- Example: We can define in OWL a class C as the intersection of the classes *Person* and *that of all whose children are doctors or have a child who is a doctor*
- In predicate logic:

$$\forall x(C(x) \equiv (Person(x) \wedge \forall y(HasChild(x,y) \rightarrow (Doctor(y) \vee \exists z(HasChild(y,z) \wedge Doctor(z))))$$

- In DL: (just for the gist, do not worry)

$$Person \sqcap \forall HasChild.(Doctor \sqcup \underbrace{\exists \underbrace{HasChild}_{role}.\underbrace{Doctor}_{concept}}_{concept})$$

- in OWL: (idem)

```
<owl:Class>
   <owl:intersectionOf rdf:parseType="collection">
          <owl:Class rdf:about="#Person"/>
          <owl:Restriction>
               <owl:onProperty rdf:resource="#hasChild"/>
               <owl:toClass>
                  <owl:unionOf rdf:parseType="collection">
                     <owl:Class rdf:about="#Doctor"/>
                     <owl:Restriction>
                          <owl:onProperty rdf:resource="#hasChild"/>
                          <owl:hasClass rdf:resource="#Doctor"/>
                     </owl:Restriction>
                  </owl:unionOf>
               </owl:toClass>
          </owl:Restriction>
   </owl:intersectionOf>
</owl:Class>
```

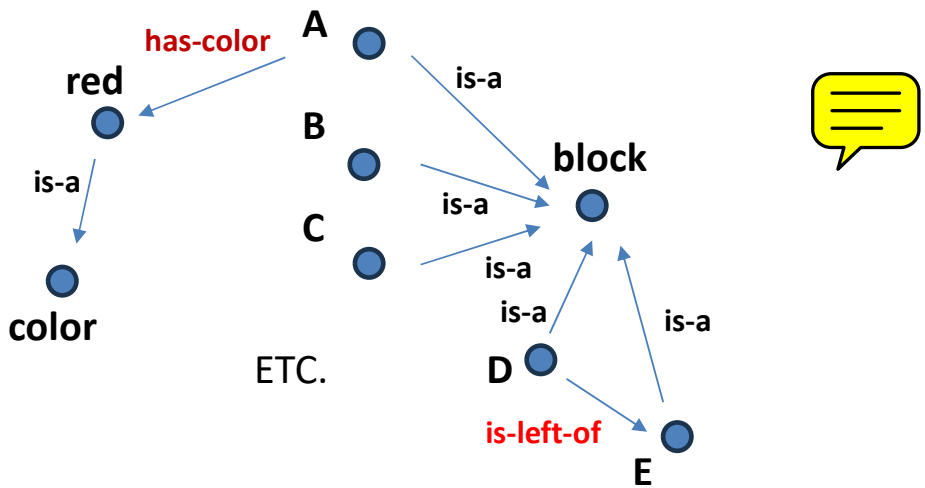
- In OWL it is possible to express axioms, i.e. general statements about the classes (top table) and properties (bottom table) being used:

OWL	Example in DL
subClassOf	$Human \sqsubseteq Animal \sqcap Biped$
equivalentClass	$Man \equiv Human \sqcap Male$
disjointWith	$Male \sqsubseteq \neg Female$
sameIndividualAs	$\{President\_Bush\} \equiv \{G\_W\_Bush\}$
differentFrom	$\{john\} \sqsubseteq \neg\{peter\}$
subPropertyOf	$HasDaughter \sqsubseteq HasChild$
equivalentProperty	$Cost \equiv Price$
inverseOf	$HasChild \equiv HasParent^{-}$
transitiveProperty	$Ancestor^{\star} \sqsubseteq Ancestor$
functionalProperty	$\top \sqsubseteq \leq 1\ HasMother$
inverseFunctionalProperty	$\top \sqsubseteq \leq 1\ HasSSN^{-}$

- The entries on the LHS have a fixed semantics, which is useful

  No need to define their meaning

- The big new thing today: Knowledge Graphs
- Many applications in business
- The old Blocks World:



- Easily stored, processed and queried in a computer
- One can add rules, e.g. $LeftOf(x, y) \wedge LeftOf(y, z) \rightarrow LeftOf(x, z)$
  Transitivity of *LeftOf*
- Here all of the above fits in ...

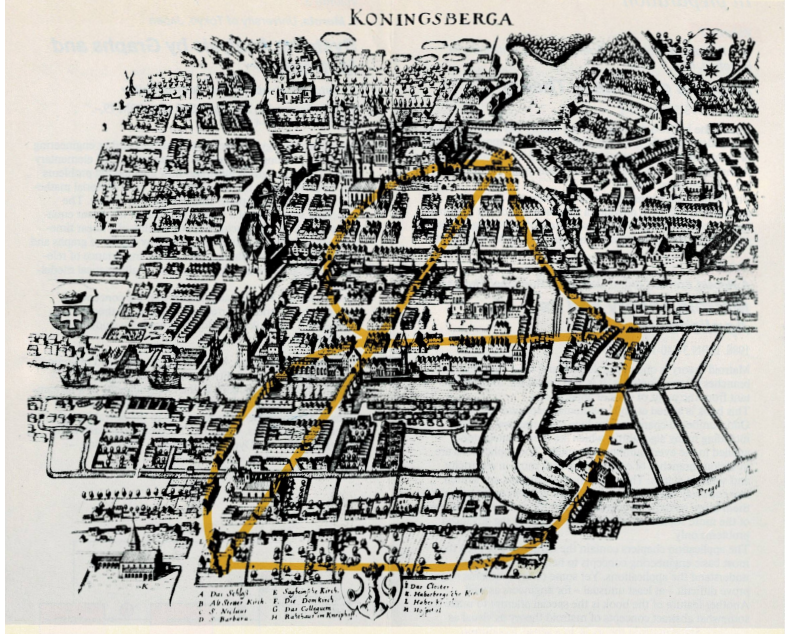
EXTRA SLIDES

Knowledge Graphs

- The knowledge representation languages we saw (and others) allow to create a Knowledge Base

  Think of an extension of a database (relational or not) with "knowledge"
- Many years of research in AI and Data Management, going through ontologies, semantic web, RDF databases, etc., convinced people that a natural way to represent data and knowledge is through graphs
- Well known and researched in Mathematics and Computer Science since

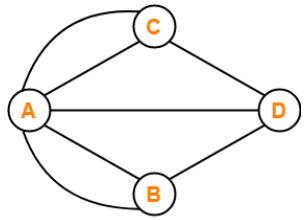
  Leonhard Euler
  (1707-1783)



Leonhard Euler (1707–1783) found that it is impossible to cross all seven bridges over the river Pregel in Königsberg without crossing at least one twice. This seems to have been the starting point of graph theory.

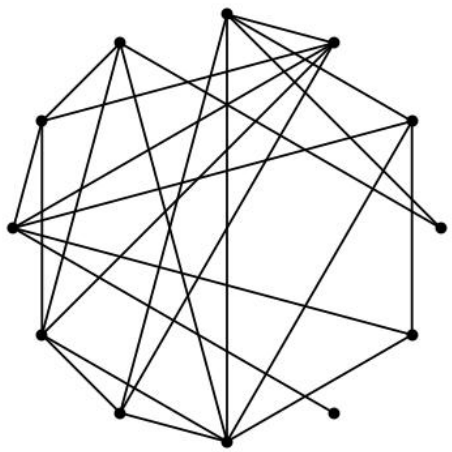
- Graphs are a natural and very useful mathematical abstraction

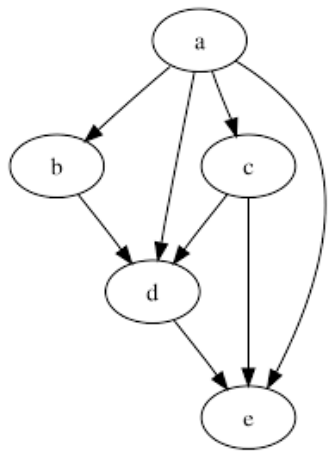
- With multiple applications in multiple disciplines
- We saw already Bayesian Networks as acyclic, directed graphs; and RDF-S as directed graphs



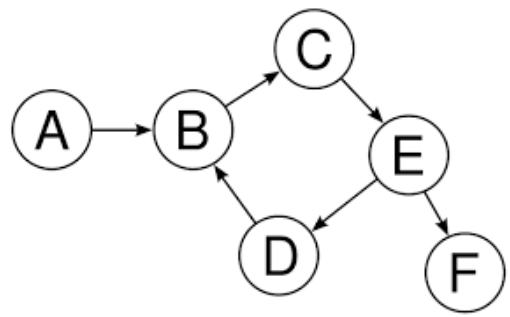
Euler's Graph

Markov Networks are probabilistic graphical models that use undirected graphs


undirected



directed acyclic

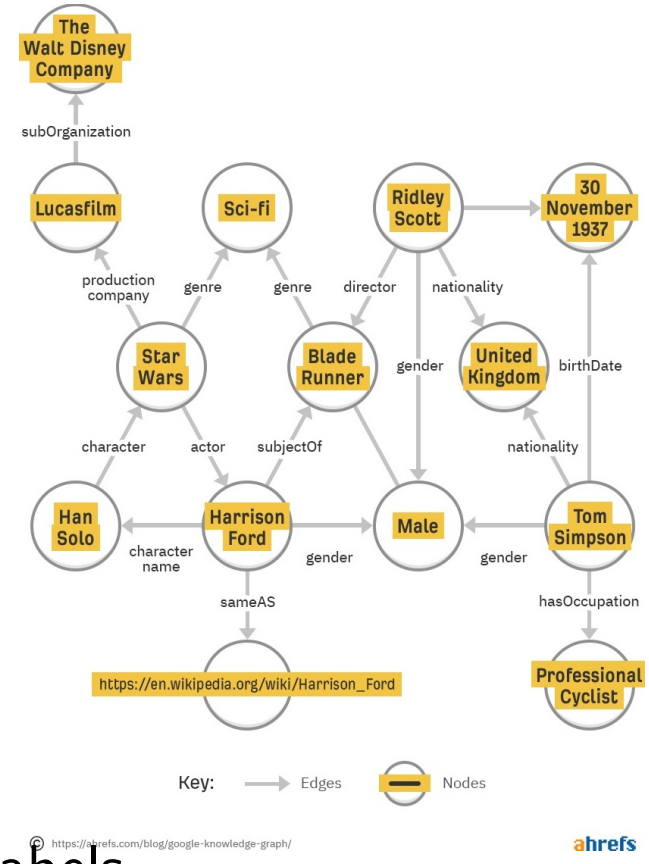


directed cyclic

- One distinguished between the nodes (vertices) and the edges

Labels at nodes and edges are not essential to graphs, but useful in many applications

- Knowledge Graphs are graphs
- They gained popularity through Google's KG-based representation of web data on which it performs searches
- Representation of data and knowledge
- KGs can be queried, navigated, analyzed, reasoned about ...
- A form of "linked data"



- Nodes have names, and edges have labels
- Some nodes correspond to "web resources" (bottom here)
- A huge "graph database" can be created

  Commonly in the form "triple stores" (recall RDF)

  Of "semi-structured" data (as opposed to structured as in relational DBs)

