateusz
Rząd Przemysław
Siry Roman
Waszkiewicz Piotr
Zawadzka Anna

14 czerwca 2016

1 Opis projektu

Tematem testowanego przez nas projektu są programy działań z efektami domyślnymi. Rozpatrywana klasa systemów dynamicznych spełnia następujące warunki:

- Prawo inercji
- Niedeterminizm i sekwencyjność działań
- Pełna informacja o wszystkich akcjach i wszystkich ich skutkach bezpośrednich
- Z każdą akcją związany jest:
 1. Warunek początkowy (ew. true)
 2. Efekt akcji
 3. Jej wykonawca
- Skutki akcji:
 1. Pewne (zawsze występują po zakończeniu akcji)
 2. Domyślne (preferowane. Zachodzą po zakończeniu akcji, o ile nie jest wiadomym, że nie występują)
- Efekty akcji zależą od jej stanu, w którym akcja się zaczyna i wykonawcy tej akcji
- W pewnych stanach akcje mogą być niewykonalne przez pewnych (wszystkich) wykonawców

Opracowywany język kwerend ma za zadanie umożliwić tworzenie zapytań, pozwalających na uzyskanie odpowiedzi na następujące pytania:

- Czy podany program działań jest wykonywalny zawsze/kiedykolwiek?
- Czy wykonanie podanego programu działań z dowolnego stanu spełniającego warunek π prowadzi zawsze/kiedykolwiek/na ogół do stanu spełniającego warunek celu γ ?
- Czy z dowolnego stanu spełniającego warunek π cel γ jest osiągalny zawsze/kiedykolwiek/na ogół?
- Czy wskazany wykonawca jest zaangażowany w realizację programu zawsze/kiedykolwiek?

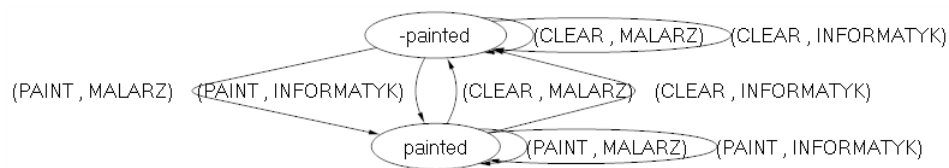
2 Przeprowadzone testy

2.1 Test 1

Zdefiniowana dziedzina:

```
initially ¬painted  
(PAINT, (MALARZ)) causes painted  
(CLEAR, (MALARZ, INFORMATYK)) causes ¬ painted
```

Otrzymany w wyniku zbudowania dziedziny graf wyglądał następująco:



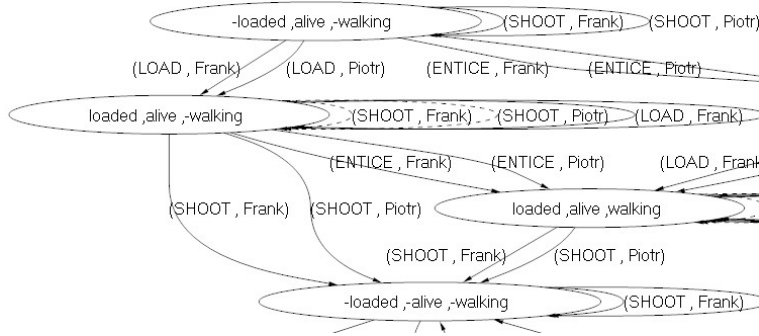
Jak widać zdefiniowane w nim zostały przejścia dla akcji PAINT wykonanej przez aktora INFORMATYK, prowadzące ze stanu w którym obraz nie był namalowany (\neg ainted) do stanu w którym już istniał (painted). Jest to błąd w programie.

2.2 Test 2

Zdefiniowana dziedzina:

initially $\neg \text{loaded} \wedge \neg \text{walking} \wedge \text{alive}$
 $(\text{LOAD}, (\epsilon))$ **causes** loaded
 $(\text{SHOOT}, (\epsilon))$ **typically causes** $\neg \text{alive} \wedge \neg \text{loaded}$ **if** loaded
 $(\text{ENTICE}, (\epsilon))$ preserves alive
 $(\text{ENTICE}, (\text{Frank}))$ **causes** walking
 $(\text{ENTICE}, (\text{Piotr}))$ **typically causes** walking

Otrzymana w wyniku zbudowania dziedziny część grafu wyglądała następująco



Znajdujący się tutaj błąd polega na braku przejścia w najwyższym stanie ($\neg \text{loaded}$, alive, $\neg \text{walking}$) dla akcji ENTICE wykonywanej przez aktora Piotr. Po zdefiniowaniu kwerendy:

ever accessible $\neg \text{walking}$ **after** $(\text{ENTICE}, (\text{PIOTR}))$ **if** alive

otrzymaną odpowiedzią było FALSE.

