REGUŁOWO-MODELOWE SKORUPOWE SYSTEMY EKSPERTOWE

Część 2: Systemy rozwinięte dokładne

Antoni Niederliński

Uniwersytet Ekonomiczny

w Katowicach

antoni. niederlinski@ue.katowice. pl

Rozwinięte dokładne bazy wiedzy

1) Baza reguł rozwinięta dokładna
2) Baza modeli rozwinięta dokładna
3) Baza ograniczeń rozwinięta dokładna
4) Baza rad rozwinięta dokładna
5) Pliki rad
6) Baza grafiki
7) Pliki grafiki
8) Baza dźwięków
9) Pliki dźwięków
10) Plik opisu wiedzy dziedzinowej

Baza reguł rozwinięta dokładna

```
reguła(Nr, "Wniosek",
          Lista_warunków, Semafor)

fakt("Warunek dopytywalny")

fakt("nWarunek dopytywalny")
```

Baza reguł rozwinięta dokładna

reguła(Nr, "Wniosek",
Lista_warunków, Semafor)

Wniosek jest prawdą, jeżeli wszystkie warunki są prawdą.

Wszystkie reguły mają wyłącznie niezanegowane wnioski

Baza reguł rozwinięta dokładna

reguła(Nr, "Wniosek",
Lista_warunków, Semafor)

Jeżeli którykolwiek z warunków nie jest prawdą, wniosek jest uważany za nieprawdziwy.

Jest to tzw. założenie zamkniętego świata.

Założenie zamkniętego świata

Dla wszystkich systemów ekspertowych rozwiniętych dokładnych zakłada się, że:

uważa się za nieprawdę to, co nie wynika:

- z reguł bazy reguł,
- z ograniczeń bazy ograniczeń,
- z modeli bazy modeli,
- z faktów i wartości argumentów zadeklarowanych przez użytkownika

Założenie zamkniętego świata

Wszystko, co nie wynika z bazy reguł, bazy ograniczeń i bazy modeli oraz z deklaracji użytkownika , uważa się za nieprawdziwe.

Założenie zamkniętego świata leży u podstaw różnicy pomiędzy implikacją logiki a implikacją regułową rozwiniętego dokładnego systemu ekspertowego:

Implikacja logiki: $q \Rightarrow p$

q	p	$q \Rightarrow p$
Prawda	Prawda	Prawda
Nieprawda	Prawda	Prawda
Nieprawda	Nieprawda	Prawda

Uwzględnienie całej wiedzy dziedzinowej sprawia, że nieprawda warunku pociąga za sobą nieprawdę wniosku

Implikacja regułowa: $q \rightarrow p$

q	p	$q \rightarrow p$
Prawda	Prawda	Prawda
Nieprawda	Nieprawda	Prawda

Dlatego interpretujemy implikację regułową następująco:

„Założenie zamkniętego świata” a prawo rzymskie

„in dubio pro reo”

wątpliwości przemawiają na rzecz oskarżonego

Jeżeli brak dowodów winy, oskarżonego uznaje się za niewinnego

Baza reguł rozwinięta dokładna

Jeżeli warunek dopytywalny:

Warunek

nie jest prawdą, to uważa się za prawdę warunek zanegowany:

nWarunek

Przedrostek ’n’ oznacza negację

Baza reguł rozwinięta dokładna

Jeżeli wniosek reguły:

reguła(Nr, "Wniosek",
Lista_warunków, Semafor)

nie jest prawdą, to uważa się za prawdę wniosek zanegowany:

nWniosek

Przedrostek ’n’ oznacza negację

Baza reguł rozwinięta dokładna

Bazy reguł mogą zawierać zanegowane warunki niedopytywalne.

Innymi słowy: dopuszczalne jest zagnieżdżanie z negacją.

Brak potrzeby definiowanie reguł dla zanegowanych wniosków.

Zagnieżdżanie bez negacji : wniosek reguły jest warunkiem innej reguły

reguła(N, "Wniosek_N", [..., "ABCD",....],1)

reguła(M, "ABCD", Lista_warunków_M,1)

Zagnieżdżanie z negacją : zanegowany wniosek reguły jest warunkiem innej reguły

reguła(N, "Wniosek_N", [..., "nABCD",....],1)

reguła(M, "ABCD", Lista_warunków_M,1)

Zagnieżdżanie z negacją

reguła(1,”pojadę na wycieczkę zagraniczną”,
[”dostanę urlop”,”będę miał pieniądze”],1)

Reguła podstawowa

reguła(2,”Narastające zmęczenie”,
[”npojadę na wycieczkę zagraniczną”], 1)

Reguła potrzebująca negacji

Brak potrzeby definiowanie reguł dla zanegowanych wniosków

Korzyści zagnieżdżania z negacją

* Mniejsza baza reguł

* Reguły bardziej zrozumiałe

* Reguły łatwiejsze do modyfikacji

Przykład: reguła *i* z zagnieżdżeniem (1)

```
reguła(1, "Wniosek_i",
        ["nWarunek_i1", "nWarunek _i2"],1)
```

```
reguła(2, "Warunek_i1",
        ["Warunek_m1", "Warunek _m2"],1)
reguła(3, "Warunek _i2",
        ["Warunek_n1", "Warunek _n2"],1)
```

Przykład: reguła *i* bez zagnieżdżenia (2)

```
reguła (11, "Wniosek_i",
        ["nWarunek_m1", "nWarunek _n1"],1)

reguła (12, "Wniosek_i",
        ["nWarunek _ m2,  "nWarunek _ n1],1)

reguła(13, "Wniosek_i",
        ["nWarunek _ m1,  "nWarunek _ n2],1)

reguła(14, "Wniosek_i",
        ["nWarunek_m2", "nWarunek _ n2 "],1)
```

Przykład: reguła *i* bez zagnieżdżenia (3)

Pojedyncza zagnieżdżona reguła *i* =

Cztery płaskie reguły *i_1, i_2, i_3, i_4*

Praktycznie rozmiary baz reguł płaskich mogą być bardzo duże w porównaniu z odpowiednimi bazami zagnieżdżającymi się.

Baza ograniczeń rozwinięta dokładna

Baza ograniczeń zawiera zbiory warunków dopytywalnych wykluczających się, np.:

("temperatura jest wyższa od 40ºC", "temperatura jest niższa od 10 ºC", "temperatura jest pomiędzy 10ºC i 40ºC").

lub:

("dostanę urlop", "nie dostanę urlopu")

Baza ograniczeń rozwinięta dokładna

Pary dychotomicznych warunków dopytywalnych mogą być deklarowane w bazie ograniczeń rozwiniętej dokładnej, np. :

("dostanę urlop", "nie dostanę urlopu")

Można tego jednak uniknąć wprowadzając zanegowane warunki dopytywalne :

"ndostanę urlop"

Przedrostek 'n' oznacza negację

Baza ograniczeń rozwinięta dokładna

Klauzule bazy ograniczeń rozwiniętej dokładnej:

ograniczenie(Nr_ograniczenia,
Lista_wykluczających_się_
warunków_dopytywalnych)

Tak samo jak dla baz elementarnych

Baza modeli rozwinięta dokładna

- Modele rozwinięte podstawowe dokładne
- Modele rozwinięte rozszerzone dokładne
- Modele rozwinięte liniowe dokładne
- Modele rozwinięte wielomianowe dokładne

Baza modeli

	arytmetyczne	relacyjne
podstawowe	X	X
rozszerzone	X	X
liniowe	X	
wielomianowe	X	

Rodzaje zmiennych łańcuchowych w modelach:

- logiczne zmienne łańcuchowe, np.:

"Relacja spełniona",
"Wartość zmiennej w zakresie dopuszczalnym",

- rzeczywiste i całkowite zmienne łańcuchowe, np.:

"Wartość ciśnienia", "56.79", "56"

Modele unikatowe i wielokrotne

Model rozwinięty dokładny jest unikatowy, jeżeli baza modeli nie zawiera innego modelu o tym samym Wyniku/Wniosku

W przeciwnym przypadku model jest modelem wielokrotnym.

Dla wszystkich modeli rozwiniętych dokładnych (1)

Numer_Modelu jest liczbą całkowitą, różną dla różnych modeli i różną od numerów reguł.

Stosowanie numerów różnych wynika stąd, że rady odwołują się do numerów reguł lub numerów modeli relacyjnych.

Dla wszystkich modeli rozwiniętych dokładnych (2)

- "Warunek startowy" jest logiczną zmienną łańcuchową;
- Wynik/Wniosek modelu jest wyznaczany tylko gdy Warunek startowy jest prawdą
- Warunek startowy = bez warunku jest zawsze prawdą

Dla wszystkich modeli rozwiniętych dokładnych (3)

- Warunek startowy różny od bez warunku wskazuje tylko model wielokrotny, który w danej sytuacji należy zastosować.
- Warunek startowy różny od bez warunku nie może wskazywać modelu unikatowego; Stosowanie dla modelu unikatowego takiego warunku startowego może doprowadzić do przerwania wnioskowania.
- Warunek startowy może być zanegowanym wnioskiem reguły lub modelu relacyjnego

Dla wszystkich modeli rozwiniętych dokładnych (4)

Semafor_wyświetlania

= 0 informacja o stosowaniu modelu nie jest wyświetlana w trakcie wnioskowania

= 1 informacja o stosowaniu modelu jest wyświetlana w trakcie wnioskowania

Modele rozwinięte podstawowe dokładne

```
model(Numer_modelu,
        "Warunek startowy",
        "Wynik"/"Wniosek",
        "Pierwszy Argument",
       "Operacja"/"Relacja",
       "Drugi Argument",
        Semafor_wyświetlania)
```

Model rozwinięty podstawowy dokładny:

"Wynik"/"Wniosek"

- **"Wynik"**– rzeczywista zmienna łańcuchowa dla modelu arytmetycznego

- **"Wniosek"** - logiczna zmienna łańcuchowa dla modelu relacyjnego

Model elementarny rozwinięty dokładny:

"Pierwszy_Argument", "Drugi_Argument”

rzeczywiste lub całkowite zmienne łańcuchowe, zarówno dla modelu arytmetycznego jak i dla modelu relacyjnego

Model elementarny rozwinięty dokładny:

"Operacja"

- modele arytmetyczne z dwoma argumentami wykonują operacje:

"+", " – ", "*", "/", "div", "mod",

"min", "max", "%", "A^N",

"zaokrąglenie_do_N"

N – liczba naturalna

Model elementarny rozwinięty dokładny:

"Operacja"

- modele arytmetyczne z jednym argumentem wykonują operacje

"sqrt", "sin", "cos", "tan", "arctan",

"log", "ln", "exp", "round", "trunc",

"abs", "="

przy czym **"Drugi_Argument" = "0"**

Model elementarny rozwinięty dokładny:

"Relacja"

- modele relacyjne testują relacje:

">", "==", "<", ">=",

"<=", "><", "<>"

Sens modelu elementarnego rozwiniętego arytmetycznego dokładnego:

model(Nr_Modelu, "Start",
"Wynik", "X1", "+", "X2",
Semafor_wyświetlania)

Jeżeli **Start** jest prawdą

to **Wynik = X1 + X2**

Sens modelu elementarnego rozwiniętego arytmetycznego dokładnego:

model(Nr_Modelu, "Start",
"Wynik", "X1", "+", "X2",
Semafor_wyświetlania)

Jeżeli **Start** nie jest prawdą

to **Wynik** jest nieokreślony

Sens modelu elementarnego rozwiniętego relacyjnego dokładnego:

```
model(Nr_Modelu, "Start",
                 "Wniosek",
                 "X1", "<= ", "X2 ",
                 Semafor_wyświetlania)
```

Jeżeli Start jest prawdą

i X1 <= X2

to Wniosek jest prawdą

Sens modelu elementarnego rozwiniętego relacyjnego:

```
model(Nr_Modelu, "Start",
                 "Wniosek",
                 "X1", "<= ", "X2 ",
                 Semafor_wyświetlania)
```

Jeżeli Start nie jest prawdą

to Wniosek jest nieokreślony

Sens modelu elementarnego rozwiniętego relacyjnego:

model(Nr_Modelu, "Start",
"Wniosek",
"X1", "<= ", "X2 ",
Semafor_wyświetlania)

Jeżeli Start jest prawdą

i X1 > X2

Istotna różnica w porównaniu z modelami baz elementarnych dokładnych

to Wniosek nie jest prawdą, lecz prawdą jest nWniosek

Modele rozszerzone

model_r(Numer_Modelu,
"Warunek_startowy",
"Wynik" /"Wniosek",
"Operacja"/"Relacja",
Lista_Argumentów,
Semafor_wyświetlania)

Model rozszerzony:

"Wynik"/"Wniosek"

- **"Wynik"**– rzeczywista zmienna łańcuchowa dla modelu arytmetycznego

- **"Wniosek"** - logiczna zmienna łańcuchowa dla modelu relacyjnego

Model rozszerzony:

"Operacja"

- modele arytmetyczne wykonują operacje

"+", "*", "max_list", "min_list"

Model rozszerzony:

"Relacja"

- modele relacyjne testują relacje:

"<,<", "<,<=", "<=,<", "<=,<="

Model rozszerzony arytmetyczny:

Lista_Argumentów

["Argument_1","Argument_2",....., "Argument_n"]

"Argument_i" = rzeczywista lub całkowita zmienna łańcuchowa

n – „dowolnie” duże

Sens modelu rozszerzonego arytmetycznego:

```
model_r(Nr_Modelu,
      "Start", "Wynik", "+",
     ["X1", "X2 ,...., "Xn"],
            Semafor_wyświetlania)
```

