CS 295A/395D: Artificial Intelligence

Bayes Nets as Causal Graphs

Prof. Emma Tosch

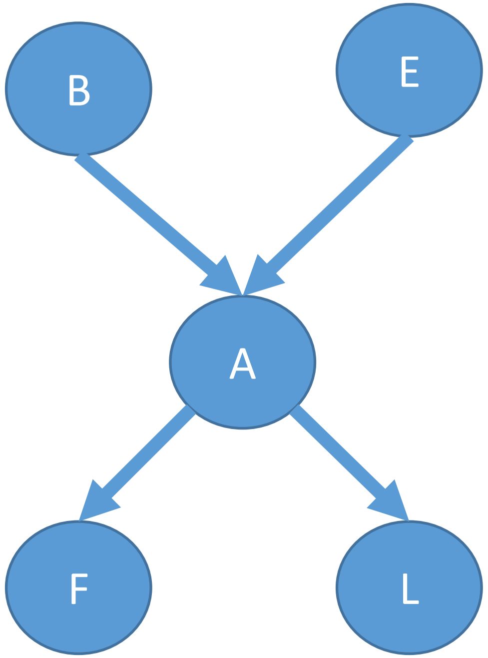
21 March 2022

The University of Vermont

Agenda

- Reminder: student hours today until Noon (Innovation E456)
- *Recap*: Bayes Nets & Independence
- *New*: Bayes Nets as *Causal Graphs*

Recap: Bayes Nets as factorization

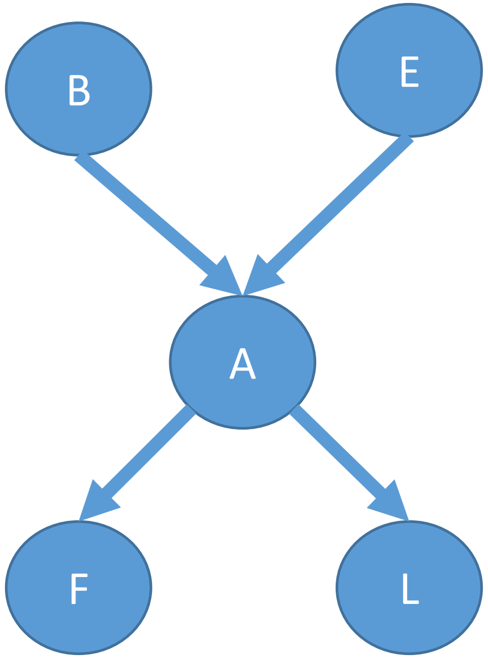


1. Reverse topologically order the nodes, e.g.
    1. F, L, A, B, E **or**
    2. L, F, A, B, E, etc.
2. Factorize joint distribution using graph semantics of $\mathcal{G} =< \mathcal{V}, \mathcal{E} >, \quad \mathcal{V} = \{V_1, \ldots Vn\}$:

$$P(V_1, \ldots Vn) = \prod P(V_i | Parents(V_i))$$

here, P(B, E, A, F, L) = P(F | A)P(L | A)P(A | B, E)P(B)P(E)

Recap: d-separation



Classical definition (Pearl):

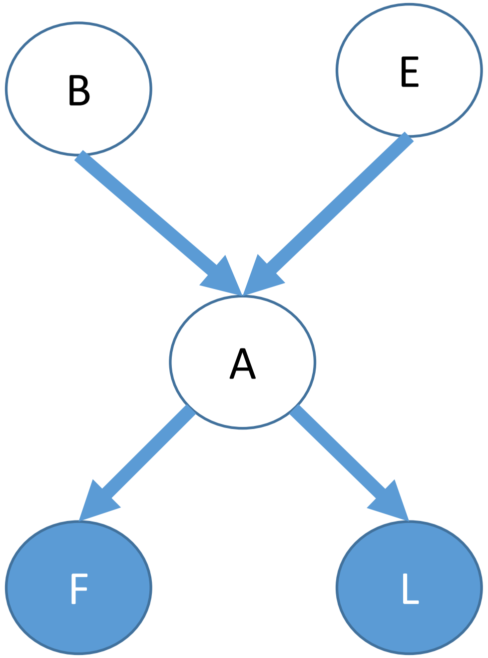
A set Z is said to d-separate X from Y iff Z blocks every path from a node in X to a node in Y.

A path p is blocked by Z iff:

1. p contains a chain i -> m -> j or a fork I <- m -> j such that m is in Z, or
2. p contains a collider i -> m <- j such that m is NOT in Z and *no descendant of m* is in Z.

Independence gives us useful, fast queries.

Recap: Partial Observability



We may need to reason about *latent* or *unobserved* nodes.

- Because they are unmeasured, we cannot reason about their *specific values*.

Depending on the task, we either:

1. *Marginalize* over them (inference).
2. Compute their *expected values* (decision making).

Both are forms of integration!

Bayes Nets beyond factorizations

Scenario: 3 variable, no independence relations

Example: height (H), weight (W), success (S)

Possible factorizations of $P(H, W, S)$:

$P(S \mid H, W)P(H \mid W)P(W)$

$P(H \mid S, W)P(S \mid W)P(W)$

$P(W \mid H, S)P(H \mid S)P(S)$

$P(S \mid H, W)P(W \mid H)P(H)$

$P(H \mid S, W)P(W \mid S)P(S)$

$P(W \mid H, S)P(S \mid H)P(H)$

All are equivalent factorizations (i.e., same probability distribution)

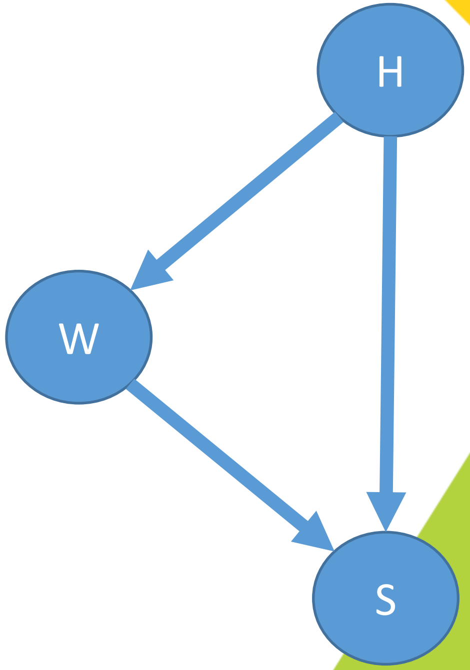
Bayes Nets beyond factorizations

Scenario: 3 variable, no inde

Example: height (H), wei

Possible factorizations of P

Purely algorithmic interpretation of Bayes Nets encoding factorizations.



P(S | H, W)P(H| W)P(W)

P(H | S, W)P(S | W)P(W)

P(W | H, S)P(H | S)P(S)

$\boxed{\text{P(S | H, W)P(W | H)P(H)}}$

P(H | S, W)P(W | S)P(S)

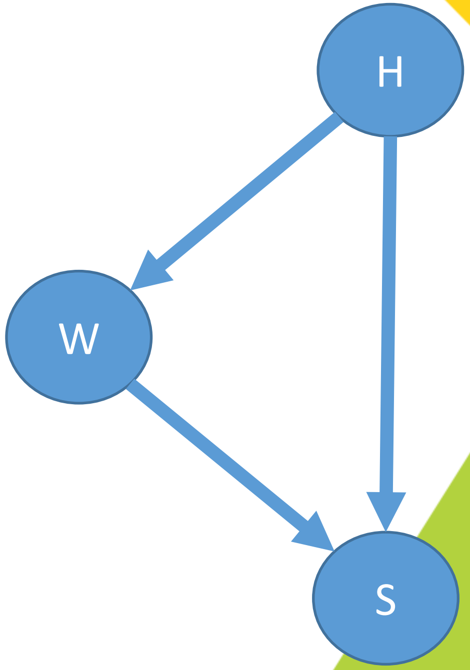
P(W | H, S)P(S | H)P(H)

Bayes Nets beyond factorizations

Scenario: 3 variable, no independence relations

Example: height (H), weight (W), success (S)

Possible factorizations of P(H, W, S):



P(S | H, W)P(H| W)P(W)	$\boxed{\text{P(S \| H, W)P(W \| H)P(H)}}$
P(H | S, W)P(S | W)P(W)	P(H | S, W)P(W | S)P(S)
P(W | H, S)P(H | S)P(S)	P(W | H, S)P(S | H)P(H)

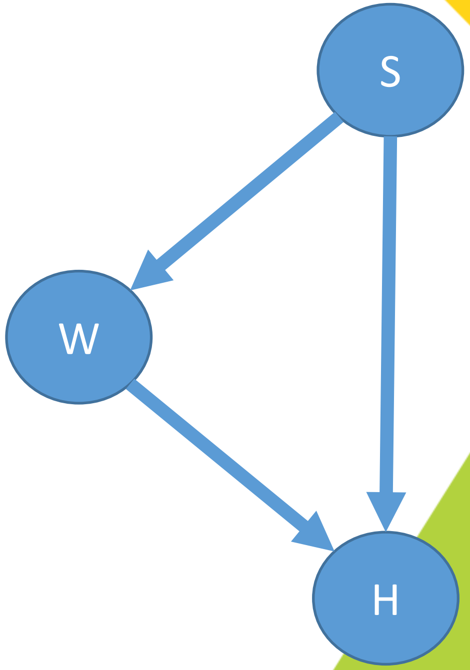
Qualitative difference?

Bayes Nets beyond factorizations

Scenario: 3 variable, no independence relations

Example: height (H), weight (W), success (S)

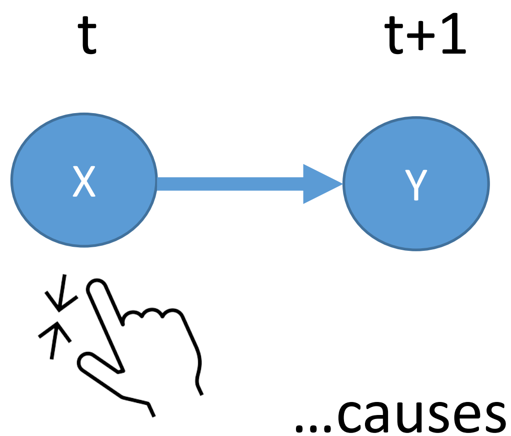
Possible factorizations of P(H, W, S):



$P(S \mid H, W)P(H \mid W)P(W)$	$P(S \mid H, W)P(W \mid H)P(H)$
$P(H \mid S, W)P(S \mid W)P(W)$	$\boxed{P(H \mid S, W)P(W \mid S)P(S)}$
$P(W \mid H, S)P(H \mid S)P(S)$	$P(W \mid H, S)P(S \mid H)P(H)$

Does this factorization seem different? Why?

Causal relations



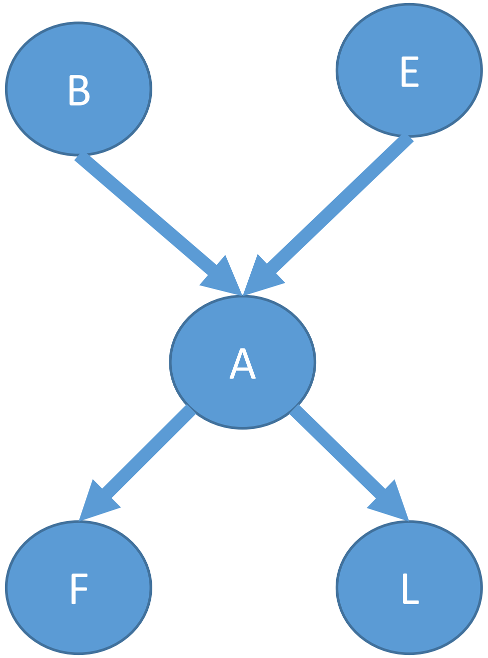
Setting X := x
at time t…

…causes P(Y) at time t+1
to not equal P(Y) at time t

X *causes* Y iff:

1. X happens before Y
2. Manipulating X leads to a change in Y (probabilistically)

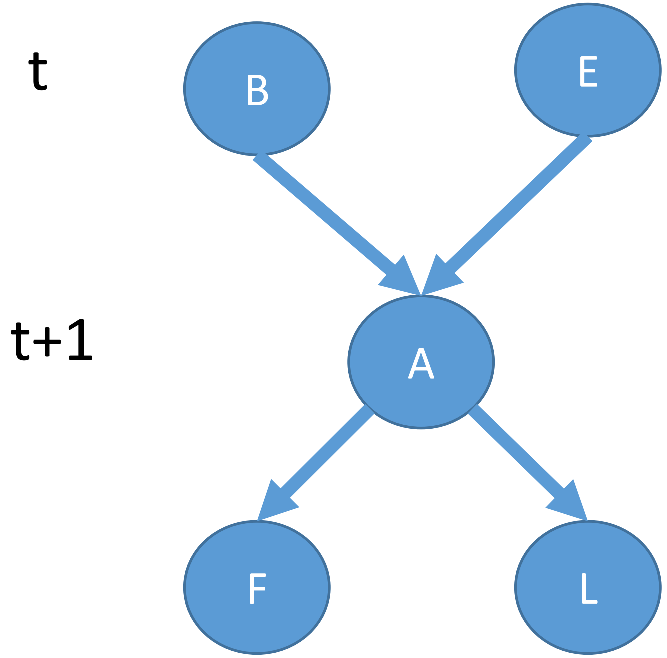
We have intuitive notions of causality



Used informal background knowledge about temporal precedence and causality to encode independence

