

# SZTUCZNA INTELIGENCJA I SYSTEMY DORADCZE

## SIECI BAYESSOWSKIE 2

## Rodzaje zadan dla sieci bayessowskiej

Zapytania proste: oblicz brzegową wartość warunkową  $P(X_i|E=e)$

np.  $P(NoGas|Gauge=empty, Lights=on, Starts=false)$

Zapytania koniunkcyjne:  $P(X_i, X_j|E=e) = P(X_i|E=e)P(X_j|X_i, E=e)$

Decyzje optymalizacyjne: sieć zawiera informację o użyteczności;

np. wnioskowanie probabilistyczne dla  $P(outcome|action, evidence)$

Wartość informacji: która przesłankę sprawdzić jako następną?

Analiza wrażliwości: które wartości prawdopodobieństwa są najbardziej krytyczne?

Wyjaśnienie: Dlaczego potrzebuję nowego rozrusznika?

# Wnioskowanie w sieci bayesowskiej

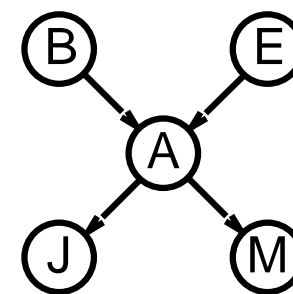
- ◇ Wnioskowanie dokładne
  - Przez wyliczanie wartości
  - Przez eliminację zmiennych
- ◇ Wnioskowanie aproksymacyjne
  - Przez symulację stochastyczną
  - metodą Monte Carlo z łańcucha Markowa

## Wnioskowanie przez wyliczanie wartosci

Sumowanie iloczynów z prawdopodobieństw brzegowych bez faktycznego konstruowania ich jawnej reprezentacji, przy użyciu prawdopodobieństw warunkowych z sieci bayessowskiej

Proste zapytanie w sieci z alarmem domowym:

$$\begin{aligned} &P(B|j, m) \\ &= P(B, j, m) / P(j, m) \\ &= \alpha P(B, j, m) \\ &= \alpha \sum_e \sum_a P(B, e, a, j, m) \end{aligned}$$



Przechodząc po zmiennych w kolejności zgodnej z siecią (np.  $B, E, A, J, M$ ) wyciągamy sumowanie po kolejnych zmiennych na zewnątrz wyrażenia i używamy wartości prawdopodobieństw z tablic TPW:

$$\begin{aligned} &P(B|j, m) \\ &= \alpha \sum_e \sum_a P(B) P(e) P(a|B, e) P(j|a) P(m|a) \\ &= \alpha P(B) \sum_e P(e) \sum_a P(a|B, e) P(j|a) P(m|a) \end{aligned}$$

## Wyliczanie wartosci: algorytm

**function** ENUMERATION-ASK( $X, \mathbf{e}, bn$ ) **returns** a distribution over  $X$

**inputs:**  $X$ , the query variable

$\mathbf{e}$ , observed values for variables  $\mathbf{E}$

$bn$ , a Bayesian network with variables  $\{X\} \cup \mathbf{E} \cup \mathbf{Y}$

$Q(X) \leftarrow$  a distribution over  $X$ , initially empty

**for each** value  $x_i$  of  $X$  **do**

    extend  $\mathbf{e}$  with value  $x_i$  for  $X$

$Q(x_i) \leftarrow$  ENUMERATE-ALL(VARS[ $bn$ ],  $\mathbf{e}$ )

**return** NORMALIZE( $Q(X)$ )

---

**function** ENUMERATE-ALL( $vars, \mathbf{e}$ ) **returns** a real number

**if** EMPTY?( $vars$ ) **then return** 1.0

$Y \leftarrow$  FIRST( $vars$ )

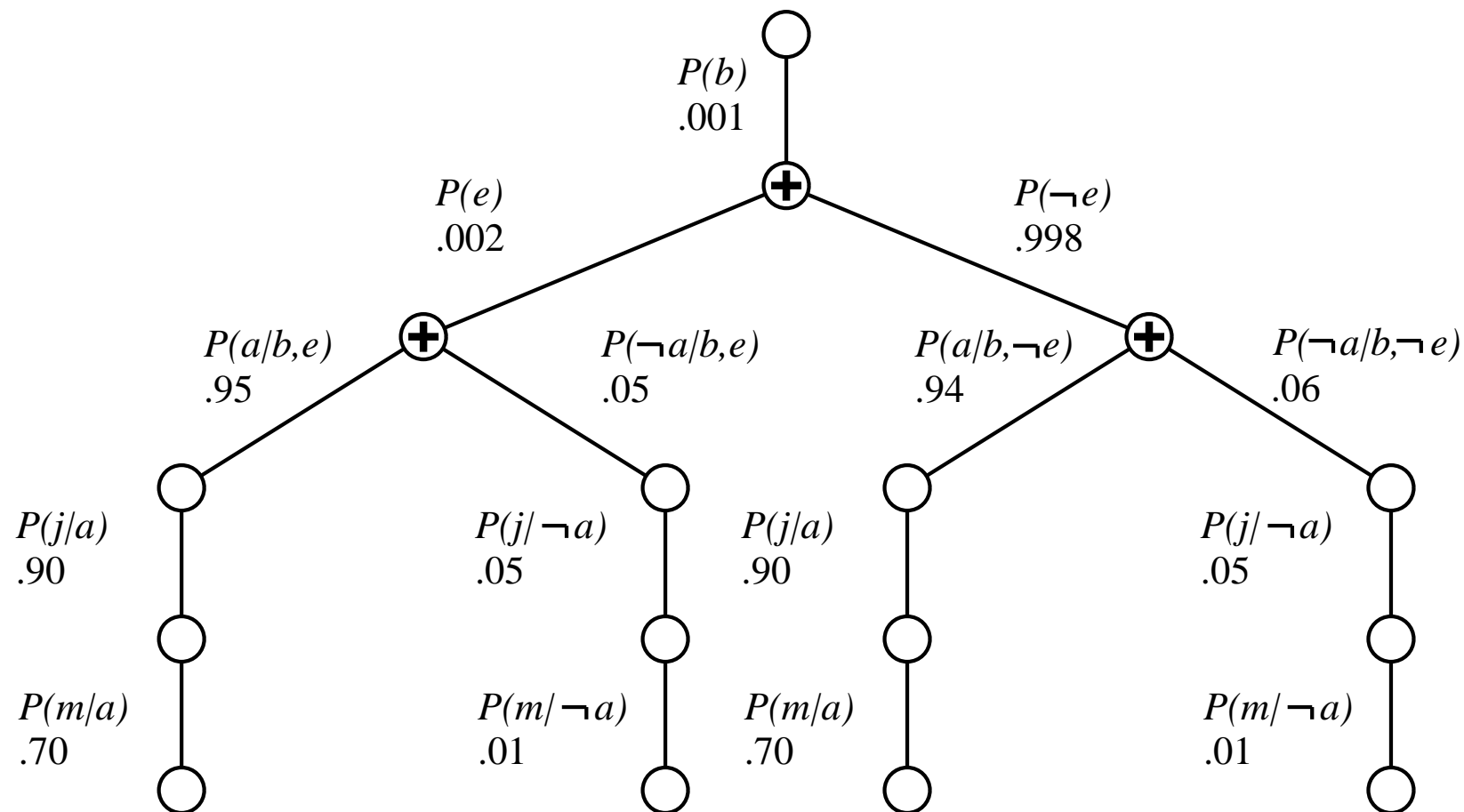
**if**  $Y$  has value  $y$  in  $\mathbf{e}$

**then return**  $P(y \mid \text{Parent}(Y)) \times \text{ENUMERATE-ALL}(\text{REST}(vars), \mathbf{e})$

**else return**  $\sum_y P(y \mid \text{Parent}(Y)) \times \text{ENUMERATE-ALL}(\text{REST}(vars), \mathbf{e}_y)$

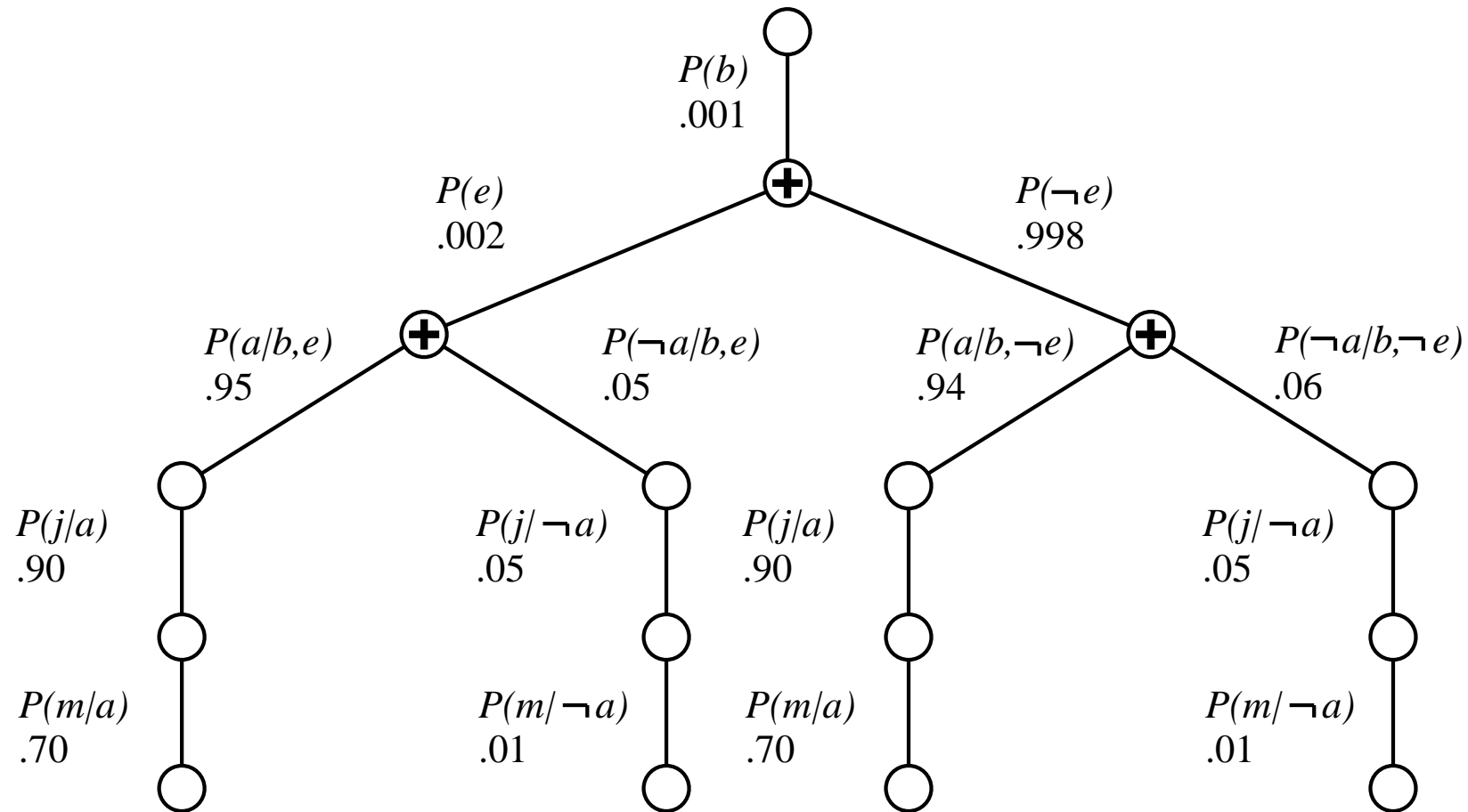
        where  $\mathbf{e}_y$  is  $\mathbf{e}$  extended with  $Y = y$