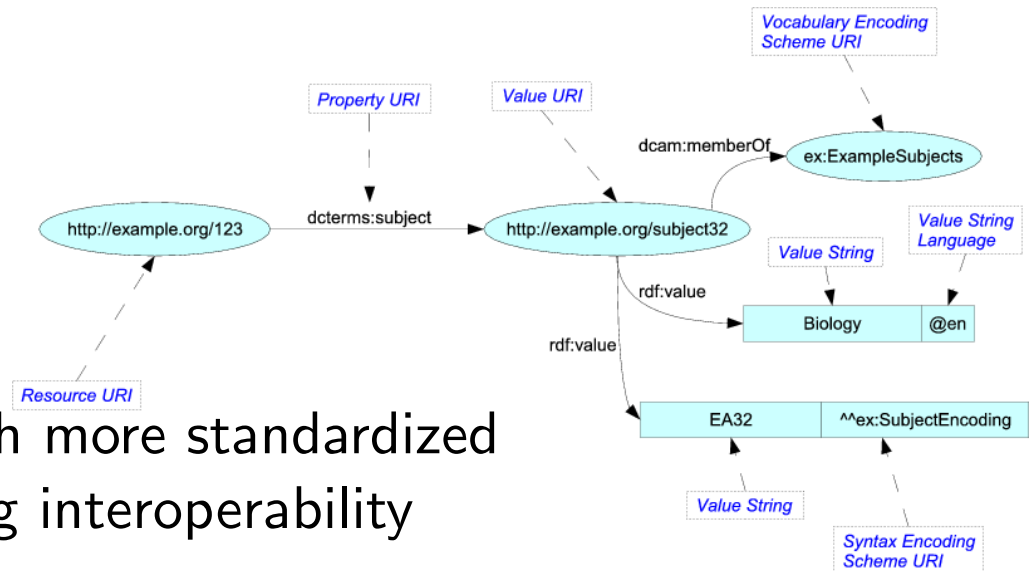
Revisiting RDF

- In many aspects, KGs are more general than RDF-Graphs

  BTW: RDF stands for "resource description framework"

  That is, metadata for web resources ...

URI stands for "Uniform Resource Identifier"



- RDF-Graphs are much more standardized and aimed at enabling interoperability

  So that different (usually web-based) data repositories can communicate with each other, and integrate data

- There are standardized query languages for RDF-S: SPARQL

Taken from:
https://www.wisecube.ai/blog/knowledge-graphs-rdf-or-property-graphs-which-one-should-you-pick/

What Can We Do with a KG?

- We can represent data and knowledge

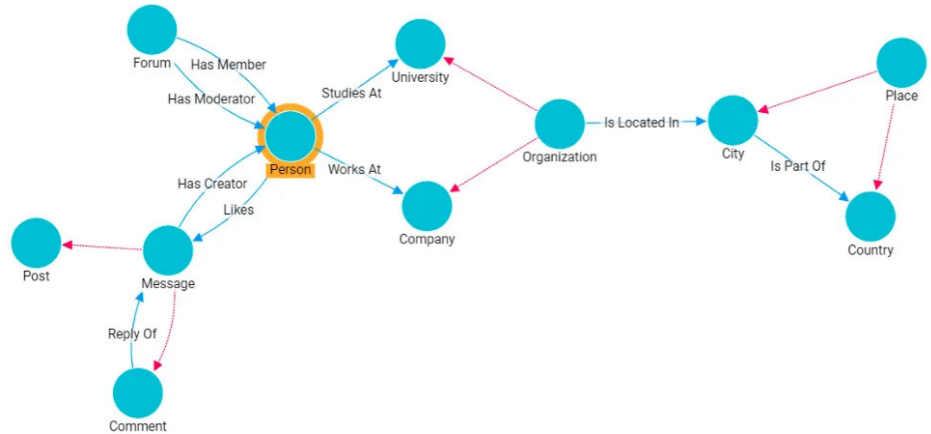
  Particulary if we extend the KG with an ontology (or consider the latter as an integral part of the KG)
- The absence of a rigid structure (as that provided by a relational DB schema) allows for more flexible representations

  Remember the fixed DB schema of Chapter 2:
  $Supply(\cdot, \cdot, \cdot)$, $Articles(\cdot, \cdot)$

  One can postpone thinking about an initial schema
- Relatively simple to updating the KG: locally add/remove nodes or edges
- The KG as a DB is very close to the data- and conceptual model (basically the same)

  Remember the greater difference between the ER model and the associated relational DB

- A social network can be modeled as a KG



  There are centrality measures that identify most relevant nodes

  Clustering and summarization (collapsing graph parts)

  Graph analytics to gain insight into application domain

- Easily represent dimensions and hierarchies through paths (remember multidimensional DBs)
- Aggregation supported by path navigation

  For data analytics

- Pattern and navigation-based query languages

  Symbolic and machine processable

  To retrieve subgraphs as query answers

  To find entities connected by paths (possibly very long)

- Relational DBs (RDBs) are still used by far the most in companies

  They are increasingly being stored and accesses in "the cloud"

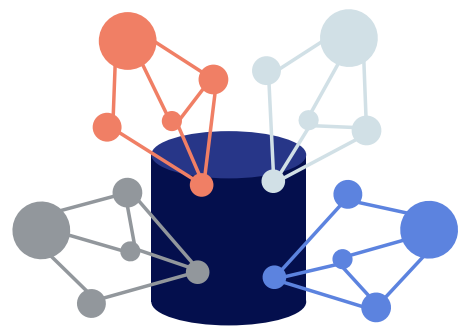
  Internet based and provided by large companies, e.g. Amazon, Google, Microsoft, etc.
- KGs are also increasingly gaining ground in all kinds of companies, and public offices

  They are and will become even more increasingly important
- KGs do not have to compete with RDBs, but cooperate

  RDBs may feed KGs with data

  Different, application dependent KGs can be defined on top (from) of a RDB
  KG data may be stored in RDBs

  Taking advantage of both worlds ...

