Politechnika Warszawska

MSI

C2. System logicznego wnioskowania

Włodzimierz Kasprzak

Zad. 1. Rachunek sytuacji

Dla „świata Wumpusa” zdefiniować w rachunku sytuacji:

- warunki początkowe,
- aksjomaty miejsc (topologię środowiska) i
- aksjomaty ukrytych własności miejsc (związek obserwacji z własnościami miejsc).

Rozwiązanie zad. 1

Warunki początkowe. Agent zna swoje początkowe położenie: $Jest(Agent,[1,1],S_0)$. Zna też kierunek, w którym patrzy. Np.: $Kierunek(Agent,S_0)= 90)$.

Aksjomaty miejsc reprezentują topologię środowiska. Są one niezależne od sytuacji, gdyż są niezmienne w czasie. Zdefiniujemy funkcję, np. PK, zwracającą kratkę przyległą do aktualnej kratki w zależności od zorientowania agenta:

$$\forall x,y\ PK([x,y], 0) =[x+1, y]\ ;\ \forall x,y\ PK([x,y], 90) =[x, y+1]$$

$$\forall x,y\ PK([x,y], 180) =[x-1, y]\ ;\qquad \forall x,y\ PK([x,y], 270) =[x, y-1]$$

Zdefiniujemy też relację sąsiedztwa kratki – „sąsiedzi kratki [x,y] to [a,b]”:

$$\forall x,y,a,b\ Sąsiedzi([x,y],[a,b]) \Leftrightarrow [a,b] \in \{[x+1,y], [x-1,y],[x,y+1],[x,y-1]\}$$

Rozwiązanie 1. (c.d.) aksjomaty ukrytych własności miejsc

Ukryte własności miejsc łączą obserwacje z przyczynami. Najpierw należy przejść z aktualnej obserwacji (zależnej od sytuacji) do charakterystyki miejsca (niezależnej od sytuacji). Następnie możemy wykorzystać reguły przyczynowo-skutkowe lub diagnostyczne do wyrażenia ukrytych własności związanych z tą charakterystyką.

Wyznaczenie charakterystyki miejsc niezależnej od sytuacji.

Np. „*W pobliżu jamy kwadrat jest wietrzny*”:

$$\forall \mathbf{x}, s \;\; Jest(\text{Agent}, \mathbf{x}, s) \wedge Wiatr(s) \Rightarrow Wietrzny(\mathbf{x})$$

Powyższa reguła jest synchroniczna - ta **sama sytuacja** dla predykatów występujących w regule.

Reguła diagnostyczna – wnioskuje ukrytą przyczynę (tu: własność miejsca) z jej obserwowanego efektu:

- $\forall \mathbf{x}\; Wietrzny(\mathbf{x}) \Rightarrow \exists\, \mathbf{y}\; Sąsiedzi(\mathbf{x},\mathbf{y}) \wedge Jama(\mathbf{y})$

Reguła przyczynowa (ważniejsza, ale trudniejsza w użyciu) - wnioskuje obserwowany efekt z ukrytej przyczyny:

- $\forall \mathbf{x}\; Jama(\mathbf{x}) \Rightarrow [\forall \mathbf{y}\; Sąsiedzi(\mathbf{x}, \mathbf{y}) \Rightarrow Wietrzny(\mathbf{y})\,]$

Zadanie 2. Wybór akcji w rachunku sytuacyjnym

W rachunku sytuacyjnym zdefiniować wiedzę potrzebną do wyboru akcji przez agenta „świata Wumpusa”.

Rozwiązanie 2. Reprezentacja wiedzy dla wyboru akcji

Najpierw określimy predykaty dla reprezentowania „ważności" akcji:

- *Wspaniała* – podniesienie złota lub wyjście z jaskini ze złotem.
- *Dobra* – wejście do bezpiecznego kwadratu, który nie był jeszcze odwiedzany.
- *Przeciętna* - wejście do bezpiecznego kwadratu, który był już odwiedzany lub oddanie strzału do Wumpusa.
- *Ryzykowna* – wejście do kwadratu, w którym może być Wumpus lub jama lub oddanie strzału przy niepewności położenia Wumpusa.
- *Śmiertelna* – wejście do kwadratu, w którym na pewno jest Wumpus lub jama.

Wprowadzamy 2-argumentowy predykat *Akcja*:

$Akcja(a,s)$ interpretujemy: *„w sytuacji s wybierz akcję a"*.

Rozwiązanie 2 (c.d.)

Wprowadzamy reguły dla wyboru akcji według priorytetu:

$\forall a,s \;\; Wspaniała(a, s) \Rightarrow Akcja(a, s)$

$\forall a,s \;\; Dobra(a, s) \wedge [\neg \exists \mathrm{b}\; Wspaniała(b, s)] \Rightarrow Akcja(a, s)$

$\forall a,s \;\; Przeciętna(a, s) \wedge [\neg \exists \mathrm{b}\; (Wspaniała(b, s) \vee Dobra(b, s))] \Rightarrow Akcja(a, s)$

$\forall a,s \;\; Ryzykowna(a, s) \wedge [\neg \exists \mathrm{b}\; (Wspaniała(b, s) \vee Dobra(b, s) \vee Przeciętna(b, s))] \Rightarrow Akcja(a, s).$

Zmiana sytuacji powinna wpływać na **zmianę** ważności poszczególnych akcji.

Należy zdefiniować reguły nadające akcjom priorytety zależnie od relacji spełnionych w danej sytuacji.

Np.

$\forall \mathbf{x},s \;\; PrzyZłocie(s) \wedge \neg TrzymaZłoto(\mathbf{x},s) \Rightarrow Wspaniała(Podnieś, s)$

$\forall \mathbf{x},s \;\; \exists \mathbf{y} \;\; Jest(Agent, \mathbf{x}, s) \wedge Sąsiedzi(\mathbf{x}, \mathbf{y}) \wedge \neg Wumpus(\mathbf{y}) \wedge \neg Jama(\mathbf{y}) \wedge Nieodwiedzany(\mathbf{y}, s) \Rightarrow Dobra(Wprzód, s)$

Zad. 3 Skolemizacja formuły

Dokonać skolemizacji następującej formuły:

$$\forall x \forall y \, (P(x,y) \Rightarrow \exists z (\forall y \; Q(y,z) \wedge \neg R(x,z))) \vee \forall z \; Q(x,y,z)$$

Rozwiązanie 3.

1. Standaryzacja rozłączna (przemianowanie) zmiennych – każdy kwantyfikator dotyczy innej zmiennej.

$$\forall x \forall y (P(x,y) \Rightarrow \exists z (\forall v \; Q(v,z) \wedge \neg R(x,z))) \vee \forall w \, Q(x,y,w)$$

2. Przeniesienie kwantyfikatorów w lewo przed formułę

$$\forall x \forall y \exists z \forall v \forall w (P(x,y) \Rightarrow (Q(v,z) \wedge \neg R(x,z))) \vee Q(x,y,w)$$

3. Właściwy krok skolemizacji formuły (→)

Rozwiązanie 3 (c.d.)

3. Skolemizacja formuły polega teraz na zastąpieniu wystąpień zmiennych związanych kwantyfikatorami szczegółowymi (egzystencjalnymi) przez tzw. „stałą Skolema” (jeśli ta zmienna nie zależy od innych zmiennych) lub „funkcją Skolema” (jeśli zależy od innych zmiennych w formule). Ta operacja nie zmienia niczego w zachodzeniu formuły – przechodzimy na równoważną postać. „Stała” względnie „funkcja Skolema” to nieokreślona stała (wartość, obiekt) względnie wynik funkcji i wyraża to samo co określenie „istnieje jakieś z”.

$$\forall x \forall y \forall v \forall w (P(x,y) \Rightarrow (Q(v, F1(x,y)) \wedge \neg R(x, F1(x,y)))) \vee Q(x,y,w)$$

4. Teraz możemy opuścić kwantyfikatory ogólne

$$(P(x,y) \Rightarrow (Q(v, F1(x,y)) \wedge \neg R(x, F1(x,y)))) \vee Q(x,y,w)$$

Zadanie 4. Unifikacja formuł

Dokonać unifikacji następujących par formuł:

1) $P(A, f(g(x))) \wedge Q(g(y), B) \Rightarrow R(x, C)$

$P(y, f(v)) \wedge Q(z, B) \Rightarrow R(g(z), z)$

2)

$\neg P(z, A, f(y)) \wedge (Q(y, B) \Rightarrow R(C, g(z))) \vee S(f(A), g(B), z)$

$\neg P(A, v, f(A)) \wedge (Q(z, x) \Rightarrow R(w, g(A))) \vee S(f(z), g(x), y)$

Rozwiązanie 4

Zakładamy, że x, y, z są zmiennymi a A, B, C – stałymi.