# Wyliczanie wartosci: dzialanie



Rekurencyjne wyliczanie zmiennych w głąb sieci:  $O(n)$  pamięci,  $O(d^n)$  czasu

## Wyliczanie wartosci: dzialanie



Wyliczanie jest nieefektywne: powtarza obliczenia  
 np. liczy  $P(j|a)P(m|a)$  dla każdej wartości  $e$

## Wnioskowanie przez eliminacje zmiennych

Eliminacja zmiennych: wykonuje sumowanie z prawej do lewej, pamięta wyniki pośrednie (czynniki) w celu uniknięcia powtórzeń

$$\begin{aligned} \mathbf{P}(B|j, m) &= \alpha \underbrace{\mathbf{P}(B)}_B \underbrace{\sum_e P(e)}_E \underbrace{\sum_a \mathbf{P}(a|B, e)}_A \underbrace{P(j|a)}_J \underbrace{P(m|a)}_M \\ &= \alpha \mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e) P(j|a) f_M(a) \\ &= \alpha \mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e) f_{JM}(a) \\ &= \alpha \mathbf{P}(B) \sum_e P(e) f_{\bar{A}JM}(b, e) \\ &= \alpha \mathbf{P}(B) f_{\bar{E}\bar{A}JM}(b) \\ &= \alpha f_B(b) \times f_{\bar{E}\bar{A}JM}(b) \end{aligned}$$

$$f_M(A) = \begin{pmatrix} P(m|a) \\ P(m|\neg a) \end{pmatrix}, \quad f_{JM}(A) = f_J(A) \times f_M(A) = \begin{pmatrix} P(j|a)P(m|a) \\ P(j|\neg a)P(m|\neg a) \end{pmatrix}$$

$f_A(A, B, E)$  jest macierzą  $2 \times 2 \times 2$  dla wszystkich wartości  $A, B, E$

$$f_{\bar{A}JM}(B, E) = f_A(a, B, E) \times f_{JM}(a) + f_A(\neg a, B, E) \times f_{JM}(\neg a)$$

$$f_{\bar{E}\bar{A}JM}(B, E) = f_E(e) \times f_{\bar{A}JM}(B, e) + f_E(\neg e) \times f_{\bar{A}JM}(B, \neg e)$$



## Eliminacja zmiennych: algorytm

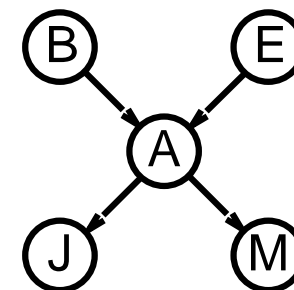
```
function ELIMINATION-ASK( $X, e, bn$ ) returns a distribution over  $X$   
inputs:  $X$ , the query variable  
          $e$ , evidence specified as an event  
          $bn$ , a belief network specifying joint distribution  $P(X_1, \dots, X_n)$   
 $factors \leftarrow []$ ;  $vars \leftarrow \text{REVERSE}(\text{VARS}[bn])$   
for each  $var$  in  $vars$  do  
     $factors \leftarrow [\text{MAKE-FACTOR}(var, e) | factors]$   
    if  $var$  is a hidden variable then  $factors \leftarrow \text{SUM-OUT}(var, factors)$   
return  $\text{NORMALIZE}(\text{POINTWISE-PRODUCT}(factors))$ 
```

## Eliminacja zmiennych: zmienne nieistotne

Rozważmy zapytanie  $P(\text{JohnCalls} | \text{Burglary} = \text{true})$

$$P(J|b) = \alpha P(b) \sum_e P(e) \sum_a P(a|b, e) P(J|a) \sum_m P(m|a)$$

Suma po  $m$  jest równa 1;  $M$  jest **nieistotne** dla zapytania  
 $\Rightarrow$  Można pominąć sumowanie po zmiennych nieistotnych



Tw 1:  $Y$  jest nieistotne jeśli  $Y \notin \text{Ancestors}(\{X\} \cup \mathbf{E})$

Tutaj  $X = \text{JohnCalls}$ ,  $\mathbf{E} = \{\text{Burglary}\}$ , i

$\text{Ancestors}(\{X\} \cup \mathbf{E}) = \{\text{Alarm}, \text{Earthquake}\}$

więc  $M$  jest nieistotne

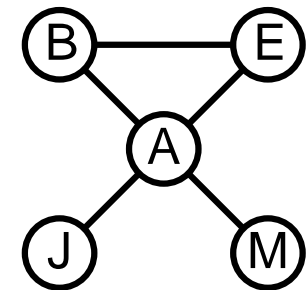
## Eliminacja zmiennych: zmienne nieistotne

Def: moralny graf sieci bayessowskiej (nieskierowany): zawiera krawędzie z oryginalnej sieci bez kierunku oraz krawędzie pomiędzy każdą parą rodziców mającą wspólne dziecko

Def: **A** jest m-odseparowane od **B** przez **C** wtw gdy jest odseparowane przez **C** w grafie moralnym

Tw 2: **Y** jest nieistotne jeśli jest m-odseparowane od **X** przez **E**

Dla  $P(\textit{JohnCalls} | \textit{Alarm} = \textit{true})$ , obie *Burglary* i *Earthquake* są nieistotne



# Złożoność dokładnego wnioskowania

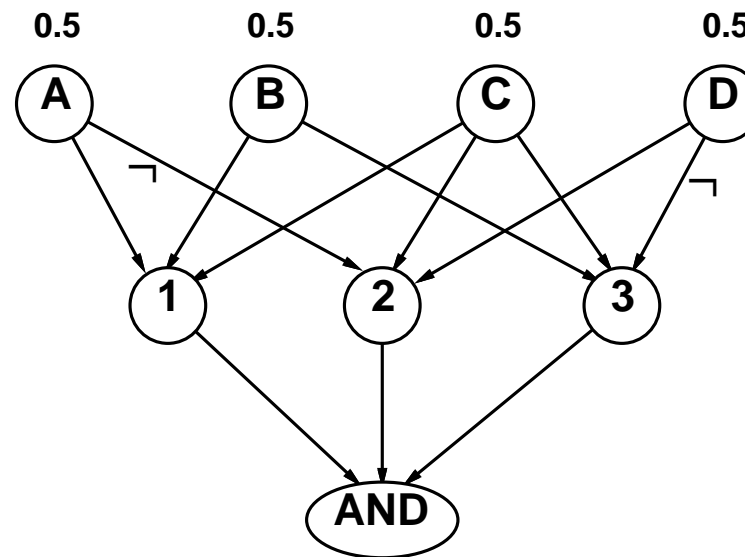
Sieci pojedynczych połączeń (polidrzewa):

- każde dwa wierzchołki połączone są co najwyżej jedną ścieżką
- złożoność czasowa i pamięciowa algorytmu eliminacji zmiennych  $O(d^k n)$

Sieci wielokrotnych połączeń:

- można zredukować 3SAT do dokładnego wnioskowania  $\Rightarrow$  NP-trudne
- równoważne **zliczaniu** modeli 3SAT  $\Rightarrow$  #P-zupełne

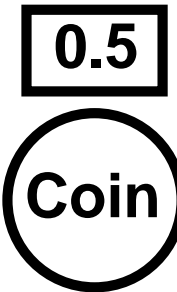
1.  $A \vee B \vee C$
2.  $C \vee D \vee \neg A$
3.  $B \vee C \vee \neg D$



# Wnioskowanie przez symulacje stochastyczna

Podstawowy pomysł:

- 1) Losuj  $N$  próbek z rozkładem próbkowym  $S$
- 2) Oblicz aproksymacyjne prawdopodobieństwo wynikowe  $\hat{P}$
- 3) Udowodnij zbieżność do prawdopodobieństwa faktycznego  $P$



Wnioskowanie stochastyczne bezwarunkowe (bez przesłanek):

- Próbkowanie bezpośrednie

Wnioskowanie stochastyczne warunkowe (z przesłankami):

- Próbkowanie z odrzucaniem: odrzuca próbki niezgodne z przesłankami
- Wążenie prawdopodobieństwa próbek:
  - używa przesłanek do wążenia prawdopodobieństwa próbek
- Monte Carlo z łańcucha Markowa (MCMC):
  - próbkuje z procesu stochastycznego, w którym prawdopodobieństwo stacjonarne jest rzeczywistym prawdopodobieństwem warunkowym

# Probkowanie bezpośrednie

**function** DIRECT-SAMPLING( $X, bn, N$ ) **returns** an estimate of  $P(X)$

**local variables:**  $\mathbf{N}$ , a vector of counts over  $X$ , initially zero

**for**  $j = 1$  to  $N$  **do**

$\mathbf{x} \leftarrow \text{PRIOR-SAMPLE}(bn)$

$\mathbf{N}[x] \leftarrow \mathbf{N}[x] + 1$  where  $x$  is the value of  $X$  in  $\mathbf{x}$