- Earthquakes and burglaries both *cause* the alarm to trigger.
- Earthquakes don't cause burglaries & vice versa
- Fry and Leela never call in response to burglaries, nor earthquakes

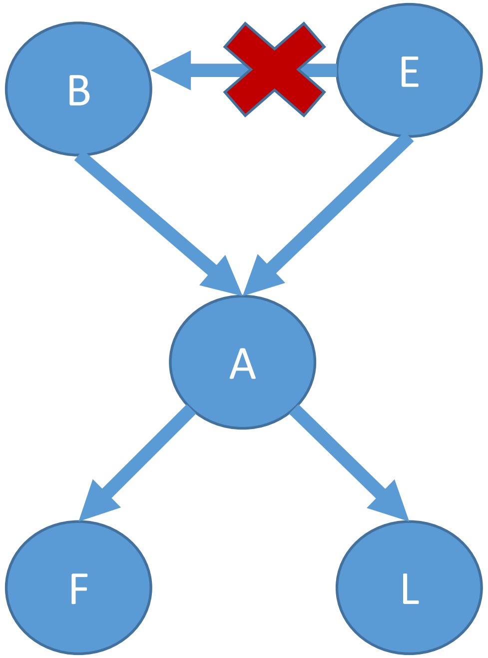
We have intuitive notions of causality



Used informal background knowledge about temporal precedence and causality to encode independence

- **Earthquakes and burglaries both *cause* the alarm to trigger.**
- Earthquakes don’t cause burglaries & vice versa
- Fry and Leela never call in response to burglaries, nor earthquakes

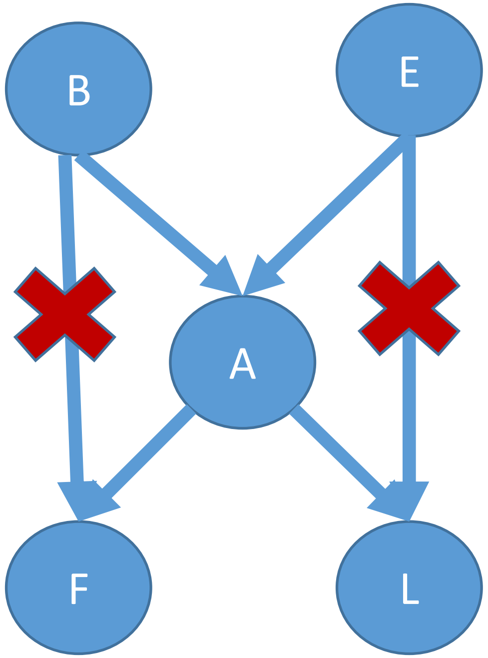
We have intuitive notions of causality



Used informal background knowledge about temporal precedence and causality to encode independence

- Earthquakes and burglaries both *cause* the alarm to trigger.
- **Earthquakes don’t cause burglaries & vice versa.**
- Fry and Leela never call in response to burglaries, nor earthquakes

We have intuitive notions of causality



Used informal background knowledge about temporal precedence and causality to encode independence

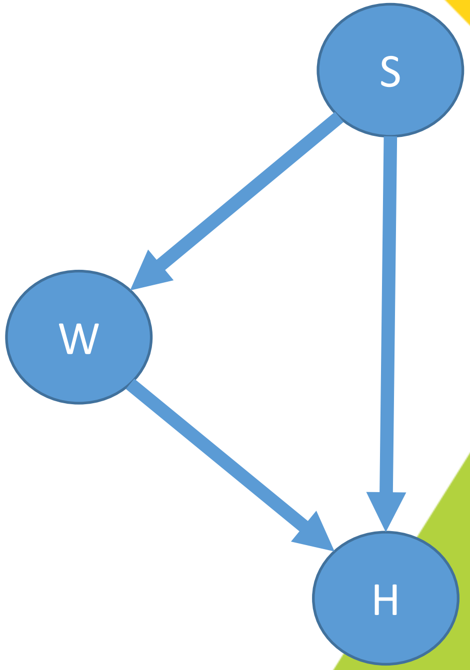
- Earthquakes and burglaries both *cause* the alarm to trigger.
- Earthquakes don’t cause burglaries & vice versa.
- **Fry and Leela never call in response to burglaries, nor earthquakes.**

Bayes Nets vs. Causal Graphs

Scenario: 3 variable, no independence relations

Example: height (H), weight (W), success (S)

Possible factorizations of P(H, W, S):



$P(S \mid H, W)P(H \mid W)P(W)$

$P(H \mid S, W)P(S \mid W)P(W)$

$P(W \mid H, S)P(H \mid S)P(S)$

$P(S \mid H, W)P(W \mid H)P(H)$

$\boxed{P(H \mid S, W)P(W \mid S)P(S)}$

$P(W \mid H, S)P(S \mid H)P(H)$

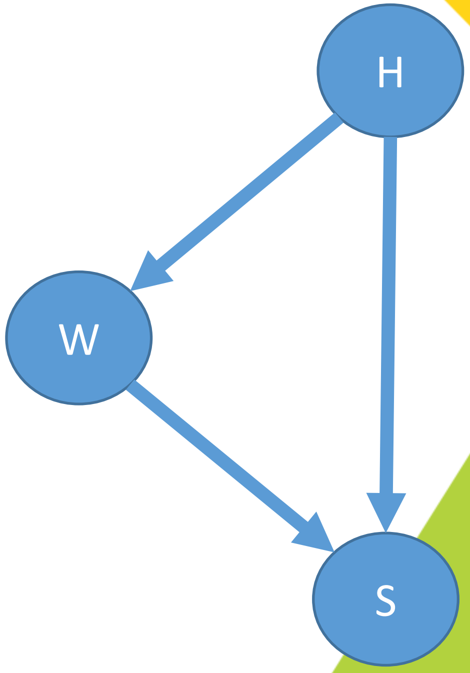
Not causal. Why?

Bayes Nets beyond factorizations

Scenario: 3 variable, no independence relations

Example: height (H), weight (W), success (S)

Possible factorizations of P(H, W, S):



$P(S \mid H, W)P(H \mid W)P(W)$	$\boxed{P(S \mid H, W)P(W \mid H)P(H)}$
$P(H \mid S, W)P(S \mid W)P(W)$	$P(H \mid S, W)P(W \mid S)P(S)$
$P(W \mid H, S)P(H \mid S)P(S)$	$P(W \mid H, S)P(S \mid H)P(H)$

Possibly causal.

