

Inference in Belief Networks

CMPUT 261: Introduction to Artificial Intelligence

P&M §8.4

Assignments

- **Assignment #1:** Marking should be done by next week
 - Final grades are not yet entered
 - **Don't Panic:** Some people may have gotten partial grades for one question due to a Canvas glitch
- **Assignment #2** will be posted **today** by midnight
 - Due **October 21/2025** at **11:59pm**

Lecture Outline

1. Recap
2. Factor Objects
3. Variable Elimination
4. Further Optimizations

After this lecture, you should be able to:

- encode a factoring of a joint distribution as a collection of factor objects for variable elimination
- define the factor operations used in variable elimination
- describe the high-level steps of variable elimination
- compare efficiency of different variable orderings for variable elimination
- trace an execution of variable elimination

Recap: Belief Networks

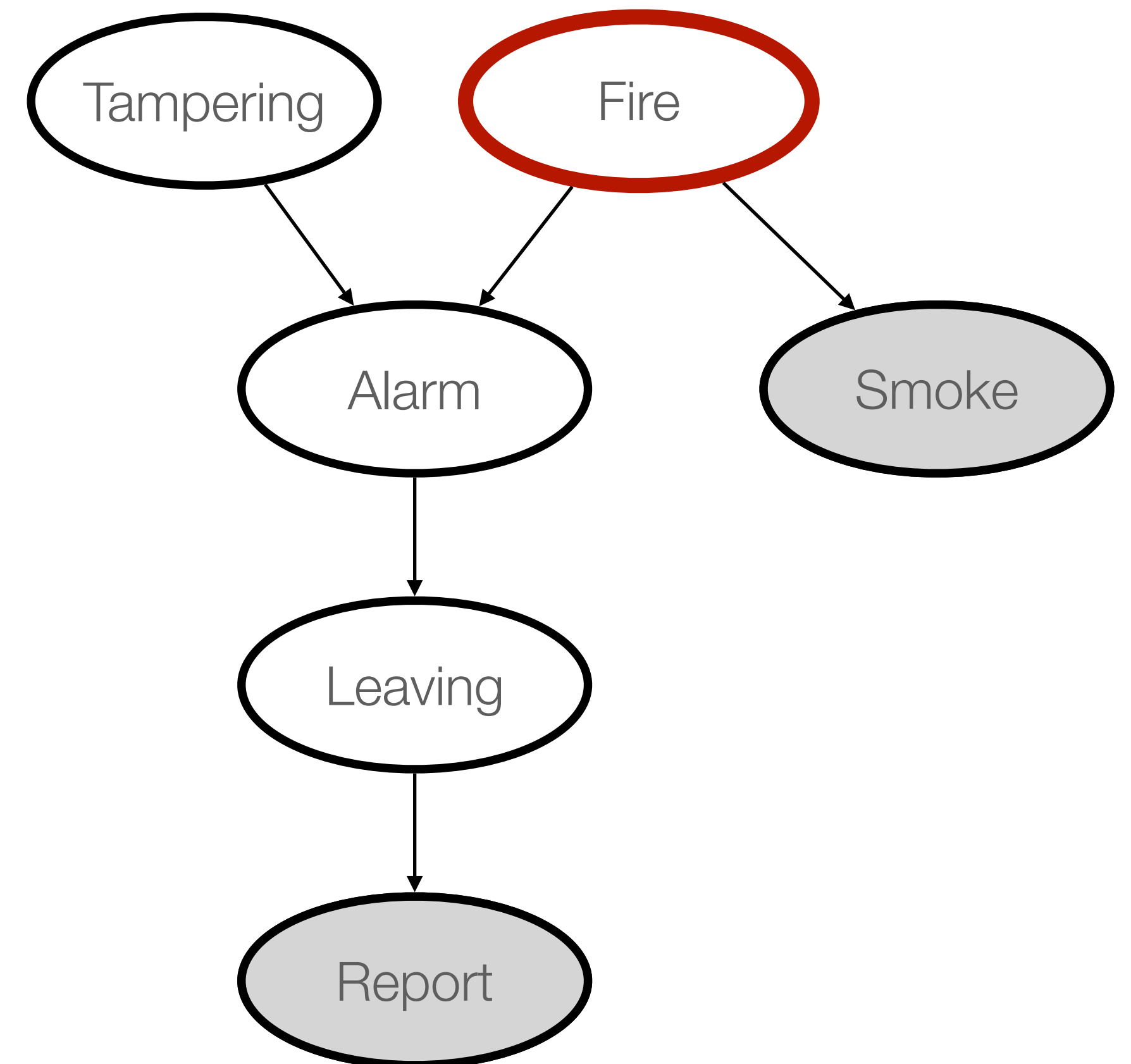
Definition:

A **belief network** (or **Bayesian network**) consists of:

1. A directed acyclic graph, with each node labelled by a **random variable**
 2. A **domain** for each random variable
 3. A **conditional probability table** for each variable given its **parents**
- The graph represents a specific **factorization** of the full **joint distribution**
 - **Key Property:**
Every node is **independent** of its **non-descendants**, **conditional** on its **parents**

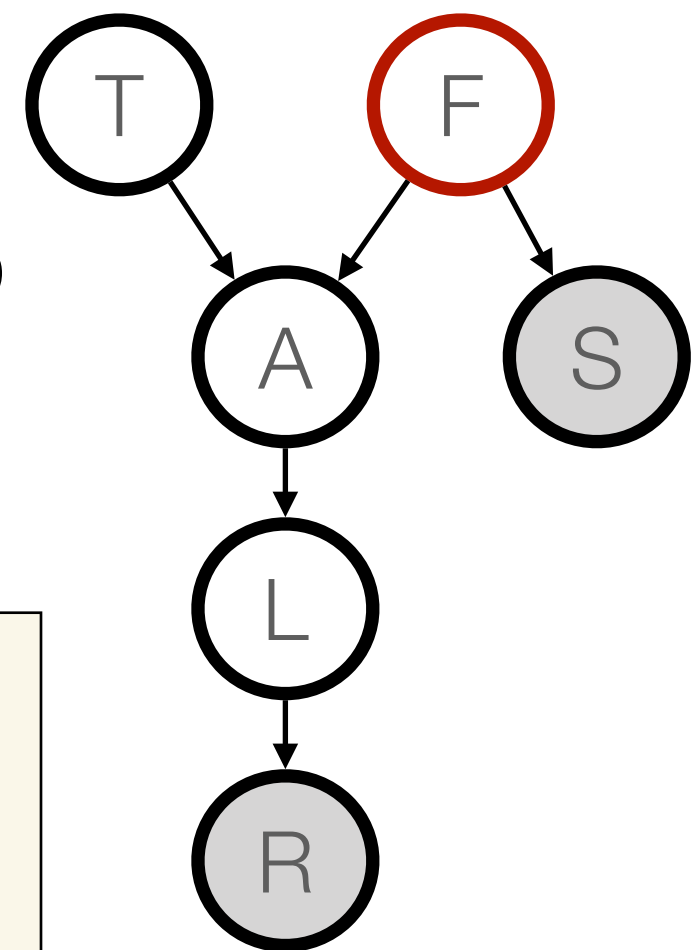
Recap: Queries

- The most common task for a belief network is to query **posterior probabilities** given some **observations**
- **Easy cases:**
 - Posteriors of a **single variable** conditional only on **parents**
 - **Joint distributions** of variables early in a **compatible variable ordering**
- Typically, the observations have **no straightforward relationship** to the target
- **This lecture:** mechanical procedure for computing **arbitrary queries**



Bayes Net: Report has parent Leaving, Leaving has parent Alarm, Alarm has parents Tampering and Fire, