Introduction to Bayesian networks

Applications to network intrusion detection

Kamil Charkiewicz

Presentation plan

- Bayesian network – definition
- How to build? How to use?
- Applications of Bayesian networks to intrusion detection

Bayesian networks

- Joint probability: $P(A_1, A_2, \ldots, A_n)$
- Conditional probability: $P(A|B) = \frac{P(A,B)}{P(B)}$
- Bayes theorem: $P(A|B) = \frac{P(B|A)*P(A)}{P(B)}$
- Path: in graph, list of edges in a way from one node to another.
  - Undirected – without specified directions on edges
  - Directed – with specified directions of the edges

Bayesian networks

- Data:
  - Attributes
  - Values for examples

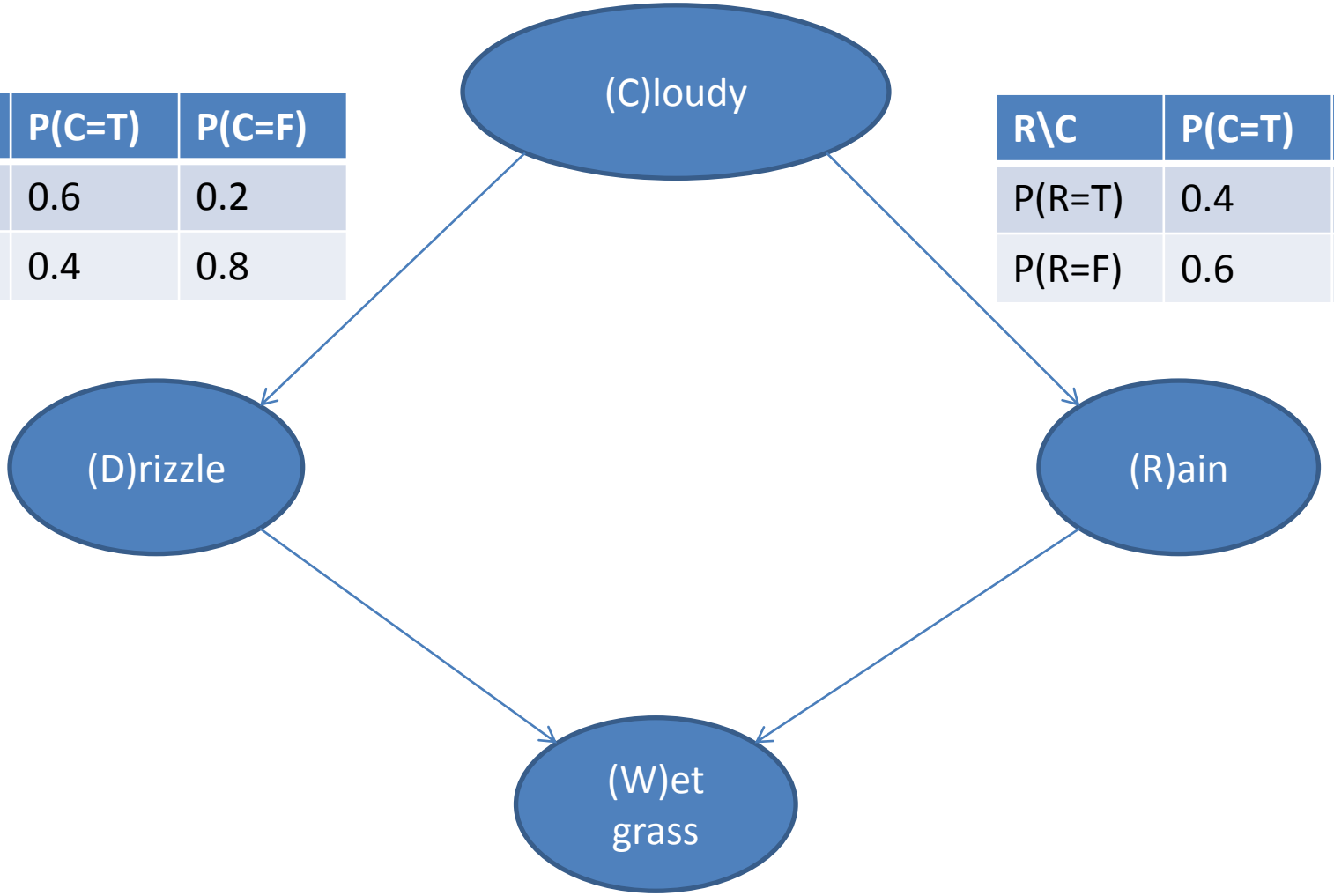
Wind	Precipitation	Minus temperature	Clouds
S	Snow	True	True
W	None	True	True
W	None	False	False
E	Rain	False	True
S	None	False	False
W	Drizzle	True	True

