

REPREZENTACJA WIEDZY

## REALIZACJE SCENARIUSZY DZIAŁAŃ

PROJEKT NR 5

SZEF:

**ROBERT JAKUBOWSKI**

AUTORZY:

MARIUSZ AMBROZIAK

PAWEŁ BIELICKI

KAROL BOCIAN

HANNA DZIEGCIAR

KAROL DZITKOWSKI

MATEUSZ JANKOWSKI

WIKTOR RYCIUK

WARSZAWA, 26 MARCA 2014

## Spis treści

<b>1. Opis zadania</b>	<b>3</b>
<b>2. Opis języka akcji</b>	<b>3</b>
2.1. Sygnatura języka . . . . .	4
2.2. Opis domeny . . . . .	4
2.3. Scenariusze działań . . . . .	4
2.4. Semantyka . . . . .	5
<b>3. Opis języka kwerend</b>	<b>6</b>
<b>4. Przykłady</b>	<b>7</b>
4.1. Pytanie czy dany scenariusz może wystąpić . . . . .	7
4.2. Pytanie czy dany warunek zachodzi w danym czasie . . . . .	9
4.3. Pytanie czy dana akcja jest wykonywana w pewnym czasie . . . . .	10
4.4. Brak integralności . . . . .	11

# 1 Opis zadania

Zadaniem projektu jest opracowanie i zaimplementowanie:

- języka akcji pewnej klasy systemów dynamicznych,
- język kwerend zapewniającego uzyskanie odpowiedzi na określone pytania.

Szczegółowy opis klasy systemów dynamicznych oraz języka akcji jest opisany w rozdziale 'Opis języka akcji', natomiast język kwerend oraz zadawane pytania znajdują się w rozdziale 'Opis języka kwerend'. W tym dokumencie znajdują się również przykłady. Pokazują one konkretne przypadki użycia oraz oczekiwane wyniki działania programu.

# 2 Opis języka akcji

Język akcji zaprojektowany na potrzeby zadania spełnia następujące warunki:

1. Prawo inercji.
2. Sekwencyjność działań.
3. Możliwe akcje niedeterministyczne.
4. Liniowy model czasu - czas dyskretny.
5. Pełna informacja o wszystkich:
  - (a) akcjach,
  - (b) skutkach bezpośrednich.
6. Akcja posiada:
  - (a) warunek początkowy,
  - (b) czas trwania  $t \geq 1, t \in \mathbb{N}$ ,
  - (c) efekt akcji.
7. Podczas trwania akcji, wartości zmiennych, na które ona wpływa, nie są znane.
8. Występujące rodzaje efektów:
  - (a) środowiskowe: zmiany wartości zmiennych systemu,
  - (b) dynamiczne: wystąpienie innych akcji po  $d \geq 0$  jednostkach czasu od zakończenia akcji przyczynowej.
9. W pewnych stanach akcje mogą być niewykonalne, przy czym stany takie podane są albo przez podanie konkretnych punktów czasowych, albo przez określenie warunków logicznych.
10. Stany opisywane częściowo przez obserwacje.
11. Pewne stany mogą rozpocząć wykonywanie pewnych akcji.

Językiem odpowiadającym powyższym założeniom jest język *AL* opisujący domeny akcji z czasem liniowym.

## 2.1 Sygnatura języka

Sygnaturą języka jest następująca trójka zbiorów:  $\psi = (F, Ac, \mathbb{N})$ , gdzie:

$F$  – zbiór zmiennych inercji (fluentów)

$Ac$  – zbiór akcji

$\mathbb{N}$  – zbiór liczb naturalnych (czas trwania akcji)

## 2.2 Opis domeny

Rodzaje zdań występujących w projektowanym języku (domena języka):

Oznaczenia:

$f$  – fluent

$Ac_i, Ac_j \in Ac$

$\alpha, \pi \in Forms(F)$

$d_i, d \in \mathbb{N}$

- initially  $\alpha$   
Określa stan początkowy fluentów w formule  $\alpha$ .
- $(Ac_i, d_i)$  causes  $\alpha$  if  $\pi$   
Akcja  $Ac_i$  trwająca  $d_i$  chwil powoduje stan  $\alpha$ , jeśli zachodzi warunek  $\pi$ .
- $(Ac_i, d_i)$  invokes  $(Ac_j, d_j)$  after  $d$  if  $\pi$   
Akcja  $Ac_i$  trwająca  $d_i$  chwil powoduje wykonanie akcji  $Ac_j$  trwającej  $d_j$  chwil po  $d$  chwilach od zakończenia akcji  $Ac_i$ , jeśli zachodzi warunek  $\pi$ .
- $(Ac_i, d_i)$  releases  $f$  if  $\pi$   
Akcja  $Ac_i$  trwająca  $d_i$  chwil powoduje uwolnienie  $f$  po zakończeniu akcji  $Ac_i$ , jeśli zachodzi warunek  $\pi$ .
- $\pi$  triggers  $(Ac_i, d_i)$   
Akcja  $Ac_i$  trwająca  $d_i$  chwil jest wykonywana, jeśli zajdzie warunek  $\pi$ .
- impossible  $(Ac_i, d_i)$  at  $d$   
Akcja  $Ac_i$  trwająca  $d_i$  chwil jest niewykonalna w chwili  $d$ .
- impossible  $(Ac_i, d_i)$  if  $\pi$   
Akcja  $Ac_i$  trwająca  $d_i$  chwil jest niewykonalna, jeśli zachodzi warunek  $\pi$ .
- always  $\alpha$   
Każdy stan spełnia warunek  $\alpha$ .

## 2.3 Scenariusze działań

Scenariusze działań opisane są w następujący sposób:

- $Sc = (OBS, ACS)$
- $OBS = \{(\gamma_1, t_1), \dots, (\gamma_m, t_m)\}$ , gdzie:  
 $m \geq 0$  – obserwacje, gdzie każda obserwacja jest stanem częściowym (stanem spełniającym warunek  $\gamma$  w pewnym punkcie czasu  $t$ ).  
 $\gamma$  – zbiór (np.  $x_1 = True, x_2 = True, x_3 = False$ ).

- $ACS = \{((Ac_1, d_1), t_1), \dots, ((Ac_n, d_n), t_n)\}$ , gdzie:  
 $n \geq 1$ ,  
 $Ac_i$  – akcja,  
 $d_i$  – czas trwania akcji,  
 $t_i$  – punkt w czasie (rozpoczęcie akcji).

## 2.4 Semantyka

**Definicja 2.1.** Semantyczną strukturą języka  $AL$  nazywamy system  $S = (H, O, E)$  taki, że:

- $H : F \times \mathbb{N} \rightarrow \{0, 1\}$  jest funkcją historii, pozwala ona stwierdzić, jaki stan ma pewny fluent w danej chwili czasu.
- $O : Ac \times \mathbb{N} \rightarrow 2^F$  jest funkcją okluzji. Dla pewnej ustalonej akcji  $A$  i chwili czasu  $t \in \mathbb{N}$  funkcja  $O(A, t)$  zwraca zbiór fluentów, na który akcja  $A$  ma wpływ, jeśli zostanie zakończona od czasu  $t - 1$  do  $t$ .
- $E \subseteq Ac \times \mathbb{N} \times \mathbb{N}$  jest relacją wykonań akcji. Trójka  $(A, t, d)$  należy do relacji  $E$  jeśli akcja  $A$  trwająca  $d$  czasu jest rozpoczęta w czasie  $t$ . W naszym modelu zakładamy warunek sekwencyjności działań. Oznacza on, że tylko jedną akcję możemy wykonać w danym czasie tak, więc jeśli  $(A, t, d) \in E$  oraz  $(B, t, d) \in E$ , to  $A = B$ .