1) Uwzględniamy kolejne pary literałów:

- $\text{UNIFY}(\theta = \varnothing, P(.), P(.)) = \{y/A,\ v/g(x)\} = \theta_1$
- $\text{UNIFY}(\theta_1, Q(.), Q(.)) = \{y/A,\ v/g(x),\ z/g(y)\} = \theta_2$
- $\text{UNIFY}(\theta_2, R(.), R(.)) = \{y/A,\ v/g(x),\ z/g(y),\ x/g(z),\ z/C\} = fail$

2) $\theta = \{z/A; v/A; y/A; y/z; x/B; w/C; z/A; z/A; x/B; z/y\} =$

$\{z/A; v/A; y/A; x/B; w/C\}$

Zadanie 5. Postać normalna CNF

Przekształcić poniższą formułę do postaci normalnej dla wnioskowania **metodą rezolucji**:

$$\forall x \quad B(x) \Leftrightarrow (\exists y\ P(y,x) \vee L(x,y))$$

- Podać **krok-po-kroku** realizowane przekształcenia.
- **Wyjaśnić** realizowane przekształcenia.

Rozwiązanie 5

1. Standaryzacja rozłączna zmiennych: każdy kwantyfikator korzysta z innej zmiennej – w tym przypadku nie ma potrzeby przemianowania zmiennych:

$$\forall x \quad B(x) \Leftrightarrow (\exists y\ P(y,x) \vee L(x,y))$$

2. Skolemizacja: każda egzystencjalna zmienna (i jej kwantyfikator) jest zastępowana przez **funkcję Skolema** dla zmiennej należącej do obejmującego kwantyfikatora uniwersalnego:

$$\forall x \quad B(x) \Leftrightarrow (P(F(x),x) \vee L(x,F(x))$$

3. Opuszczamy kwantyfikator uniwersalny:

$$B(x) \Leftrightarrow (P(F(x),x) \vee L(x,F(x))$$

Rozwiązanie 5 (c.d.).

4. Zastępujemy równoważność $\Leftrightarrow$ przez dwie implikacje $\Rightarrow$:

$$(B(x) \Rightarrow (P(F(x),x) \vee L(x,F(x))) \wedge ((P(F(x),x) \vee L(x,F(x)) \Rightarrow B(x))$$

5. Stosujemy dwukrotnie regułę eliminacji implikacji:

$$(\neg B(x) \vee (P(F(x),x) \vee L(x,F(x))) \wedge (\neg(P(F(x),x) \vee L(x,F(x)) \vee B(x))$$

6. Przesuwamy „zewnętrzne" negacje do symboli atomowych:

$$(\neg B(x) \vee P(F(x),x) \vee L(x,F(x)) \wedge ((\neg P(F(x),x) \wedge \neg L(x,F(x))) \vee B(x))$$

7. Rozdzielamy $\vee$ względem $\wedge$:

$$(\neg B(x) \vee P(F(x),x) \vee L(x,F(x)) \wedge (\neg P(F(x),x) \vee B(x)) \wedge (\neg L(x,F(x)) \vee B(x))$$

Uwaga: Stosując następnie regułę eliminacji koniunkcji dla formuł uznanych za prawdziwe otrzymujemy zbiór klauzul dodawanych do bazy wiedzy:

$$(\neg B(x) \vee P(F(x),x) \vee L(x,F(x)),$$

$$(\neg P(F(x),x) \vee B(x)),$$

$$(\neg L(x,F(x)) \vee B(x))$$

Zadanie 6. Wnioskowanie w przód

Zakładamy regułę modelowanego świata:

• *Według prawa przestępstwo popełnia Amerykanin, który sprzedaje broń wrogim narodom*.

Znamy fakty:

• *Kraj Nono, wróg Ameryki, posiada rakiety, które sprzedał im pułkownik West, będący Amerykaninem.*

Sprawdzić, czy *„pułkownik West jest przestępcą”*, stosując procedurę wnioskowania w przód w logice predykatów.

Rozwiązanie 6

Najpierw należy wyrazić regułę i fakty w języku predykatów. Następnie należy dodać ogólną wiedzę (aksjomaty) o modelowanej dziedzinie, na tyle, aby móc wyrazić predykaty występujące w regule za pomocą predykatów reprezentujących fakty.

Rozwiązanie 6 (c.d.)

Baza wiedzy zawiera regułę - implikację o postaci Horna:

... *jest przestępstwem, gdy Amerykanin sprzedaje broń wrogim narodom*:

(1) *Amerykanin*(*x*) ∧ *Broń*(*y*) ∧ *Sprzedaje*(*x,y,z*) ∧ *Wrogi*(*z*) ⇒ *Przestępca*(*x*)

Dalsze implikacje w postaci Horna – wiedza o świecie:

… *wszystkie ich rakiety sprzedał im płk.* West:

(2) *Rakieta*(*x*) ∧ *Posiada*(Nono, *x*) ⇒ *Sprzedaje*(West, *x*, Nono)

Rakiety to broń:

(3) *Rakieta*(*x*) ⇒ *Broń*(*x*)

Wróg Ameryki jest „wrogi”:

(4) *Wróg*(*x*, Ameryka) ⇒ *Wrogi*(*x*)

Teraz po kolei dodajemy 4 formuły atomowe:

- *West jest Amerykaninem …* : *Amerykanin*(West) (5)
- *Kraj Nono jest wrogiem Ameryki …* : *Wróg*(Nono, Ameryka) (6)
- *Nono … posiada rakiety*, tzn. ∃x *Posiada*(Nono,*x*) ∧ *Rakieta*(*x*)
  - Tu wymagana jest redukcja kwantyfikatora egzystencjalnego - skolemizacja: *Posiada*(Nono, M1), (7); *Rakieta*(M1) (8)

Rozwiązanie 6 (c.d.)

Następnie realizujemy przemianowanie zmiennych (standaryzacja rozłączna). W naszej bazie wiedzy formuły (1) – (4) posługują się wszystkie zmienną *x*. Pozostawiamy ją jedynie w formule (1) a wprowadzamy w jej miejsce *x2* (w 2), *x3* (w 3) i *x4* (w 4).

Proces wnioskowania w przźód

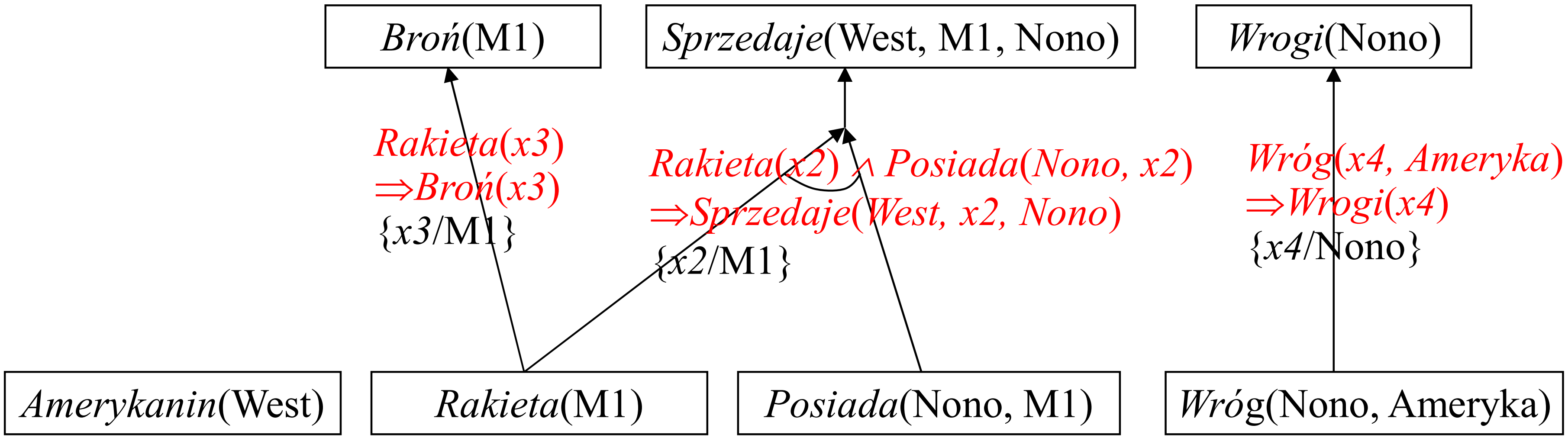
Krok 1) Rozpoczynamy proces wnioskowania wprzód od literałów atomowych - pozbawionych zmiennych:

Amerykanin(West)	*Rakieta*(M1)	*Posiada*(Nono, M1)	*Wróg*(Nono, Ameryka)

Rozwiązanie 6 (c.d.)