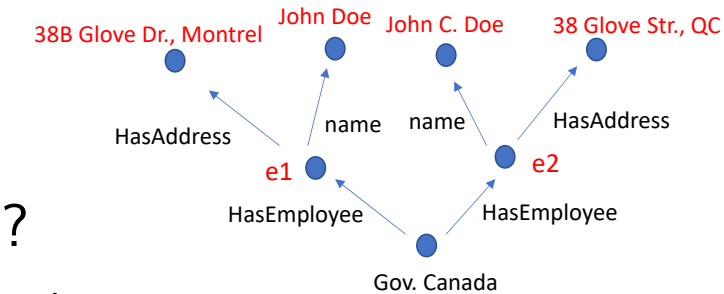


- A KG is by nature incomplete, i.e. contrary to the CWA of RDBs, there may be "true" knowledge not explicitly represented in KG

  E.g. missing triples, missing labels, missing links, etc., we may add

- There may be uncertainty about whether two entities are the same (but explicitly represented differently)

  The old problem of entity resolution:
  a same external entity
  represented in multiple ways

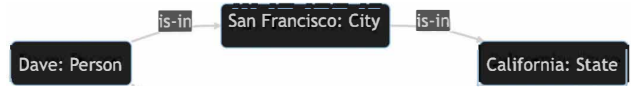


- How we deal with those problems?
  - Deductive reasoning (logic-based), specially with a rich ontology (as in previous chapter)
  - ML-based "reasoning", i.e. learning from data and inferring new data (or disambiguating existing data)

    A.k.a. "inductive" reasoning

Some Deductive Reasoning

- Consider an entity diagram that may be part of a KG
  (KG is also a property graph, with properties at the nodes)

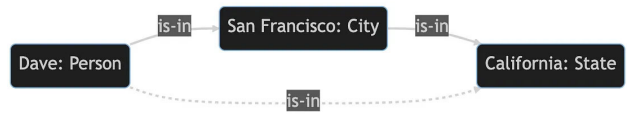


  It can be easily transformed into one of those we have seen so far

- With this only, we cannot deduce that Dave is also in California
- We need the "is-in" relation to be transitive
- Something like this:

  A Datalog-like rule

  (*) $(x)\mathit{is\text{-}in}(z) \leftarrow (x)\mathit{is\text{-}in}(y),\ (y)\mathit{is\text{-}in}(z)$



- With this rule in an ontology on top of the KG, we can deduce the new triple $(\mathit{Dave})\mathit{is\text{-}in}(\mathit{California})$
- Which we obtain with the query $\text{:-}\ (x?)\mathit{is\text{-}in}(y?)$
  Even without adding it to the KG

- We will not go into specifics of any query language in particular

  There are a few: SparQL, GraphQL, ... (being SQL the standard query language for RDBs)

  There are languages for creating graph documents, e.g. RDF-S, Json (for semistructured data), ...

  And popular systems, e.g. Neo4J, etc.

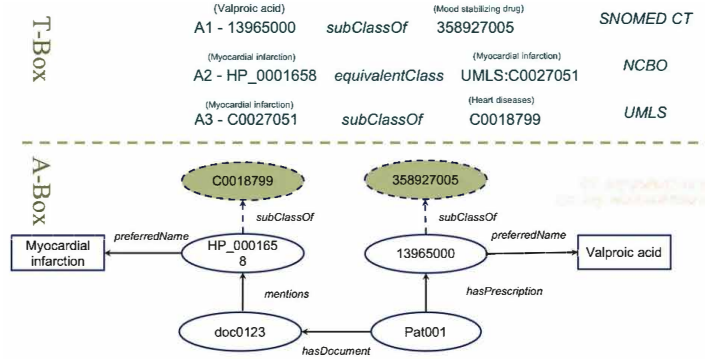
- Where should the ontology go?
- The ontology may be a KB interacting with the "data-graph"
- Ee can use any ontological languages, e.g. Datalog$^{\pm}$, DL, etc.

  But we would have to implement the interaction

- However, we can combine in the same KG the "data part" with the "ontological part"
- Recall that in RDF-S graphs we had data and also some basic ontological (or schema) "data"

- A standardized language such as OWL (basically extends RDF-S) can specify the transitivity of the relation

  As a part of the same KG



- Data and metadata are at the same level ...
- The "OWL engine" that manages this knowledge "understands" the meaning of "TransitiveObjectProperty"

  The engine has the definition "built-in" (that it means (*))
- Then, a "definition" of KG like this makes sense:

  " A knowledge graph is a schema-free, graph-based data structure/database, consisting of nodes, edges, and labels, together with a defining ontology

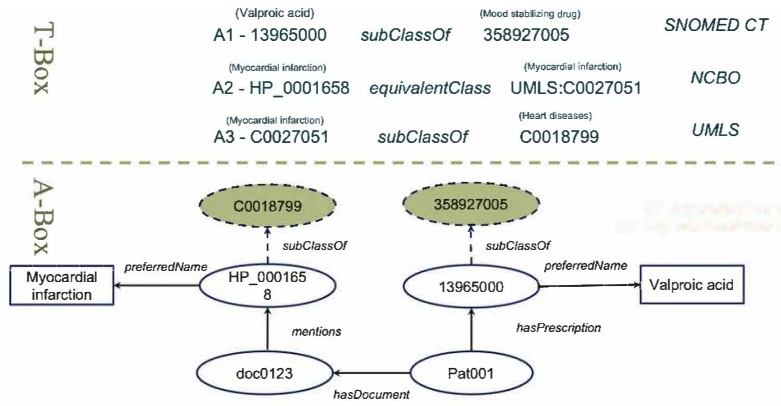
  The nodes define entities, the edges define relationships between those entities, and an ontology provides a flexible, formal definition of the entities to be found in the knowledge graph

  Ontologies themselves can have a graphlike structure, defining relationships between entity and property types (classes)"

- Often KGs are presented as "data + ontologies"

Ontology "terms" →

Data "assertions" →



← axioms from different integrated ontologies

Read:

https://towardsdatascience.com/derive-insights-from-health-data-using-knowledge-graph-technologies-b6cf2b742cd6