Causal graphical models (CGMs) are an *interpretation of Bayes Nets.*

What do CGMs give us?

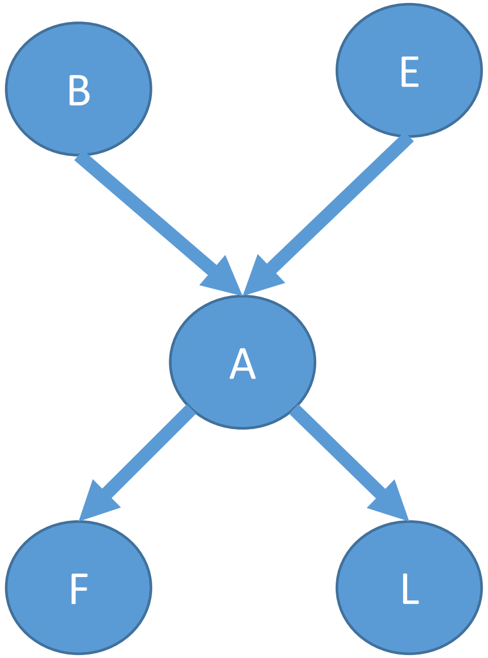
Semantics of a Bayes Net = factorization.

Semantics of a CGM = factorization + *intervention*.

i.e., how to reason from first principles about the statement: *setting X := x at time y causes P(Y) at time t+1 to not equal P(Y) at time t*

Deriving the interventional distribution w/do-calculus

B
0.01



E
0.02

A	B	E
0.01	0	0
0.90	0	1
0.95	1	0
0.99	1	1

F	A
0.05	0
0.75	1

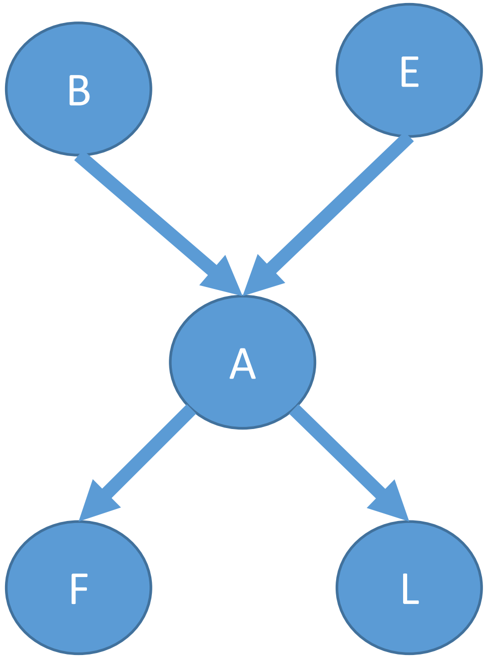
L	A
0.10	0
0.95	1

What does it look like to "intervene" on A?

Deriving the interventional distribution w/do-calculus

B
0.01

E
0.02



A	B	E
0.01	0	0
0.90	0	1
0.95	1	0
0.99	1	1

F	A
0.05	0
0.75	1

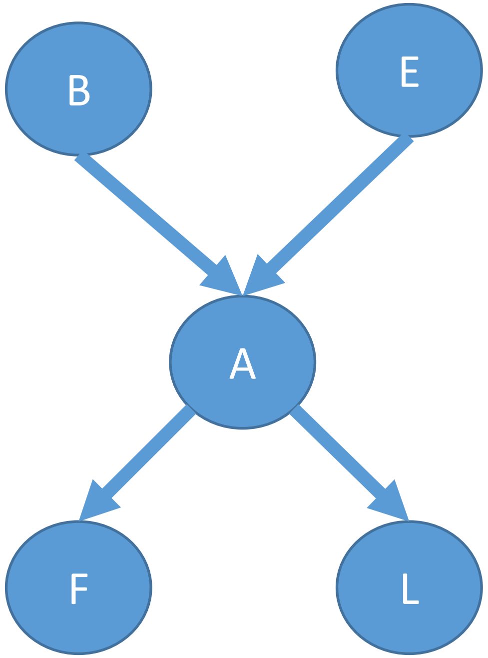
L	A
0.10	0
0.95	1

What does it look like to "intervene" on A?

1. Set A=1 with probability 1

Deriving the interventional distribution w/do-calculus

B
0.01



E
0.02

A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.05	0
0.75	1

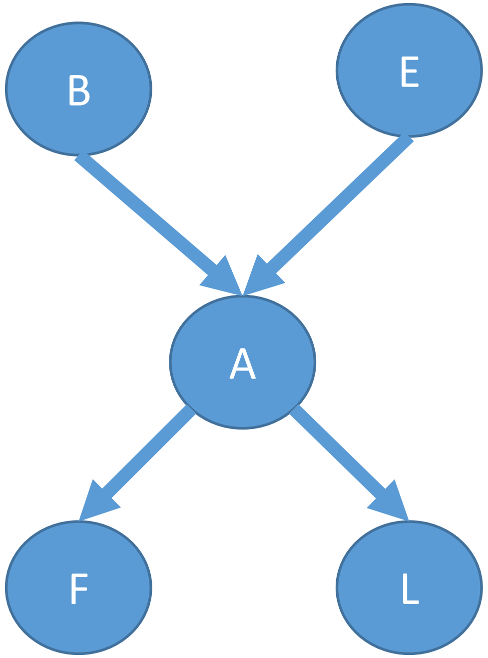
L	A
0.10	0
0.95	1

What does it look like to "intervene" on A?

1. Set A=1 with probability 1

Deriving the interventional distribution w/do-calculus

B
0.01



E
0.02

A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.05	0
0.75	1

L	A
0.10	0
0.95	1

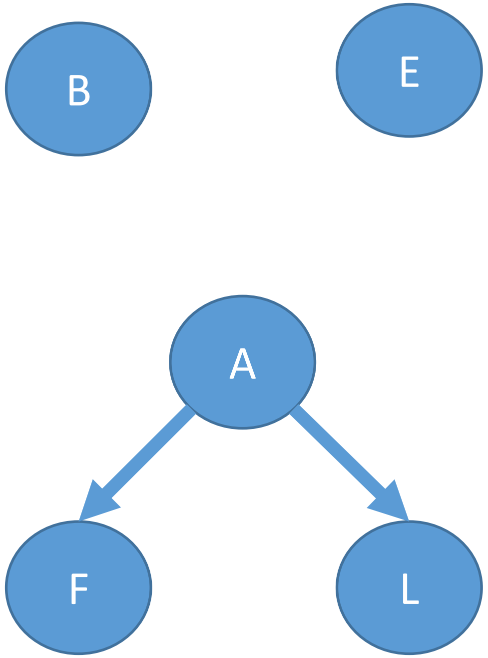
What does it look like to "intervene" on A?