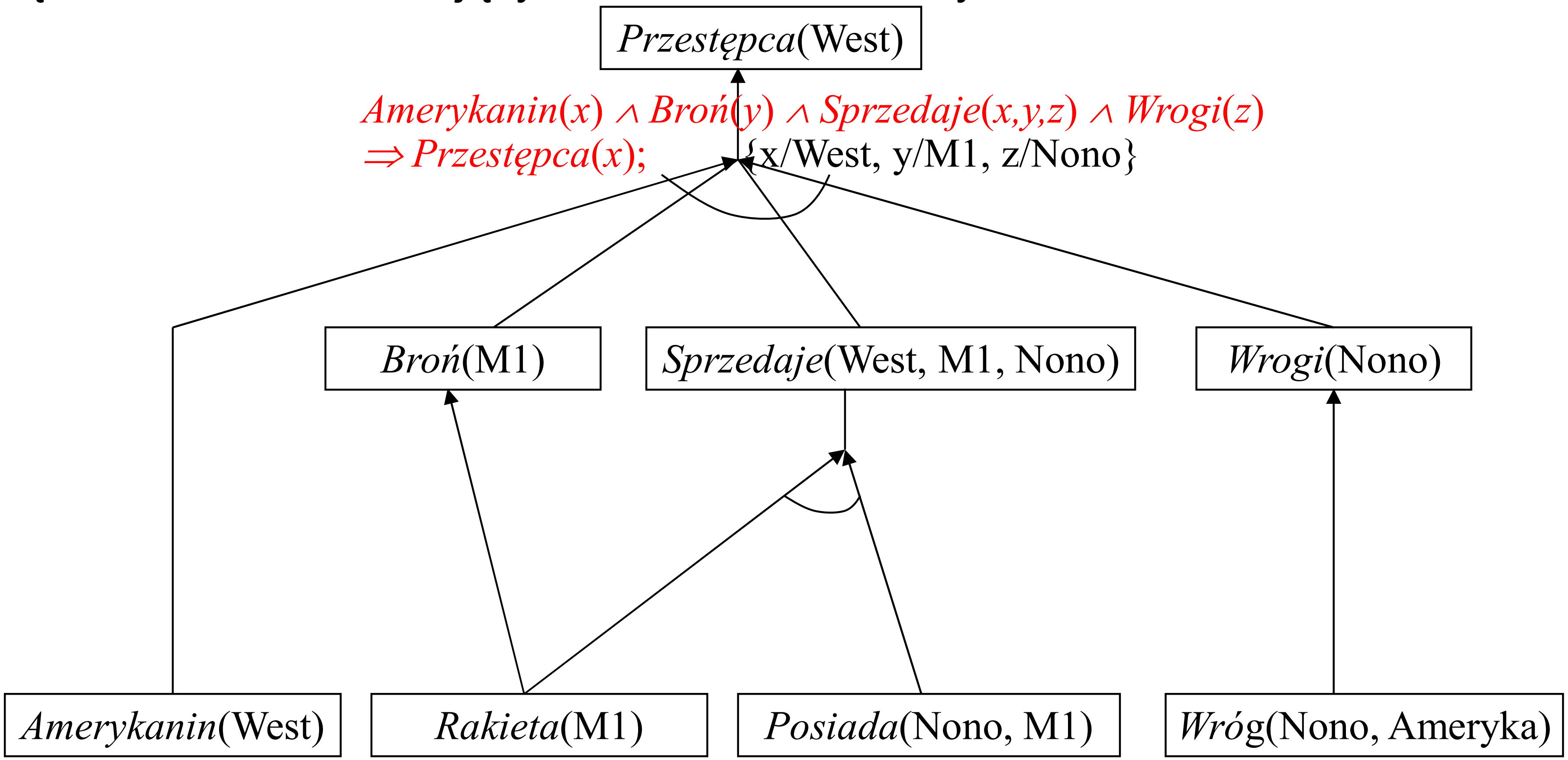
Kroki 2-4: Wybierane są po kolei trzy reguły (3, 2 i 4), których literały w poprzedniku dają się zunifikować z istniejącymi w KB literałami atomowymi.

Każda unifikacja zwykle dodaje podstawienie dla kolejnej zmiennej.



Rozwiązanie 6 (c.d.)

Krok 5: Wybierana jest reguła (1), której wszystkie literały w poprzedniku dają się zunifikować z istniejącymi literałami atomowymi.



Koniec: więcej formuł nie da się już wyprowadzić. Formuła zapytania znajduje się wśród wyprowadzonych literałów, czyli jest spełniona, przy podstawieniu:
{x/West, y/M1, z/Nono, x2/M1, x3/M1, x4/Nono}

Zadanie 7. Wnioskowanie wstecz

Zilustrować działanie funkcji **wnioskowania wstecz** w logice predykatów na przykładzie zbioru formuł z zadania 6 i zapytania „ ∃x *Przestępca*(x)”?

Rozwiązanie 7

Krok 1: Pierwszym celem funkcji wnioskowania wstecz jest formuła zapytania.

BC_ASK(*KB*, 1,)

wywołuje

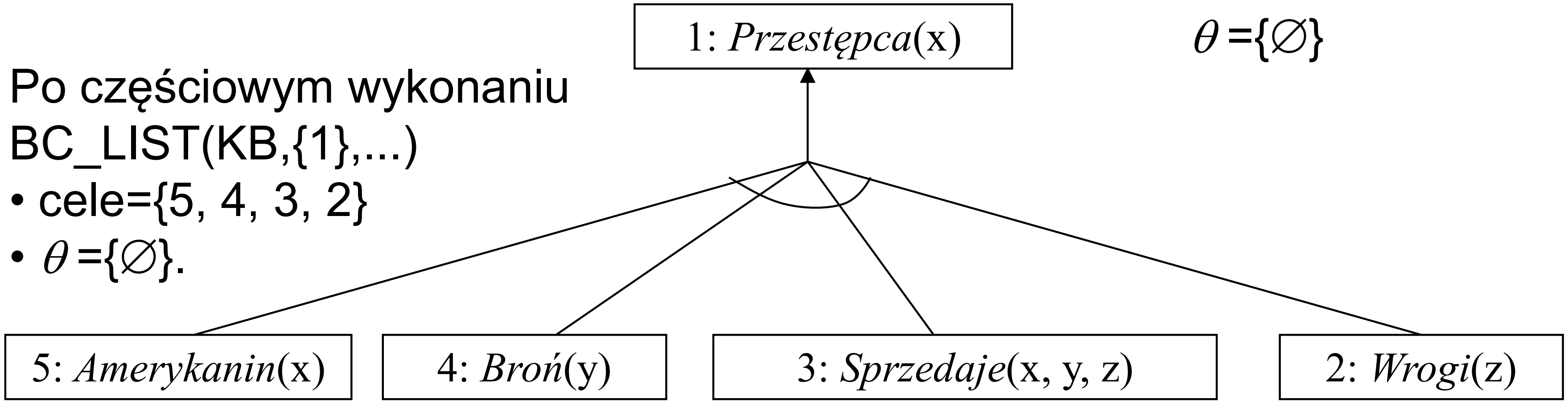
BC_LIST(*KB*,{1}, ...)

cele={1}, θ={∅},

$\boxed{1\text{: } \textit{Przestępca}(\mathrm{x})}$

Rozwiązanie 7 (c.d.)