**return** NORMALIZE( $\mathbf{N}[X]$ )

---

**function** PRIOR-SAMPLE( $bn$ ) **returns** an event sampled from  $bn$

**inputs:**  $bn$ , a belief network specifying joint distribution  $\mathbf{P}(X_1, \dots, X_n)$

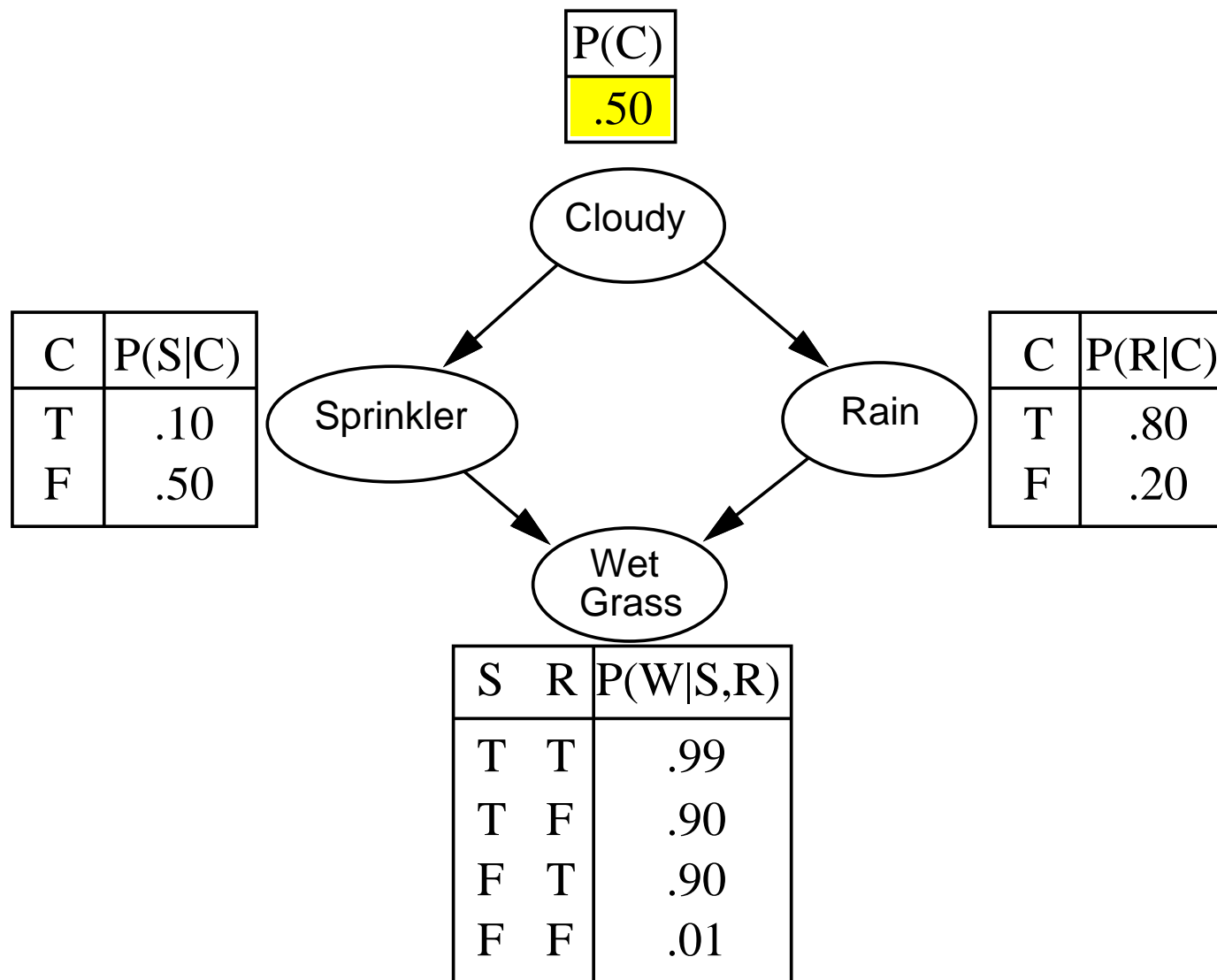
$\mathbf{x} \leftarrow$  an event with  $n$  elements

**for**  $i = 1$  to  $n$  **do**

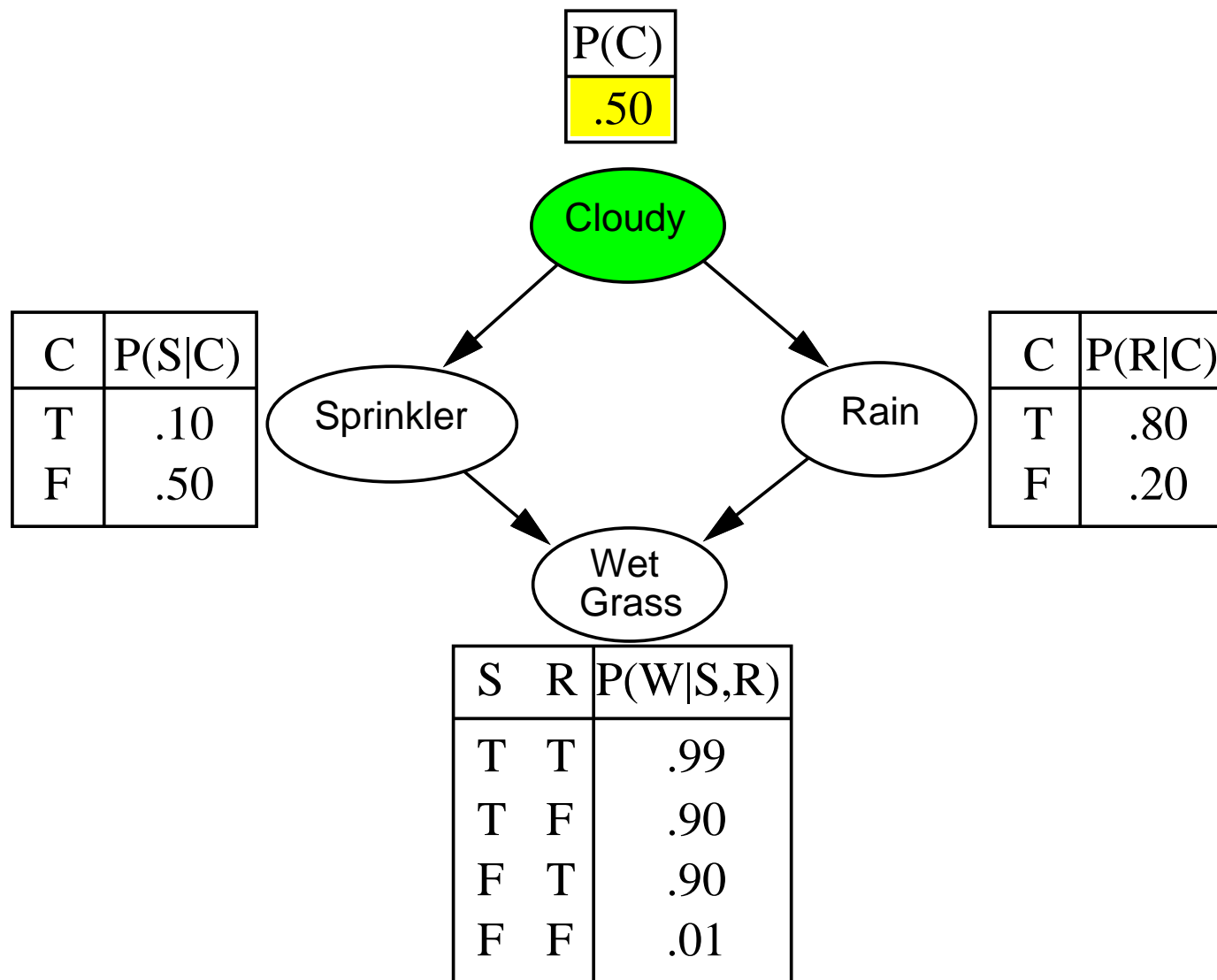
$x_i \leftarrow$  a random sample from  $\mathbf{P}(X_i \mid \text{Parents}(X_i))$