2.3 Test 3

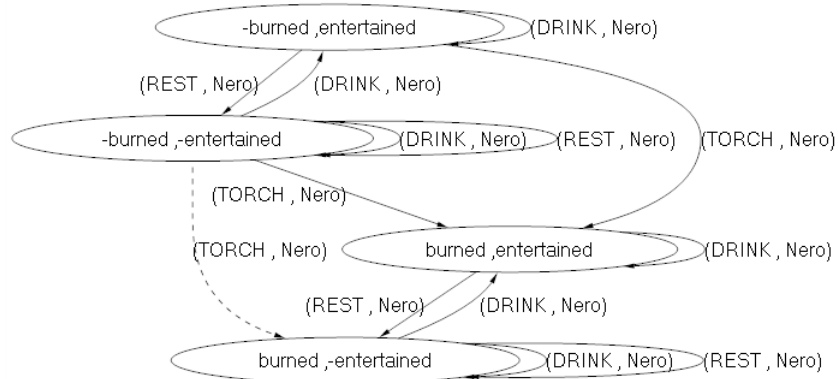
Zdefiniowana dziedzina:

```

initially  $\neg$ burned  $\wedge$   $\neg$ entertained
(DRINK,(Nero)) releases entertained if  $\neg$ entertained
(TORCH,(Nero)) typically causes entertained
(TORCH,(Nero)) causes burned
(REST,(Nero)) causes  $\neg$ entertained
impossible (TORCH,(Nero)) if burned

```

Otrzymany graf:



Kwerendy:

```

ever executable (DRINK, Nero) , (REST, Nero) , (TORCH, Nero)
always executable (DRINK, Nero) , (REST, Nero) , (TORCH, Nero)

```

zwracają kolejno *True* i *False*. W istocie, Neron może podpalić Rzym tylko raz.

Kwerendy:

```

ever accessible  $\neg$ burned when (TORCH, Nero)
always accessible  $\neg$ burned when (TORCH, Nero)
typically accessible  $\neg$ burned when (TORCH, Nero)

```

zwracają kolejno *False*, *False* i *True*. Patrząc na graf widać, że trzecia z kwerend zwraca niepoprawny wynik.

2.4 Test 4

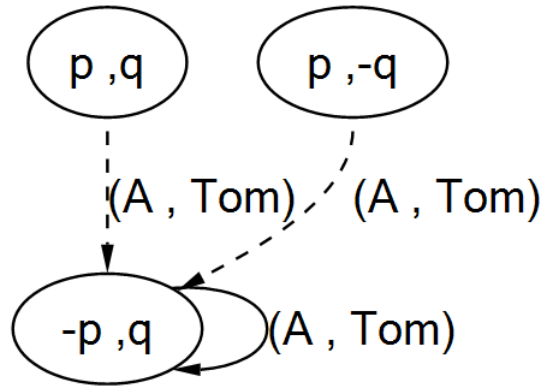
Zdefiniowana dziedzina:

```

initially p ∧ q
always p ∨ q
(A,(Tom)) causes ¬p
(A,(Tom)) typically causes ¬q if p

```

Otrzymany graf:



Kwerendy:

```

always accessible ¬p ∧ q if p when (A,Tom)
ever accessible ¬p ∧ q if p when (A,Tom)
typically accessible ¬p ∧ q if p when (A,Tom)

```

zwracają odpowiedź *True*, co jest poprawne dla dwóch pierwszych kwerend, natomiast patrząc na graf widać, że trzecia z kwerend zwraca niepoprawny wynik.

2.5 Test 5

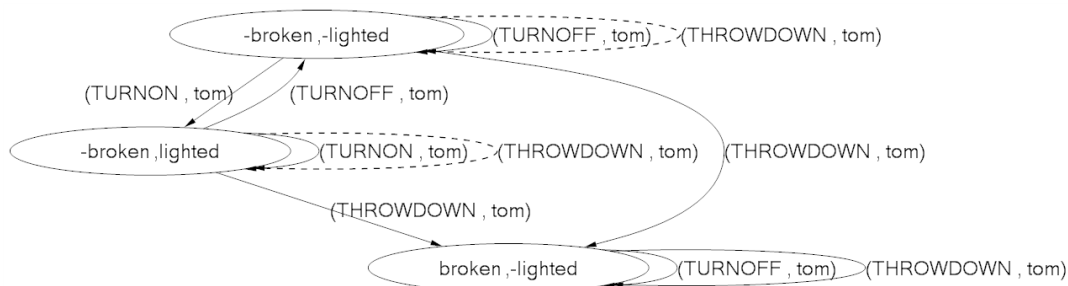
Zdefiniowana dziedzina:

```

initially lighted ∧ ¬ broken
always lighted → ¬ broken
(THROWDOWN,(Tom)) typically causes broken
(TURNON,(Tom)) causes lighted
(TURNOFF,(Tom)) causes ¬lighted
(TURNON,(Tom)) preserves broken

```

Otrzymany graf:



Przejścia i stany w grafie są poprawne.

Zadano następujące kwerendy:

```

always accessible lighted if lighted  $\wedge \neg$  broken
when (ThrowDown, Tom), (TurnOn, Tom)

ever accessible lighted if lighted  $\wedge \neg$  broken
when (ThrowDown, Tom), (TurnOn, Tom)

typically accessible lighted if lighted  $\wedge \neg$  broken
when (ThrowDown, Tom), (TurnOn, Tom)

```

Uzyskane odpowiedzi:

- **always** - *True* - OK - wynik nie jest do końca intuicyjny, ale zakładając, że dany scenariusz wykonał się w całości, jedynym możliwym stanem końcowym jest taki, w którym lampka jest zapalona
- **ever** - *True* - OK
- **typically** - *False* - OK - uzyskany efekt jest nietypowy - lampka spadła, ale dalej się świeci

2.6 Test 7

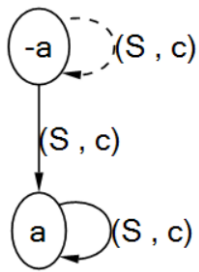
Dziedzina:

```

Ac = {c}, F = {a}, A = {S}
initially  $\neg$ a
(S, (c)) typically causes a

```

Otrzymany graf:



ever accessible $\neg a$

Zwrócona odpowiedź *FALSE*
 Oczekiwana odpowiedź: *TRUE*