Bayesian networks - definition

- Bayesian network:
  - Acyclic directed graph
  - Vertices (nodes) represent attributes
  - Edges (arcs) represent dependencies between attributes
  - Represents joint distribution over data – this can be used to infer about some unknown attribute value

C,D,R,W ∈ {T, F}

P(C=T)	P(C=F)
0.4	0.6



D\C	P(C=T)	P(C=F)
P(D=T)	0.6	0.2
P(D=F)	0.4	0.8

R\C	P(C=T)	P(C=F)
P(R=T)	0.4	0.1
P(R=F)	0.6	0.9

T\D,R	P(D=T,R=T)	P(D=T,R=F)	P(D=F,R=T)	P(D=F,R=F)
P(W=T)	0.9	0.7	0.8	0.1
P(W=F)	0.1	0.3	0.2	0.9

Conditional independence

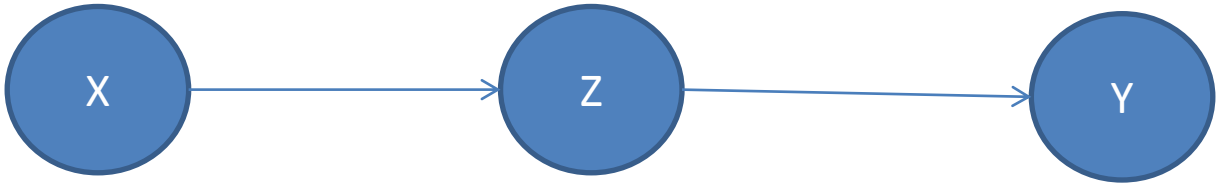
- Topic very useful when explaining structure building of Bayesian network
- Independence given a observation of some other variables: $Ind(X; Y|Z)$
- Knowledge about X gives us no additional information about Y, if we know Z

C.I. and Bayesian Networks

- Types of connections and their characterization:
  - Chain
  - Common cause
  - Common effect

Chain

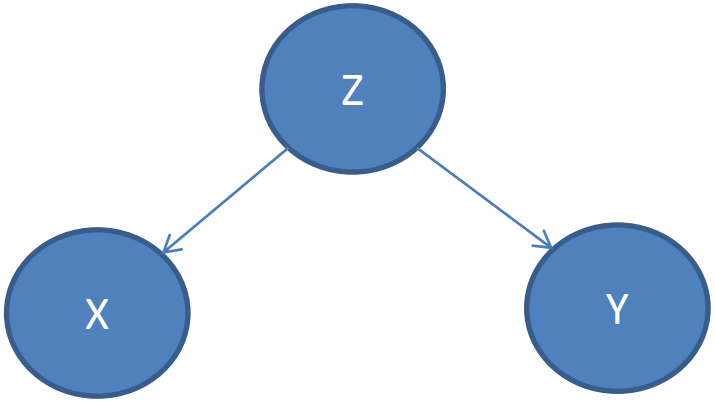
- X – power is offline
- Z – fridge is not freezing
- Y – food will go bad



- Ind(X;Y|Z)