Niech:  $A, B$  będą akcjami,  $f$  - fluentem,  $\alpha, \pi$  - będą formułami,  $d, d_2, d_3$  - liczbami naturalnymi (oznaczającymi czas trwania akcji) oraz  $fl(\alpha)$  będzie zbiorem fluentów występujących w  $\alpha$ . Wtedy dla zdań języka  $AL$  muszą być spełnione następujące warunki:

- Dla każdego wyrażenia (*initially*  $\alpha$ ) mamy  $H(\alpha, 1) = 1$ .
- Dla każdego wyrażenia  $((A, d) \text{ causes } \alpha \text{ if } \pi) \in D$  i dla każdego momentu w czasie  $t \in \mathbb{N}$ , jeżeli  $H(\pi, t) = 1$  oraz  $(A, t, d) \in E$ , wtedy  $H(\alpha, t + d) = 1$  i  $fl(\alpha) \subseteq O(A, t + d)$ .
- Dla każdego wyrażenia  $((A, d) \text{ release } f \text{ if } \pi) \in D$  i dla każdego momentu czasu  $t \in \mathbb{N}$ , jeżeli  $H(\pi, t) = 1$  oraz  $(A, t, d) \in E$ , wtedy  $f \in O(A, t + d)$ .
- Dla każdego wyrażenia  $(\pi \text{ triggers } (A, d)) \in D$  i dla każdego momentu czasu  $t \in \mathbb{N}$ , jeżeli  $H(\pi, t) = 1$ , wtedy  $(A, t, d) \in E$ .
- Dla każdego wyrażenia  $((A, d_1) \text{ invokes } (B, d_2) \text{ after } d \text{ if } \pi) \in D$  i dla każdego momentu czasu  $t \in \mathbb{N}$ , jeżeli  $H(\pi, t) = 1$  oraz  $(A, t, d_1) \in E$ , wtedy  $(B, t + d + d_1, d_2) \in E$ .
- Dla każdego wyrażenia (*impossible*  $(A, d)$  at  $t$ ) mamy  $(A, t, d) \notin E$ .
- Dla każdego wyrażenia (*impossible*  $(A, d)$  if  $\pi$ ) i dla każdego momentu czasu  $t \in \mathbb{N}$ , jeżeli  $H(\pi, t) = 1$ , wtedy  $(A, t, d) \notin E$ .
- Dla każdego wyrażenia (*always*  $\alpha$ ) i dla każdego momentu czasu  $t \in \mathbb{N}$ , mamy  $H(\alpha, t) = 1$ .

**Definicja 2.2.** Niech  $S = (H, O, E)$  będzie strukturą języka  $AL$ ,  $Sc = (OBS, ACS)$  będzie scenariuszem, oraz  $D$  domeną. Powiem, że  $S$  jest strukturą dla  $Sc$  zgodnym z opisem domeny  $D$  jeśli:

- Dla każdej obserwacji  $(\alpha, t) \in OBS$ ,  $H(\alpha, t) = 1$
- $ACS \subseteq E$

**Definicja 2.3.** Niech  $O_1, O_2: X \rightarrow 2^Y$ , mówimy, że  $O_1 \prec O_2$  jeżeli  $\forall x \in X \ O_1(x) \subseteq O_2(x)$  oraz  $O_1 \neq O_2$ .

**Definicja 2.4.** Niech  $S = (H, O, E)$  będzie strukturą dla scenariusza  $Sc = (OBS, ACS)$  zgodną z opisem domeny  $D$ . Mówimy, że  $S$  jest  $O$ -minimalną strukturą, jeżeli nie istnieje struktura  $S' = (H', O', E')$  dla tego samego scenariusza i domeny taka, że  $O' \prec O$ .

**Definicja 2.5.** Niech  $S = (H, O, E)$  będzie strukturą dla scenariusza  $Sc = (OBS, ACS)$  zgodną z opisem domeny  $D$ .  $S$  będziemy nazywać modelem  $Sc$  zgodnym z opisem  $D$  jeżeli:

- $S$  jest  $O$ -minimalny
- Dla każdego momentu w czasie  $t, d \in \mathbb{N}$ , jeżeli istnieje  $f \in F$ : takie, że  $H(f, t) \neq H(f, t + d)$  to istnieje pewna akcja  $A \in Ac$  trwająca  $d$  czasu, taka, że  $f \in O(A, t + d)$ .
- Nie istnieje, żadna struktura  $S' = (H', O', E')$  dla  $Sc$  zgodna z opisem  $D$  która spełnia poprzednie warunki oraz taka, że  $E' \subset E$ .