Po częściowym wykonaniu
BC_LIST(KB,{1},...)
- cele={5, 4, 3, 2}
- θ ={∅}.

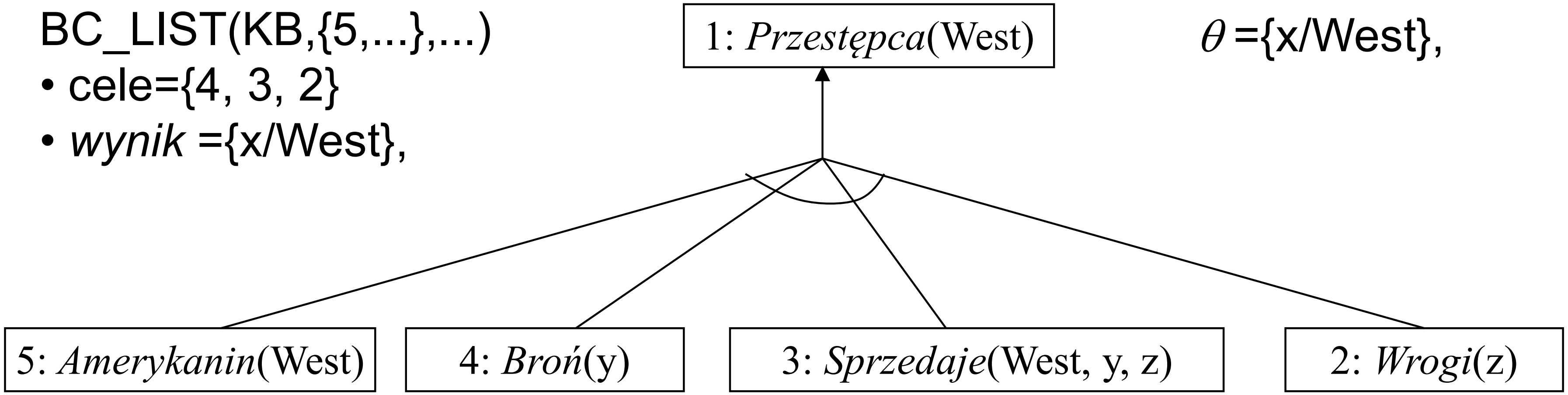


Z nowymi celami 5, 4, 3 i 2 jako parametrem wywoływana jest kolejno rekursywna funkcja BC_LIST (KB, {cel}, …)

Rozwiązanie 7 (c.d.)

Po wykonaniu
BC_LIST(KB,{5,...},...)
- cele={4, 3, 2}
- *wynik* ={x/West},

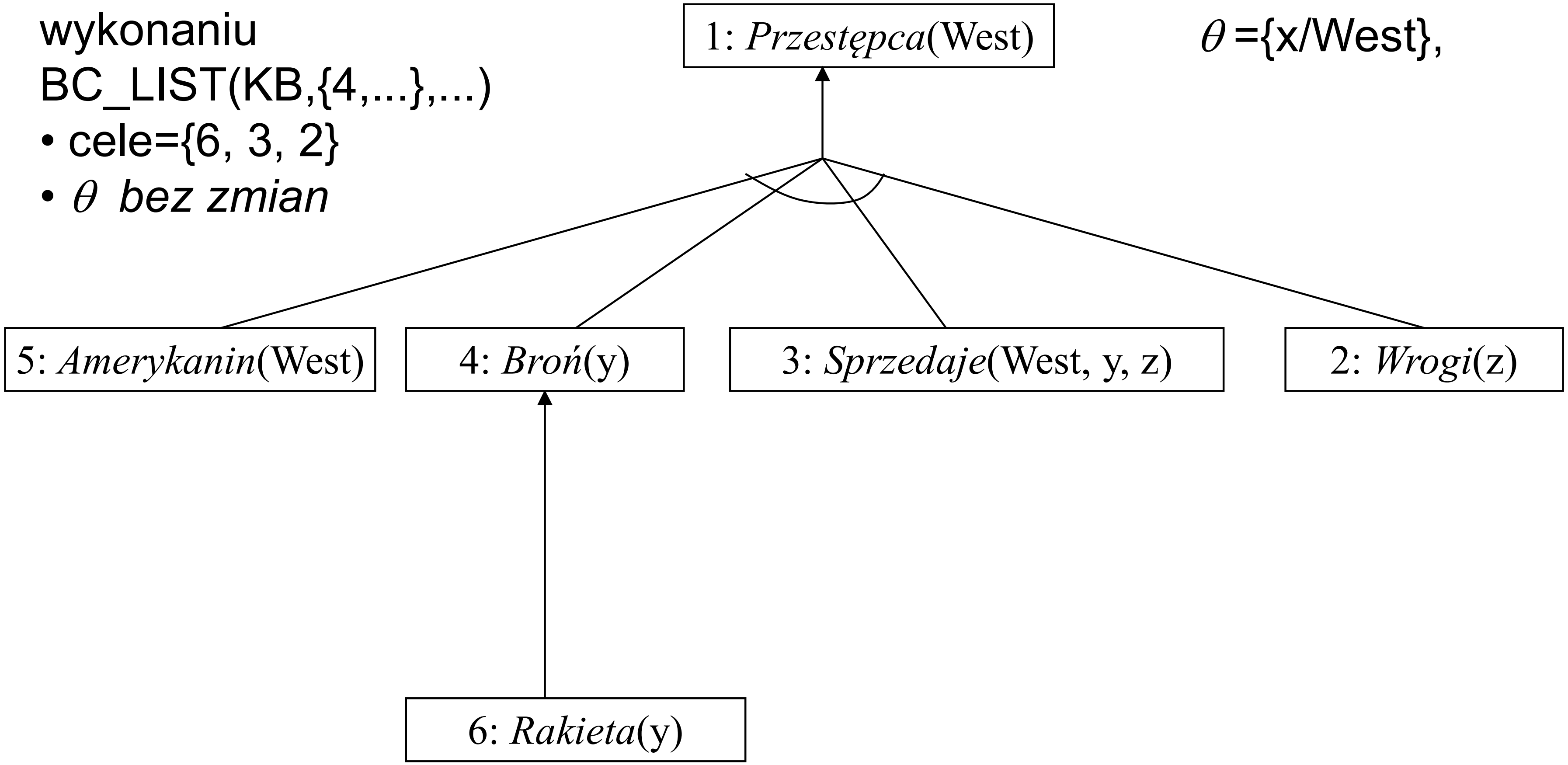
θ ={x/West},



{ x/West}

Rozwiązanie 7 (c.d.)