Jeżeli Start jest prawdą

to Wynik = X1 + X2 + ... +Xn

Sens modelu rozszerzonego arytmetycznego:

model_r(Nr_Modelu,
"Start", "Wynik", "+",
["X1", "X2 ,...., "Xn"],
Semafor_wyświetlania)

Jeżeli Start nie jest prawdą

to Wynik jest nieokreślony

Model rozszerzony relacyjny:

Lista_Argumentów

["Ograniczenie_dolne", "Wielkość_testowana", "Ograniczenie_górne"]

"Ograniczenie_dolne"

"Wielkość_testowana"

"Ograniczenie_górne"

} Rzeczywiste zmienne łańcuchowe

Sens modelu rozszerzonego relacyjnego:

model_r(Nr_Modelu,
"Start", " Wniosek ",
" <,<= ", ["Ogr_d", "X", "Ogr_g"],
Semafor_wyświetlania)

Jeżeli Start jest prawdą

i Ogr_d < X <= Ogr_g

to Wniosek jest prawdą

Sens modelu rozszerzonego relacyjnego:

```
model_r(Nr_Modelu,
            "Start", " Wniosek ",
            " <,<= ", ["Ogr_d", "X", "Ogr_g"],
                Semafor_wyświetlania)
```

Jeżeli Start nie jest prawdą

to Wniosek jest nieokreślony

Sens modelu rozszerzonego relacyjnego:

model_r(Nr_Modelu,
"Start", " Wniosek ",
" <,<= ", ["Ogr_d", "X", "Ogr_g"],
Semafor_wyświetlania)

Jeżeli Start jest prawdą

i X <= Ogr_d lub Ogr_g < X

to Wniosek nie jest prawdą,
prawdą jest nWniosek

Istotna różnica w porównaniu z modelami baz elementarnych dokładnych

Model liniowy:

```
model_liniowy(Numer_Modelu,
    "Warunek_startowy",
        "Wynik",
        Lista_współczynników,
        Lista_Zmiennych,
        Semafor_wyświetlania)
```

Model liniowy:

"Wynik"

rzeczywista zmienna łańcuchowa

Model liniowy:

"Lista_współczynników"

["A_1", "A_2",..., "A_n"]

gdzie

"A_i" - rzeczywiste zmienne łańcuchowe

Model liniowy:

Lista_zmiennych

["X_1", "X_2",..., "X_n"]

gdzie

"X_i" - rzeczywiste zmienne łańcuchowe

Sens modelu liniowego:

```
model_liniowy(Numer_Modelu,
      "Start", "Wynik",
            ["A_1", "A_2", "A_3"],
            ["X_1", "X_2", "X_3"],
           Semafor_wyświetlania)
```

jeżeli Start jest prawdą

to

Wynik =

$$A\_1 * X\_1 + A\_2 * X\_2 + A\_3 * X\_3$$

Sens modelu liniowego:

```
model_liniowy(Numer_Modelu,
      "Start", "Wynik",
            ["A_1", "A_2", "A_3"],
            ["X_1", "X_2", "X_3"],
           Semafor_wyświetlania)
```

jeżeli Start nie jest prawdą

to

Wynik jest nieokreślony

Model wielomianowy:

```
model_wielomianowy(Numer_Modelu,
                   "Warunek_startowy",
                   "Wynik",
                   "Wartość_zmiennej",
                       Lista_współczynników,
                       Lista_Potęg,
                       Semafor_wyświetlania)
```

Model wielomianowy:

"Wynik"

rzeczywista zmienna łańcuchowa

Model wielomianowy:

"Wartość_Zmiennej"

rzeczywista zmienna łańcuchowa, dla której jest wyznaczana **wartość wielomianu**

Model wielomianowy:

Lista_współczynników

["A_1", "A_2",..., "A_n"]

gdzie:

"A_i" - rzeczywista zmienna łańcuchowa

Model wielomianowy:

Lista_potęg

[0,1,2,...,n-1]

Lista **liczb całkowitych** (niekoniecznie kolejnych)

n – „dowolnie” duże

Sens modelu wielomianowego:

model_wielomianowy(Numer_Modelu, " Start ",
" Wynik ",
" 3 ", ["A_0", "A_2", ”A_5”] ,
[0,2,5] ,
Semafor_wyświetlania)

Jeżeli Start jest prawdą

to Wynik =
A_0 * 3^0 + A_2 * 3^2 + A_5 * 3^5

Sens modelu wielomianowego:

model_wielomianowy(Numer_Modelu, " Start ",
" Wynik ",
" 3 ", ["A_0", "A_2", ”A_5”] ,
[0,2,5] ,
Semafor_wyświetlania)

Jeżeli Start nie jest prawdą

to Wynik jest nieokreślony

Modele mogą się zagnieżdżać:

Zagnieżdżanie modeli arytmetycznych: wynik jednego modelu może być argumentem innego modelu:

model(N, St_N, "Wynik_N", A, O_N, B, 1)

model (M, St_M, Wynik_M, "Wynik_N" , O_M, G, 1)

Modele mogą się zagnieżdżać:

Zagnieżdżanie modeli relacyjnych i innych: wniosek jednego modelu relacyjnego może być warunkiem stosowania innego modelu:

model(N, _ , "Wniosek_N", A, R_N, B, 1)

model (M, "Wniosek_N", Wynik_M, C, O_M, D, 1)

Modele mogą się zagnieżdżać:

Zagnieżdżanie modeli relacyjnych i innych: zanegowany wniosek jednego modelu relacyjnego może być warunkiem stosowania innego modelu:

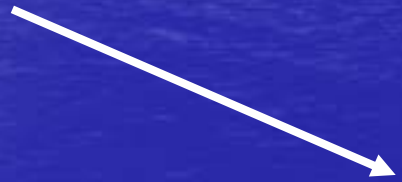
model(N, _ , "Wniosek_N", A, R_N, B, 1)

model (M, "nWniosek_N", Wynik_M, C, O_M, D, 1)

Modele relacyjne i reguły mogą się zagnieżdżać:

Zagnieżdżanie modeli relacyjnych i reguł: wniosek modelu relacyjnego może być warunkiem reguły

model(N, _, "Wniosek_N", A, R_N, B, 1])

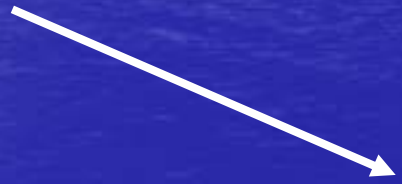


reguła(M, Wniosek_M, [..., "Wniosek_N",..])

Modele relacyjne i reguły mogą się zagnieżdżać:

Zagnieżdżanie modeli relacyjnych i reguł: zanegowany wniosek modelu relacyjnego może być warunkiem reguły

model(N, _, "Wniosek_N", A, R_N, B, 1])



reguła(M, Wniosek_M, [..., "nWniosek_N",..])

Zagnieżdżanie reguł i modeli:

wniosek reguły może być warunkiem startowym modelu

reguła(N, "Wniosek_N", Lista_warunków,1)

model (M, "Wniosek_N", Wynik_M, A ,”op_M”, B, 1)

Zagnieżdżanie reguł i modeli:

zanegowany wniosek reguły może być warunkiem startowym modelu

reguła(N, "Wniosek_N", Lista_warunków,1)

model (M, "nWniosek_N", Wynik_M, A ,”op_M”, B, 1)

Zagnieżdżanie modeli

Zagnieżdżające się modele mogą mieć argumenty dwojakiego rodzaju:

Argumenty dopytywalne: nie są wynikami modeli. Wartość tych argumentów jest określana przez użytkownika systemu ekspertowego

Argumenty niedopytywalne: są wynikami modeli. Wartość liczbowa tych argumentów jest określana przez system ekspertowy

Zagnieżdżanie modeli

Zagnieżdżające się modele mogą mieć warunki startowe dwojakiego rodzaju:

Warunki startowe dopytywalne: nie są wnioskami modeli relacyjnych ani wnioskami reguł. Wartość tych warunków startowych jest określana przez użytkownika systemu ekspertowego

Warunki startowe niedopytywalne: są wnioskami modeli relacyjnych lub wnioskami reguł. Wartość logiczna tych warunków startowych jest określana przez system ekspertowy

BAZA RAD

Każdej regule i każdemu modelowi relacyjnemu mogą być przyporządkowane dwie rady:

- **rada udzielana dla spełnionej reguły (spełnionego modelu relacyjnego)**
- **nrada udzielana dla niespełnionej reguły (niespełnionego modelu relacyjnego)**

BAZA RAD

Klauzule bazy rad:

rada(numer_reguły/modelu,
"nazwa_pliku_rady_dla_reguły_spełnionej/
modelu_relacyjnego_spełnionego.hlp")

nrada(numer_reguły/modelu,
nazwa_pliku_rady_dla_reguły_niespełnionej
modelu_relacyjnego_niespełnionego.hlp")

Istotne różnice pomiędzy wnioskowaniem elementarnym dokładnym i rozwiniętym dokładnym

- **Dla wnioskowania elementarnego dokładnego zapamiętywano tylko wnioski prawdziwe.**
- **Dla wnioskowania rozwiniętego dokładnego zapamiętuje się zarówno prawdziwe jak i nieprawdziwe wnioski.**

Nigdy bowiem nie wiemy, czy zanegowany wniosek nie będzie potrzebny jako warunek jakiejś reguły

Istotne różnice pomiędzy wnioskowaniem elementarnym dokładnym i rozwiniętym dokładnym

Wnioskowanie elementarne dokładne jest **monotoniczne**: wniosek uznany za prawdę nie mógł w trakcie dalszego wnioskowania, przy niezmienionych warunkach dopytywalnych będących faktami, stać się nieprawdą.

Istotne różnice pomiędzy wnioskowaniem elementarnym dokładnym i rozwiniętym dokładnym

Wnioskowanie rozwinięte dokładne jest **niemonotoniczne**: wniosek uznany za prawdę może w trakcie dalszego wnioskowania, przy niezmienionych warunkach dopytywalnych będących faktami, stać się nieprawdą.

Mechanizm wnioskowania rozwiniętego

Dla zrozumienia mechanizmu wnioskowania rozwiniętego, zaczniemy od prostego przykładu:

Wnioskowanie rozwinięte dokładne w przód

Wnioskowanie rozwinięte dokładne w przód jest wnioskowaniem w przód dla baz reguł rozwiniętych dokładnych, tzn. baz dokładnych mogących zawierać zanegowane warunki niedopytywalnych.

Wnioskowanie to można stosować również dla baz reguł elementarnych dokładnych

Konwencja: **W - warunek niezanegowany**

nW - warunek zanegowany

Wnioskowanie rozwinięte dokładne w przód

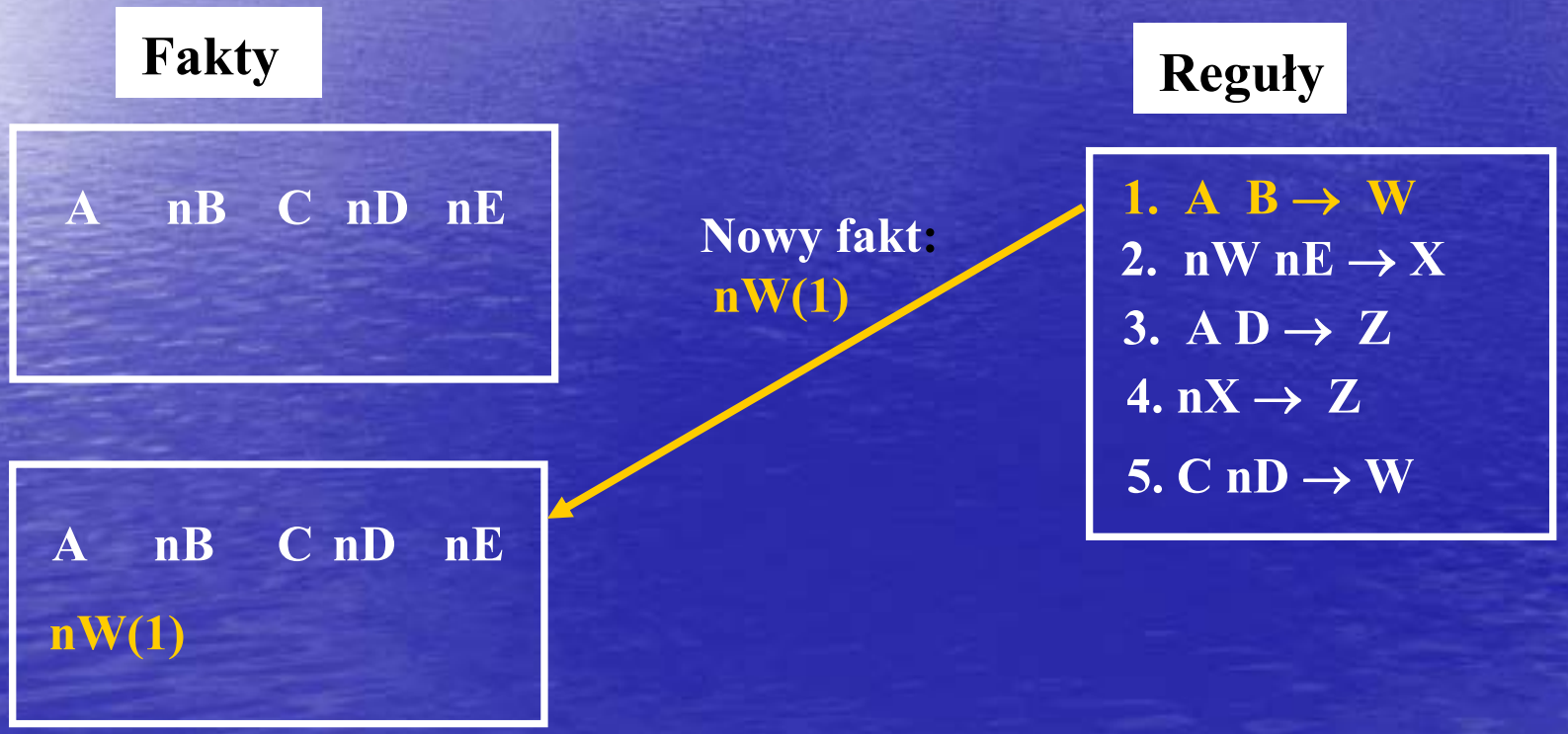
Cel wnioskowania:

Wyznaczenie wszystkich wniosków prawdziwych i wszystkich wniosków nieprawdziwych dla:

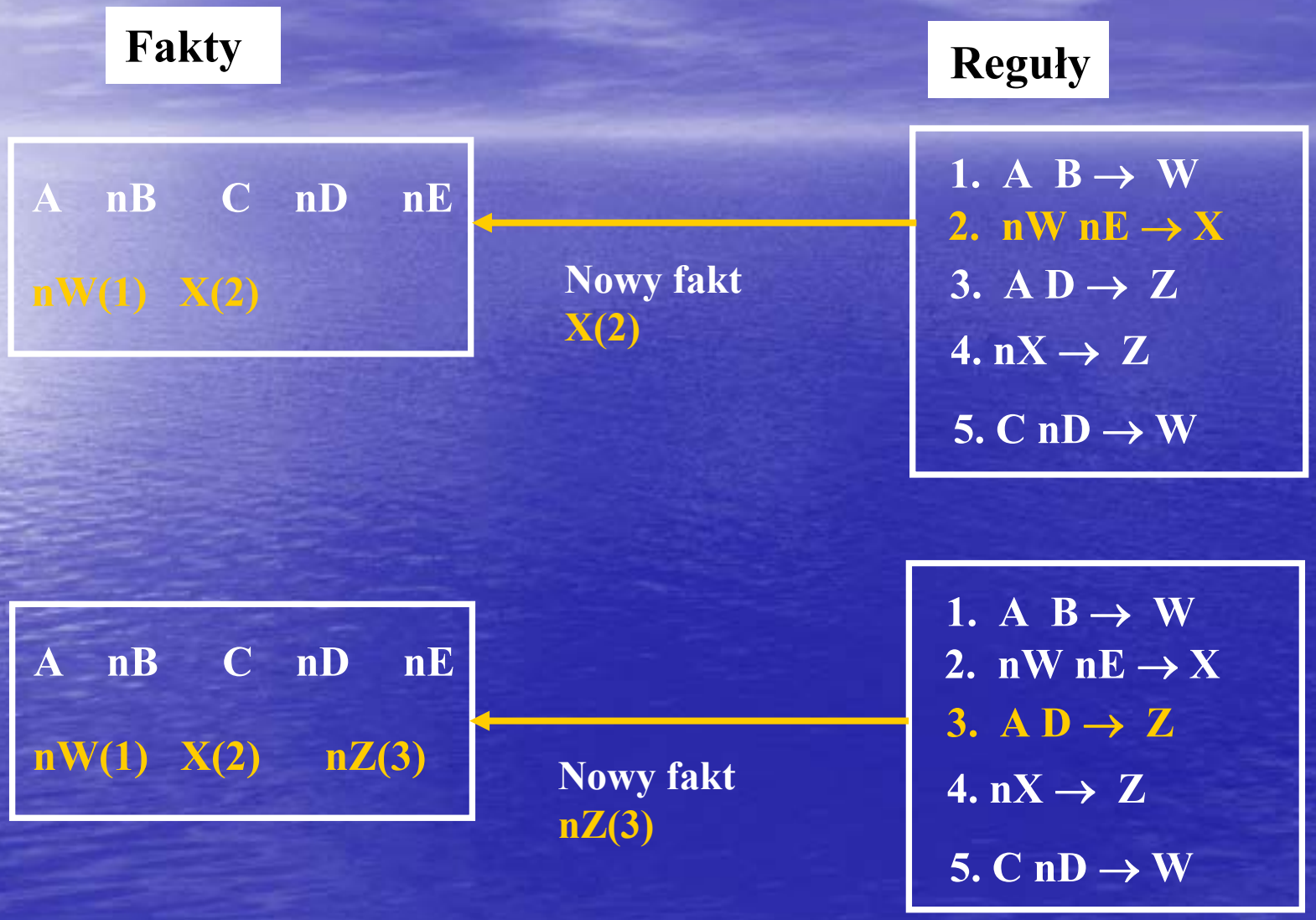
- danego początkowego zbioru prawdziwych i nieprawdziwych warunków dopytywalnych
- danej bazy reguł rozwiniętej dokładnej i odpowiadającej jej bazy ograniczeń
- danego początkowego zbioru wartości argumentów
- danej bazy modeli dokładnej

Wnioskowanie rozwinięte dokładne w przód: przykład (a)

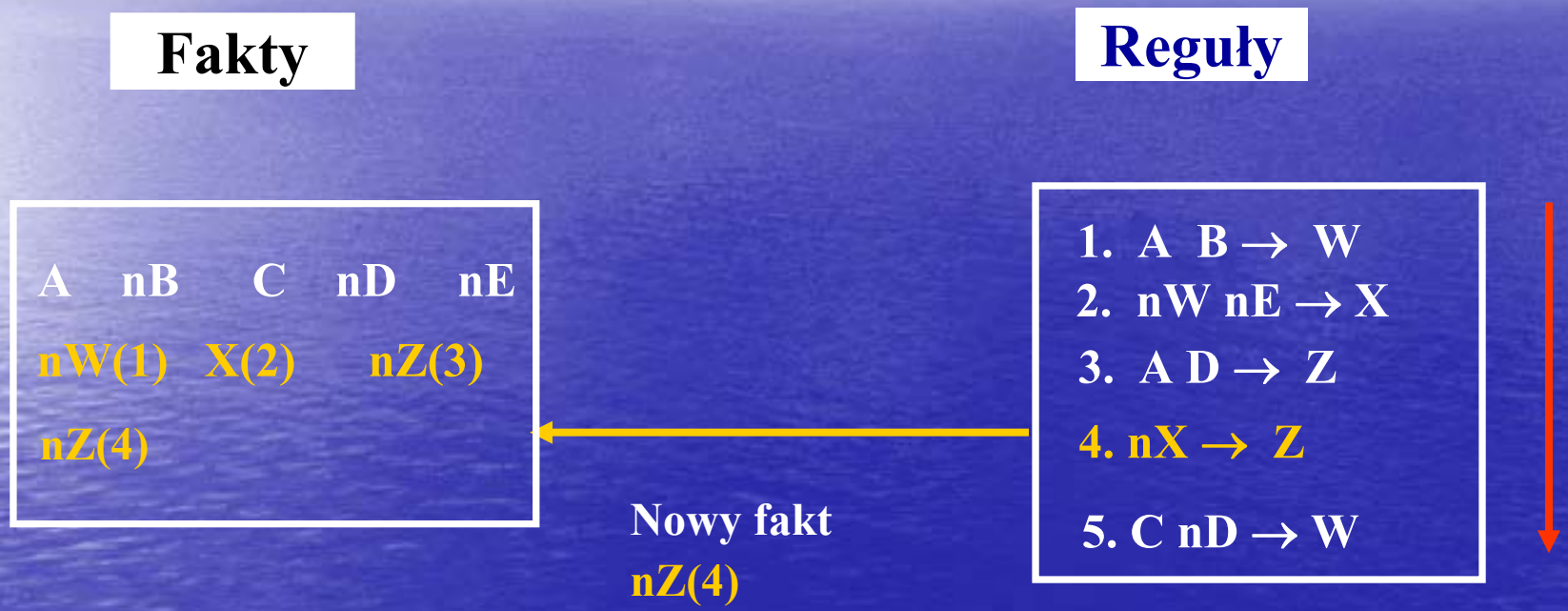
Przykład: dla danej bazy reguł wyznaczyć wszystkie fakty wynikające z faktów **A , nB, C, nD i nE**:



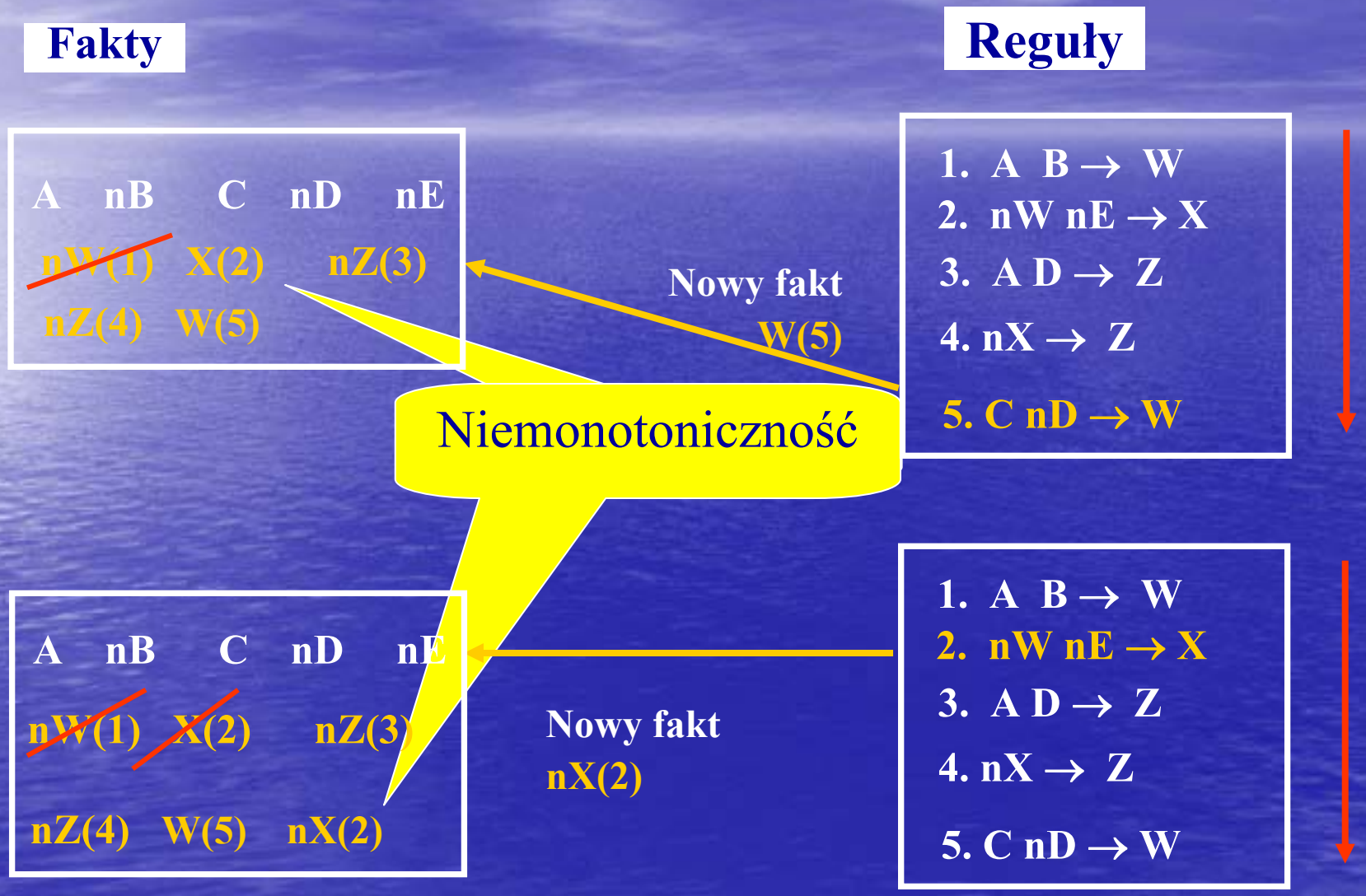
Wnioskowanie rozwinięte dokładne w przód: przykład (b)



Wnioskowanie rozwinięte dokładne w przód: przykład (c)



Wnioskowanie rozwinięte dokładne w przód: przykład (d)



Wnioskowanie rozwinięte dokładne w przód: przykład (e)

Fakty

A nB C nD nE

~~nW(1)~~ ~~X(2)~~ ~~nZ(3)~~

~~nZ(4)~~ W(5) nX(2)

Z(4)