**return**  $\mathbf{x}$

## Probkowanie bezpośrednie: przykład

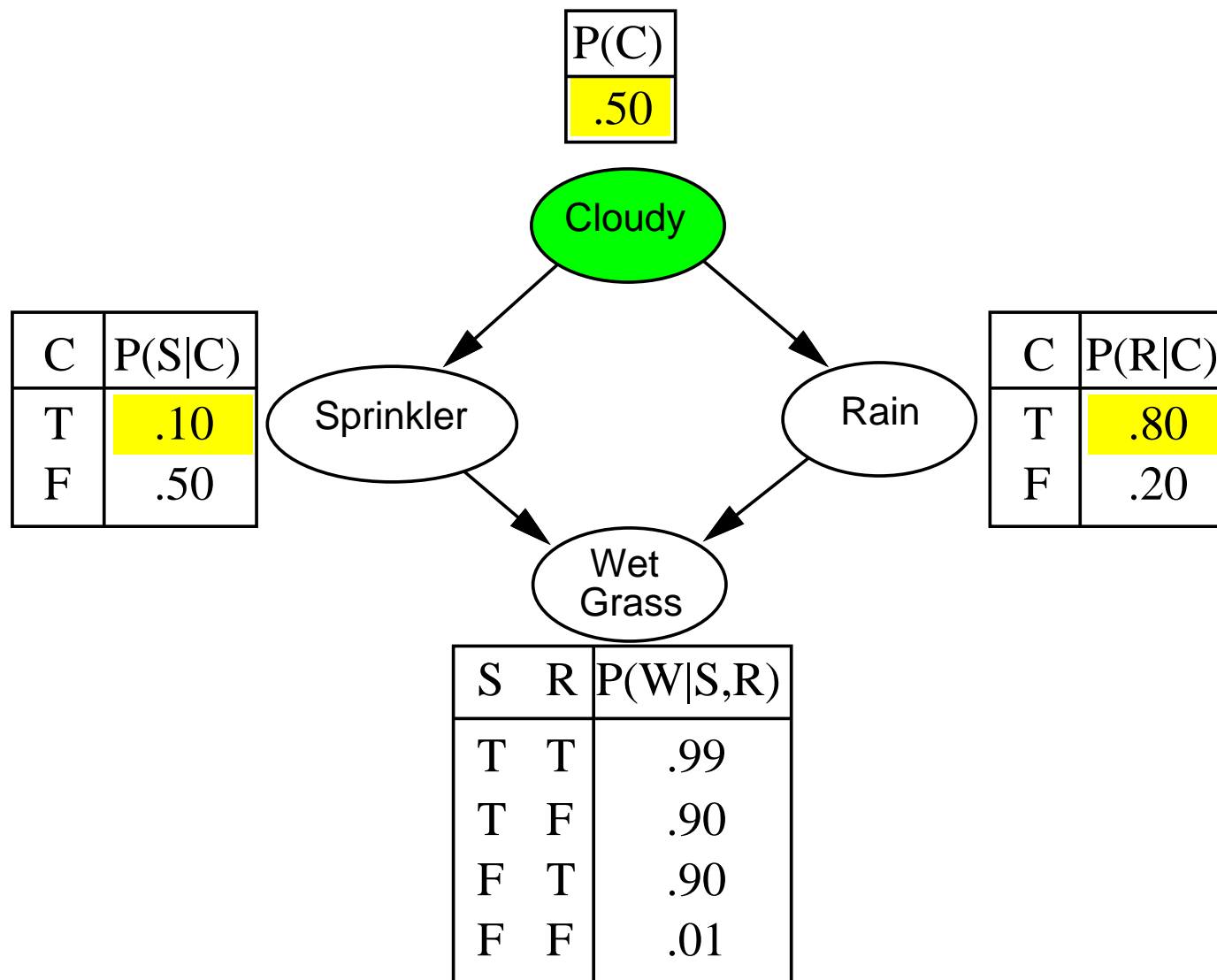


# Probkowanie bezposrednie: przyklad

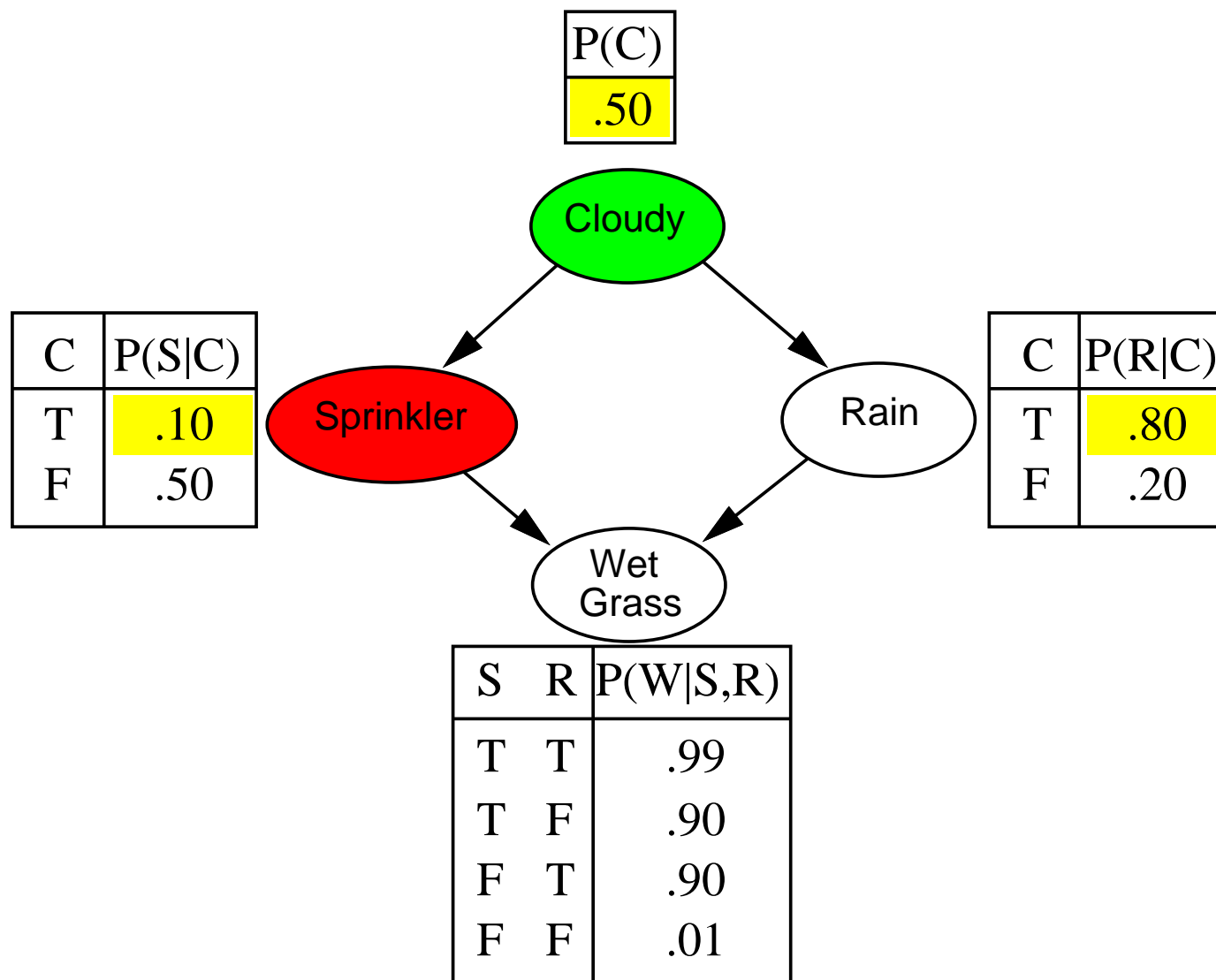




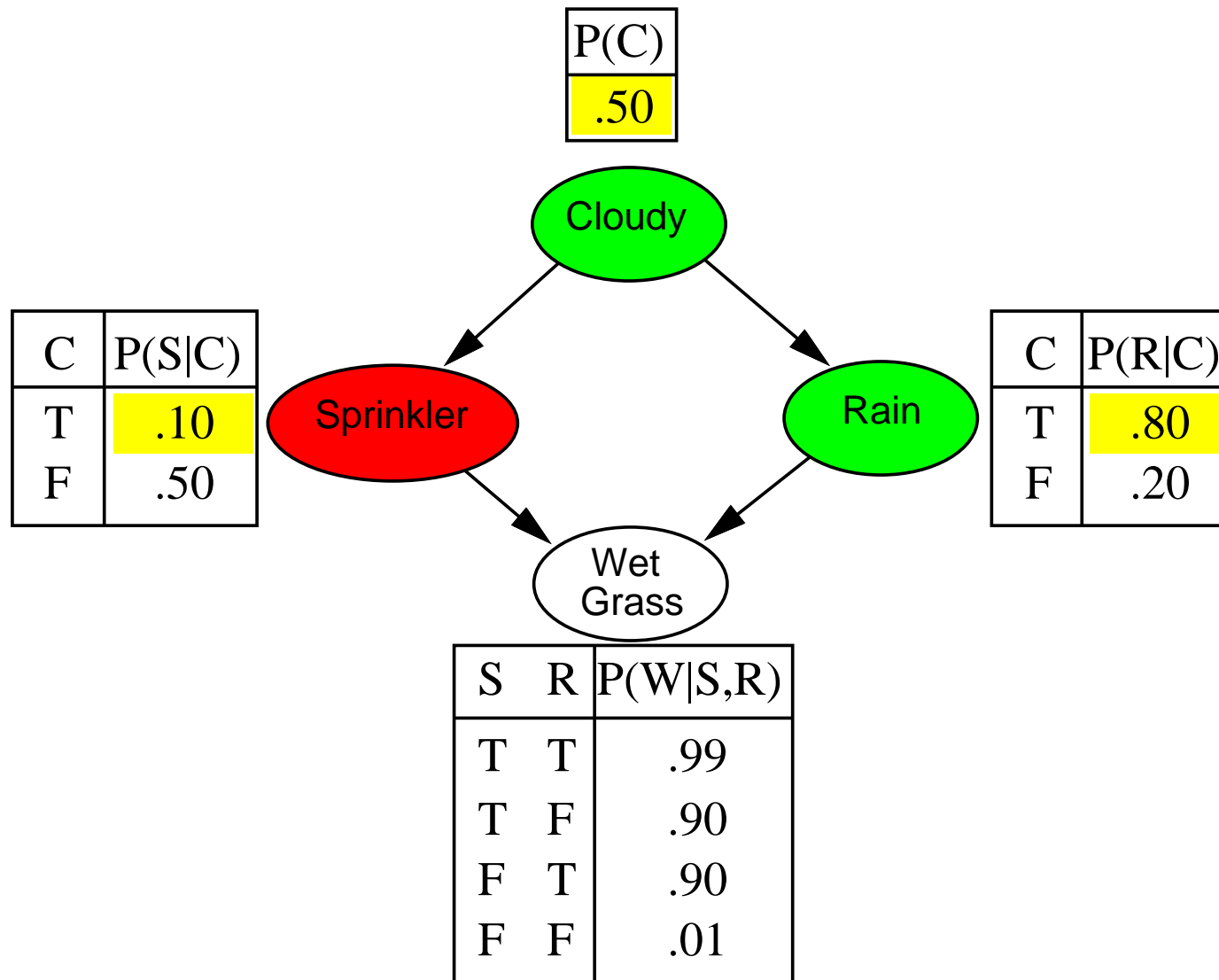
## Probkowanie bezpośrednie: przykład



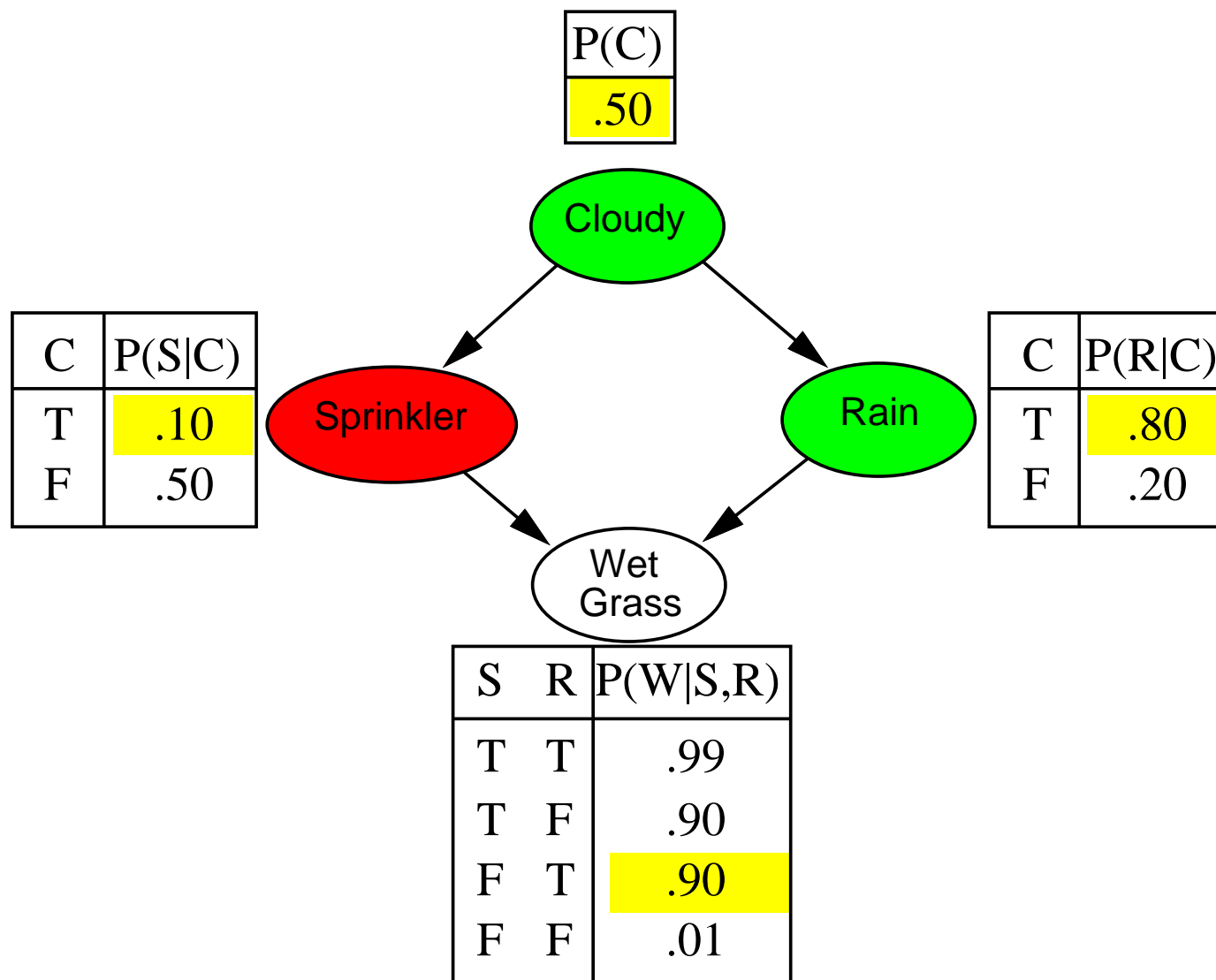
# Probkowanie bezposrednie: przyklad



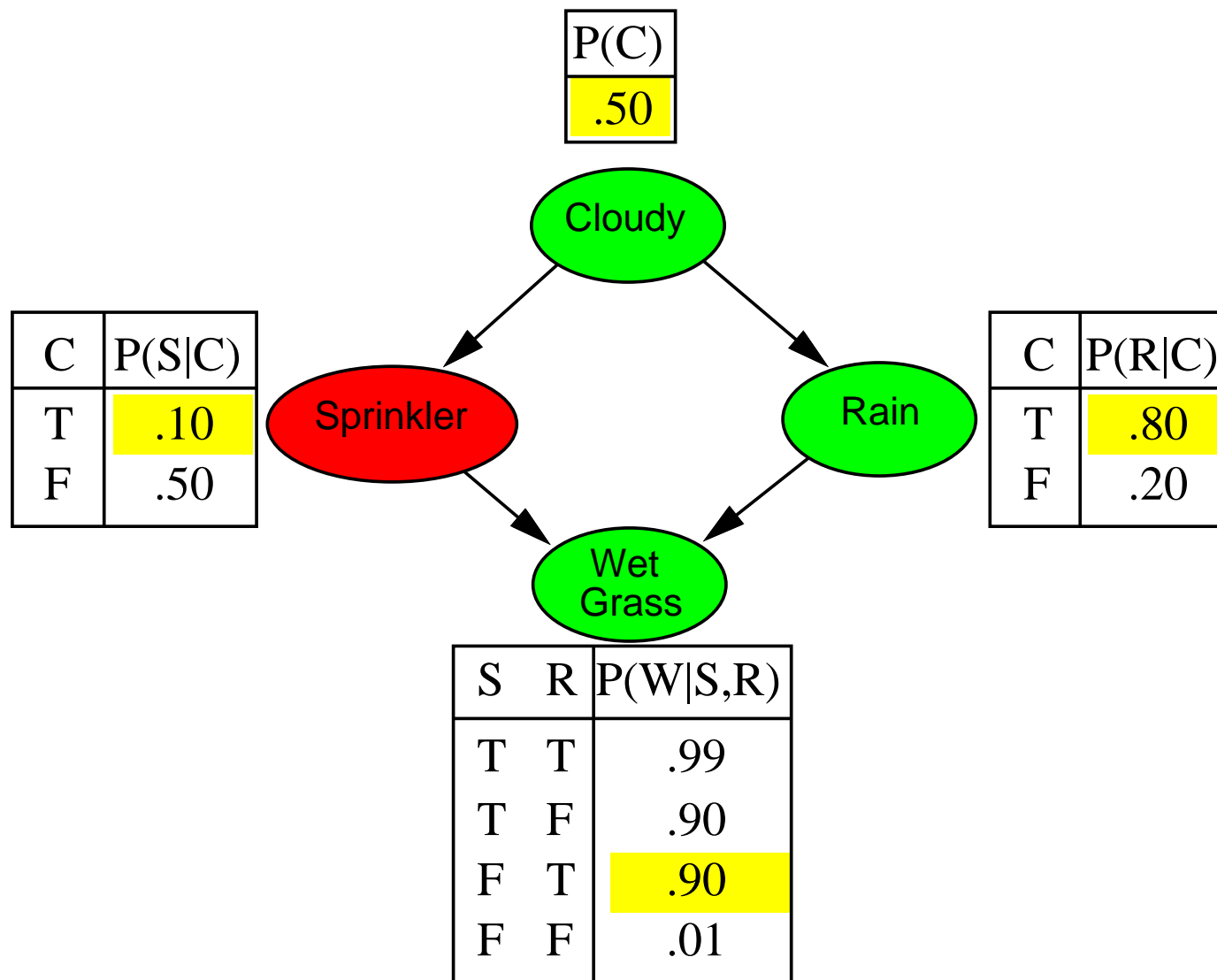
# Probkowanie bezposrednie: przyklad



# Probkowanie bezposrednie: przyklad



## Probkowanie bezpośrednie: przykład



## Probkowanie bezpośrednio: własności

Prawdopodobieństwo, że PRIORSAMPLE generuje dane zdarzenie

$$S_{PS}(x_1 \dots x_n) = \prod_{i=1}^n P(x_i | \text{Parents}(X_i)) = P(x_1 \dots x_n)$$

to odpowiada prawdopodobieństwu faktycznemu tego zdarzenia

Np.  $S_{PS}(t, f, t, t) = 0.5 \times 0.9 \times 0.8 \times 0.9 = 0.324 = P(t, f, t, t)$

$N_{PS}(x_1 \dots x_n)$  — liczbą próbek wygenerowanych dla zdarzenia  $x_1, \dots, x_n$

Wtedy

$$\begin{aligned} \lim_{N \rightarrow \infty} \hat{P}(x_1, \dots, x_n) &= \lim_{N \rightarrow \infty} N_{PS}(x_1, \dots, x_n) / N \\ &= S_{PS}(x_1, \dots, x_n) \\ &= P(x_1 \dots x_n) \end{aligned}$$

Powyższą własność algorytmu DIRECTSAMPLING nazywamy **spójnością**

Notacja:  $\hat{P}(x_1, \dots, x_n) \approx P(x_1 \dots x_n)$

## Probkowanie z odrzucaniem

$\hat{P}(X|e)$  szacowane z próbek zgodnych z przesłankami  $e$

```
function REJECTION-SAMPLING( $X, e, bn, N$ ) returns an estimate of  $P(X|e)$ 
  local variables:  $N$ , a vector of counts over  $X$ , initially zero
  for  $j = 1$  to  $N$  do
     $x \leftarrow \text{PRIOR-SAMPLE}(bn)$ 
    if  $x$  is consistent with  $e$  then