- The ontology part of a KG conveys *meaning about the data*, i.e. semantics

  This is why the area falls under what is called "semantic techniques"

- Data with semantics, specially when the latter is understandable by the user, become closer to the user

  Recall the discussion about having an ER model together with a RDB

Recommended Reading: Gartner Report on KGs

Read and watch an important player: (non-mandatory)

RAI docs and presentations

Chapter 5: Uncertain Knowledge and Probabilistic Graphical Models

Leopoldo Bertossi

Uncertain Knowledge

- So far we have been using classical symbolic logic (and variations of it) to represent knowledge
- Knowledge is true, false or undetermined (open, incomplete)

  But never uncertain in that it is only partially true or false

  No numerical degrees of truth ...
- For that we need different representations "languages" and models
- Probability is a natural choice
- There are many ways to represent probabilistic (stochastic) knowledge about variables and their stochastic dependencies

  Depending on what assumptions we make about them
- We will briefly present only one common model that is useful in many applications

We will assume basic knowledge about the notions of:[1]

- Random variable (RV)
- Probability, density function of a RV:

  $f_X(x) := P(X = x)$
- Conditional probability:

  $P(X = x|Y = y) := \frac{P(X=x,Y=y)}{P(Y=y)}$

  Equivalently:

  $P(X = x, Y = y) = P(X = x|Y = y) \times P(Y = y)$
- Joint distribution of several RVs:

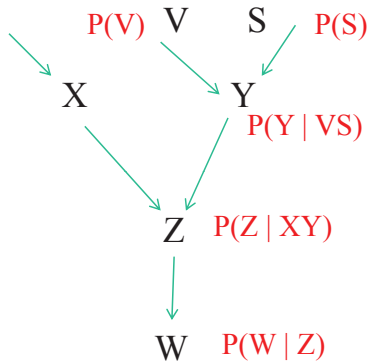
  $f_{XY}(x, y) := P(X = x, Y = y)$
- Marginal distributions:

  $P_Y(y) := P(Y = y) = \sum_x P(X = x, Y = y)$
- Independence of two RVs:

  $X \perp\!\!\!\perp Y \;:\Leftrightarrow\; P(X = x, Y = y) = P_X(x) \times P_Y(y)$, for all x, y

[1] Mandatory Reading on Probability: here

Bayesian Networks



- Random variables (RV) become nodes of a directed and acyclic graph

  The graph represents some known (assumed) dependencies and independencies among RVs

- Source nodes V, S have absolute distributions

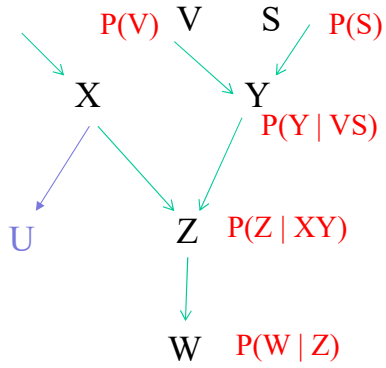
  The RVs at the other nodes, have conditional distributions

- The RVs are all defined on the same probability space, and they have a joint distribution $P(W, Z, X, Y, V, S)$

  The latter is not known (or explicitly given) and can be obtained from the BN (how?)

  Actually the distributions given with the BN are manifestations of it

- BNs are one class of so-called probabilistic graphical models



- Usually BNs have Bernoulli (or binary) RVs, i.e. they are propositional random variables that take values 0 or 1 only (think of "true" and "false")

  For example, Y has value 1 if the coin gives head, and 0 if tail
  S has value 1 if urn A is chosen, and 0 is urn B (from which the coin is taken)

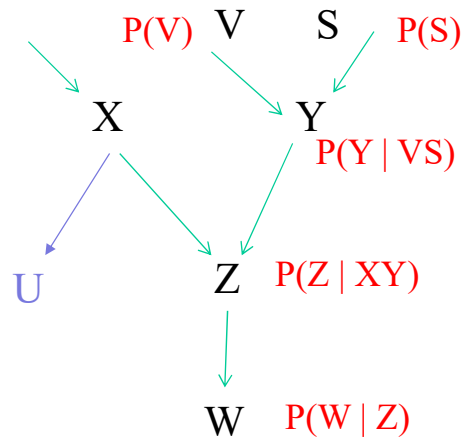
- The BN encodes (represents) an important assumption that is assumed to be true

  Markov Condition: Given its parents (i.e. values for them), a RV is independent of the non-descendant RVs

- For example:
  1. Z has X, Y as parents
  2. V is not a descendant of Z
  3. Then, $Z|\{X, Y\} \perp\!\!\!\perp V$

Similarly: $(U \perp\!\!\!\perp Z)|X$, $V \perp\!\!\!\perp S$

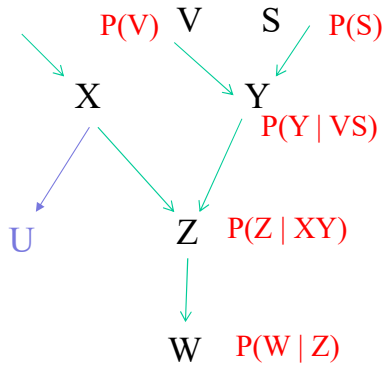
- More precisely: $Z|\{X = x, Y = y\} \perp\!\!\!\perp V$
- With the initial distributions plus the MC, the BN allows us to infer other distributions



  - The joint distribution: for $w, u, z, x, y, v, s \in \{0, 1\}$
    $P(W = w, U = u, Z = z, X = x, Y = y, V = v, S = s)$
  - Marginal distributions: $P(W = w)$, e.g. $P(W = 1) =?$
  - Joint marginal distributions:
    $P(W = w, Z = z, Y = y, V = v)$
  - Conditional distributions: $P(X = x|W = w)$