Po częściowym wykonaniu BC_LIST(KB,{4,...},...)
- cele={6, 3, 2}
- *θ bez zmian*

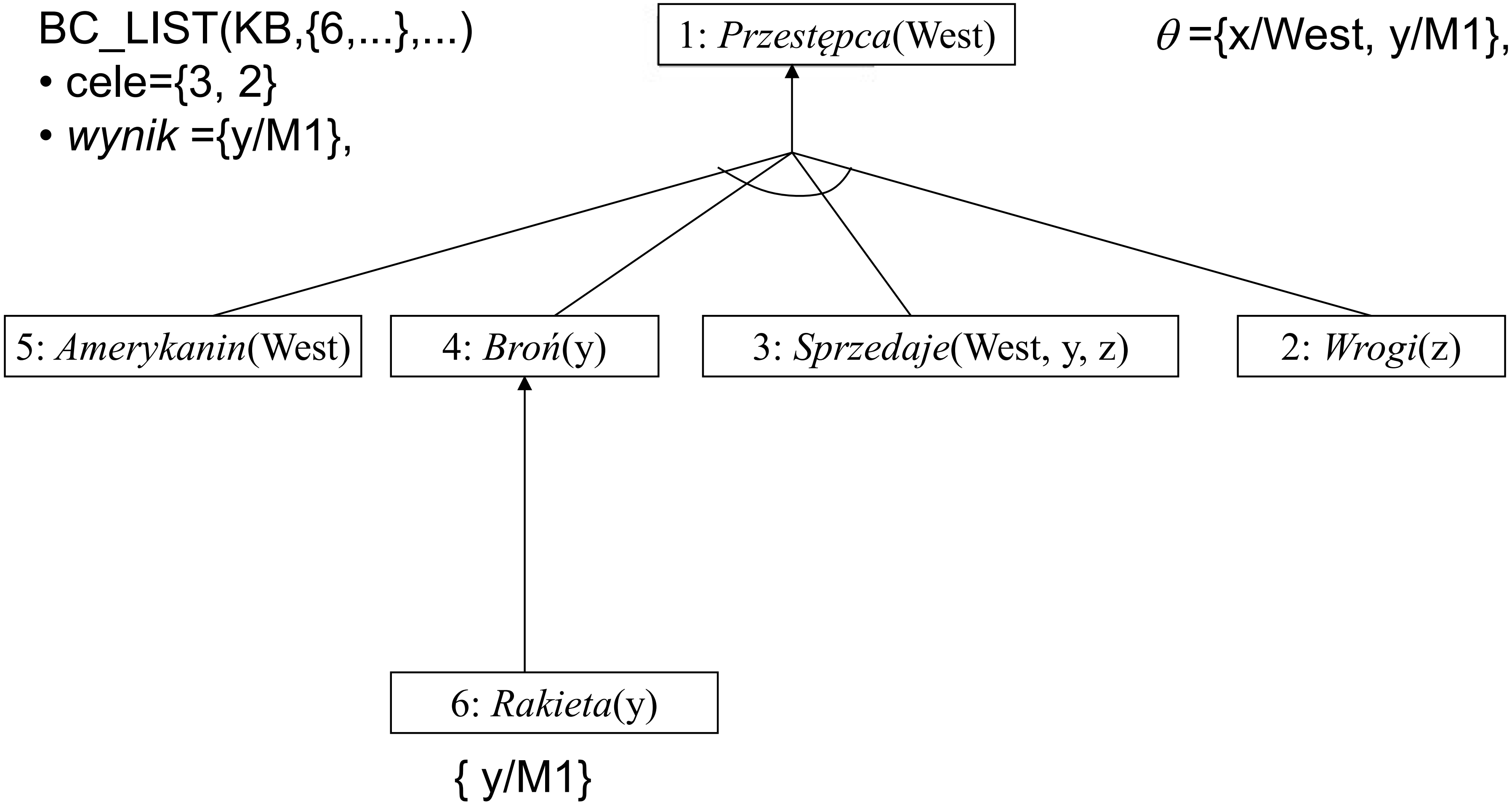


θ ={x/West},

Rozwiązanie 7 (c.d.)

Po wykonaniu
BC_LIST(KB,{6,...},...)
- cele={3, 2}
- *wynik* ={y/M1},

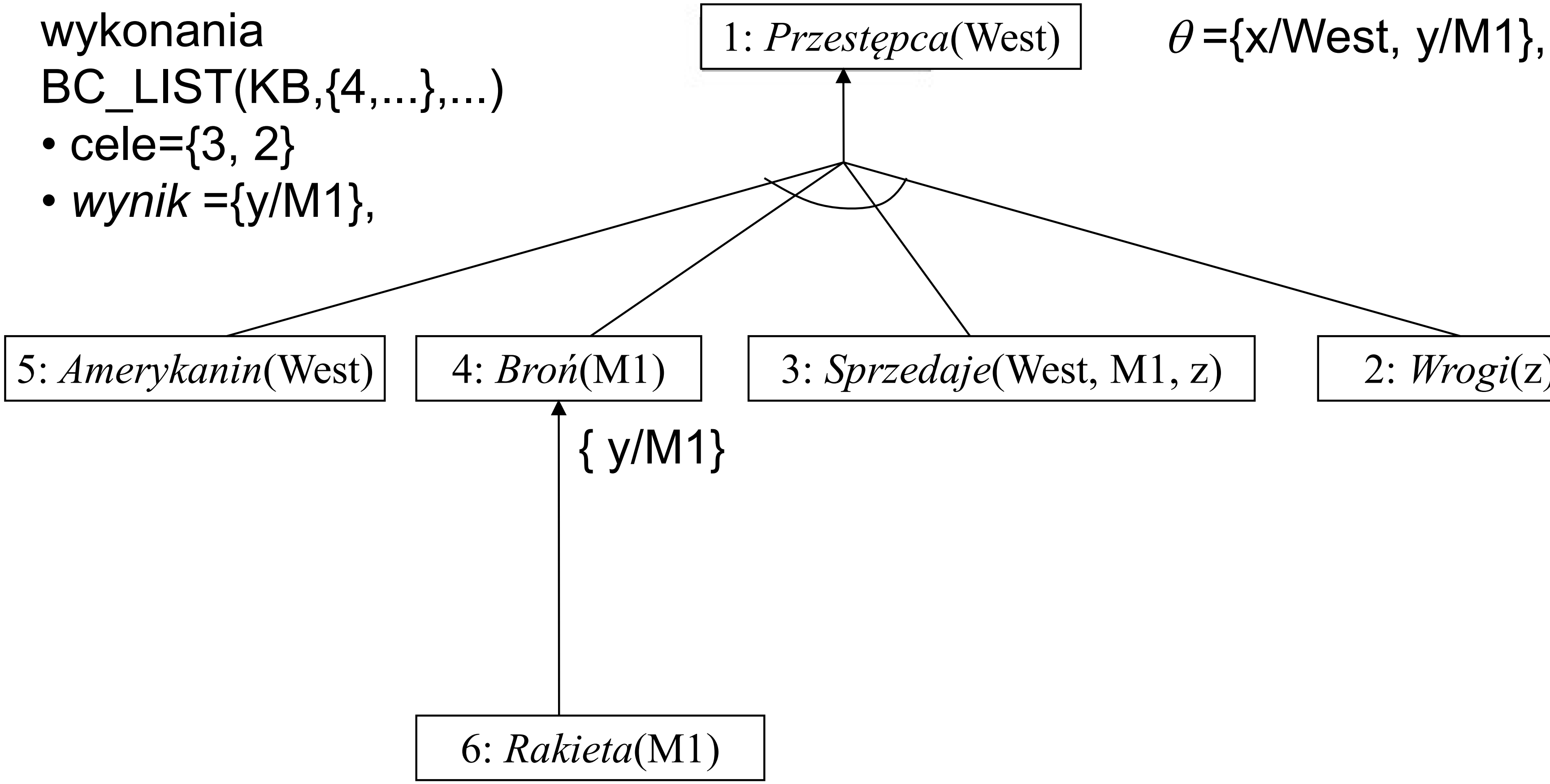
θ ={x/West, y/M1},



Rozwiązanie 7 (c.d.)