       $N[x] \leftarrow N[x] + 1$  where  $x$  is the value of  $X$  in  $x$ 
  return NORMALIZE( $N[X]$ )
```

Np. oszacowanie  $P(Rain|Sprinkler = true)$  przy użyciu 100 próbek

27 próbek ma  $Sprinkler = true$

Z tego, 8 ma  $Rain = true$  i 19 ma  $Rain = false$ .

$\hat{P}(Rain|Sprinkler = true) = \text{NORMALIZE}(\langle 8, 19 \rangle) = \langle 0.296, 0.704 \rangle$

## Probkowanie z odrzucaniem: własności

$$\begin{aligned}\hat{P}(X|e) &= \alpha N_{PS}(X, e) && \text{(wynik algorytmu REJECTION SAMPLING)} \\ &= N_{PS}(X, e) / N_{PS}(e) && \text{(normalizowane przez } N_{PS}(e) \text{)} \\ &\approx P(X, e) / P(e) && \text{(własność PRIOR SAMPLE)} \\ &= P(X|e) && \text{(prawdopodobieństwo faktyczne)}\end{aligned}$$

Zatem próbkowanie z odrzucaniem ma własność spójności  
tzn. oszacowanie zbiega do faktycznego prawdopodobieństwa warunkowego

Problem: bardzo kosztowne jeśli  $P(e)$  jest małe

$P(e)$  rozpada się wykładniczo wraz z liczbą zmiennych!



## Wazenie prawdopodobienstwa probek

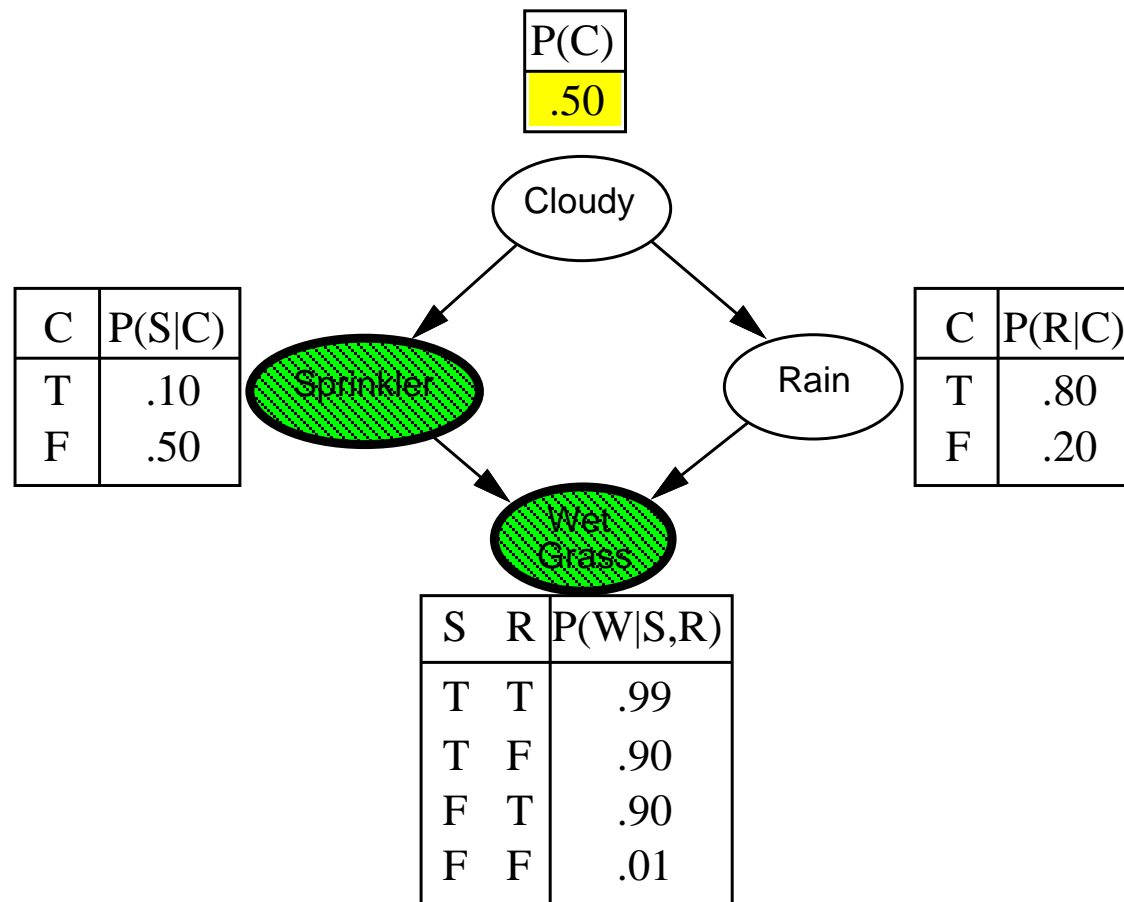
Pomysł: ustala zmienne z przesłanek, próbkuję tylko zmienna spoza przesłanek, i wazy prawdopodobienstwo kazdej próbki stosownie do przesłanek

```
function LIKELIHOOD-WEIGHTING( $X, \mathbf{e}, bn, N$ ) returns an estimate of  $P(X|\mathbf{e})$   
  local variables:  $\mathbf{W}$ , a vector of weighted counts over  $X$ , initially zero  
  for  $j = 1$  to  $N$  do  
     $\mathbf{x}, w \leftarrow \text{WEIGHTED-SAMPLE}(bn)$   
     $\mathbf{W}[x] \leftarrow \mathbf{W}[x] + w$  where  $x$  is the value of  $X$  in  $\mathbf{x}$   
  return  $\text{NORMALIZE}(\mathbf{W}[X])$ 
```