Common cause

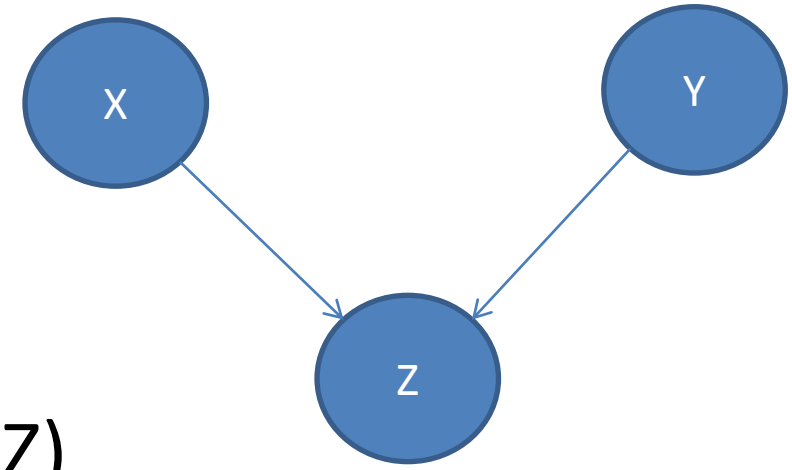
- X – it's snowing
- Z – minus temperature
- Y – ice is present



- Ind(X;Y|Z)

Common effect

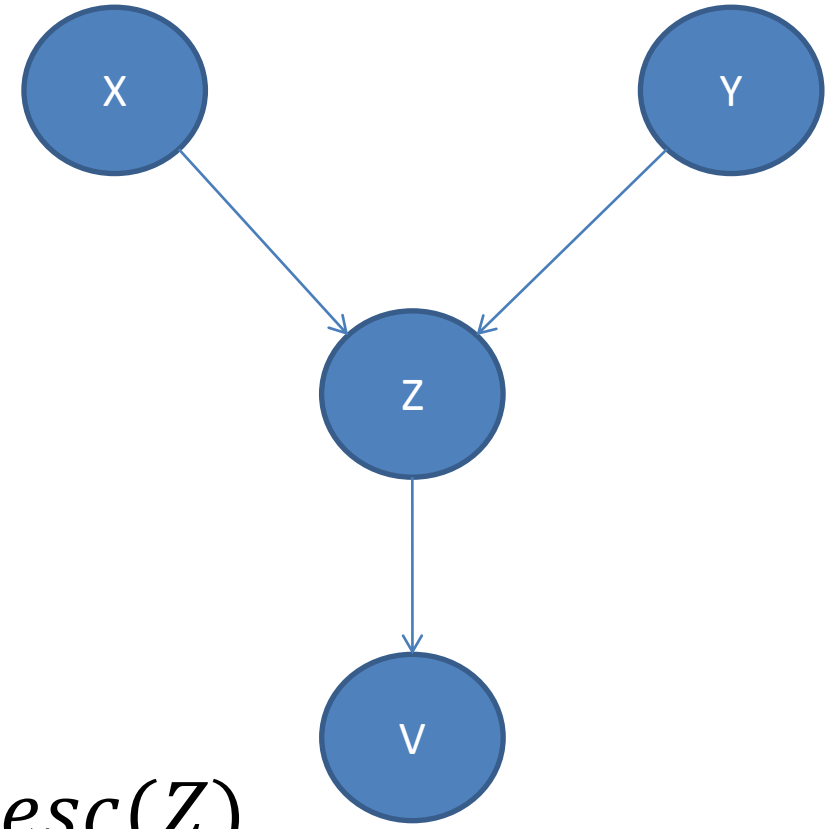
- X – electricity failure
- Z – car isn't starting
- Y – engine failure



- $\neg Ind(X;Y|Z)$

Common effect

- X – electricity failure
- Z – car isn't starting
- Y – engine failure
- V – go to car service