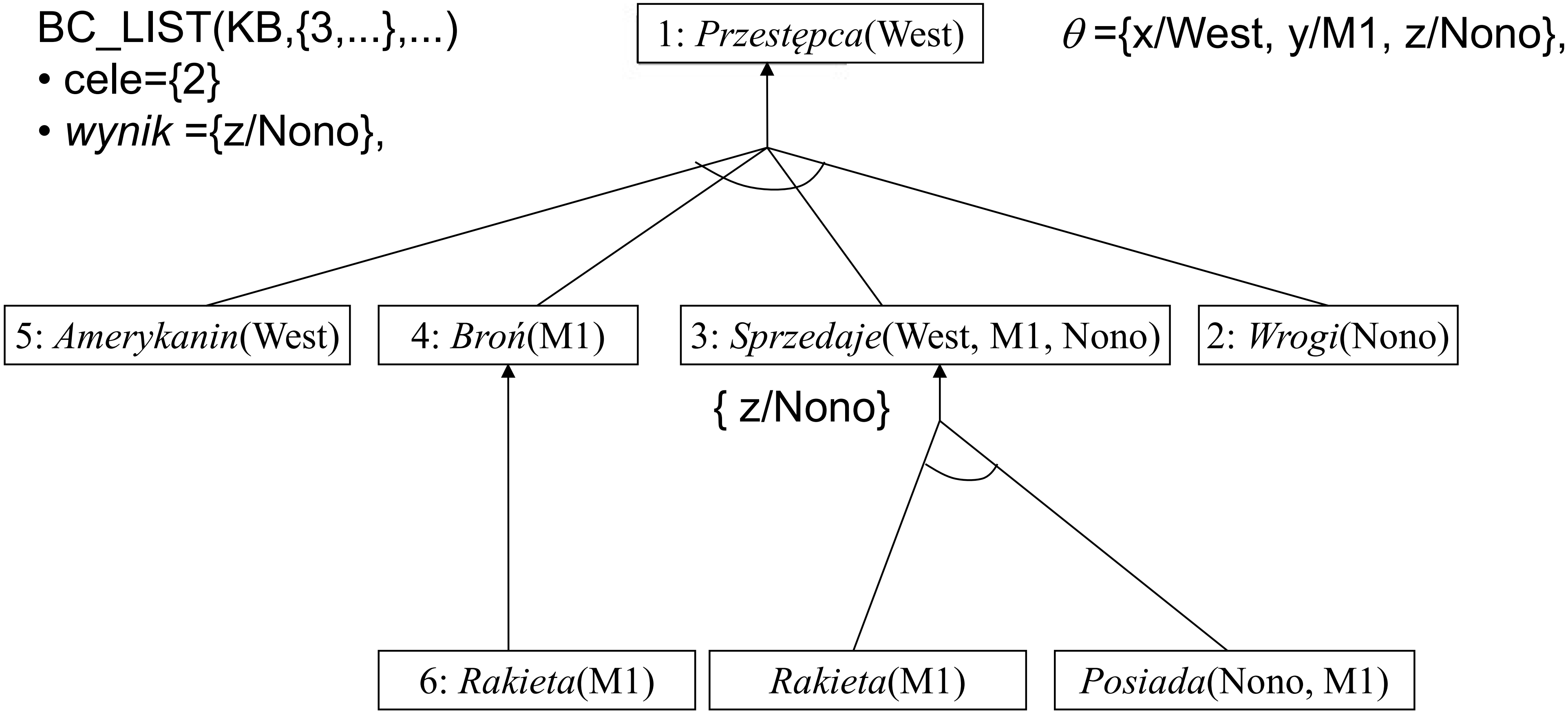
Po dokończeniu wykonania BC_LIST(KB,{4,...},...)

- cele={3, 2}
- *wynik* ={y/M1},



Rozwiązanie 7 (c.d.)

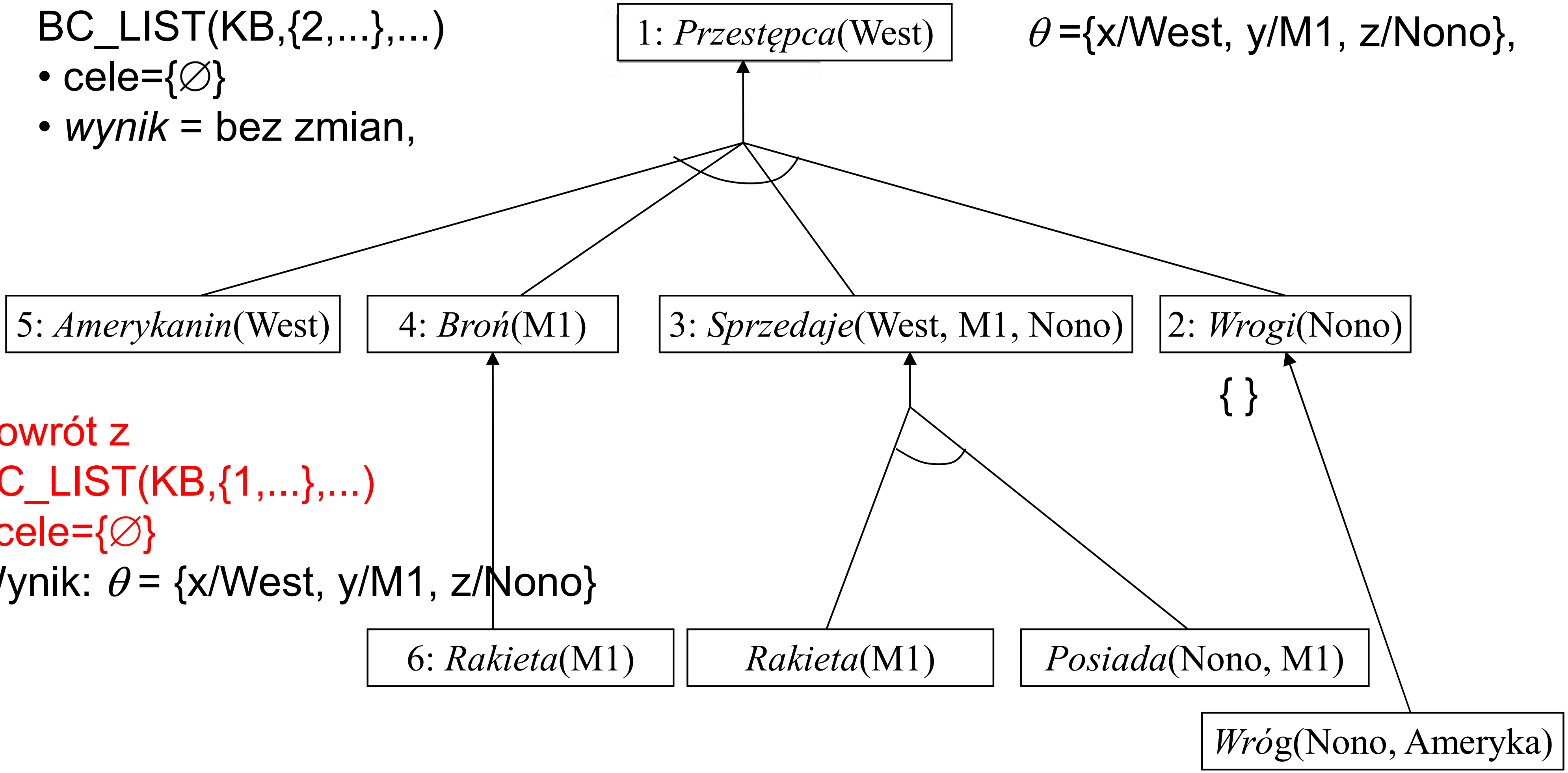
Po wykonaniu
BC_LIST(KB,{3,...},...)
- cele={2}
- *wynik* ={z/Nono},



Rozwiązanie zad. 7 (c.d.)

Po wykonaniu
BC_LIST(KB,{2,...},...)
• cele={∅}
• *wynik* = bez zmian,

θ ={x/West, y/M1, z/Nono},



Powrót z
BC_LIST(KB,{1,...},...)
• cele={∅}
Wynik: θ = {x/West, y/M1, z/Nono}

Zad. 8 Wnioskowanie metodą rezolucji

Zilustrować działanie funkcji **wnioskowania metodą rezolucji** w logice predykatów na przykładzie zbioru formuł z zad. 6 i zapytania „*Przestępca*(West)”?

Rozwiązanie 8

W zad. 6 podaliśmy przykład bazy wiedzy: „... *przestępstwo popełnia Amerykanin, który sprzedaje broń wrogim narodom”.*

Najpierw formuły o postaci klauzul Horna należy przekształcić na postać normalną CNF, właściwą dla wnioskowania metodą rezolucji. Następnie należy przemianować zmienne (*standaryzacja rozłączna*) w zdaniach (2-4):

(1) $\mathit{Amerykanin}(x) \wedge \mathit{Broń}(y) \wedge \mathit{Sprzedaje}(x,y,z) \wedge \mathit{Wrogi}(z) \Rightarrow \mathit{Przestępca}(x)$

$\neg\, \mathit{Amerykanin}(x) \vee \neg\, \mathit{Broń}(y) \vee \neg\, \mathit{Sprzedaje}(x,y,z) \vee \neg\, \mathit{Wrogi}(z) \vee \mathit{Przestępca}(x)$

(2) $\mathit{Rakieta}(x) \wedge \mathit{Posiada}(\text{Nono}, x) \Rightarrow \mathit{Sprzedaje}(\text{West}, x, \text{Nono})$

$\neg\, \mathit{Rakieta}(x1) \vee \neg\, \mathit{Posiada}(\text{Nono}, x1) \vee \mathit{Sprzedaje}(\text{West}, x1, \text{Nono})$

(3) $\mathit{Rakieta}(x) \Rightarrow \mathit{Broń}(x) \longrightarrow \neg\, \mathit{Rakieta}(x2) \vee \mathit{Broń}(x2)$

(4) $\mathit{Wróg}(x, \text{Ameryka}) \Rightarrow \mathit{Wrogi}(x) \longrightarrow \neg\, \mathit{Wróg}(x3, \text{Ameryka}) \vee \mathit{Wrogi}(x3)$

Rozwiązanie 8 (c.d.)