---

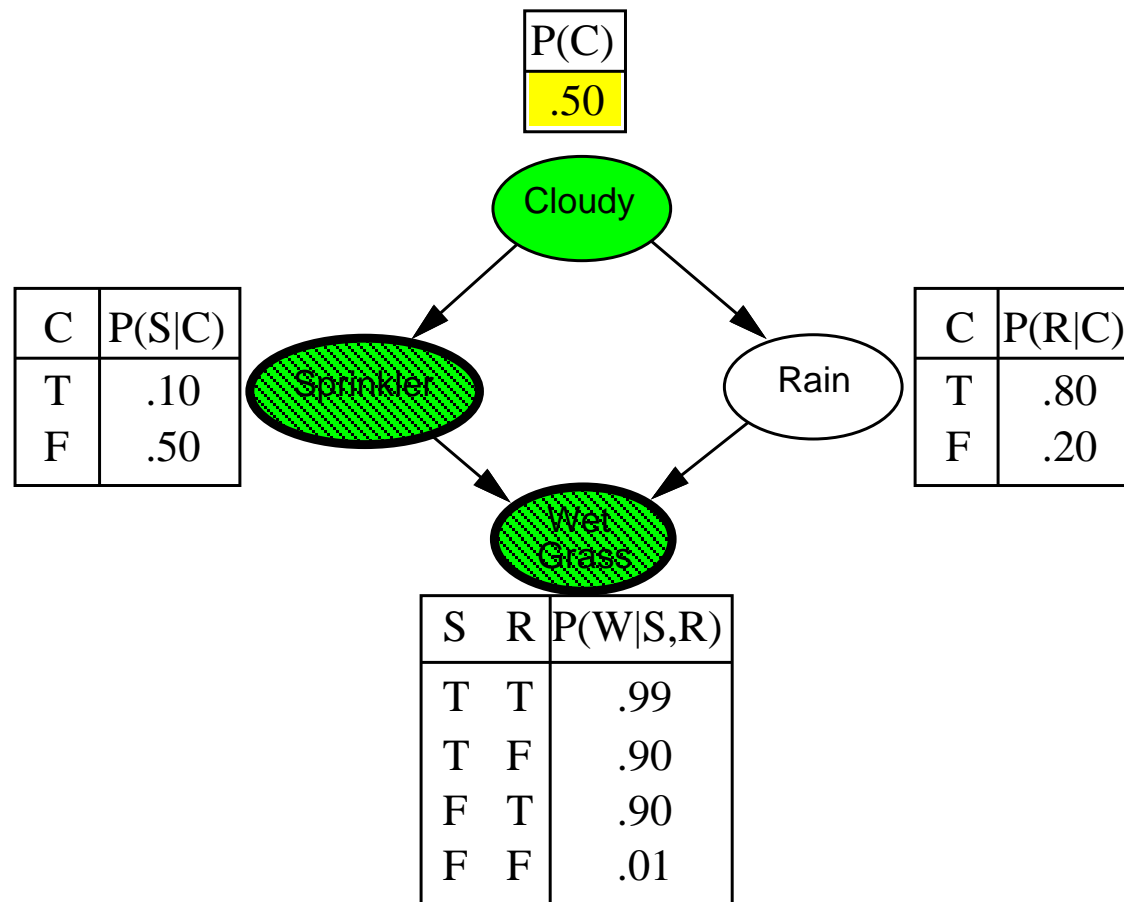
```
function WEIGHTED-SAMPLE( $bn, \mathbf{e}$ ) returns an event and a weight  
   $\mathbf{x} \leftarrow$  an event with  $n$  elements;  $w \leftarrow 1$   
  for  $i = 1$  to  $n$  do  
    if  $X_i$  has a value  $x_i$  in  $\mathbf{e}$   
      then  $w \leftarrow w \times P(X_i = x_i \mid \text{Parents}(X_i))$   
      else  $x_i \leftarrow$  a random sample from  $\mathbf{P}(X_i \mid \text{Parents}(X_i))$   
  return  $\mathbf{x}, w$ 
```