1. Set A=1 with probability 1
2. Remove incoming edges on A.

Deriving the interventional distribution w/do-calculus

B
0.01

E
0.02



A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.05	0
0.75	1

L	A
0.10	0
0.95	1

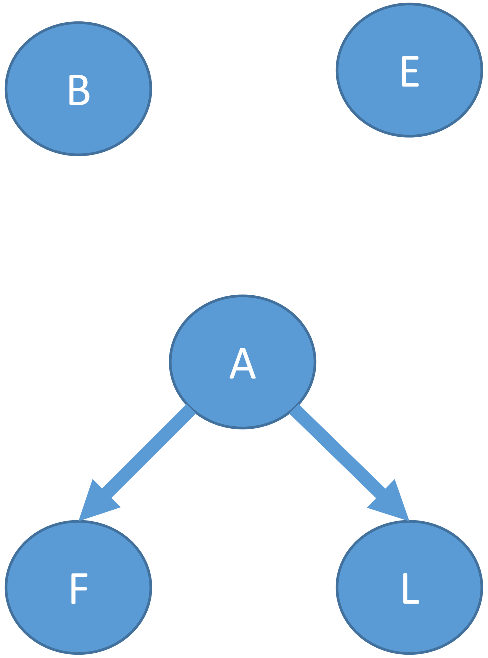
What does it look like to "intervene" on A?

1. Set A=1 with probability 1
2. Remove incoming edges on A.

Deriving the interventional distribution w/do-calculus

B
0.01

E
0.02



A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.05	0
0.75	1

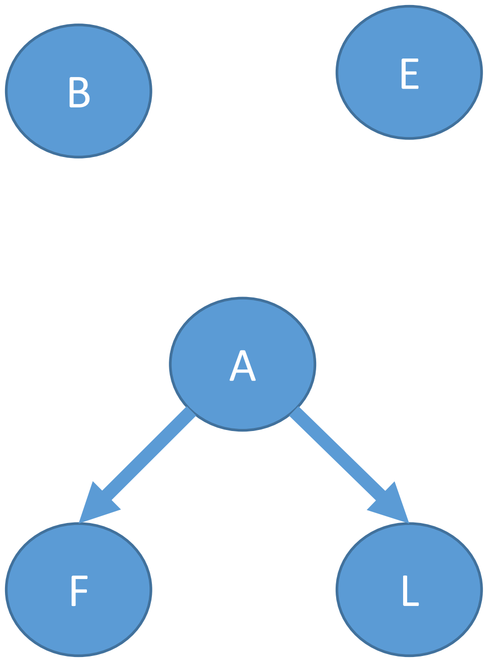
L	A
0.10	0
0.95	1

What does it look like to "intervene" on A?

1. Set A=1 with probability 1
2. Remove incoming edges on A.
3. Update dependent conditional probability distributions.

Deriving the interventional distribution w/do-calculus

B
0.01



E
0.02

A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.75	1

L	A
0.95	1

What does it look like to "intervene" on A?

1. Set A=1 with probability 1
2. Remove incoming edges on A.
3. Update dependent conditional probability distributions.

Compare P(F) vs. P(F) under intervention

```python
# use indices to denote values that the variable takes on
B = [0.99, 0.01]
E = [0.98, 0.02]
# outer index is B; inner is E. don't do this at home.
A = [[[0.99, 0.01], [0.10, 0.90]], [[0.05, 0.95], [0.01, 0.99]]]
# outer index is value of A
F = [[0.95, 0.05], [0.25, 0.75]]
L = [[0.90, 0.10], [0.05, 0.95]]

# P(F) -- naive
PF = [0, 0]
for f in [0, 1]:
  for b in [0,1]:
    for e in [0,1]:
      for a in [0,1]:
        for l in [0,1]:
          # P(F|A)P(L|A)P(A|B,E)P(B)P(E)
          PF[f] += F[a][f]*L[a][l]*A[b][e][a]*B[b]*E[e]

print(PF) #[0.924079, 0.075921]
print(PF[0] + PF[1]) # 1.0
```

```python
# use indices to denote values that the variable takes on
B = [0.99, 0.01]
E = [0.98, 0.02]
# outer index is B; inner is E. don't do this at home.
A = [[[0, 1], [0, 1]], [[0, 1], [0, 1]]]
# outer index is value of A
F = [[0.95, 0.05], [0.25, 0.75]]
L = [[0.90, 0.10], [0.05, 0.95]]

# P(F | do(A := 1)) -- naive
PF = [0, 0]
for f in [0, 1]:
  for b in [0,1]:
    for e in [0,1]:
      for a in [0,1]:
        for l in [0,1]:
          # P(F|A)P(L|A)P(A|B,E)P(B)P(E)
          PF[f] += F[a][f]*L[a][l]*A[b][e][a]*B[b]*E[e]

print(PF) #[0.25, 0.74999999999]
print(PF[0] + PF[1]) # 0.99999999999
```

```
# use indices to denote values that the variable takes on
B = [0.99, 0.01]
E = [0.98, 0.02]
# outer index is B; inner is E. don't do this at home.
A = [[[0, 1], [0, 1]], [[0, 1], [0, 1]]]
# outer index is value of A
F = [[0.95, 0.05], [0.25, 0.75]]
L = [[0.90, 0.10], [0.05, 0.95]]

# P(F | do(A := 1))
PF = [0, 0]
for f in [0, 1]:
  for l in [0,1]:
    # P(F|A)P(L|A)
    PF[f] += F[a][f]*L[a][l]

print(PF) #[0.25, 0.74999999999]
print(PF[0] + PF[1]) # 0.99999999999
```

```python
# use indices to denote values that the variable takes on
B = [0.99, 0.01]
E = [0.98, 0.02]
# outer index is B; inner is E. don't do this at home.
A = [[[0, 1], [0, 1]], [[0, 1], [0, 1]]]
# outer index is value of A
F = [[0.95, 0.05], [0.25, 0.75]]
L = [[0.90, 0.10], [0.05, 0.95]]

# P(F | do(A := 1))
PF = [0, 0]
for f in [0, 1]:
  for l in [0,1]:
    # P(F|A)P(L|A)
    PF[f] += F[1][f]*L[1][l]

print(PF) #[0.25, 0.74999999999]
print(PF[0] + PF[1]) # 0.99999999999
```