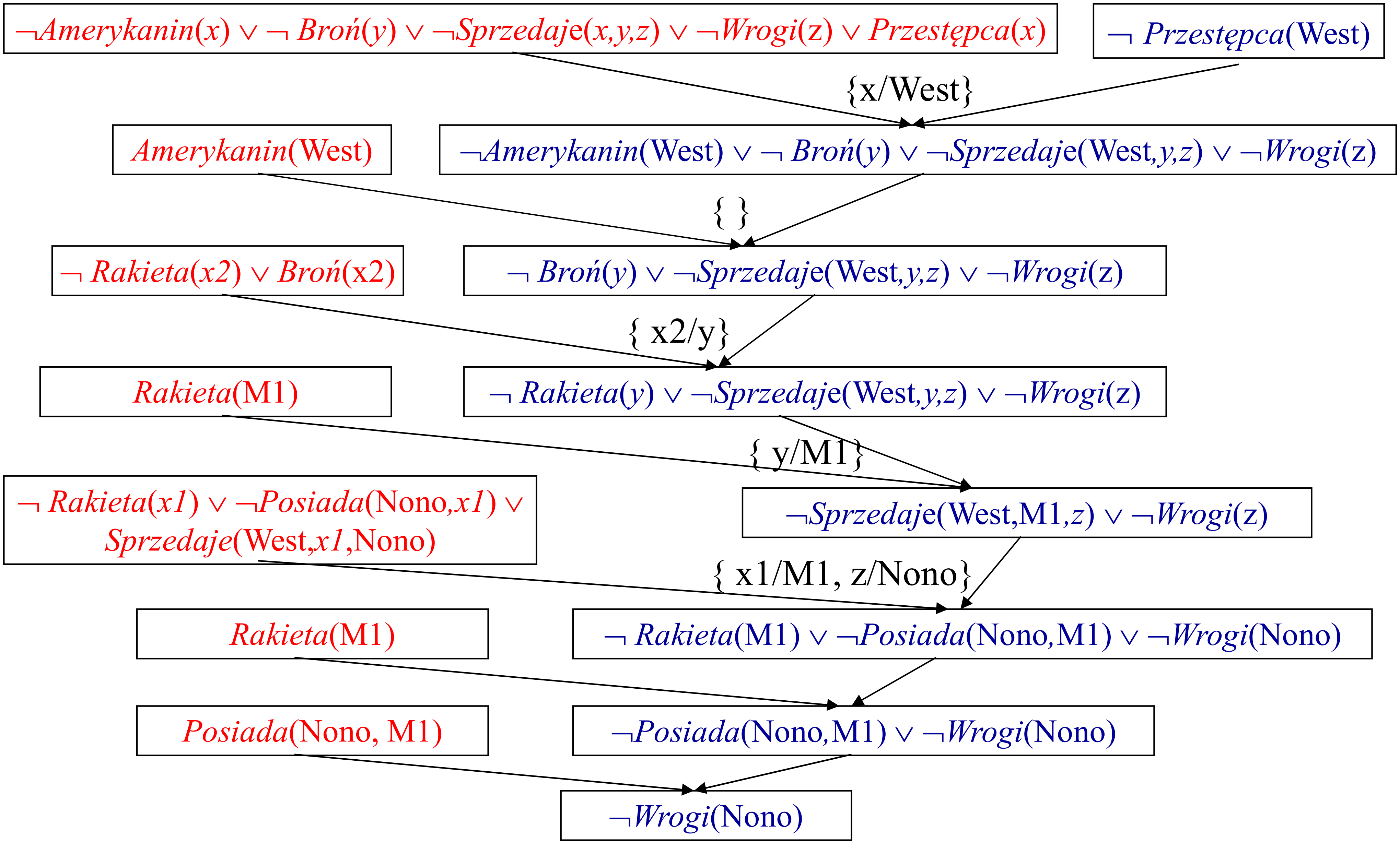
W bazie wiedzy istnieją też 4 formuły atomowe reprezentujące znane nam fakty:

(5) $\mathit{Amerykanin}(\mathrm{West})$
(6) $\mathit{Wróg}(\mathrm{Nono, Ameryka})$
(7) $\mathit{Posiada}(\mathrm{Nono, M1})$
(8) $\mathit{Rakieta}(\mathrm{M1})$

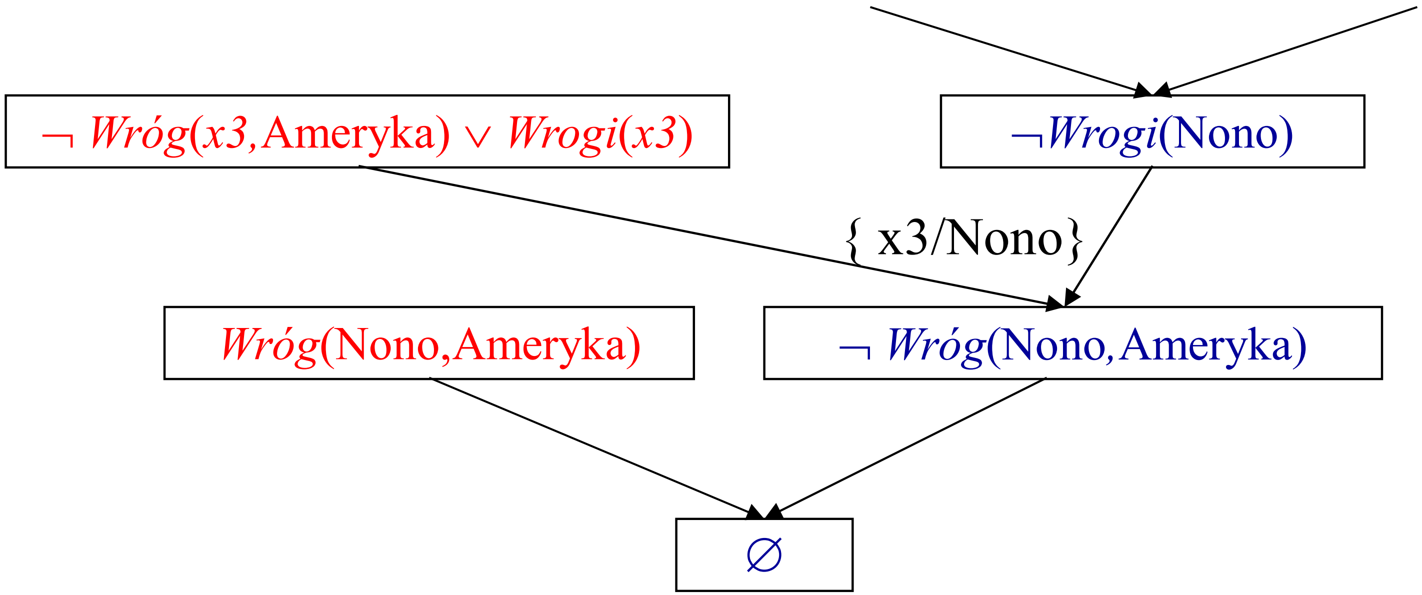
Formuła zapytania jest postaci klauzuli: $\mathit{Przestępca}(\mathrm{West})$.

Dla sprawdzenia, czy wynika ona z bazy wiedzy *KB* (tzn. czy jest spełniona w modelu *KB*), w metodzie rezolucji dodajemy zaprzeczenie formuły zapytania do *KB* oraz sprawdzamy czy doprowadza to bazę wiedzy do sprzeczności i przy jakim wartościowaniu zmiennych.

Rozwiązanie 8 (c.d.)



Rozwiązanie 8 (c.d.)