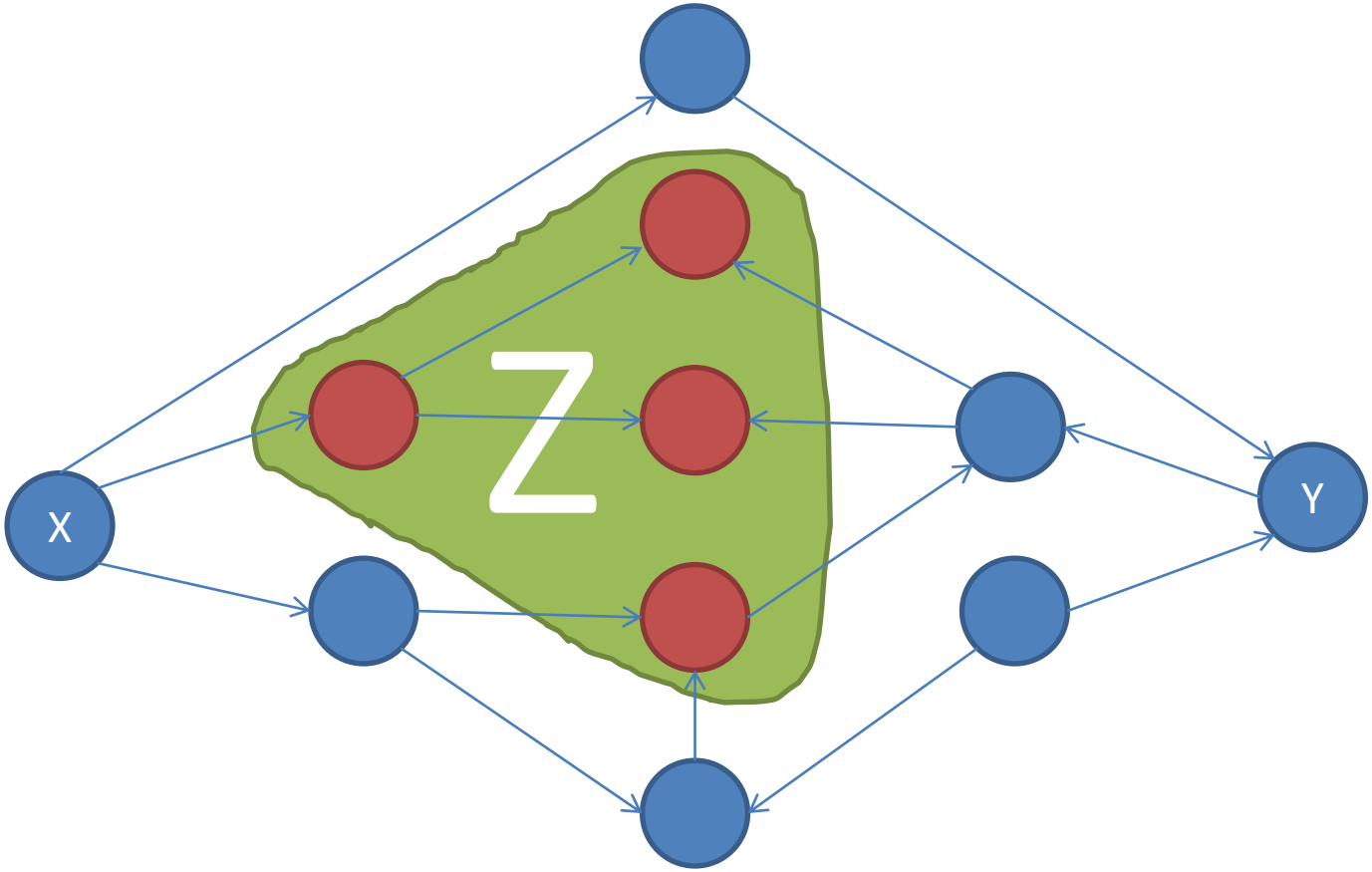


- Also $\neg Ind(\mathrm{X};\mathrm{Y}|\mathrm{V})$, $V \in Desc(Z)$

D-separation

- Common effect - blocking node
- Undirected path P between nodes X and Y is blocked given observation Z when at least one of the following conditions holds:
  - Exists non-blocking node on P that belongs to Z
  - Exists blocking node which and whose descendants do not belong to Z