# Wazenie prawdopodobienstwa probek: przyklad



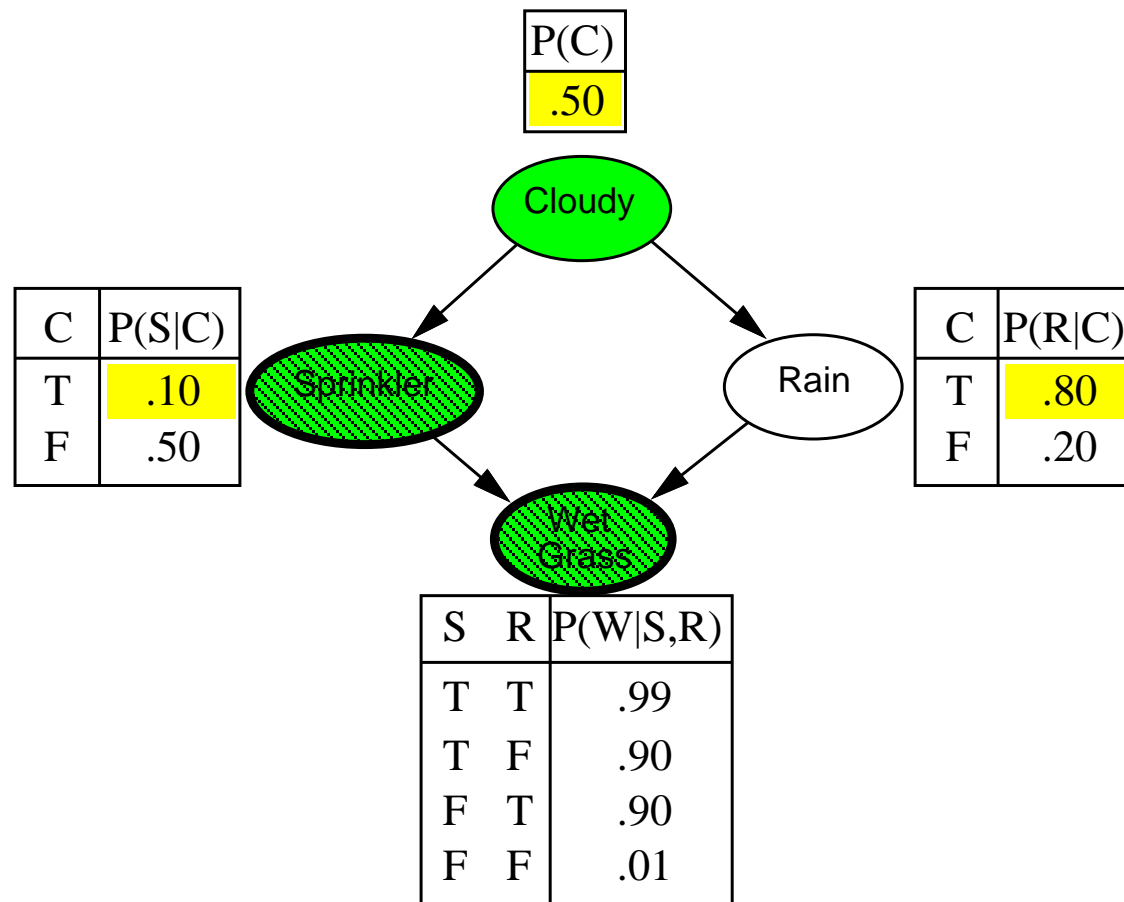
$w = 1.0$

# Wazenie prawdopodobienstwa probek: przyklad



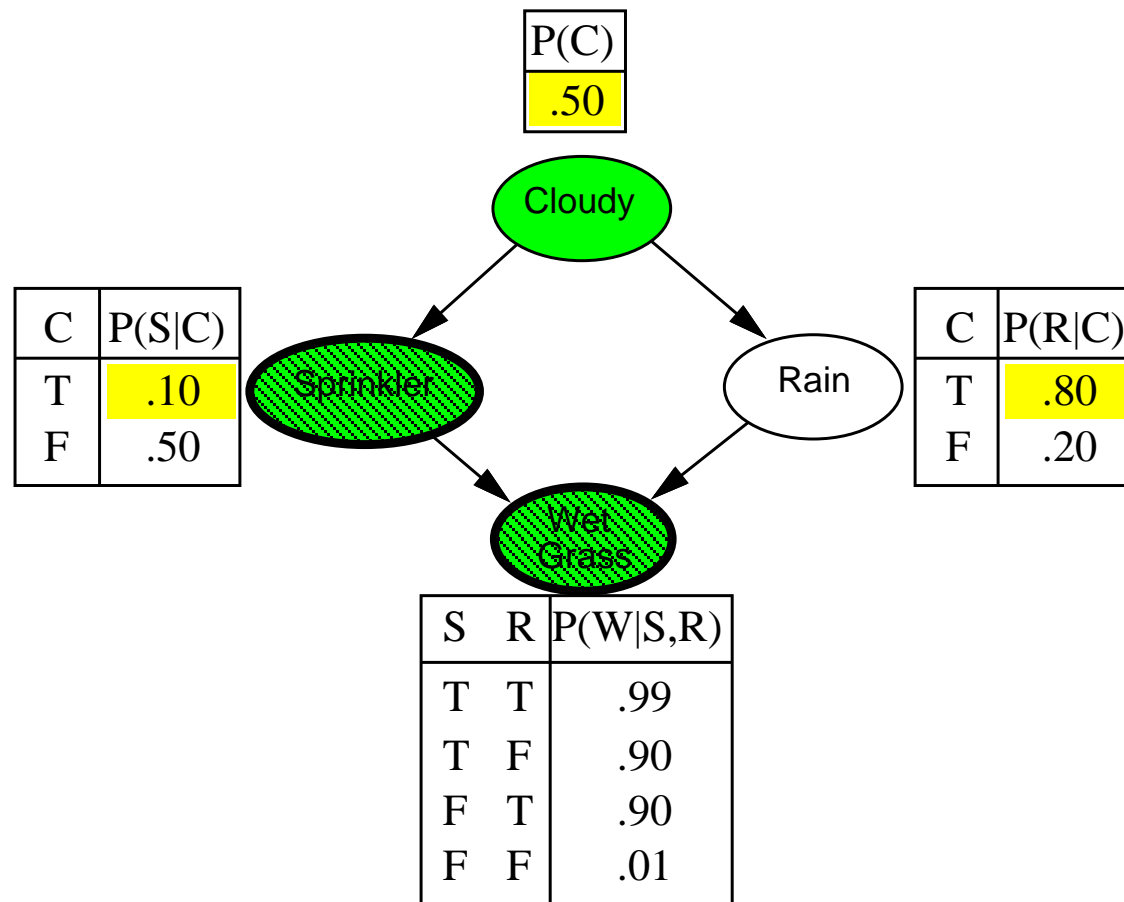
$w = 1.0$

# Wazenie prawdopodobienstwa probek: przyklad



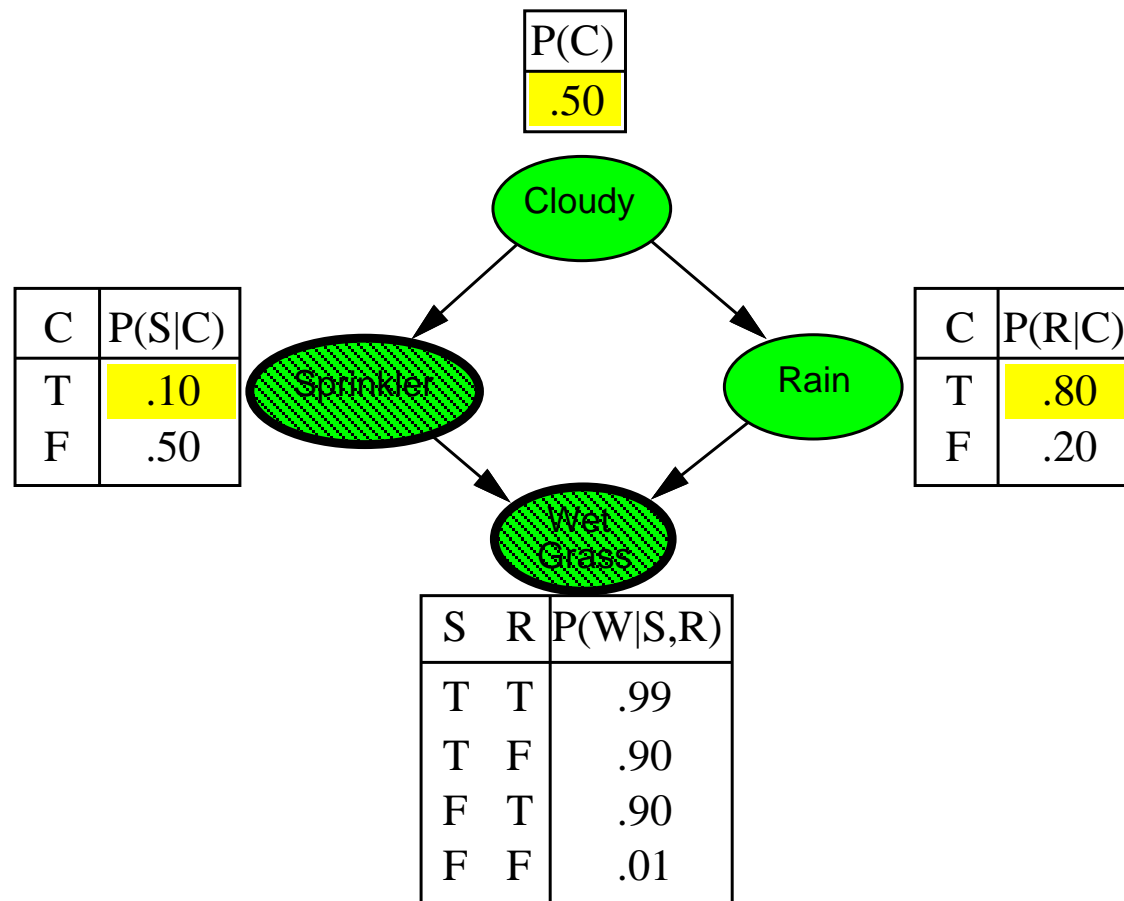
$w = 1.0$

# Wazenie prawdopodobienstwa probek: przyklad



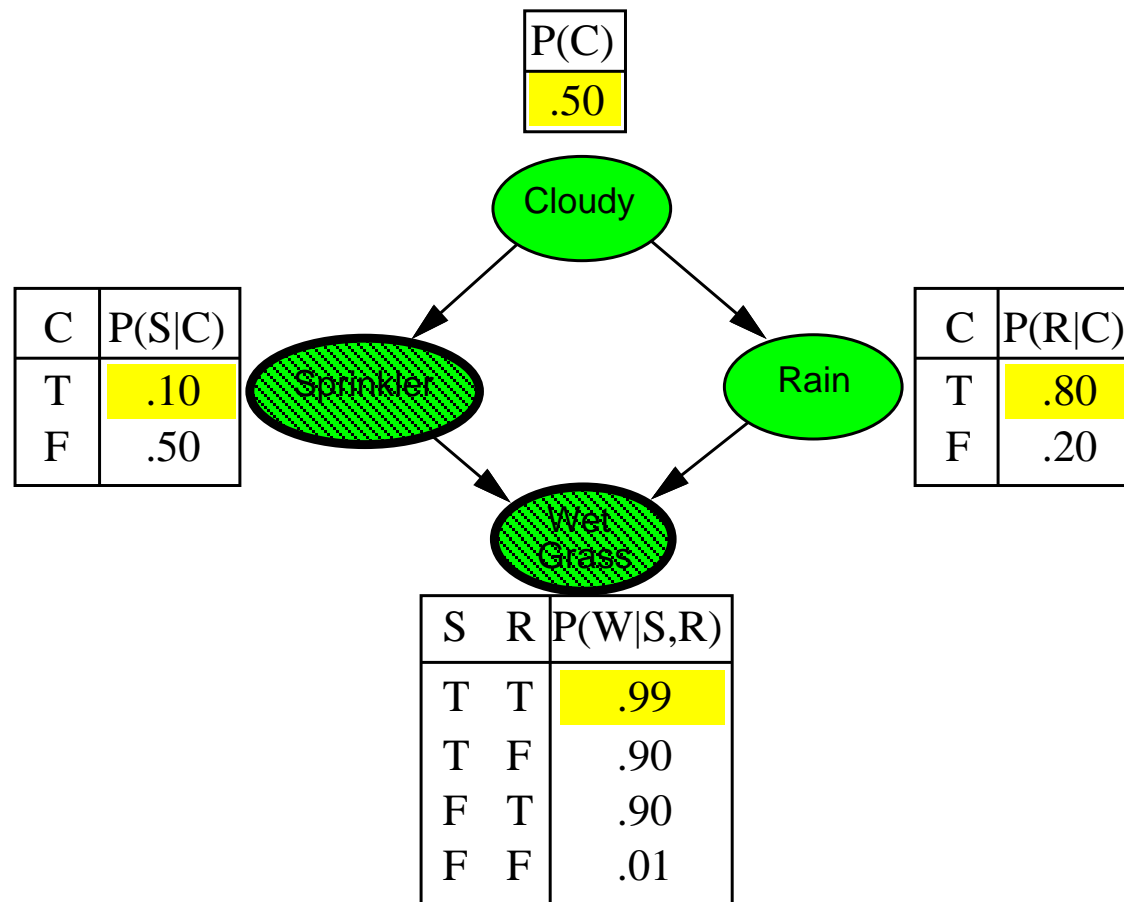
$$w = 1.0 \times 0.1$$

# Wazenie prawdopodobienstwa probek: przyklad



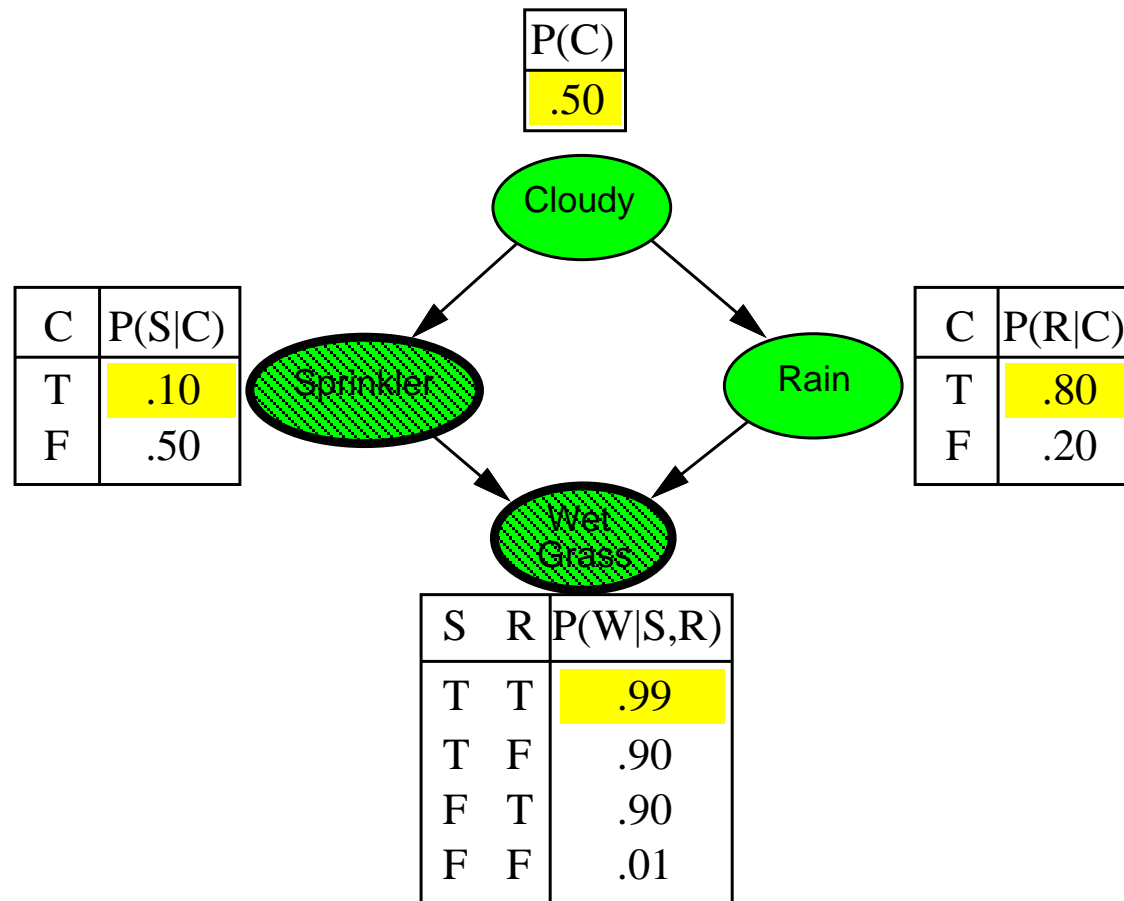
$$w = 1.0 \times 0.1$$

# Wazenie prawdopodobienstwa probek: przyklad



$$w = 1.0 \times 0.1$$

# Wazenie prawdopodobienstwa probek: przyklad



$$w = 1.0 \times 0.1 \times 0.99 = 0.099$$



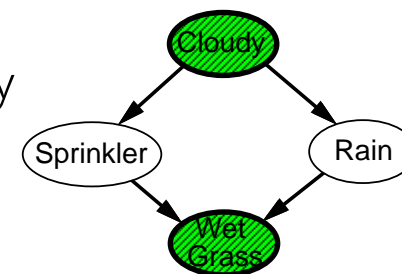
## Ważenie prawdopodobieństwa próbek: własności