Reguły

1. A B → W
2. nW nE → X
3. A D → Z
4. nX → Z
5. C nD → W

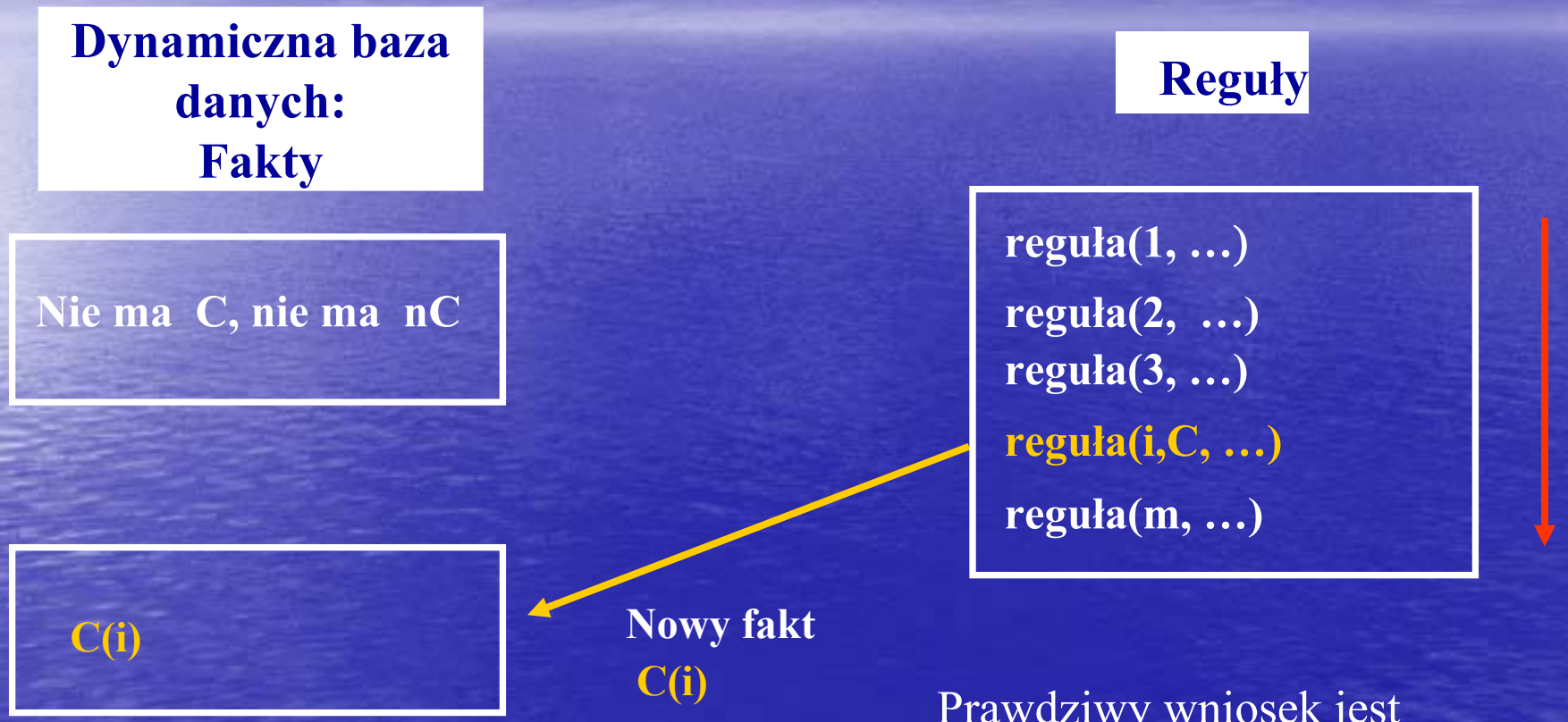
Nowy fakt
Z(4)

Z faktów A, nB, C, nD i nE wynikają fakty W(5), nX(2) , Z(4).

Ogólne zasady wnioskowania w przód dla baz reguł rozwiniętych dokładnych

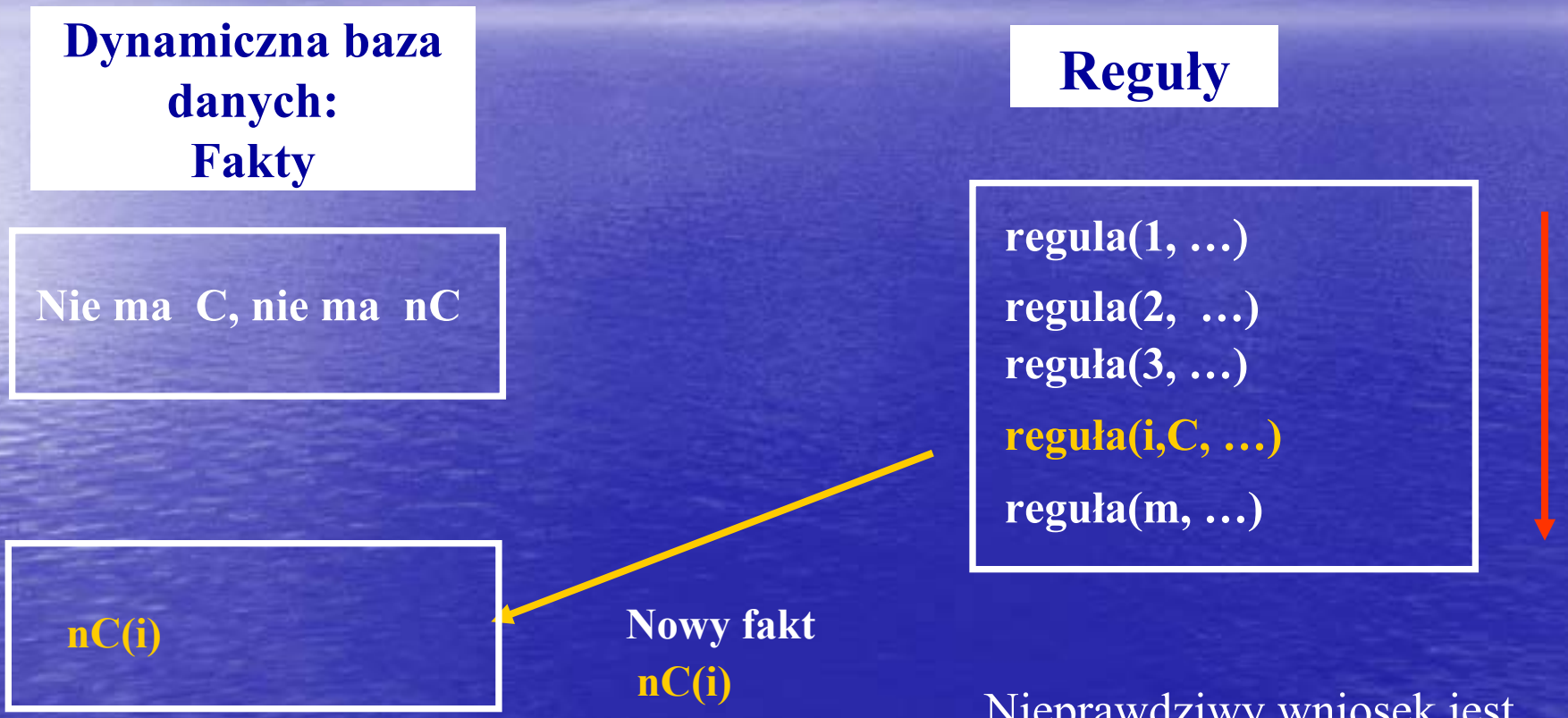
Z poprzedniego przykładu wynikają następujące zasady ogólne:

Ogólne zasady wnioskowania w przód dla baz reguł rozwiniętych dokładnych (1)



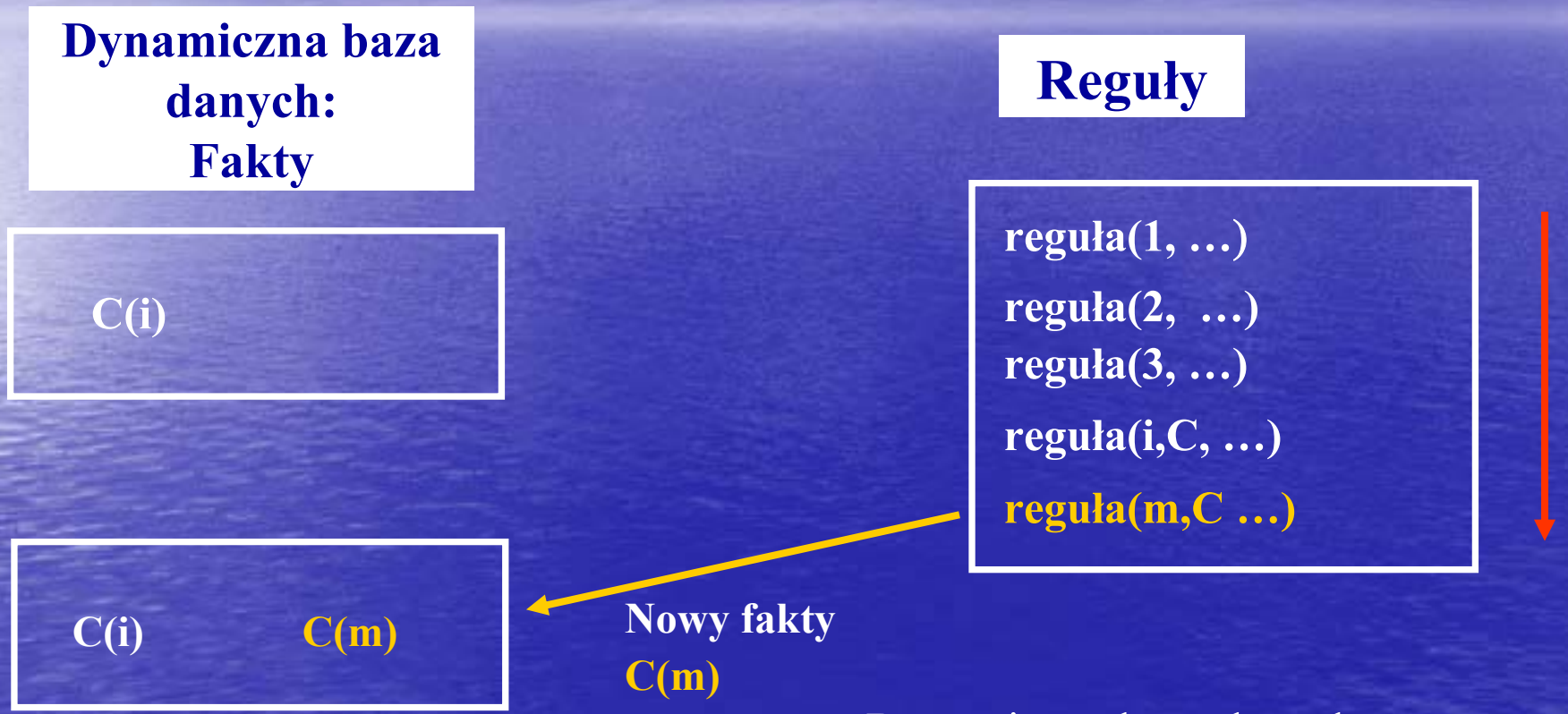
Prawdziwy wniosek jest zapisywany w dynamicznej bazie danych

Ogólne zasady wnioskowania w przód dla baz reguł rozwiniętych dokładnych (2)



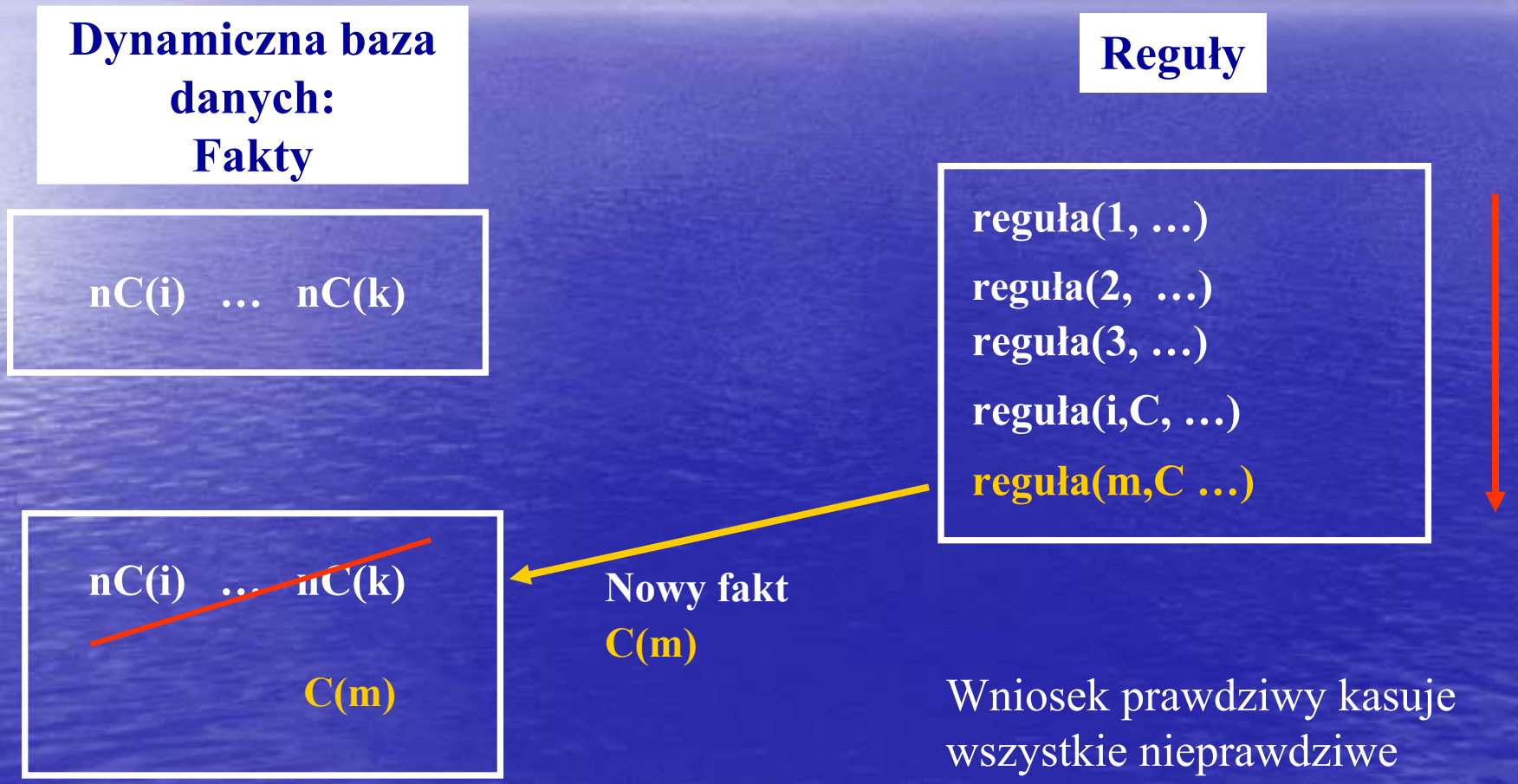
Nieprawdziwy wniosek jest zapisywany w dynamicznej bazie danych