- They are expected to be expressed in terms of the initial distributions

- Example: $\underbrace{P(Y = y, V = v, S = s)}_{\text{unknown marginal joint distr.}} = ?$



$$P(Y = y, (V = v, S = s)) =$$

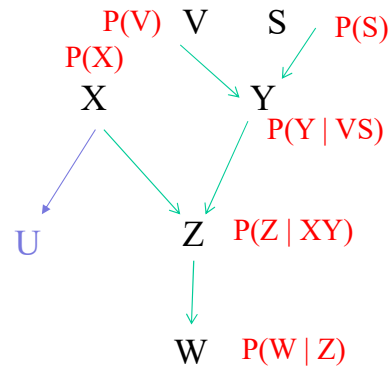
$$\underbrace{P(Y = y \mid V = v, S = s) \times P(V = v, S = s)}_{\text{def. of condic. prob.}} =$$

$$P(Y = y \mid V = v, S = s) \times \underbrace{P(V = v) \times P(S = s)}_{\text{independence}}$$

- We expressed the joint distribution in terms of given distributions
- This factorization makes sense: look at the BN

  Go upwards iteratively one level at a time towards the source nodes
- This is a form of probabilistic inference
- Exercise: $P(Z, X, Y, V, S) = ?$ $P(Z, Y, V, S) = ?$

- All you need are the:
  - The given conditional and absolute distributions
  - The definition of conditional probability
  - Markov conditions on independence



- Intuitively: Markov Condition tells us that for a given variable all we need to know are the immediately preceding variables in the BN, and not the whole history (i.e. trajectories in the graph leading to it)
- Actually, we have a general formula:

$$P(X_1, \ldots, X_n) = \Pi_{i=1}^{n} P(X_i \mid Par(X_i))$$

$Par(X)$ = parents of X

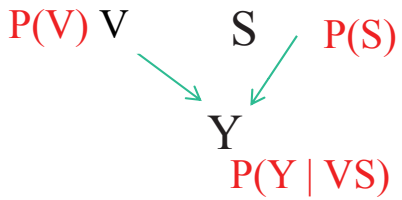
Special case for the sources: $P(X|\emptyset) := P(X)$

- In the example:

$$P(W, U, Z, X, Y, V, S) = P(W|Z)P(Z|XY)P(U|X)P(Y|VS)P(X)P(V)P(S)$$

Example: We have binary RVs:

(true/false or 1/0)



Y	V	S	$P(V)$	$P(S)$	$P(Y\|V,S)$
0	0	0	0.9	0.8	0.2
0	1	0	0.1	0.8	0.3
0	0	1	0.9	0.2	0.4
0	1	1	0.1	0.2	0.05
1	0	0	0.9	0.8	0.8
1	1	0	0.1	0.8	0.7
1	0	1	0.9	0.2	0.6
1	1	1	0.1	0.2	0.95

- Y : *"Client books at beach resort"*
  V : *"Client is top manager"*
  S : *"Client plays at the lottery"*
- Y depends on V, S, and we have the *joint distribution*:

$$P(Y = y, V = v, S = s) = P(Y = y|V = v, S = s) \times P(V = v) \times P(S = s)$$

(a) Compute the probability that *"Client simultaneously: books at a beach resort, is not a top manager, and plays at the lottery"*

We want: $P(Y = 1, S = 1, V = 0) = ?$

$$= P(Y = 1|V = 0, S = 1) \times P(V = 0) \times P(S = 1) = ?$$

$$= 0.6 \times 0.9 \times 0.2 = 0.108$$

(b) Compute the probability that *"Client does not book at the beach resort"*

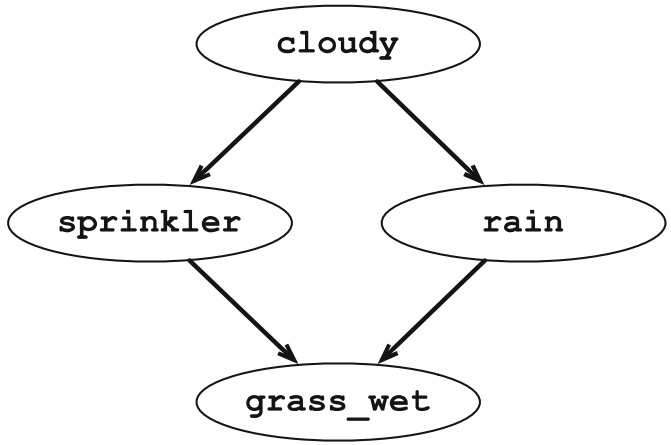
We want: $P(Y = 0) = ?$ (the marginal distribution)

$$= \sum_{v,s} P(Y = 0, V = v, S = s) =$$

$$P(Y = 0, V = 0, S = 0) + P(Y = 0, V = 0, S = 1) + P(Y = 0, V = 1, S = 0) + P(Y = 0, V = 1, S = 1)$$

Each of the terms is computed as in (a)

- Example: Boolean RVs C, S, R, G ('cloudy', 'sprinkler', etc.)