D-separation



D-separation

- Nodes X and Y are d-separated given set of nodes Z, if and only if all paths between X and Y are blocked given Z

$$Dsep(X, Y|Z)$$

- d – separation in Bayesian network is the same as conditional independence in joint probability distribution
- d – separation is useful when explaining algorithms used for networks structure building

C.I. test (1/3)

- Measure of information shows how ‚important' given information is. Less probable information are more important
- Information entropy is an expected value of measure of information over all possible values of random variable
- Information entropy is a function from information theory that gives a level of uncertainty of our knowledge of random variable value

$$H(X) = \sum_{i=1}^{n} P(x_i) * \log \frac{1}{P(x_i)}$$

C.I. test (2/3)

- Cross entropy is a function that gives us the level of uncertainty of variable X when we now Y, or uncertainty of Y when we know X

$$CE(X,Y) = \sum_{i=1,j=1}^{n} P(x_i, y_j) * \log \frac{P(x_i, y_j)}{P(x_i) * P(y_j)}$$

- Bigger CE → more dependent variables

C.I. test (3/3)

- Conditional cross entropy is a function that gives us the level of uncertainty of variable X when we know Y, and uncertainty of Y when we know X, but given the observation of Z

$$CE(X,Y|Z) = \sum_{i=1,j=1,k=1}^{n} P(x_i, y_j|z_k) * \log \frac{P(x_i, y_j|z_k)}{P(x_i|z_k) * P(y_j|z_k)}$$

- Threshold must be given to distinguish dependent and independent variables.

Structure building

- Finding directed dependencies between attributes
- Two approaches:
  - Constraint-based
  - Score-based

Constraint based structure learning: SGS Algorithm

- Theorem 1:

Nodes X and Y in BN are connected if there is no subset of nodes Z that d-separates X and Y given Z

- Theorem 2:

Nodes X,Z,Y are connected in a common effect type connection if there exists no subset of nodes that contains Z and d-separates X and Y.

- Algorithm assumes that we can provide knowledge about d-separation (or conditional independence)

Parameter learning

$$\theta_{ijk} = \frac{N_{ijk}+1}{N_{ij}+r_i}$$

R\C	P(C=T)	P(C=F)
P(R=T)	0.4	0.1
P(R=F)	0.6	0.9

Where:

- θ_{ijk} - distribution parameter for node value k of node X_i and values j of $Pa(i)$
- r_i - number of possible values of X_i in data
- N_{ijk} - number of cases in data where X_i has value k and $\mathrm{Pa(i)}$ have values j
- $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ - number of cases in data where $\mathrm{Pa}(i)$ have values j