**Uwaga 2.1.** Nie dla każdego scenariusza można ułożyć model. Mówimy, że scenariusz  $Sc$  jest zgodny jeśli istnieje do niego model zgodny z domeną  $D$ .

### 3 Opis języka kwerend

Zdefiniowany język akcji może być odpytywany przez poniżej zaprezentowany język kwerend, który zapewnia uzyskanie odpowiedzi *TRUE/FALSE* na następujące pytania:

**Q1.** Czy podany scenariusz jest możliwy do realizacji zawsze/kiedykolwiek?

- *always/ever executable Sc*  
Oznacza, że scenariusz  $Sc$  zawsze/kiedykolwiek jest możliwy do realizacji.

**Q2.** Czy w chwili  $t \geq 0$  realizacji podanego scenariusza warunek  $\gamma$  zachodzi zawsze/kiedykolwiek?

- *always/ever  $\gamma$  at  $t$  when  $Sc$*   
Oznacza, że zawsze/kiedykolwiek w chwili  $t$  realizacji scenariusza  $Sc$  zachodzi warunek  $\gamma$ .
- *always/ever  $\gamma$  when  $Sc$*   
Oznacza, że zawsze/kiedykolwiek w pewnej chwili  $t$  realizacji scenariusza  $Sc$  zachodzi warunek  $\gamma$ .

**Q3.** Czy w chwili  $t$  realizacji scenariusza wykonywana jest akcja  $A$ ?

- *performing  $A$  at  $t$  when  $Sc$*   
Oznacza, że zawsze w chwili  $t$  realizacji scenariusza  $Sc$  zachodzi akcja  $A$ .
- *performing  $A$  when  $Sc$*   
Oznacza, że zawsze w pewnej chwili  $t$  realizacji scenariusza  $Sc$  zachodzi akcja  $A$ .
- *performing at  $t$  when  $Sc$*   
Oznacza, że zawsze w chwili  $t$  realizacji scenariusza  $Sc$  zachodzi pewna akcja.

**Q4.** Czy podany cel  $\gamma$  jest osiągalny zawsze/kiedykolwiek przy zadanym zbiorze obserwacji  $OBS$ ?

- *always/ever accesible  $\gamma$  when  $Sc$*   
Oznacza, że cel  $\gamma$  jest osiągalny zawsze/kiedykolwiek przy zadanym zbiorze obserwacji  $OBS$

przy realizacji scenariusza  $Sc$ .

### Semantyka kwerend w języku

Niech  $Sc$  będzie scenariuszem,  $D$  niech będzie opisem domeny języka, wtedy powiemy, że kwerenda  $Q$  jest konsekwencją  $Sc$  zgodnie z  $D$  (ozn.  $Sc, D \models Q$ )

- zapytanie kwerendą  $Q$  postaci *always/ever executable*  $Sc$  zwróci wynik  $TRUE$  jeśli dla każdego modelu  $S = (H, O, E)$  zgodnego z  $D$  scenariusz  $Sc$  zakończy się.
- zapytanie kwerendą  $Q$  postaci  $\gamma$  at  $t$  when  $Sc$  zwróci wynik  $TRUE$  jeśli dla każdego modelu  $S = (H, O, E)$  scenariusza  $Sc$  zgodnego z  $D$  zajdzie  $H(\gamma, t) = 1$
- zapytanie kwerendą  $Q$  postaci  $\gamma$  when  $Sc$  zwróci wynik  $TRUE$  jeśli dla każdego modelu  $S = (H, O, E)$  scenariusza  $Sc$  zgodnego z  $D$  zajdzie  $\exists_{t \in N} H(\gamma, t) = 1$
- zapytanie kwerendą  $Q$  postaci *performing*  $A$  at  $t$  when  $Sc$  zwróci wynik  $TRUE$  jeśli dla każdego modelu  $S = (H, O, E)$  scenariusza  $Sc$  zgodnego z  $D$  zajdzie  $\forall_{d \in N} (A, t, d) \in E$
- zapytanie kwerendą  $Q$  postaci *performing*  $A$  when  $Sc$  zwróci wynik  $TRUE$  jeśli dla każdego modelu  $S = (H, O, E)$  scenariusza  $Sc$  zgodnego z  $D$  zajdzie  $\exists_{t \in N} \forall_{d \in N} (A, t, d) \in E$
- zapytanie kwerendą  $Q$  postaci *performing* at  $t$  when  $Sc$  zwróci wynik  $TRUE$  jeśli dla każdego modelu  $S = (H, O, E)$  scenariusza  $Sc$  zgodnego z  $D$  zajdzie  $\exists_{A \in Ac} \forall_{d \in N} (A, t, d) \in E$
- zapytanie kwerendą  $Q$  postaci *accessible*  $\gamma$  when  $Sc$  zwróci wynik  $TRUE$  jeśli dla każdego modelu  $S = (H, O, E)$  scenariusza  $Sc$  zgodnego z  $D$  zajdzie  $\exists_{t \in N} \exists_{A \in Ac} \gamma \in O(A, t)$