	P(C)
	0.5

C	P(S)
t	0.80
f	0.20

C	P(R)
t	0.10
f	0.50

S	R	P(G)
t	t	0.99
t	f	0.90
f	t	0.90
f	f	0.00

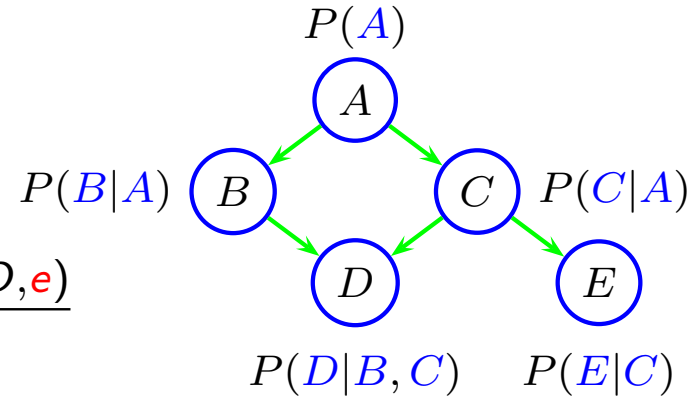
- Some implicit stochastic independencies:
  - $(G \perp\!\!\!\perp C)|\{S, R\}$ and $(S \perp\!\!\!\perp R)|C$ (*)

    C and G "separated" by S and R

- $(S \not\perp\!\!\!\perp R)|G$ (S and R not separated by G due to $\rightarrow\leftarrow$, c.f. later)

Inference: Variable Elimination

- Do we need $P(A, B, C, D, E)$ to calculate $P(A|c, e)$?
  (a posteriori distr. after evidence)



- Note: $P(A|c, e) = \frac{\sum_B \sum_D P(A,B,c,D,e)}{P(c,e)}$

  In general: $P(A|e) = \frac{\sum_{\mathcal{V}\setminus\{A\}} P(\mathcal{V},e)}{P(e)}$

$$\sum_B \sum_D P(A, B, c, D, e) = \sum_B \sum_D P(e|c)P(c|A)P(D|c, B)P(A)P(B|A)$$

$$= P(e|c)P(c|A)P(A) \sum_B \sum_D P(D|c, B)P(B|A)$$

$$= P(e|c)P(c|A)P(A) \sum_B P(B|A) \times \underbrace{\sum_D P(D|c, B)}_{\text{total sum 1 over cond. distr. of argument before |}}$$

$$= P(e|c)P(c|A)P(A)$$

- Instead of a table with 2^5 entries we only need 2 numbers!

Where From?

- The BN can be given directly
- It can also be "learned" from data (machine learning)
- This means learning the structure of the BN (the "arrows" capturing dependencies) and the distributions
- Sometimes the structure is known (assumed, given), and we have to learn the distributions (coming)
- The initial distributions (that come with the BN) can be estimated from relative frequencies
- Alternatively, and consistently with a Bayesian approach, they can be assigned subjectively, *a priori*

  Possibly updating them in the light of data (*a posteriori* distributions)

Deciding With Bayesian Networks

- Bayesian networks (BNs) can be used for classification
- The classification is read off from the value a particular, output random feature Y takes
- In general terms, as the most likely (probable) value given the input values for the other random features $X_1, \ldots, X_n$
- Under what distribution? We saw the BN defines a joint distribution $P(X_1, \ldots, X_n, Y)$

  From which $P(Y|X_1, \ldots, X_n)$ can be computed

  This holds for any network structure
- For a concrete input $\langle x_1, \ldots, x_n \rangle$ for $(X_1, \ldots, X_n)$, and possible values $y_1, \ldots, y_k$ for Y, compute and compare:

  $$P(y_1|X_1 = x_1, \cdots, X_n = x_n), \ldots, P(y_k|X_1 = x_1, \cdots, X_n = x_n)$$

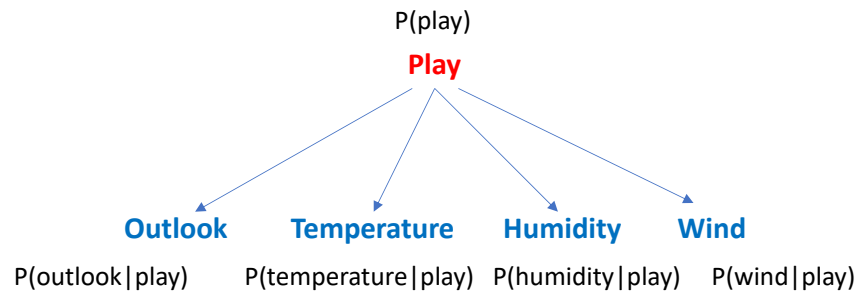
  The largest, $P(y^\star|X_1 = x_1, \cdots, X_n = x_n)$, gives the label: $y^\star$

- As we saw, building a BN can be complex, and the same applies to a BN classifier

  Unless we make simplifying (but still reasonable) assumptions

- The most simple, still useful, and sensible BN classifiers are "Naive Bayes Classifiers"

- Weather features depend on output/label/decision feature Play



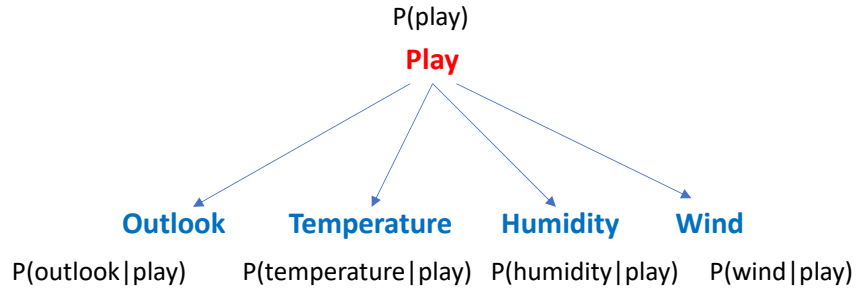
  Weather features are not assumed to be mutually independent

  They become independent given each particular value for Play

  A basic assumption about BNs in general

- ~~The distributions we need are exactly those we estimated early on (hence, those)~~

- The NBC
- We had:



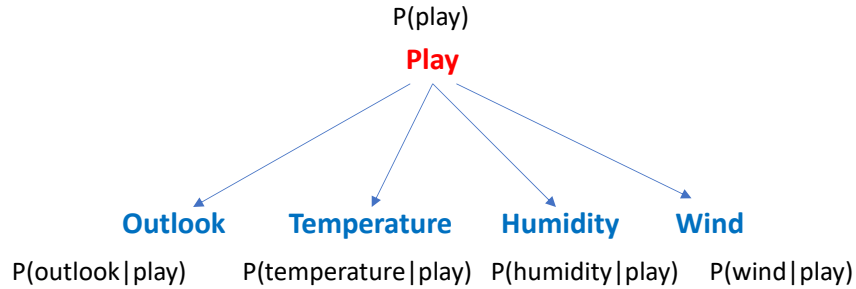
Outlook	Temperature	Humidity	Windy	Play
sunny	high	high	false	no
sunny	high	high	true	no
overcast	high	high	false	yes
rain	medium	high	false	yes
rain	low	normal	false	yes
rain	low	normal	true	no
overcast	low	normal	true	yes
sunny	medium	high	false	no
sunny	low	normal	false	yes
rain	medium	normal	false	yes
sunny	medium	normal	true	yes
overcast	medium	high	true	yes
overcast	high	normal	false	yes
rain	medium	high	true	no

$$P(play = yes) = \frac{9}{14} \quad P(play = no) = \frac{5}{14}$$

$$P(outlook = sunny|play = yes) = \frac{2}{9}$$
$$P(outlook = sunny|play = no) = \frac{3}{5}$$
$$P(outlook = overcast|play = yes) = \frac{3}{9}$$

$$P(outlook = overcast|play = no) = \frac{0}{5} = 0$$
$$P(outlook = rain|play = yes) = \frac{3}{9}$$
$$P(outlook = rain|play = no) = \frac{2}{5}$$
$$P(temp = high|play = yes) = \frac{2}{9}$$
$$P(temp = high|play = no) = \frac{2}{5}$$
$$P(temp = medium|play = yes) = \frac{4}{9}$$
$$P(temp = medium|play = no) = \frac{2}{5}$$
$$P(temp = low|play = yes) = \frac{4}{9}$$
$$P(temp = low|play = no) = \frac{1}{5}$$
$$P(humidity = high|play = yes) = \frac{3}{9}$$
$$P(humidity = high|play = no) = \frac{4}{5}$$
$$P(humidity = normal|play = yes) = \frac{6}{9}$$
$$P(humidity = normal|play = no) = \frac{1}{5}$$
$$P(windy = true|play = yes) = \frac{3}{9}$$
$$P(windy = true|play = no) = \frac{3}{5}$$
$$P(windy = false|play = yes) = \frac{6}{9}$$
$$P(windy = false|play = no) = \frac{2}{5}$$

- From the general formula for BNs:



$$P(\text{outlook, temperature, humidity, wind}|\text{play}) = P(\text{outlook}|\text{play}) \times \cdots \times P(\text{wind}|\text{play})$$

Conditional independence of weather features given the root

- To classify we need the conditional distribution of decision variable:

$$P(\text{play}|\text{outlook, temperature, humidity, wind})$$

- Given specific weather condition: $\langle \mathbf{o}, \mathbf{t}, \mathbf{h}, \mathbf{w} \rangle$

Decide according to which of these is greater:

$$P(\text{play} = \text{yes}|\mathbf{o}, \mathbf{t}, \mathbf{h}, \mathbf{w}) \quad \text{or} \quad P(\text{play} = \text{no}|\mathbf{o}, \mathbf{t}, \mathbf{h}, \mathbf{w})$$

- Deciding with weather entity

  $Outlook = rain, Temperature = high, Humidity = normal, Windy = false$

- Which of the following two is a maximum a posteriori?

  $P(Play = yes | Outlook = rain, Temp = high, Humidity = normal, Windy = false)$

  $P(Play = no | Outlook = rain, Temp = high, Humidity = normal, windy = false)$

- With a NBC, for any of these probabilities we have:

$$P(p|o,t,h,w) = \frac{\overbrace{P(p,o,t,h,w)}^{\text{big joint}}}{\underbrace{P(o,t,h,w)}_{\text{marginal}}} = \frac{P(o|p)P(t|p)P(h|p)P(w|p)P(p)}{\sum_{p\in\{\text{yes,no}\}} P(o|p)P(t|p)P(h|p)P(w|p)P(p)}$$

(cannot assume $P(O,T,H,W) = P(O)P(T)P(H)P(W)$)

- The denominator is the same, no need to compute it

- $P(play = yes | outlook = rain, temp = high, humidity = normal, windy = false)$

  $= P(outlook = rain | play = yes) \times P(temp = high | play = yes) \times P(humidity = normal | play = yes) \times$
  $P(windy = false | play = yes) \times P(play = yes) = \frac{3}{9}\frac{2}{9}\frac{6}{9}\frac{6}{9}\frac{9}{14} = \frac{4}{3\times 9\times 7}$

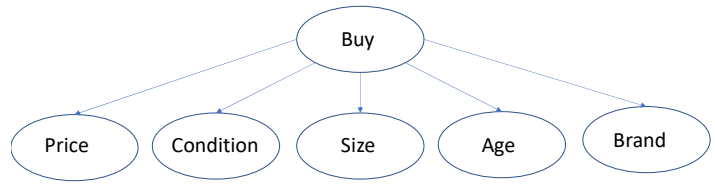
- $P(play = no | outlook = rain, temp = high, humidity = normal, windy = false) = \frac{4}{5^3\times 7}$

- The first is greater: We play!

Exercise: Compute:

$P(outlook = rain, temp = high, humidity = normal, windy = false | play = yes)$

Exercise: Compare and Disccuss



For this NBC, decision/class label variable *Buy* may be Bernoulli

Considered "the cause (hypothesis) for the effects", *Price*, etc.

Naive underlying assumptions:

(a) Class label L (root) does not depend on "attributes" $X_1, \ldots, X_n$ (leaves)

(b) By Markov property, attributes are independent given the label:

$$P(X_i X_j | L) = P(X_i | L) \times P(X_j | L), \quad i \neq j$$

The X_i "separated" by L

But what about the following non-Naive Bayesian classifier?