Inference

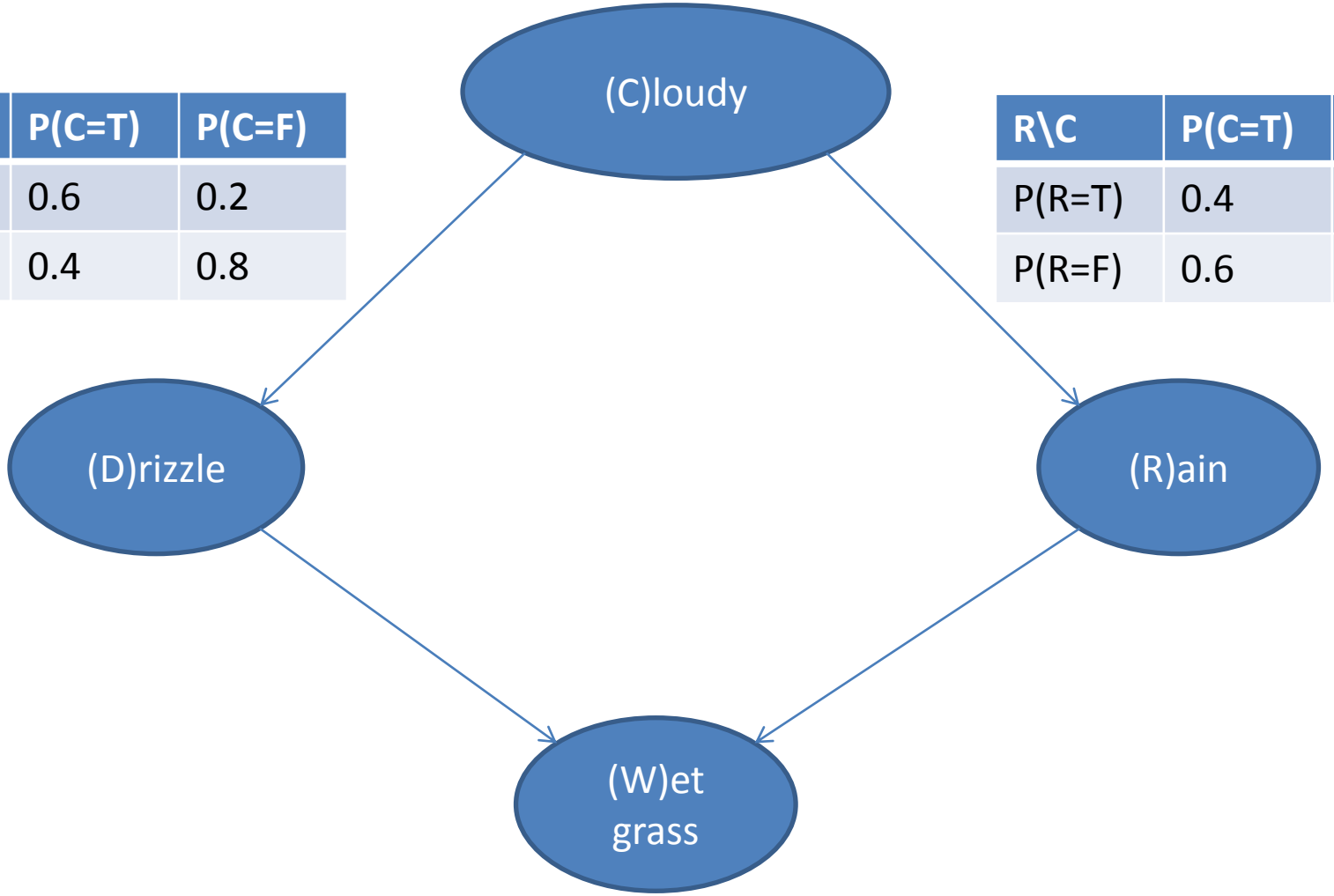
$$P(X_k, \dots, X_m | X_1, \dots, X_{k-1}, X_{m+1}, \dots, X_n) = \frac{P(X_1,\dots,X_n)}{P(X_1,\dots,X_{k-1},X_{m+1},\dots,X_n)}$$

$$P(x_1, \dots, x_n) = \prod_i P(x_i | Pa(i))$$

- Sometimes denominator can be omitted
- Numerator:
  - for complete data: simple multiplication
  - For incomplete: sum of $\prod_{k \in U} r_k$ values, where U – unobserved nodes (to cover all missing values combinations)

C,D,R,W ∈ {T, F}

P(C=T)	P(C=F)
0.4	0.6



D\C	P(C=T)	P(C=F)
P(D=T)	0.6	0.2
P(D=F)	0.4	0.8

R\C	P(C=T)	P(C=F)
P(R=T)	0.4	0.1
P(R=F)	0.6	0.9

T\D,R	P(D=T,R=T)	P(D=T,R=F)	P(D=F,R=T)	P(D=F,R=F)
P(W=T)	0.9	0.7	0.8	0.1
P(W=F)	0.1	0.3	0.2	0.9

Exact inference: example

$$P(\boldsymbol{W} = \boldsymbol{T} |\, D = T, R = T) = \frac{P(D=T,R=T,W=T)}{P(D=T,R=T)}$$