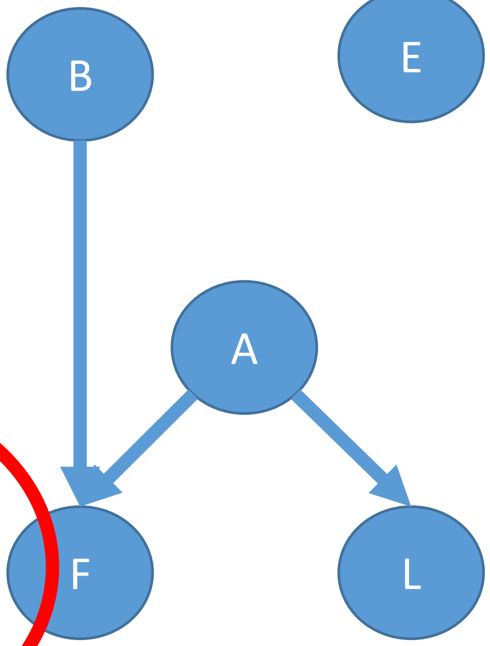
Compare with P(F|A=1)

```python
# use indices to denote values that the variable takes on
B = [0.99, 0.01]
E = [0.98, 0.02]
# outer index is B; inner is E. don't do this at home.
A = [[[0, 1], [0, 1]], [[0, 1], [0, 1]]]
# outer index is value of A
F = [[0.95, 0.05], [0.25, 0.75]]
L = [[0.90, 0.10], [0.05, 0.95]]

# P(F | do(A := 1))
PF = [0, 0]
for f in [0, 1]:
  for l in [0,1]:
    # P(F|A)P(L|A)
    PF[f] += F[a][f]*L[a][l]

print(PF) #[0.25, 0.74999999999]
print(PF[0] + PF[1]) # 0.99999999999
```

When might P(F|A=1) =/= P(F|do(A:=1))?

Deriving the interventional distribution w/do-calculus

B
0.01



E
0.02

A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.75	1

L	A
0.95	1

What does it look like to "intervene" on A?

1. Set A=1 with probability 1
2. Remove incoming edges on A.
3. Update dependent conditional probability distributions.

Do stuff on the board

```python
# use indices to denote values that the variable takes on
B = [0.99, 0.01]
E = [0.98, 0.02]
# outer index is B; inner is E. don't do this at home.
A = [[[0.99, 0.01], [0.10, 0.90]], [[0.05, 0.95], [0.01, 0.99]]]
# outer index is value of A
F = [[0.95, 0.05], [0.25, 0.75]]
L = [[0.90, 0.10], [0.05, 0.95]]
```

```
# use indices to denote values that the variable takes on
B = [0.99, 0.01]
E = [0.98, 0.02]
# outer index is B; inner is E. don't do this at home.
A = [[[0.99, 0.01], [0.10, 0.90]], [[0.05, 0.95], [0.01, 0.99]]]
# outer index is value of A; inner is value of B
F = [[[0.99, 0.01], [0.98, 0.02]], [[0.97, 0.03], [0.96, 0.04]]]
L = [[0.90, 0.10], [0.05, 0.95]]
```

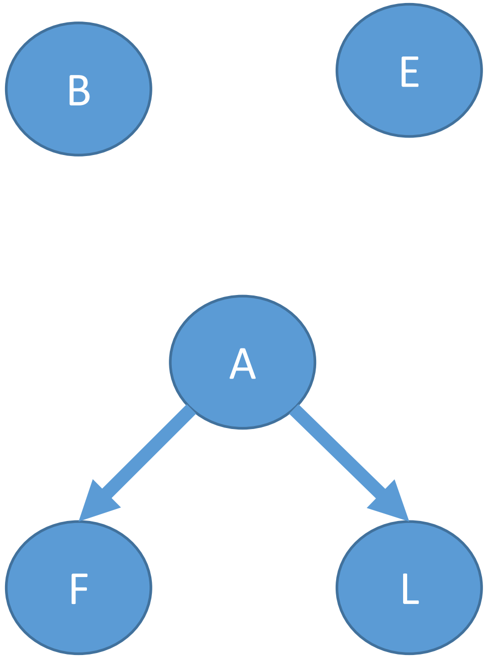
```python
# use indices to denote values that the variable takes on
B = [0.99, 0.01]
E = [0.98, 0.02]
# outer index is B; inner is E. don't do this at home.
A = [[[0.99, 0.01], [0.10, 0.90]], [[0.05, 0.95], [0.01, 0.99]]]
# outer index is value of A; inner is value of B
F = [[[0.99, 0.01], [0.98, 0.02]], [[0.97, 0.03], [0.96, 0.04]]]
L = [[0.90, 0.10], [0.05, 0.95]]


PFdoA = [[0,0], [0,0]]
for f in [0,1]:
  for a in [0,1]:
    for l in [0,1]:
      for b in [0,1]:
        # P(F|do(A))P(L|do(A))P(B)
        PFdoA[a][f] += F[a][b][f]*L[a][l]*B[b]

print(PFdoA)
# [[0.9899000000000001, 0.0101], [0.9699, 0.030099999999999995]]
```

Draw graph on board

What does this give us?

B
0.01

E
0.02



A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.75	1

L	A
0.95	1

We can compute the *effect* of setting A=0 vs. A=1 (denoted with "do"):

- Need to compute a meaningful quantity (not probability)
- Expectation!

$E[F \mid do(A:=0) - F \mid do(A:=1)]$

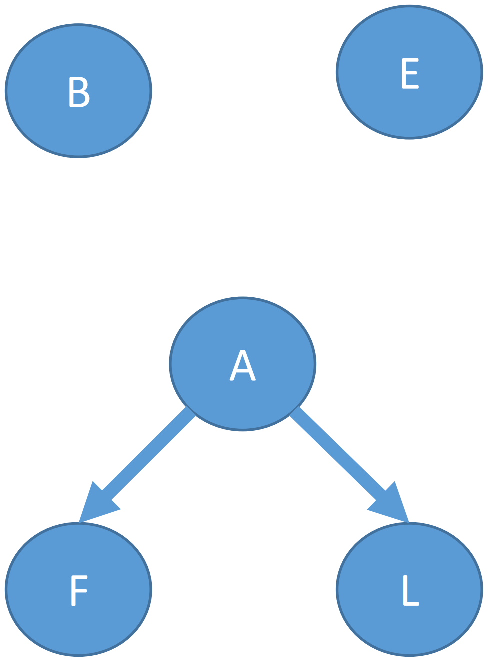
$= E[F \mid do(A:=0)] - E[F \mid do(A:=1)]$

```python
# Feel free to compute over either the original graph (here) or the
# modified one
PFdoA1 = [0,0]
for f in [0,1]:
  for l in [0,1]:
    PFdoA1[f] += F[1][f] * L[1][l]

PFdoA0 = [0,0]
for f in [0,1]:
  for l in [0,1]:
    PFdoA0[f] += F[0][f] * L[0][l]

EFdoA1 = sum([f * PFdoA1[f] for f in range(len(PFdoA1))])
EFdoA0 = sum([f * PFdoA0[f] for f in range(len(PFdoA0))])
print(EFdoA1 - EFdoA0)
```

What does this all mean?