Ogólne zasady wnioskowania w przód dla baz reguł rozwiniętych dokładnych (3)

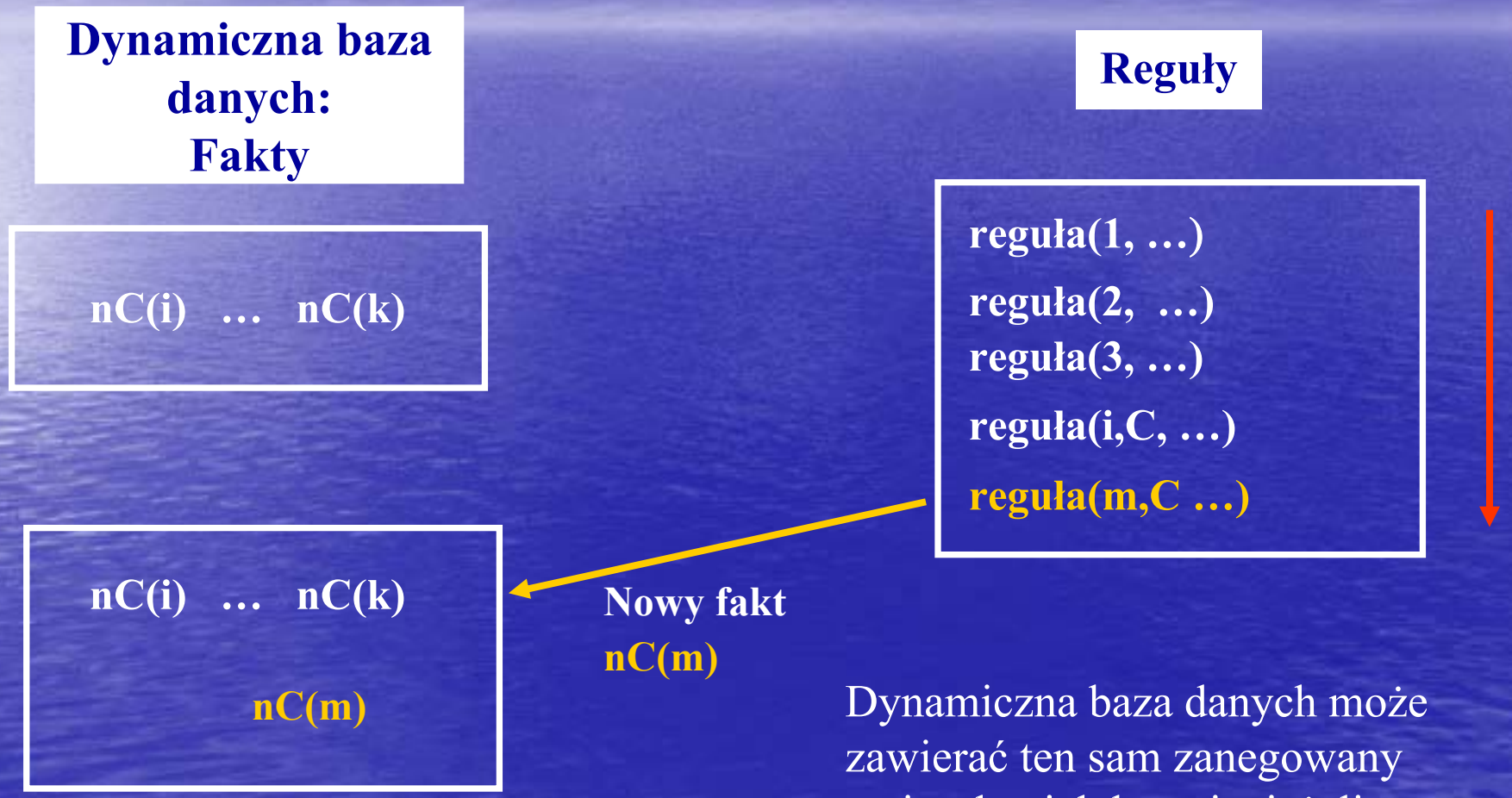


Dynamiczna baza danych może zawierać ten sam wniosek wielokrotnie, jeżeli pochodzi od różnych reguł.

Ogólne zasady wnioskowania w przód dla baz reguł rozwiniętych dokładnych (4)

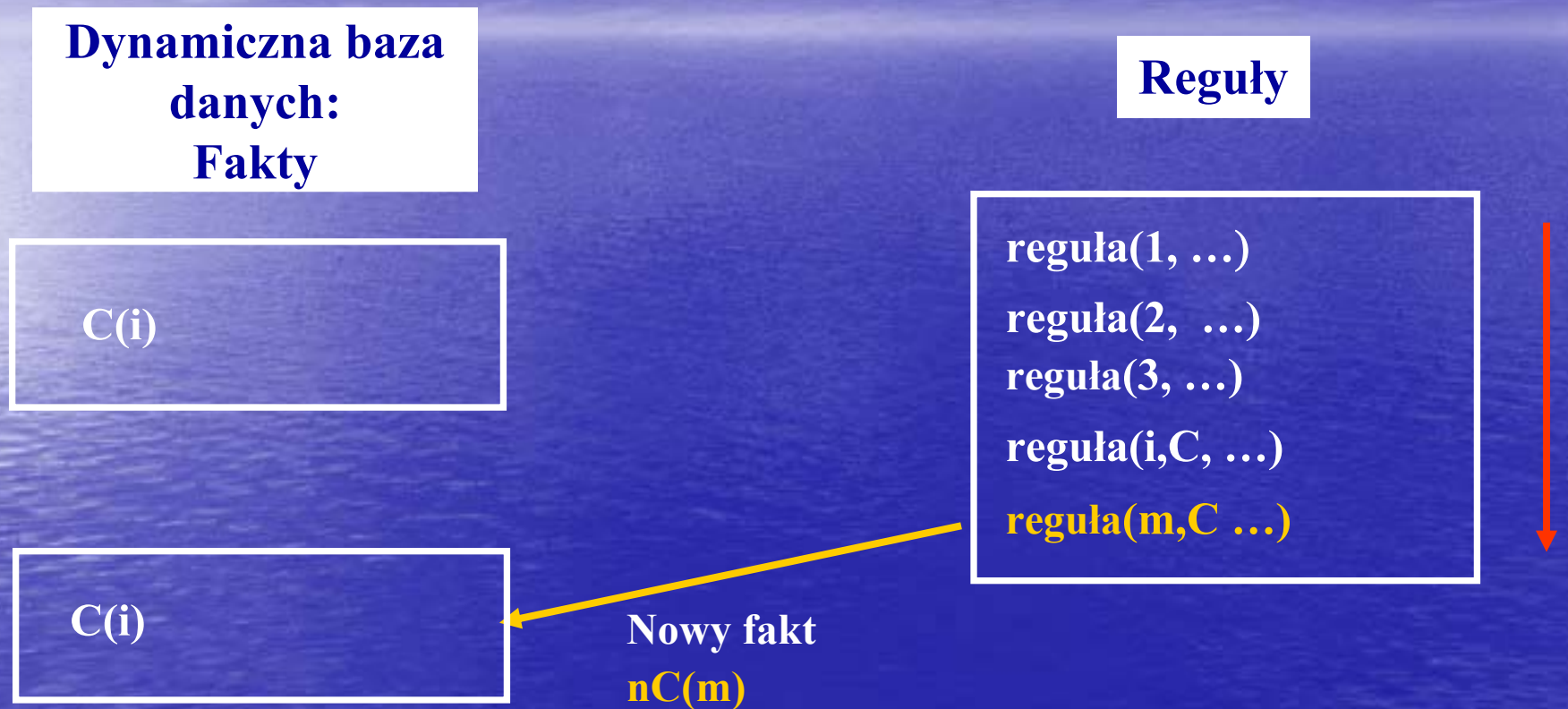


Ogólne zasady wnioskowania w przód dla baz reguł rozwiniętych dokładnych (5)



Dynamiczna baza danych może zawierać ten sam zanegowany wniosek wielokrotnie, jeżeli pochodzi od różnych reguł.

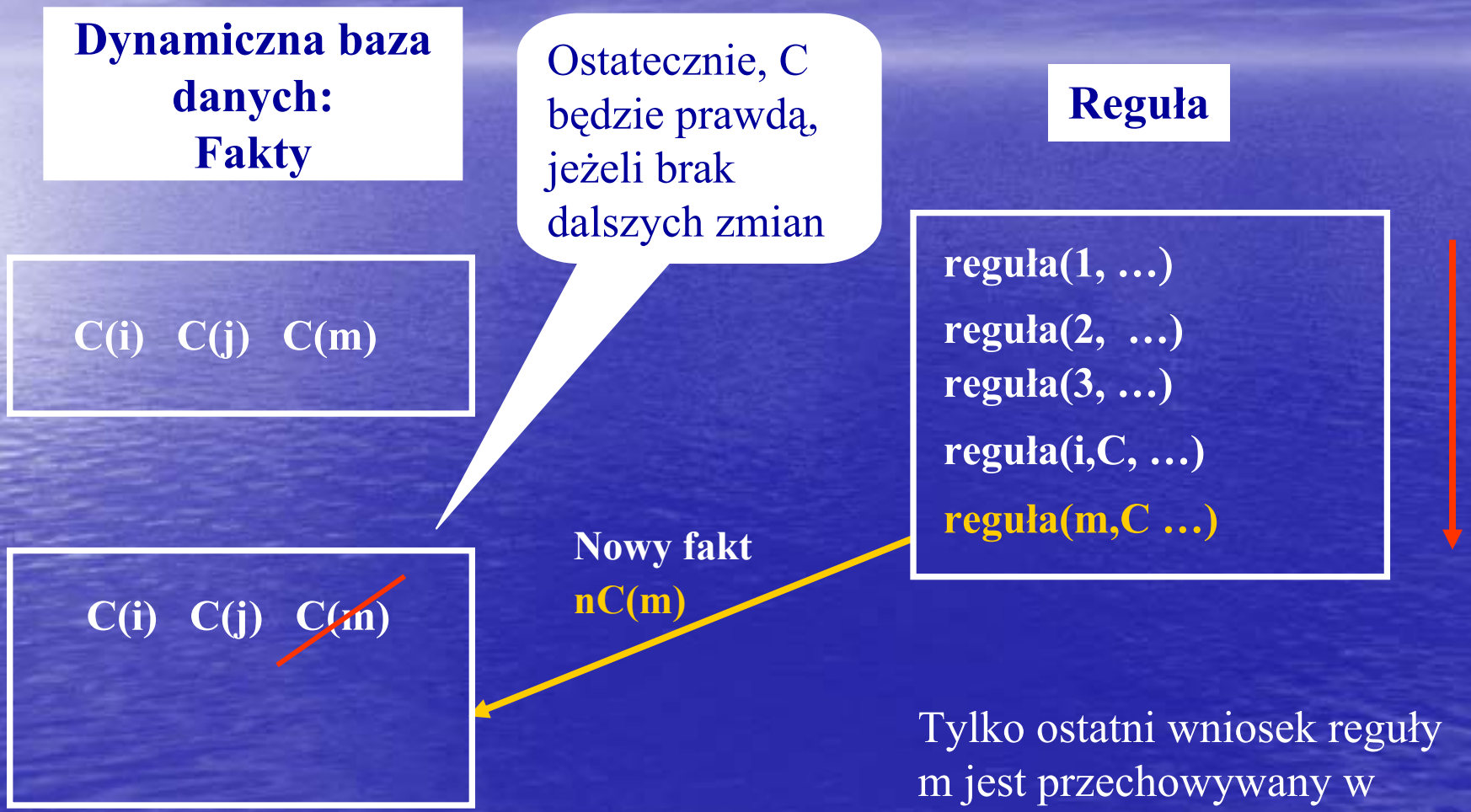
Ogólne zasady wnioskowania w przód dla baz reguł rozwiniętych dokładnych (6)



Zawartość dynamicznej bazy danych nie ulega zmianie.

Rozwiązanie konfliktu 2

Ogólne zasady wnioskowania w przód dla baz reguł rozwiniętych dokładnych (7)



Ogólne zasady wnioskowania w przód dla baz reguł rozwiniętych dokładnych (8)

Ważny przypadek szczególny zasady 7

Wnioskowanie rozwinięte dokładne w przód

(a) zaistnienie faktu **X** w sytuacji, gdy w dynamicznej bazie danych nie ma faktu **X** ani faktu zanegowanego **nX**, powoduje wpisanie faktu **X** do dynamicznej bazy danych wraz z numerem generującej go reguły. Jeżeli fakt ten odpowiada warunkowi dopytywalnemu, numerem tym jest 0;

Wnioskowanie rozwinięte dokładne w przód

(b) zaistnienie faktu **X** w sytuacji, gdy w dynamicznej bazie danych jest odpowiadający mu fakt zanegowany **nX**, powoduje dodanie faktu **X** do dynamicznej bazy danych wraz z numerem generującej go reguły i usunięcie z niej faktu zanegowanego **nX** wraz z numerem odpowiedniej reguły.