$$P(D = T, R = T, W = T) = P(\boldsymbol{C} = \boldsymbol{T}, D = T, R = T, W = T) + P(\boldsymbol{C} = \boldsymbol{F}, D = T, R = T, W = T)$$

$$= \boldsymbol{P}(\boldsymbol{W} = \boldsymbol{T} | \boldsymbol{D} = \boldsymbol{T}, \boldsymbol{R} = \boldsymbol{T}) * P(D = T|C = T) * P(R = T|C = T) * P(C = T)$$
$$+\boldsymbol{P}(\boldsymbol{W} = \boldsymbol{T} | \boldsymbol{D} = \boldsymbol{T}, \boldsymbol{R} = \boldsymbol{T}) * P(D = T|C = F) * P(R = T|C = F) * P(C = F)$$

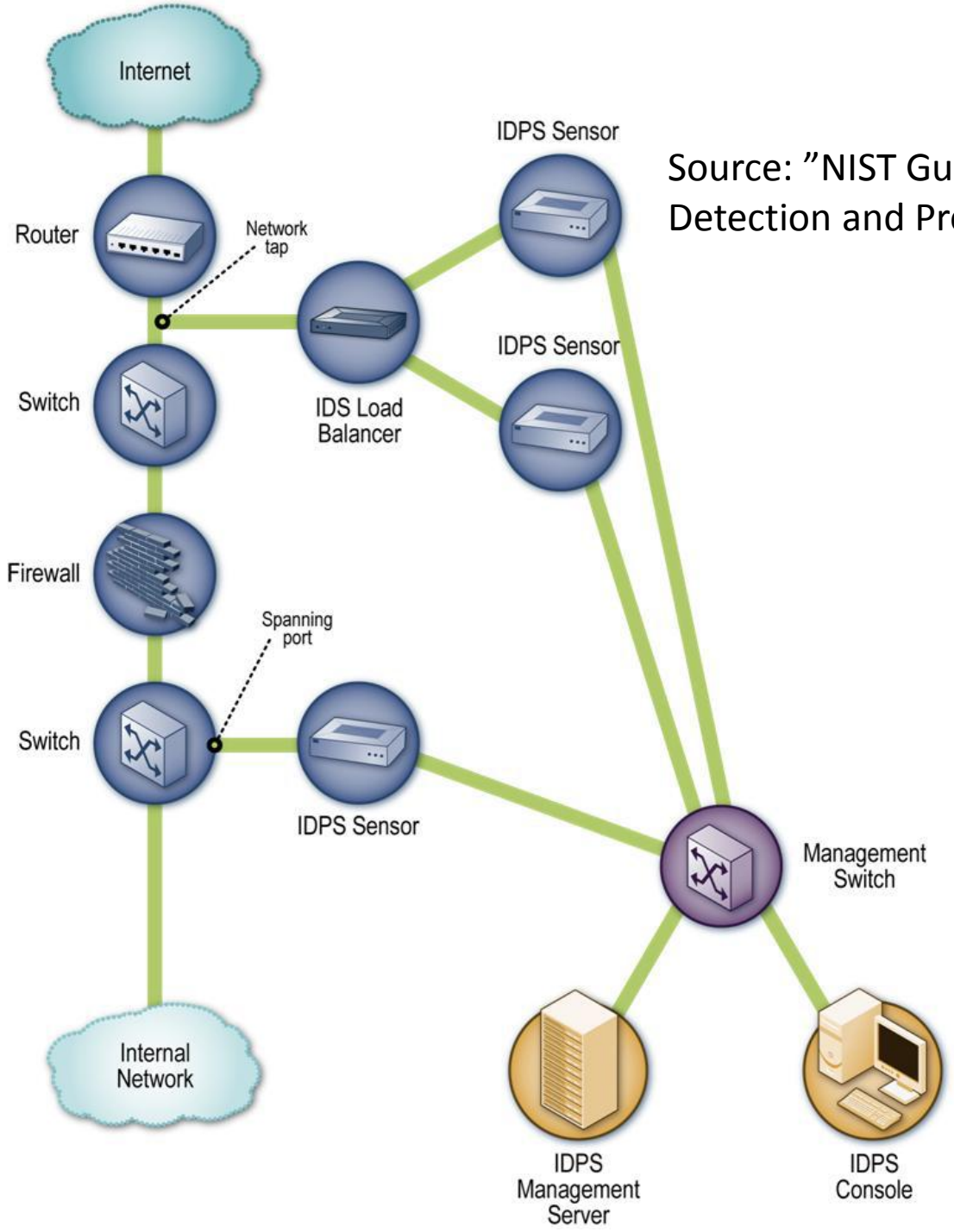
$$P(\boldsymbol{W} = \boldsymbol{F} |\, D = T, R = T) = \frac{P(D=T,R=T,W=F)}{P(D=T,R=T)}$$