B
0.01

E
0.02



A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.75	1

L	A
0.95	1

Remember: expectation can only be computed over *random variables*

Random variables are functions from outcomes to real numbers

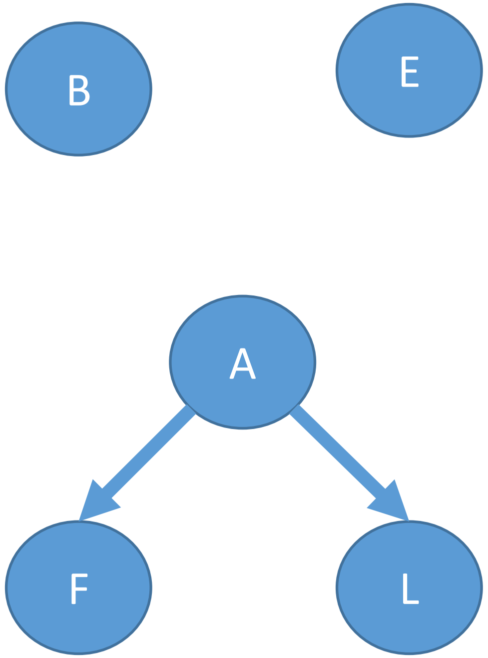
Because these are Bernoulli (i.e., from the set {0, 1}) random variables, they can be manipulated as numbers…

…but this interpretation may not be meaningful!

What does this all mean?

B
0.01

E
0.02



A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.75	1

L	A
0.95	1

Remember: expectation can only be computed over *random variables*

Random variables are functions from outcomes to real numbers

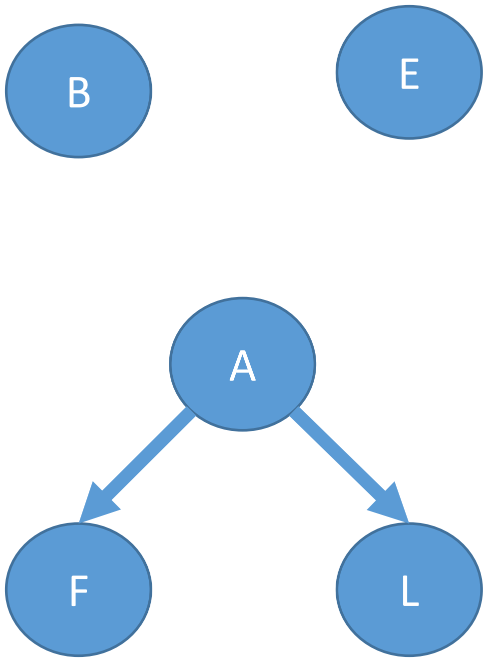
Because these are Bernoulli (i.e., from the set {0, 1}) random variables, they can be manipulated as numbers…

…but this interpretation may not be meaningful!

What does this all mean?

B
0.01

E
0.02



A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.75	1

L	A
0.95	1

Remember: expectation can only be computed over *random variables*

Random variables are functions from outcomes to real numbers

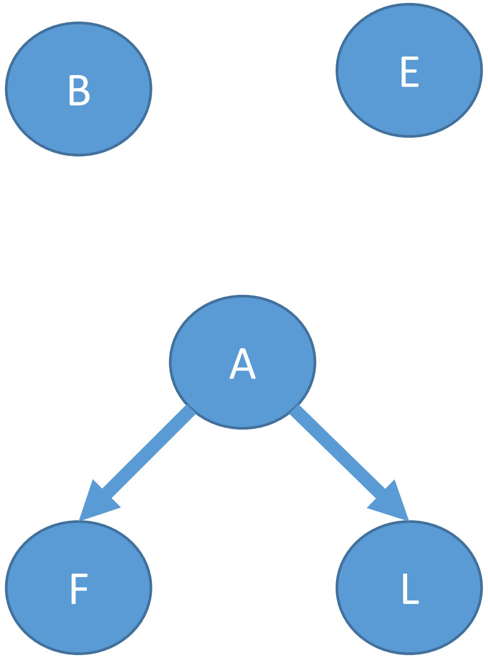
Because these are Bernoulli (i.e., from the set {0, 1}) random variables, they can be manipulated as numbers…

…but this interpretation may not be meaningful!

What does this all mean?

B
0.01

E
0.02



A	B	E
1	0	0
1	0	1
1	1	0
1	1	1

F	A
0.75	1

L	A
0.95	1

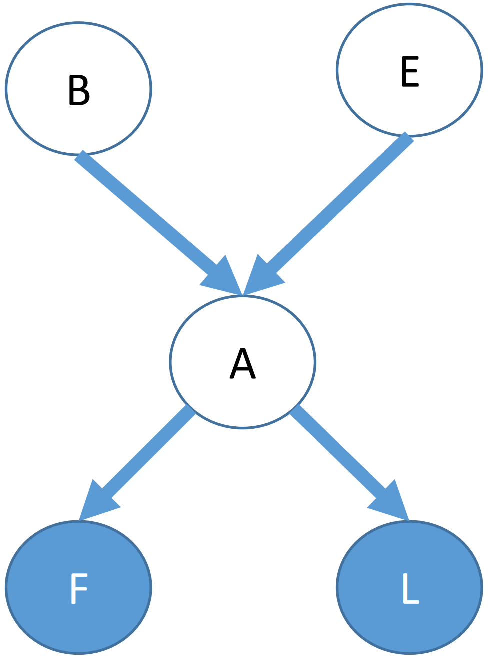
Remember: expectation can only be computed over *random variables*

Random variables are functions from outcomes to real numbers

Because these are Bernoulli (i.e., from the set {0, 1}) random variables, they can be manipulated as numbers…

…but this interpretation may not be meaningful!

What does this give us



We may need to reason about *latent* or *unobserved* nodes.

IRL we can manipulate the alarm.