**Uwaga 3.1.** Jeśli warunek nie zajdzie program zwróci wartość  $FALSE$ .

## 4 Przykłady

### 4.1 Pytanie czy dany scenariusz może wystąpić

#### 4.1.1 Historia

Mick i Sarah są parą, więc mają wspólne produkty spożywcze, ale posiłki zwykle jadają oddzielnie. Pewnego dnia Sarah chce zrobić ciasto, a Mick naleśniki. Nie mogą być one robione w tym samym czasie ze względu na konieczność użycia miksera do przygotowania obu. Ponadto, zrobienie jednego lub drugiego dania zużywa cały zapas jajek dostępnych w mieszkaniu, więc trzeba je potem dokupić.

#### 4.1.2 Opis akcji

**initially** *eggs*

*(making\_panc, 1)* **causes**  $\neg$  *eggs* **if** *eggs*

$(\text{making\_cake}, 1)$  **causes**  $\neg \text{eggs}$  **if**  $\text{eggs}$   
**impossible**  $\{\text{making\_pan}, \text{making\_cake}\}$   
 $(\text{buy\_eggs}, 2)$  **causes**  $\text{eggs}$

#### 4.1.3 Scenariusze

$Sc = (OBS; ACS)$   
 $OBS = \emptyset$   
 $ACS = ((\text{making\_pan}, ; 1), 0), ((\text{making\_cake}, 1), 2)$

$Sc2 = (OBS2; ACS2)$   
 $OBS2 = \emptyset$   
 $ACS2 = ((\text{making\_panc}, 1), 0), ((\text{buy\_eggs}, 2), 2), ((\text{making\_cake}, 1), 4), ((\text{making\_panc}, 1), 4)$

#### 4.1.4 Kwerendy

1. **performing**  $\text{making\_panc}$  **at** 1 **when**  $Sc$
2. **performing**  $\text{making\_cake}$  **at** 2 **when**  $Sc$
3. **ever executable**  $Sc$
4. **ever executable**  $Sc2$

#### 4.1.5 Analiza

Odpowiedzi na kwerendy to odpowiednio:

1. TRUE,
2. TRUE,
3. TRUE,
4. FALSE.

Zgodnie z diagramem dla scenariuszy  $Sc$  i  $Sc2$ :



	making_panc	making_cake	
Czas	1	2	3
Eggs	E	~E	~E

	making_panc	buy_eggs		making_panc	
				making_cake	
Czas	1	2	3	4	5
Eggs	E	~E	?E	E	~E

Scenariusza *Sc2* nie można wykonać, ponieważ wymaga on jednoczesnego wypełnienia akcji *making\_panc* i *making\_cake*, co jest niezgodne z warunkami zadania.

## 4.2 Pytanie czy dany warunek zachodzi w danym czasie

### 4.2.1 Historia

Mick i Sarah są parą, więc mają wspólne produkty spożywcze, ale posiłki zwykle jadają oddzielnie. Pewnego dnia Sarah chce zrobić ciasto, a Mick naleśniki. Nie mogą być one robione w tym samym czasie ze względu na konieczność użycia miksera do przygotowania obu. Ponadto, zrobienie jednego lub drugiego dania zużywa cały zapas jajek dostępnych w mieszkaniu, więc trzeba je potem dokupić.