$$P(D = T, R = T, W = F) = P(\boldsymbol{C} = \boldsymbol{T}, D = T, R = T, W = F) + P(\boldsymbol{C} = \boldsymbol{F}, D = T, R = T, W = F)$$

$$= \boldsymbol{P}(\boldsymbol{W} = \boldsymbol{F} | \boldsymbol{D} = \boldsymbol{T}, \boldsymbol{R} = \boldsymbol{T}) * P(D = T|C = T) * P(R = T|C = T) * P(C = T)$$
$$+\boldsymbol{P}(\boldsymbol{W} = \boldsymbol{F} | \boldsymbol{D} = \boldsymbol{T}, \boldsymbol{R} = \boldsymbol{T}) * P(D = T|C = F) * P(R = T|C = F) * P(C = F)$$

Intrusion detection systems

- Goal: detection of malicious computer systems usage
- Data: network traffic
- Main two types:
  - Rule based misuse detectors – database of rules – traffic is compared to the rules. This category of systems has very low rate of false positives (normal traffic marked as an attack), but is not immune to new types of attacks (fixed rules usually do not generalize)
  - Anomaly detectors and heuristic IDS – no database of rules – system learns from traffic and then compares new traffic to the trained one. If something is unusual or similar to trained malicious activity, systems makes an alert. This category of IDS has a better generalizing ability but also often generates many false positives



Source: ”NIST Guide to Intrusion Detection and Prevention Systems”

Intrusion detection systems

- Big problem with constructing IDS: testing
  - Database with representative traffic – very small number of old databases
  - Simulated attacks – could provide newer types of attacks, but will not necessarily be representative
  - Recorded traffic from a real network – more representative traffic but not necessarily to many kinds of attacks and problem with gathering (and processing) very large amount of data

Intrusion detection systems

- Most of existing network IDS systems perform a static (not connected with time relations) analysis of network traffic
- Some kinds of attacks can be only detected when probing is performed from not less than few points of the network – IDS should be distributed

Intrusion detection systems – applications of Bayesian networks

- BN have good generalization abilities and thus can be used as a heuristic model for traffic type anomaly based classification
- BN are „readable” for humans – not like in some other models (e.g. neural networks) they can easily interpreted in the terms of what is a symptom of attack, especially they can be adjusted „by hand” by a human expert

Intrusion detection systems – applications of Bayesian networks

- Time relations can be modeled by an extension of the idea of Bayesian networks called dynamic Bayesian networks and their modifications
- Bayesian networks can be easily constructed in a hierarchy, for example, where decision nodes of one layer of networks are observed random variables of another network. This can naturally model system distribution or could connect making decisions based on disjoint categories of traffic attributes

Summary

- Bayesian networks are a good and elastic way of representing the data
- BN have many possible extensions (like hierarchical structure or time relation modeling)
- BN can give a tool for composing an IDS system that improves present solutions

Thank you