Uzyskano formułę pustą, czyli wykazano sprzeczność w bazie wiedzy po dodaniu do niej zaprzeczenia formuły zapytania.

Tym samym pokazano, że formuła zapytania wynika z aktualnej bazy wiedzy, przy podstawieniu za zmienne w formułach:

{ x/West, x2/y, y/M1, x1/M1, z/Nono, x3/Nono}

Zad. 9. Wnioskowanie formuł

Sprawdzić, czy z bazy wiedzy KB wynikają (tzn. można wywnioskować) zapytania q_1 , q_2 .

KB: reguły

$$P(x,y) \wedge Q(y,z) \Rightarrow R(x,y)$$

$$R(x_1,y_1) \wedge S(z_1,v) \wedge W(y_1,v) \Rightarrow W(x_1,z_1)$$

$$\neg Q(x_2,y_2) \Rightarrow Q(y_2,x_2)$$

KB: obserwacje

$$P(Ania, Beata), \quad \neg Q(Czarek, Beata)$$

$$S(Darek, Ewa), \quad W(Czarek, Ewa)$$

Formuły zapytań:

$$q_1: \quad \exists_{a,d}\, W(a,d)$$

$$q_2: \quad \exists_{a,d,e}\, W(d,e) \Rightarrow W(a,d)$$

Rozwiązanie 9.

Wskazówka

1. Zastanowić się czy formuły zapytania są zadane we właściwej postaci i dla jakiego mechanizmu wnioskowania

(q_1 : tak – dla wnioskowania wprzód lub wstecz;

q_2 : nie bezpośrednio, ale po przekształceniu można dwukrotnie wykonać wnioskowanie wprzód lub wstecz albo też jednokrotnie wnioskowanie przez rezolucję).

2. Wykonać procedury wnioskowania – zwracać uwagę na unifikację formuł i sprawdzać prawidłowość podstawień!

Zadanie 10. Predykat is_sorted(L) w PROLOGU

Zaimplementować w języku **PROLOG** predykat **sorted**(L), który jest prawdziwy, jeśli lista L jest posortowana.

Rozwiązanie 10.

Listę można definiować wyliczając jej kolejne elementy, np.: ["a","b","c"] lub też podając jej dwa fragmenty, pierwszy element (zwany głową) i listę pozostałych elementów (zwaną ogonem): np. ["a"| ["b","c",„d"]] .

Głowę od ogona oddziela operator |.

Elementami listy mogą być dowolne wartości, w szczególności mogą być to listy, np.: [["a","b"],["c"],"d", []].

Zakładamy sortowanie w kolejności rosnącej.

Zakładamy istnienie predykatu „mniejsze niż" : X<Y

Rozwiązanie 10 (c.d.)

Będą potrzebne alternatywne definicje predykatu, uzależnione od rozmiaru listy (jeden, dwa lub więcej elementów).

Na wszelki wypadek zdefiniujemy najpierw zmienne, którymi będziemy się posługiwać:

domain(X, int). domain(X1, int). domain(X2, int). itd.

domain(Y, int). domain(Y1, int). domain(Y2, int). itd.

domain(T, int[]). domain(T1, int[]). domain(T2, int[]). itd.

Definicja sorted() dla 1-elementowej listy:

Def00: sorted([X]).

Definicja dla 2-elementowej listy:

Def01: sorted([X1,Y1]) :- X1<Y1.

Definicja dla listy o więcej niż 2 elementach:

Def02: sorted([X2],[Y2 | T]) :- X2<Y2, sorted([Y2],[T]).

Zad. 11. Predykat merge(K1, K2, K) w PROLOGU

A) Zaimplementować w języku **PROLOG** predykat **merge**(K1,K2,K), który jest prawdziwy wówczas, gdy listy K1, K2 i K są posortowane, przy czym lista K zawiera wszystkie elementy zawarte w K1 i K2.

Dla uproszczenia założyć, że K1 i L2 są w wywołaniu listami konkretnymi, a K jest zmienną.

B) Jaki będzie przebieg sprawdzania poniższego zapytania?

merge([1,3,5], [2,4], K).

Gdzie 1,2,3,4,5 są stałymi a K – zmienną.

C) Pytanie: czy definicja merge(K1,K2,K) wymaga zmian, aby zrealizować zapytanie: merge([1,3,5], [2,4], [1,2,3,4,5]) , gdzie 1,2,3,4,5 są stałymi ?

Rozwiązanie 11. Predykat **merge**

Rozwiązanie A) dla „merge(lista, lista, zmienna)"

def10 merge([],[],[]).

def20 merge(L,[],L):- sorted(L). Jeśli lista 2 jest pusta, to L jest tożsama z listą 1.

def30 merge([],L1,L1):- sorted(L1). Jeśli lista 1 jest pusta, to L1 jest tożsama z listą 2.

Jeśli obie listy 1 i 2 są niepuste, to lista 1 powinna być sklejona z listą 2, dając L4 :

def00 merge(L2, L3, L4):-
sorted(L2), sorted(L3), mergeA(L2, L3, L4).

def1 mergeA(L5, L6, L7):-
L5=[X1 | T1], L6=[Y1 |T2], X1≤Y1, L7 = [X1| L8], mergeA(T1, L6, L8).

def2 mergeA(L9,L10,L11):-
L9=[X2|T3], L10=[Y2|T4], Y2<X2, L11=[Y2|L12], mergeA(L9, T4, L12).

def3 mergeA([],[],[]).

Rozwiązanie 11. Predykat **merge** (2)

Rozwiązanie B) Cel: merge([1,3,5], [2,4], K) ?

Stosuj def00 merge(L2, L3, L4):- $\theta 1$ ={L2|[1,3,5], L3|[2,4], K|L4}

Cele: (1)sorted([1,3,5]), (2)sorted([2,4]), (3)mergeA([1,3,5],[2,4], L4)

Cel (1): stosuj def02 sorted i def01 sorted → cel (1) spełniony

Cel (2): stosuj def02 sorted i def01 sorted → cel (2) spełniony

Dla (3) stosuj def1: $\theta 2$ ={L4|L7} → X/1, T1/[3,5], L5=[1 |[3,5]], Y1/2, T2/[4], L6=[2| 4], 1≤2, L7=[1|L8]

Próbuj def1 ponownie: X1/3, T1/[5], L5=[3 | 5], Y1/2, T2/[4], L6=[2| 4], 3≤2 → porażka (nawrót)

Próbuj def2: $\theta 3$ ={L8|L11} → X2/3, T3/[5], L9=[3 | 5], Y2/2, T4/[4], L10=[2| 4], 2<3 , L11=[2 | L12]

Próbuj def1: $\theta 4$ ={L12|L7'} → X'/3, T1'/[5], L5'=[3|5], Y1'/4, T2'/[], L6'=[4], 3≤4, L7'=[3 | L8']

Próbuj def1: X''/5, T1''/[], L5''=[5], Y1''/4, T2''/[], L6''=[4], 5≤4 porażka (nawrót)

Rozwiązanie 11. Predykat **merge** (3)

Próbuj def2: θ ={L8'|L11'} → X2'/5, T2'/[], L9'=[5], Y2'/4, T4'/[], L10'=[4], 4<5 , L11'=[4|L12']

Próbuj def1: θ ={L12'|L7''} → X''/5, T1''/[], L5''=[5], Y1''/null, T2''/[], L6''=[], 5<null, L7''=[5 |L8'']

Stosuj def3: θ ={L8''|[]} → T1''=[], L6''=[], L8''=[] → cel (3) jest spełniony przy podstawieniach:

L8''= [], L7''=[5 |L8''] = [5]

{L12'|L7''} → L12' = [5], L11'=[4|L12'] = [4, 5]

{L8'|L11'} → L8' = [4, 5], L7'=[3 | L8'] = [3, 4, 5]

{L12|L7'} → L12 = [3, 4, 5], L11=[2 | L12] = [2,3,4,5]

{L8|L11} → L8 = [2,3,4,5], L7 = [1 | L8] = [1,2,3,4,5]

{L4|L7} → L4 = [1,2,3,4,5]

{K|L4} → **K = [1,2,3,4,5] → Odpowiedź**

C) Definicja nie wymaga zmian dla zapytania pozbawionego zmiennych