Prawdopodobieństwo próbki ważonej WEIGHTEDSAMPLE wynosi

$$S_{WS}(\mathbf{z}, \mathbf{e}) = \prod_{i=1}^l P(z_i | \text{Parents}(Z_i))$$

Uwaga:  $S_{WS}$  uwzględnia tylko przesłanki z **przodków**  $z_i$

⇒ daje prawdopodobieństwo pośrednie pomiędzy  
prawdopodobieństwem a priori i a posteriori



Waga dla danej próbki  $\mathbf{z}, \mathbf{e}$  wynosi

$$w(\mathbf{z}, \mathbf{e}) = \prod_{i=1}^m P(e_i | \text{Parents}(E_i))$$

Ważone prawdopodobieństwo próbkowe:

$$\begin{aligned} & S_{WS}(\mathbf{z}, \mathbf{e}) w(\mathbf{z}, \mathbf{e}) \\ &= \prod_{i=1}^l P(z_i | \text{Parents}(Z_i)) \prod_{i=1}^m P(e_i | \text{Parents}(E_i)) \\ &= P(\mathbf{z}, \mathbf{e}) \text{ (ze standardowej, globalnej semantyki sieci)} \end{aligned}$$

Stąd ważenie prawdopodobieństwa też ma własność spójności  
ale efektywność nadal maleje przy dużej liczbie przesłanek  
ponieważ bardzo mało próbek ma dużą wagę

# Monte Carlo dla lancucha Markowa

“Stan” sieci: bieżące przypisanie wszystkich zmiennych

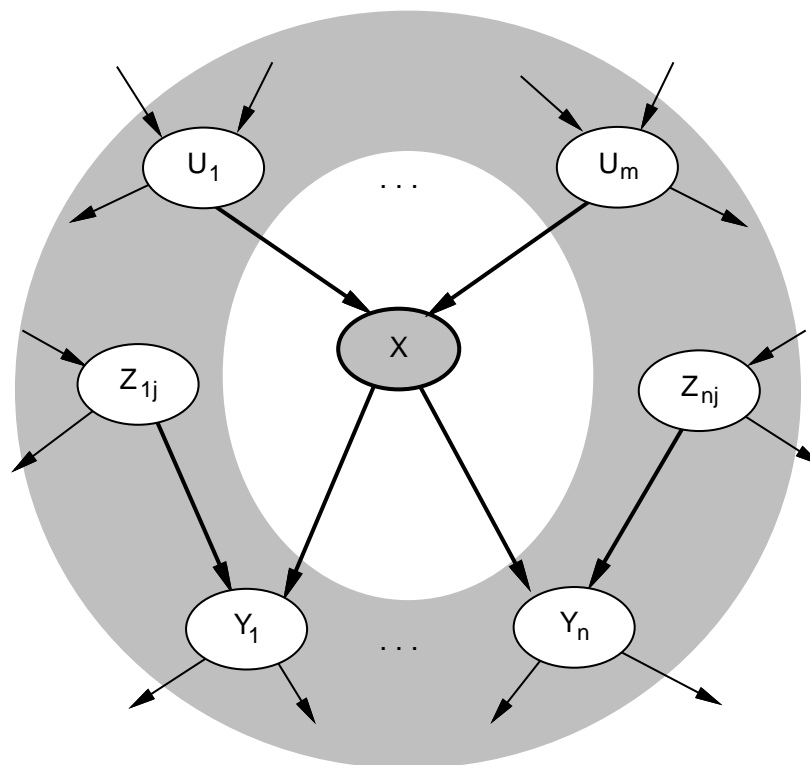
**Łańcuch Markowa**: ciąg stanów sieci, następny stan jest generowany poprzez próbkowanie jednej zmiennej nie będącej przesłanką na podstawie jej koca Markowa

```
function MCMC-ASK( $X, \mathbf{e}, bn, N$ ) returns an estimate of  $P(X|\mathbf{e})$ 
  local variables:  $\mathbf{N}[X]$ , a vector of counts over  $X$ , initially zero
                     $\mathbf{Z}$ , the nonevidence variables in  $bn$ 
                     $\mathbf{x}$ , the current state of the network, initially copied from  $\mathbf{e}$ 

  initialize  $\mathbf{x}$  with random values for the variables in  $\mathbf{Y}$ 
  for  $j = 1$  to  $N$  do
     $\mathbf{N}[x] \leftarrow \mathbf{N}[x] + 1$  where  $x$  is the value of  $X$  in  $\mathbf{x}$ 
    for each  $Z_i$  in  $\mathbf{Z}$  do
      sample the value of  $Z_i$  in  $\mathbf{x}$  from  $\mathbf{P}(Z_i|MB(Z_i))$ 
      given the values of  $MB(Z_i)$  in  $\mathbf{x}$ 
  return NORMALIZE( $\mathbf{N}[X]$ )
```

# Koc Markowa

Każdy węzeł jest warunkowo niezależny od wszystkich pozostałych przy danym jego **kocu Markowa**: rodzice + dzieci + inni rodzice dzieci



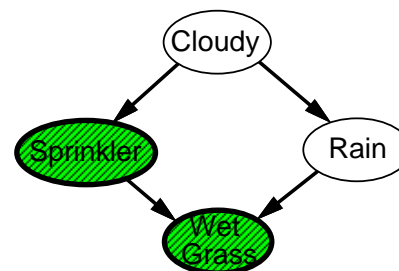
## Koc Markowa: przykład

Koc Markowa dla *Cloudy*:

*Sprinkler* i *Rain*

Koc Markowa dla *Rain*:

*Cloudy*, *Sprinkler* i *WetGrass*

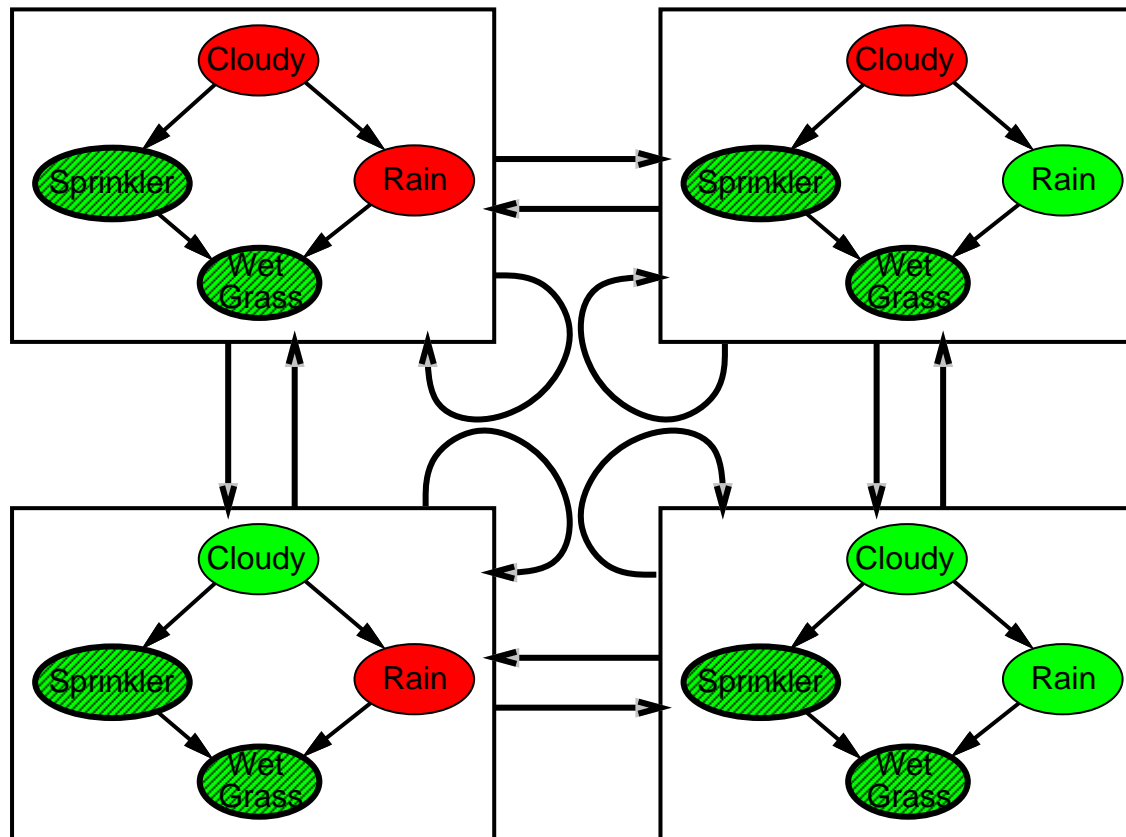


Prawdopodobieństwo warunkowe przy danym kocu Markowa:

$$P(x'_i | MB(X_i)) = P(x'_i | Parents(X_i)) \prod_{Z_j \in Children(X_i)} P(z_j | Parents(Z_j))$$

# Lancuch Markowa

Przy przesłankach  $Sprinkler = true, WetGrass = true$   
łańcuch Markowa zawiera 4 stany:



## Monte Carlo dla lancucha Markowa: przyklad

Szacowanie  $P(Rain|Sprinkler = true, WetGrass = true)$

Algorytm powtarza próbkowanie zmiennych *Cloudy* i *Rain* na podstawie ich koca Markowa. Zlicza, ile razy *Rain* było true i false w kolejnych stanach sieci.

Np. odwiedza 100 stanów

31 ma *Rain = true*, 69 ma *Rain = false*

$$\begin{aligned}\hat{P}(Rain|Sprinkler = true, WetGrass = true) \\ = \text{NORMALIZE}(\langle 31, 69 \rangle) = \langle 0.31, 0.69 \rangle\end{aligned}$$

## Monte Carlo dla łańcucha Markowa: własności

Twierdzenie: łańcuch zbiega do rozkładu stacjonarnego ( $\approx$  spójność):  
proporcja czasu spędzonego w danym stanie w czasie długiego działania sieci jest dokładnie proporcjonalna do faktycznego prawdopodobieństwa warunkowego

### ◇ Zalety

- Metoda nie jest wrażliwa na topologię sieci
- Można stosować do zmiennych dyskretnych i ciągłych

### ◇ Wady

- Zbieżność może być wolna
- Trudno określić moment, w którym algorytm daje już bliskie rozwiązanie
- Może być czasowo rozrzutny, jeśli występują duże koce Markowa:  
 $P(X_i|MB(X_i))$  nie zmienia się dużo (Prawo Wielkich Liczb)  
a jest liczone za każdym razem