Wnioskowanie rozwinięte dokładne w przód

(c) zaistnienie faktu **X** w sytuacji, gdy w dynamicznej bazie danych jest fakt **X** wygenerowany przez inną regułę, powoduje dodanie nowego faktu **X** do dynamicznej bazy danych wraz z numerem generującej go reguły

Wnioskowanie rozwinięte dokładne w przód

(d) zaistnienie faktu **nX** w sytuacji, gdy w dynamicznej bazie danych jest odpowiadający mu fakt **X** generowany przez inną regułę niż fakt **nX**, nie powoduje zmian w dynamicznej bazie danych

Wnioskowanie rozwinięte dokładne w przód

(e) zaistnienie faktu nX w sytuacji, gdy w dynamicznej bazie danych jest odpowiadający mu fakt X generowany przez tę samą regułę co fakt nX i nie ma faktów X generowanych przez inne reguły, powoduje dodanie faktu nX do dynamicznej bazy danych wraz z numerem odpowiedniej reguły i usunięcie z dynamicznej bazy danych faktu X.

Wnioskowanie rozwinięte dokładne wstecz

- Wnioskowanie rozwinięte wstecz jest wnioskowaniem wstecz dla baz reguł rozwiniętych dokładnych, tzn. mogących zawierać zanegowane warunki niedopytywalnych.

- Wnioskowanie to można stosować również dla baz reguł elementarnych dokładnych.

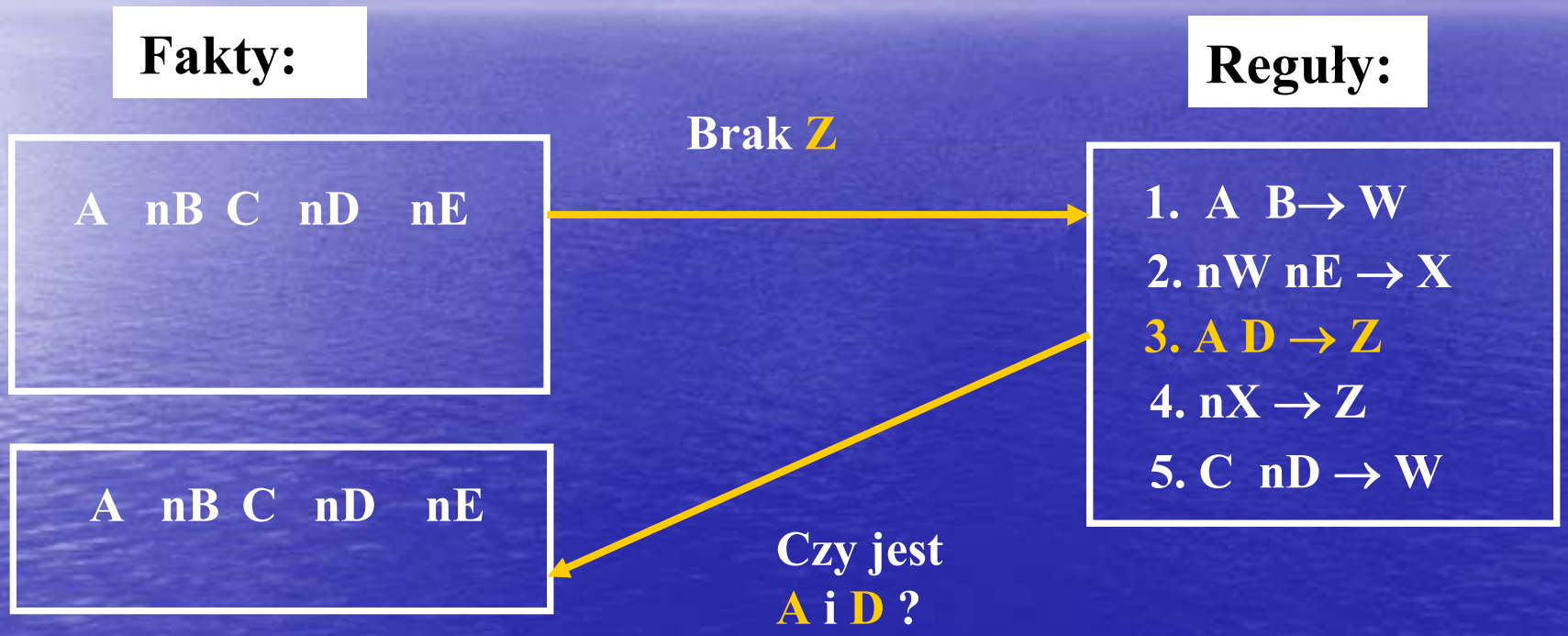
Wnioskowanie rozwinięte dokładne wstecz (1)

Cel wnioskowania:

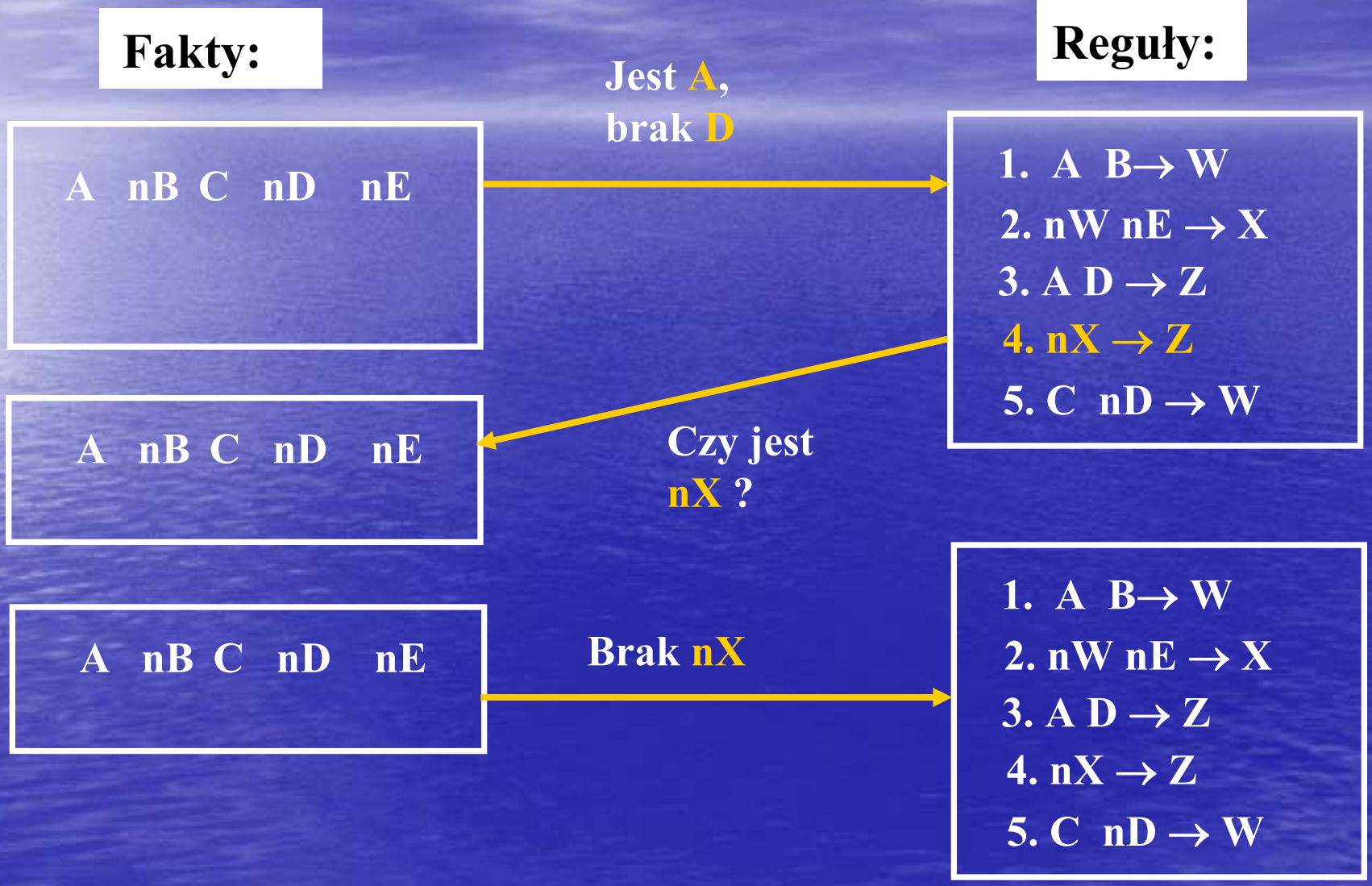
Potwierdzenie (weryfikowanie) lub zaprzeczenie (falsyfikowanie) iż dana hipoteza wynika z:

- danego początkowego zbioru prawdziwych i nieprawdziwych warunków dopytywalnych
- danej bazy reguł rozwiniętej dokładnej i odpowiadającej jej bazy ograniczeń

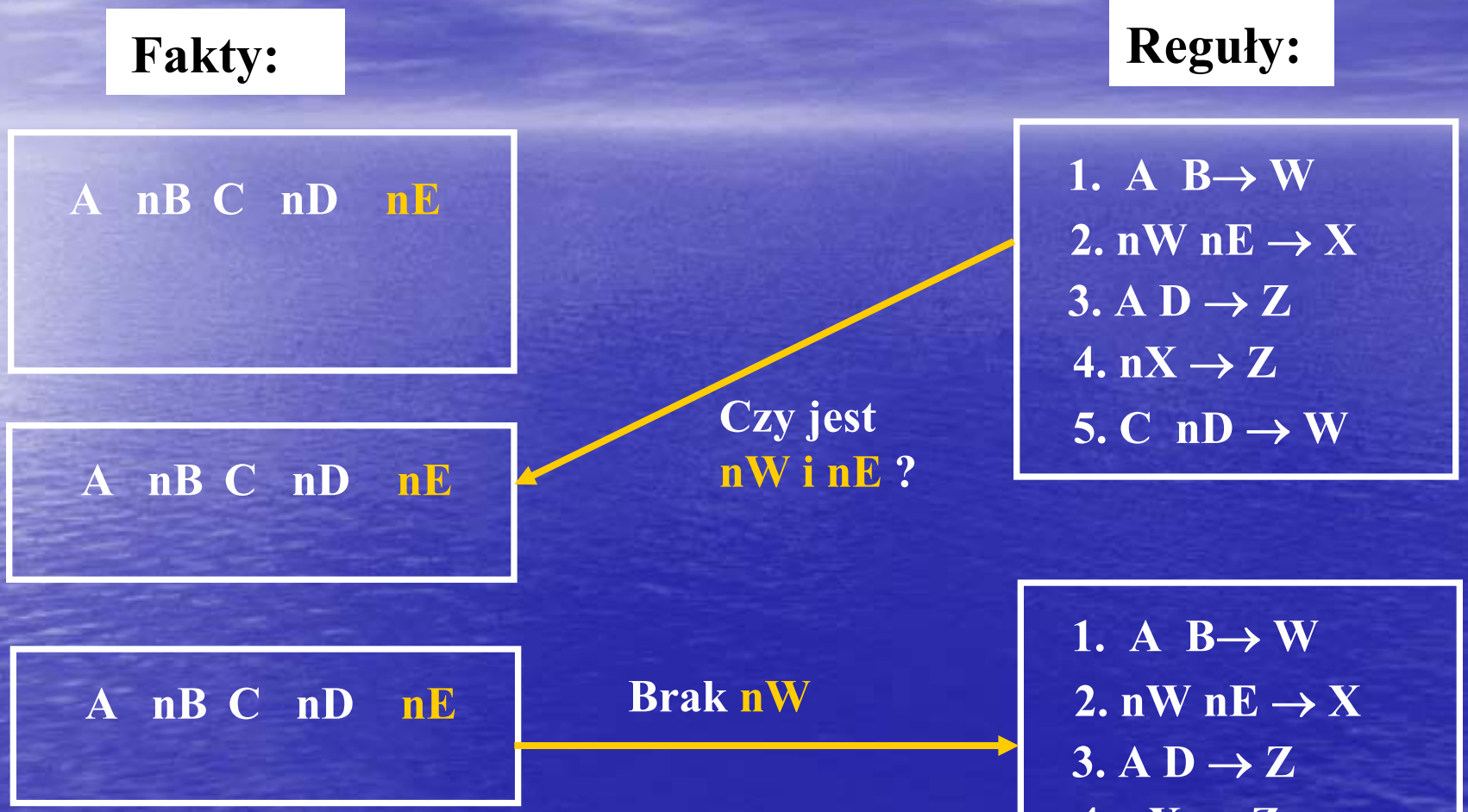
Wnioskowanie rozwinięte dokładne wstecz: weryfikacja hipotezy Z