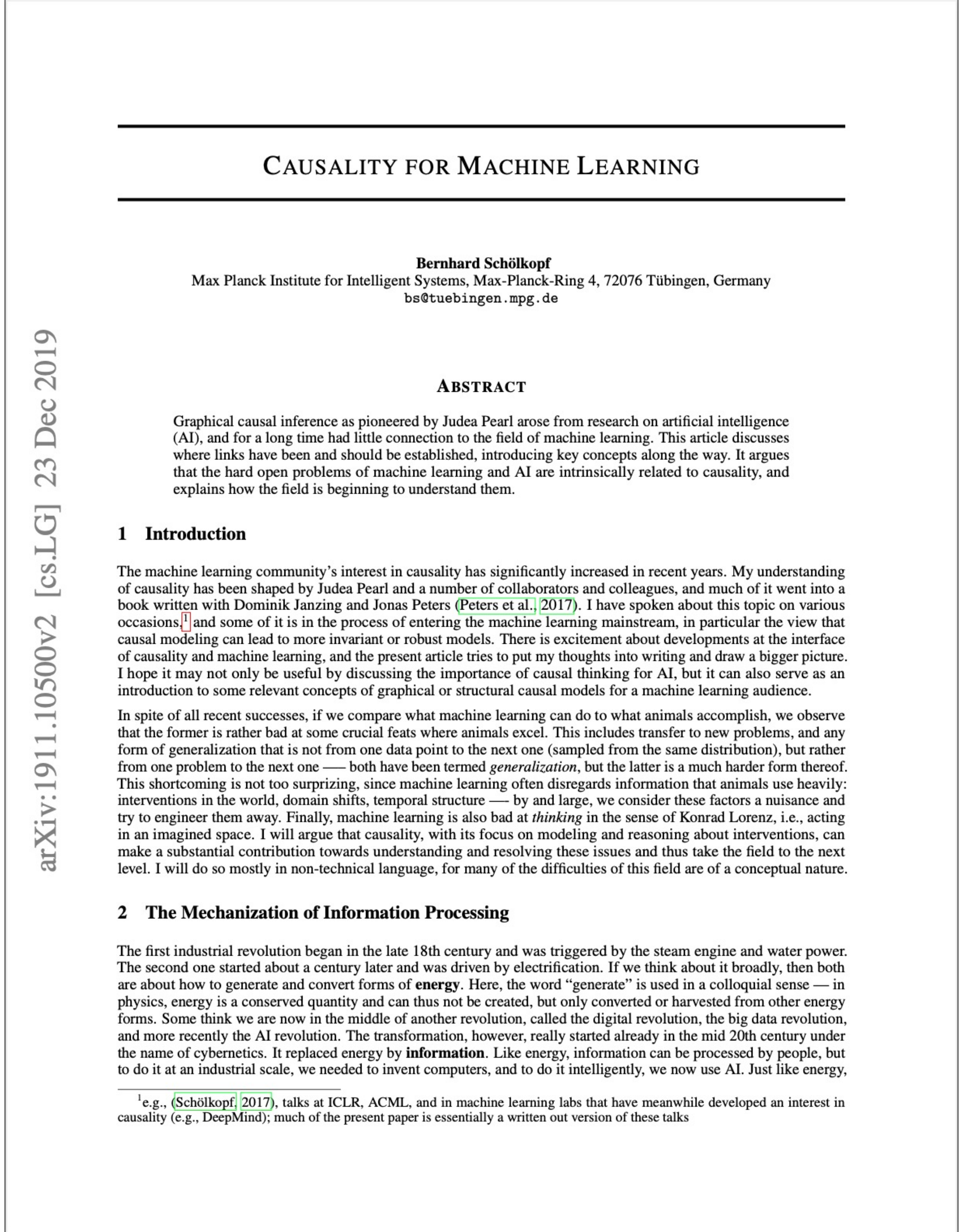
IRL we can’t (or at least shouldn’t) *cause* an earthquake.

Do-calculus gives us a mechanism for reasoning about experimentation.

Important high-level properties of CGMs

What's the big deal with intervention?

- Addresses weaknesses of classical AI logics:
  - Open world doesn't matter
  - Allows counterfactual reasoning
- Sparse representation (no cruft)
  - c.f. deep learning, where we *expect* collinearities in features
- Represents invariance (with respect to other covariates)



arXiv:1911.10500v2 [cs.LG] 23 Dec 2019

Causality for Machine Learning

Bernhard Schölkopf
Max Planck Institute for Intelligent Systems, Max-Planck-Ring 4, 72076 Tübingen, Germany
bs@tuebingen.mpg.de

Abstract

Graphical causal inference as pioneered by Judea Pearl arose from research on artificial intelligence (AI), and for a long time had little connection to the field of machine learning. This article discusses where links have been and should be established, introducing key concepts along the way. It argues that the hard open problems of machine learning and AI are intrinsically related to causality, and explains how the field is beginning to understand them.

1 Introduction

The machine learning community's interest in causality has significantly increased in recent years. My understanding of causality has been shaped by Judea Pearl and a number of collaborators and colleagues, and much of it went into a book written with Dominik Janzing and Jonas Peters (Peters et al., 2017). I have spoken about this topic on various occasions,[1] and some of it is in the process of entering the machine learning mainstream, in particular the view that causal modeling can lead to more invariant or robust models. There is excitement about developments at the interface of causality and machine learning, and the present article tries to put my thoughts into writing and draw a bigger picture. I hope it may not only be useful by discussing the importance of causal thinking for AI, but it can also serve as an introduction to some relevant concepts of graphical or structural causal models for a machine learning audience.

In spite of all recent successes, if we compare what machine learning can do to what animals accomplish, we observe that the former is rather bad at some crucial feats where animals excel. This includes transfer to new problems, and any form of generalization that is not from one data point to the next one (sampled from the same distribution), but rather from one problem to the next one — both have been termed *generalization*, but the latter is a much harder form thereof. This shortcoming is not too surprizing, since machine learning often disregards information that animals use heavily: interventions in the world, domain shifts, temporal structure — by and large, we consider these factors a nuisance and try to engineer them away. Finally, machine learning is also bad at *thinking* in the sense of Konrad Lorenz, i.e., acting in an imagined space. I will argue that causality, with its focus on modeling and reasoning about interventions, can make a substantial contribution towards understanding and resolving these issues and thus take the field to the next level. I will do so mostly in non-technical language, for many of the difficulties of this field are of a conceptual nature.

2 The Mechanization of Information Processing

The first industrial revolution began in the late 18th century and was triggered by the steam engine and water power. The second one started about a century later and was driven by electrification. If we think about it broadly, then both are about how to generate and convert forms of **energy**. Here, the word "generate" is used in a colloquial sense — in physics, energy is a conserved quantity and can thus not be created, but only converted or harvested from other energy forms. Some think we are now in the middle of another revolution, called the digital revolution, the big data revolution, and more recently the AI revolution. The transformation, however, really started already in the mid 20th century under the name of cybernetics. It replaced energy by **information**. Like energy, information can be processed by people, but to do it at an industrial scale, we needed to invent computers, and to do it intelligently, we now use AI. Just like energy,

[1] e.g., (Schölkopf, 2017), talks at ICLR, ACML, and in machine learning labs that have meanwhile developed an interest in causality (e.g., DeepMind); much of the present paper is essentially a written out version of these talks

Applications in AI

- Classically, planning
  - Post-conditions as causal
  - Problem: impedance mismatch in representation & tractability issues
- Recently: mechanism representation in machine learning
- CGMs as bridge between statistics and logic

Next Class: Epistemic Logics