### 4.2.2 Opis akcji

**initially** *eggs*

*(making\_panc, 1)* **causes**  $\neg$  *eggs* **if** *eggs*

*(making\_cake, 1)* **causes**  $\neg$  *eggs* **if** *eggs*

**impossible** {*making\_panc, making\_cake*}

*(buy\_eggs, 2)* **causes** *eggs*

### 4.2.3 Scenariusz

$Sc = (OBS; ACS)$

$OBS = \emptyset$

$ACS = ((making\_panc, 1), 0), ((making\_cake, 1), 2)$

### 4.2.4 Kwerendy

1. *eggs* at 1 **when** *Sc*
2. *eggs* at 2 **when** *Sc*

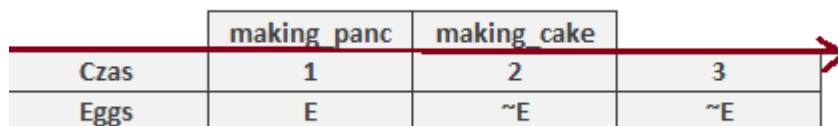
#### 4.2.5 Analiza

Odpowiedzi na kwerendy to odpowiednio:

1. TRUE,
2. FALSE.

Zgodnie z diagramem dla scenariusza  $Sc$ :

	making panc	making cake	
Czas	1	2	3
Eggs	E	~E	~E



Oczywiście warunek akcji *making-panc* nie jest spełniony w momencie 2.

### 4.3 Pytanie czy dana akcja jest wykonywana w pewnym czasie

Ten przykład pokazuje przypadek kwerendy, która pyta, czy dana akcja jest wykonywana w pewnym czasie.

#### 4.3.1 Historia

Mamy Billa i psa Maxa. Jeśli Bill idzie, to Max biegnie. Jeśli Bill gwizdże, Max szczeka. Jeśli Bill zatrzymuje się, Max również. Jeśli Bill przestaje gwizdać, to Max przestaje szczekać.

#### 4.3.2 Opis akcji

initially  $\neg go\_Bill$  and  $\neg run\_Max$  and  $\neg whistle\_Bill$  and  $\neg bark\_Max$   
 $(goes\_Bill, 2)$  **causes**  $run\_Max$   
 $(goes\_Bill, 2)$  **invokes**  $(runs\_Max, 2)$  **after** 0  
 $(runs\_Max, 2)$  **causes**  $\neg run\_Max$   
 $(whistles\_Bill, 1)$  **causes**  $bark\_Max$   
 $(whistles\_Bill, 1)$  **invokes**  $(barks\_Max, 1)$  **after** 0  
 $(whistles\_Bill, 1)$  **causes**  $\neg whistle\_Bill$

#### 4.3.3 Scenariusz

$Sc = (OBS; ACS)$

$OBS = \emptyset$

$ACS = ((goes\_Bill, 2), 1), ((whistles\_Bill, 1), 5), ((goes\_Bill, 2), 7)$

#### 4.3.4 Kwerendy

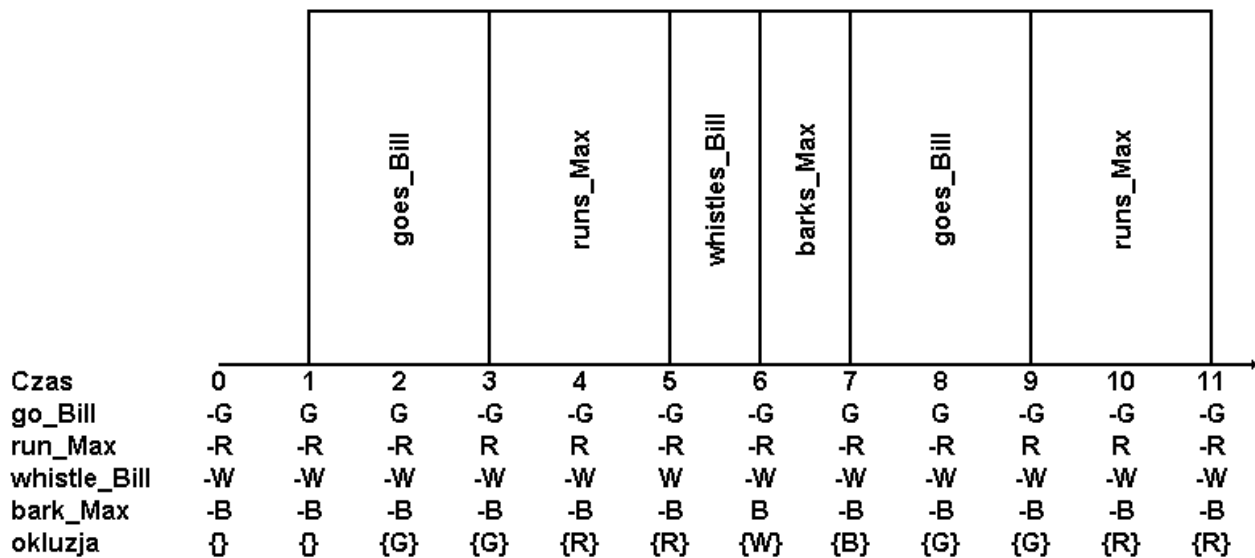
1. performing *run\_Max* at 8 when *Sc*
2. performing *run\_Max* when *Sc*
3. performing at 8 when *Sc*