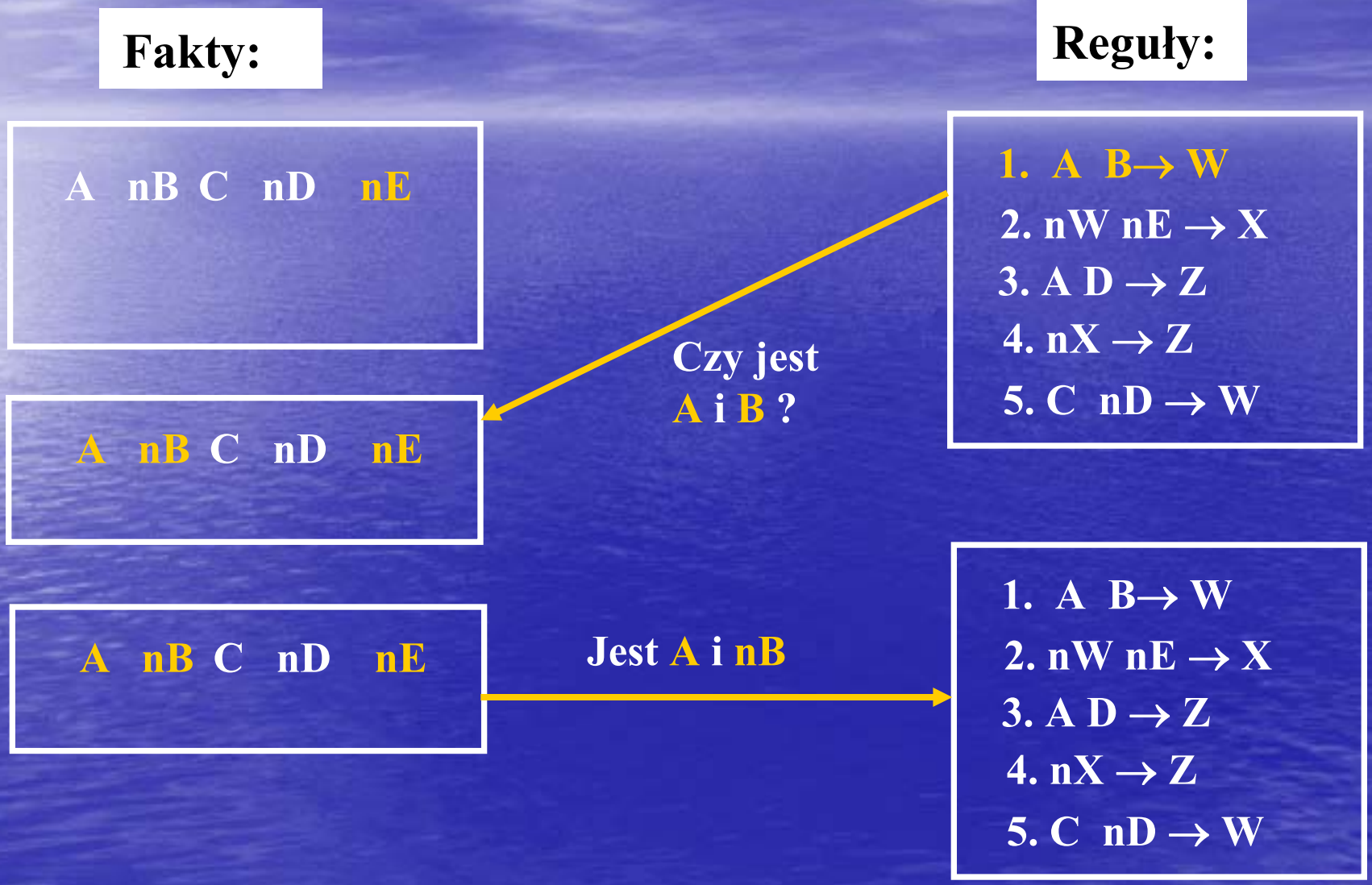
Wnioskowanie rozwinięte dokładne wstecz: weryfikacja hipotezy Z



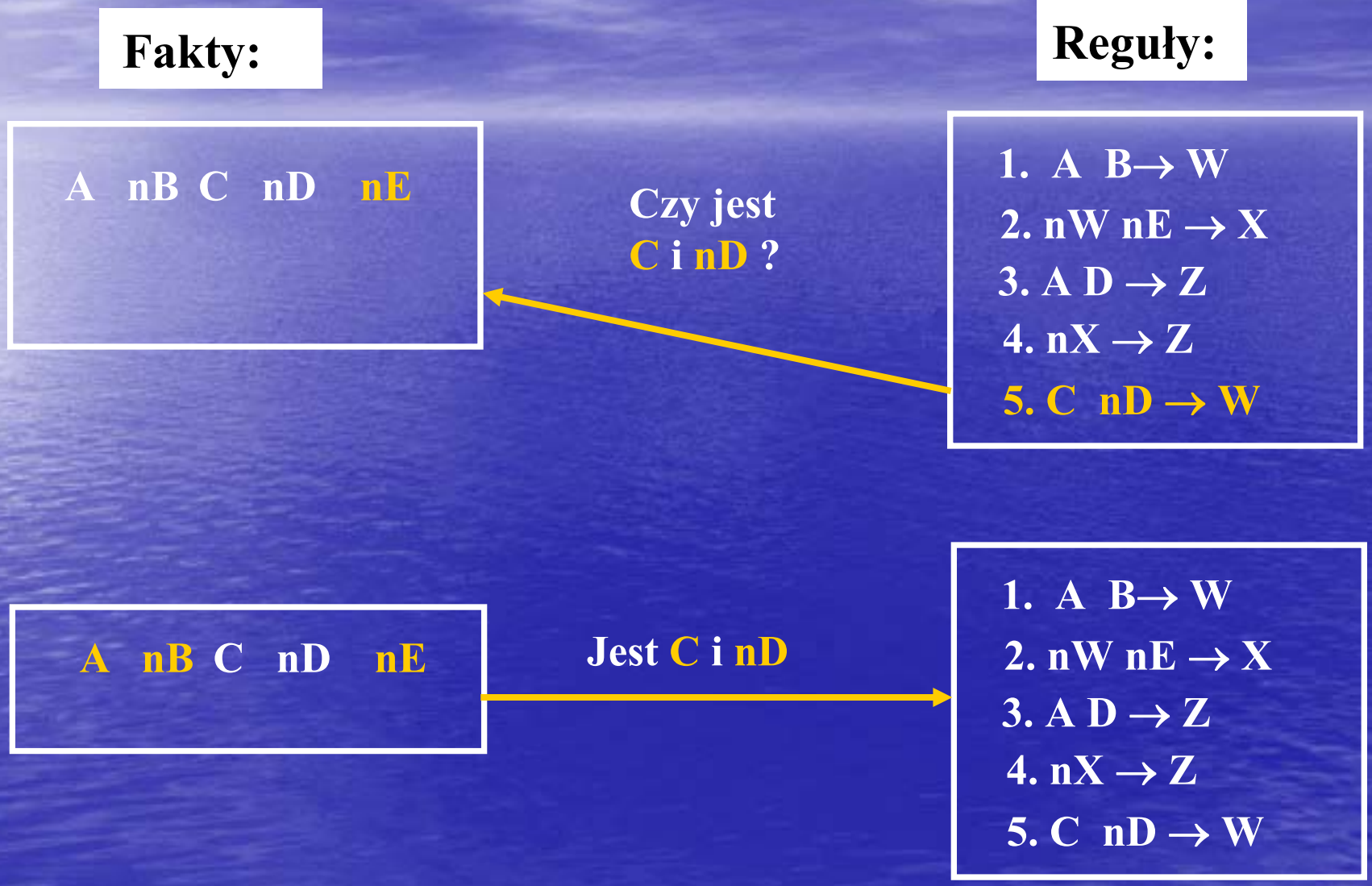
Wnioskowanie rozwinięte dokładne wstecz: weryfikacja hipotezy Z



Wnioskowanie rozwinięte dokładne wstecz: weryfikacja hipotezy Z



Wnioskowanie rozwinięte dokładne wstecz: weryfikacja hipotezy Z



Wnioskowanie rozwinięte dokładne wstecz: weryfikacja hipotezy Z

Fakty:

A nB C nD nE

Jest C i nD, a więc jest W, a więc jest nX, więc jest Z

Reguły:

1. A B→ W
2. nW nE → X
3. A D → Z
4. nX → Z
5. C nD → W

Sprzeczności w rozwiniętych dokładnych bazach reguł (1)

1. Sprzeczności typu SRD1:

- **zewnętrzne**: wniosek reguły jest sprzeczny z jej warunkiem
- **wewnętrzne**: warunki reguły są sprzeczne

SRD1 _Sprzeczność bazy Rozwiniętej Dokładnej generowana przez Pojedynczą (1) bazę

Sprzeczności w rozwiniętych dokładnych bazach reguł (2)

Sprzeczności zewnętrzne: sprzeczności pomiędzy wnioskiem a warunkami:

1. **Reguła jest zewnętrznie SRD1-samosprzeczna, jeżeli jednym z jej warunków jest jej wniosek lub negacja wniosku. Np.:**

```
reguła(1,"Wniosek",["Wniosek", "Warunek_1"],_)

reguła(2,"Wniosek",["nWniosek", "Warunek_1"],_)
```

Sprzeczności w rozwiniętych dokładnych bazach reguł (3)

- Reguła **n** jest **zewnętrznie bezpośrednio SRD1-sprzeczna** z regułą **m**, jeżeli:
  - wniosek lub zanegowany wniosek reguły **m** jest warunkiem reguły **n**, i
  - wniosek lub zanegowany wniosek reguły **n** jest warunkiem reguły **m**.

  Np.:

reguła(1, ”X”,[”M”,”N”,”nY”],1)
reguła(2, ”Y”,[”P”,”Q”,”X”],1)

Zastąpienie warunku **X** reguły **2** warunkami reguły **1** czyni z reguły **2** regułę **3** będącą zewnętrznie SRD1-samosprzeczną:

reguła(3, ” Y”,[” P”,” Q”,” M”,”N”,”nY”],1)

Sprzeczności w rozwiniętych dokładnych bazach reguł (4)

- Reguła **n** jest **zewnętrznie pośrednio SRD1-sprzeczna** z regułą **m**, jeżeli podstawienie reguły **m** do innej reguły, tej zaś do jeszcze innej itd., doprowadza do reguły bezpośrednio **SRD1**-sprzecznej z regułą **n**. Np. reguła **1** jest zewnętrznie pośrednio **SRD1**-sprzeczna z regułą **3:**

reguła(1, ”H”,[”K”,”B”],1)
reguła(2, ”K”,[” C”,”D”, ”nX”],1)
reguła(3, ”X”,[” H”,” A”],1)

Pośrednikiem jest tutaj reguła **2.** Podstawienie warunków reguły **3** w miejsce warunku **X** reguły **2** daje bowiem (w najbardziej niekorzystnym przypadku) regułę **4**:

reguła(21, ”K”,[” C”,”D”, ”nH” ,” nA”],1)

która jest zewnętrznie bezpośrednio **SRD1**-sprzeczna z regułą **1**

Sprzeczności w rozwiniętych dokładnych bazach reguł (5)

Sprzeczności wewnętrzne: sprzeczności pomiędzy warunkami reguł spłaszczonych

1. Reguła jest wewnętrznie SRD1-samosprzeczna, jeżeli ma warunki dopytywalne wykluczające się i nie ma innej reguły dla tego samego wniosku, która ma warunki dopytywalne nie wykluczające się.

Sprzeczności w rozwiniętych dokładnych bazach reguł (6)

Sprzeczności wewnętrzne: sprzeczności pomiędzy warunkami reguł spłaszczonych

Np.:

reguła(1,"X", ["nW", "W", "A"], 1)

reguła(2,"X", ["nB", "C", "B"], 1)

i nie ma innej reguły dla X, która ma warunki dopytywalne nie wykluczające się.

Sprzeczności w rozwiniętych dokładnych bazach reguł (7)

Sprzeczności wewnętrzne: sprzeczności pomiędzy warunkami reguł spłaszczonych

2. Reguła jest pośrednio wewnętrznie SRD1-sprzeczna, jeżeli ma dla wszystkich możliwych spłaszczeń warunki dopytywalne wykluczające się i nie ma innej reguły dla tego samego wniosku, która po spłaszczeniu ma warunki dopytywalne nie wykluczające się.

Sprzeczności w rozwiniętych dokładnych bazach reguł (8)

Sprzeczności wewnętrzne: sprzeczności pomiędzy warunkami reguł spłaszczonych

Np.:

reguła(1,"X", ["C", "B", "Y"], 1)
reguła(2,"Y", ["A", "nB"], 1)

po spłaszczeniu:

reguła(3,"X", ["C", "B", "A", "nB"], _)

i nie ma innej reguły dla X, która po spłaszczeniu ma warunki dopytywalne nie wykluczające się.

Sprzeczności w rozwiniętych dokładnych bazach reguł (9)

3. Reguła pośrednio wewnętrznie SRD1-sprzeczna nie będzie nigdy spełniona, lecz może być niespełniona:

Np.:

reguła(1,"X", ["C", "B", "Y"], 1)
reguła(2,"Y", ["A", "nB"], 1)

Jeżeli nA,
to nY,
to nX.

Sprzeczności w rozwiniętych dokładnych bazach reguł i bazach ograniczeń (10)

2. Sprzeczności typu SRD2: warunki reguły są sprzeczne w wyniku interakcji bazy reguł i bazy ograniczeń.

Np.:

reguła(1,"X", ["A", "C", "B"], 1)

ograniczenie(1, ["A", "C"])

ograniczenie sprawia, ze reguła ta nie będzie nigdy stosowana

Nadmiarowości w rozwiniętych dokładnych bazach reguł (1)

1. **Nadmiarowości typu NRD1**: występowanie reguł o jednakowych wnioskach i jednakowych warunkach. Np. w przypadku reguł:

```
reguła(1,"W", ["nA", "B"], 1)
    reguła(2,"W", ["E", "F" ,"nC", "nD"],
1)
reguła(3,"A", ["C", "D"], 1)
reguła(4,"B", ["E", "F"], 1)
```

reguła **2** ma takie same warunki i taki sam wniosek jak reguła **1** wraz z regułami **3** i **4**.

Nadmiarowości w rozwiniętych dokładnych bazach reguł (2)

2. **Nadmiarowości typu NRD1_2**: występowanie **reguł subsumowanych**. Np.w przypadku reguł

reguła(1,"W", ["A", "B", "C", "nX"], 1)
reguła(2,"W", ["A", "B"], 1)

reguła 1 jest **subsumowana** (zawarta) w regule **2**, gdyż obydwie mają taki sam wniosek, a warunki reguły **2** są podzbiorem warunków reguły **1**. Regułę **1** można więc usunąć z bazy reguł.

Nadmiarowości w rozwiniętych dokładnych bazach reguł (3)

2. **Nadmiarowości typu NRD1_3**: występowanie **reguł o niepotrzebnych warunkach**. Np.w przypadku reguł

reguła(1,"W", ["A", "B", "C"], 1)
reguła(2,"W", ["A", "B", "nC"], 1)

Reguły **te** można zastąpić jedną regułą.

reguła(3,"W", ["A", "B"], 1)

Nadmiarowości w rozwiniętych dokładnych bazach reguł (4)

2. **Nadmiarowości typu NRD2**: Źródłem nadmiarowości typu **NRD2** jest interakcja bazy reguł i bazy ograniczeń. Np

Baza reguł:

reguła (1, "W", ["A", "nF"] ,1)
reguła (2, "W", ["B", "nF"] ,1)

Baza ograniczeń:

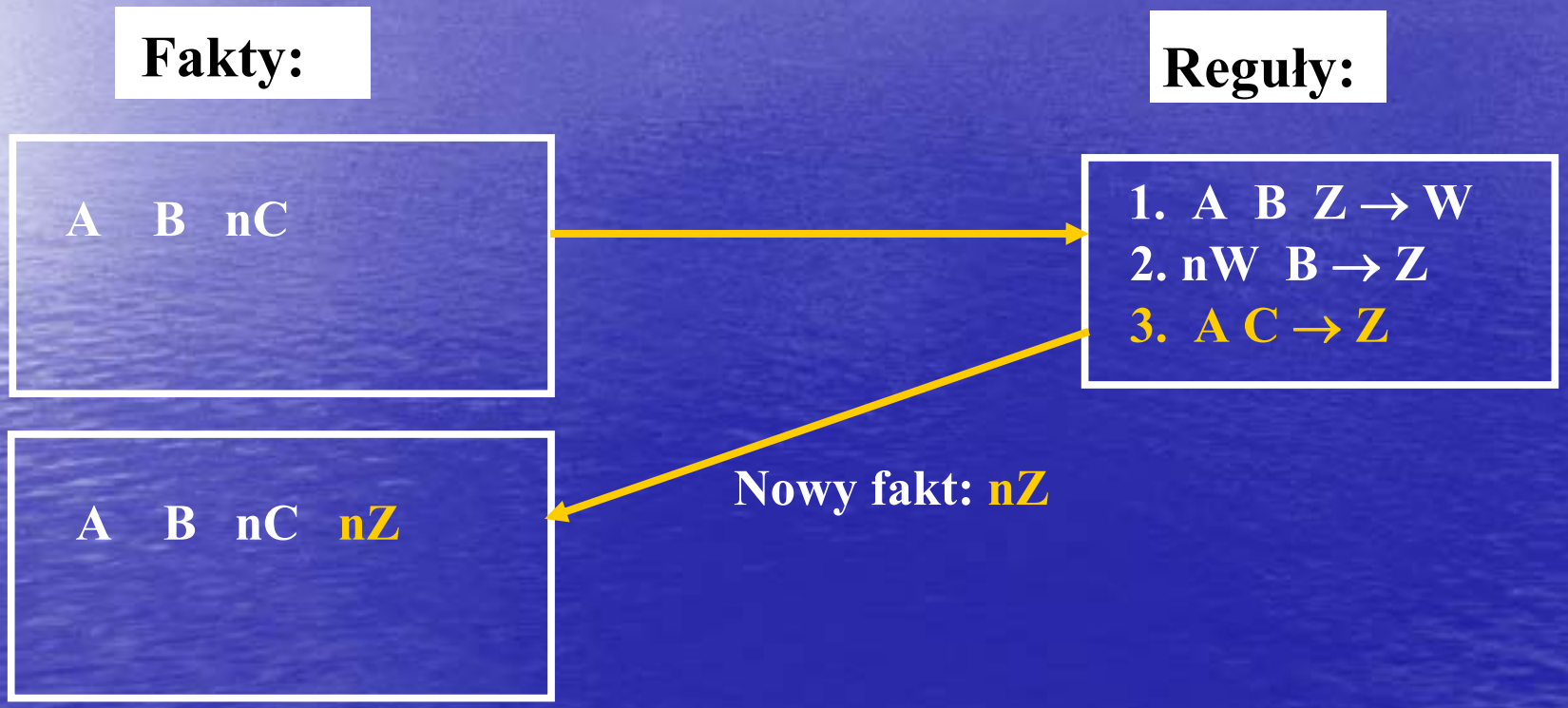
ograniczenie(1,["A" , "B"])

redukują się do:

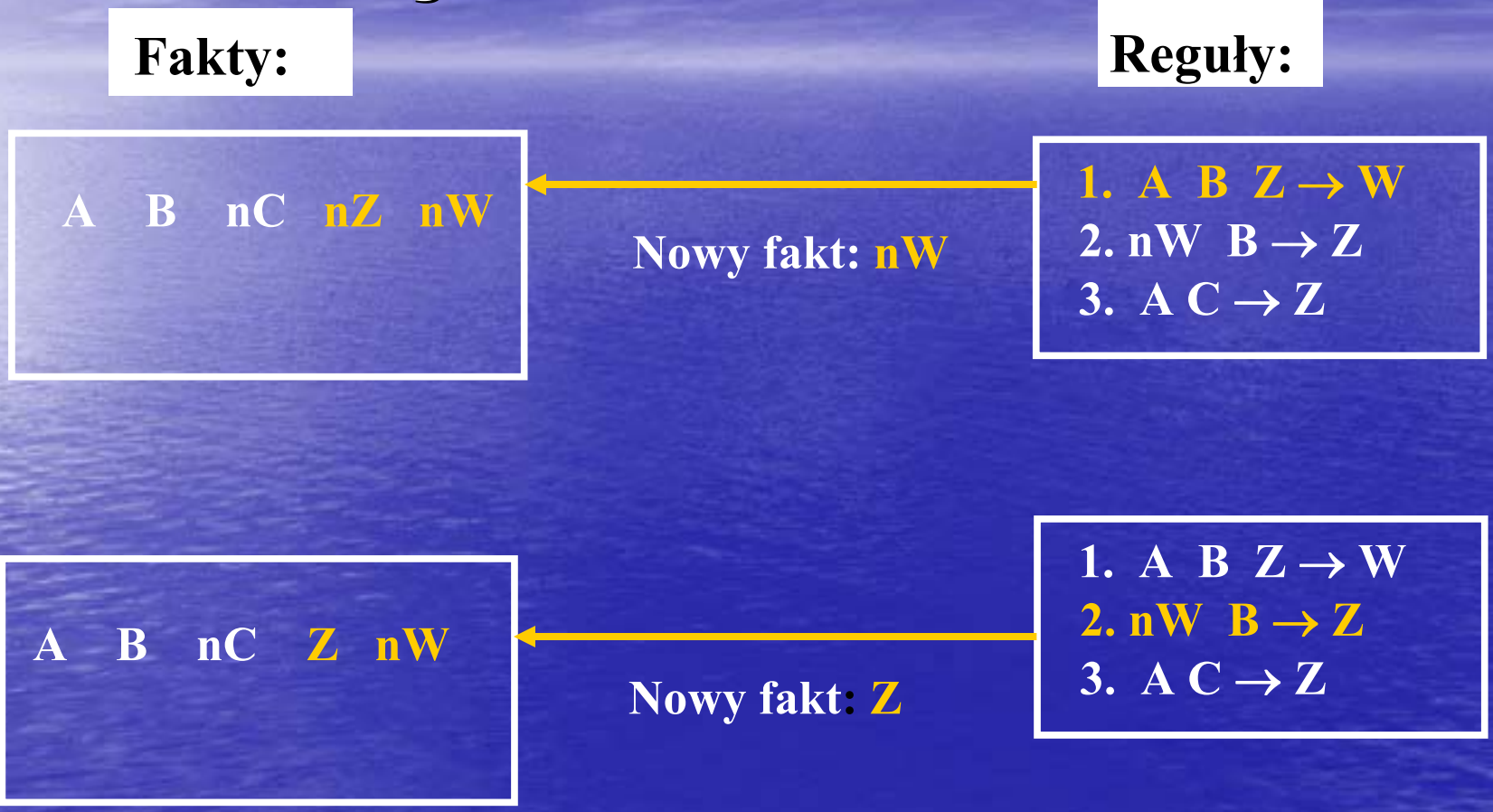
Baza Reguł

reguła (3, "W", ["nF"] ,1)

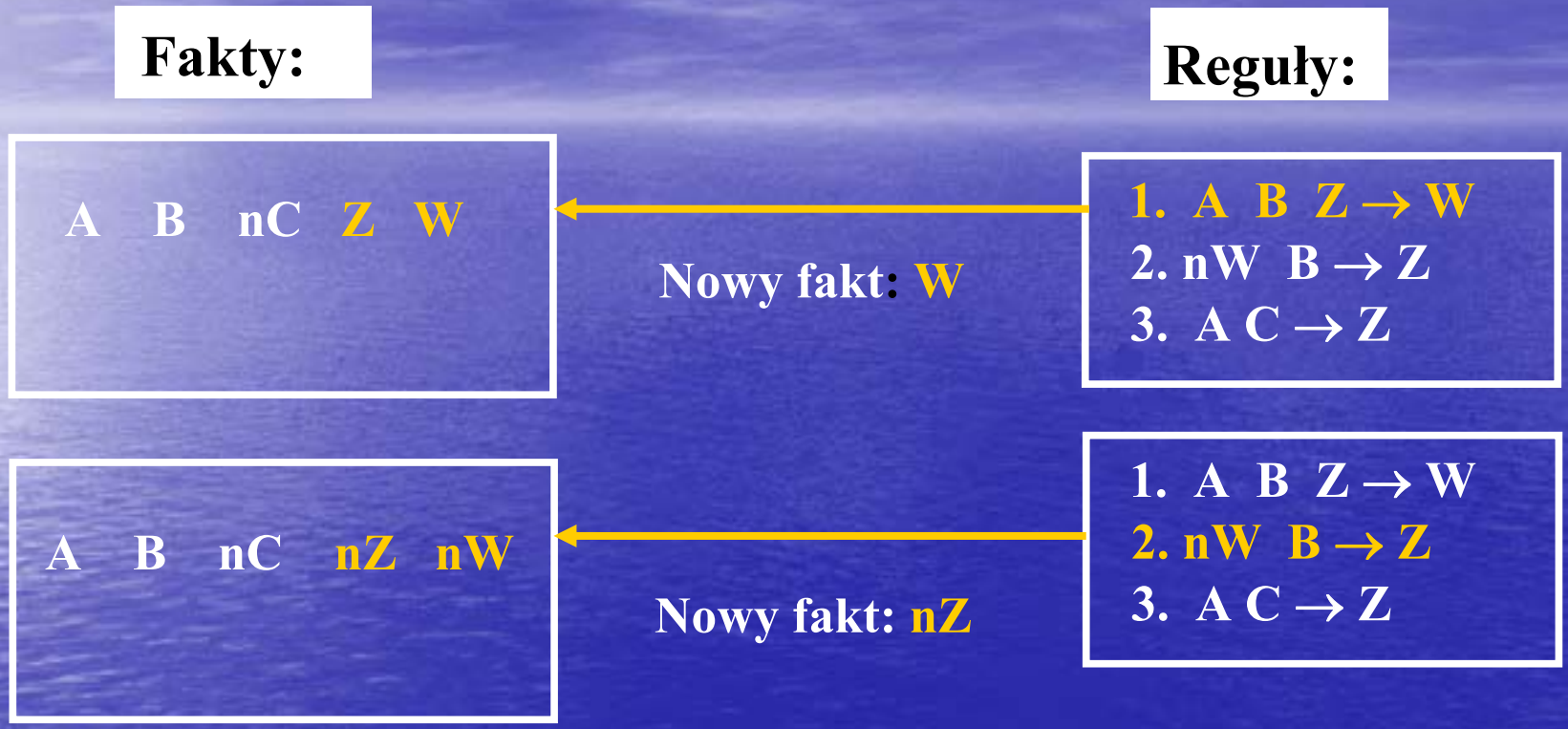
Wnioskowanie rozwinięte dokładne w przód: przykład (b)
sprzeczna baza reguł



Wnioskowanie rozwinięte dokładne w przód: przykład (b)
sprzeczna baza reguł



Wnioskowanie rozwinięte dokładne w przód: przykład (b)
sprzeczna baza reguł



itd. itd. Zmiany W → nZ → nW → Z → W są niekończącymi się oscylacjami, świadczącymi o sprzeczności bazy reguł.

Wnioskowanie rozwinięte dokładne w przód: przykład (c) sprzeczna baza reguł