#### 4.3.5 Analiza

Odpowiedzi na powyższe kwerendy są następujące:

1. FALSE,
2. TRUE,
3. TRUE.

Ilustruje to poniższy diagram:



#### 4.4 Brak integralności

Przykład *Brak integralności* pokazuje scenariusz, który mimo zgodności z warunkami zadania, jest sprzeczny z logiką *common sense* (z powodu braku warunków integralności).

##### 4.4.1 Historia

Mamy Billa oraz komputer. Bill może nacisnąć przycisk *Włącz* lub odłączyć komputer od zasilania. Komputer jest wyłączony i podłączony do zasilania. Jeżeli zostanie naciśnięty jego przycisk *Włącz*, to komputer włącza się.

#### 4.4.2 Opis akcji

**initially**  $\neg on\_computer$  **and**  $connect\_power\_computer$  **and**  $\neg switch\_on\_computer$   
*(clicks\_button\_on, 1)* **causes**  $switch\_on\_computer$   
*(clicks\_button\_on, 1)* **invokes** *(switches\_on\_computer, 2)* **after** 0  
*(switches\_on\_computer, 1)* **causes**  $on\_computer$   
*(disconnects\_power, 1)* **causes**  $on\_computer$  **and**  $\neg switch\_on\_computer$

#### 4.4.3 Scenariusz

$Sc = (OBS; ACS)$

$OBS = \emptyset$

$ACS = ((clicks\_button\_on, 1), 1), ((disconnects\_power, 1), 4), ((clicks\_button\_on, 1), 5)$

#### 4.4.4 Kwerendy

1.  $switch\_on\_computer$  **at**  $6 + 2$  **when**  $Sc$
2.  $switch\_on\_computer$  **and**  $\neg on\_computer$  **at**  $6 + 2$  **when**  $Sc$

#### 4.4.5 Analiza

Powyższy scenariusz jest prawidłowy, lecz zawiera pewną niezgodność. W chwili  $t = 4 + 1$  komputer zostaje odcięty od zasilania. Powinien więc wyłączyć się. Bill chwili  $t = 5 + 1$  naciska przycisk *Włącz*. Komputer zacznie włączać się mimo iż jest odcięty od zasilania. Zachodzą dwa sprzeczne ze sobą stany, tj.  $switch\_on\_computer = T$  i  $on\_computer = T$ . Odpowiedzi na powyższe kwerendy będą odpowiednio: 1. TRUE i 2. FALSE. Należy zaznaczyć, że odpowiedzi zgodnie ze zdrowym rozsądkiem powinny być sobie równe.

		clicks_button	switches_on_computer	disconnects_power	clicks_button	switches_on_computer			
Czas	0	1	2	3	4	5	6	7	8
on_computer	F	F	F	F	-F	?F	?F	?F	?F
connect_power_computer	T	T	T	T	T	-T	-T	-T	-T
switch_on_computer	G	G	-G	-G	-G	G	G	G	G
okluzja	{}	{}	{F}	{G}	{G}	{F,T}	{F}	{G}	{G}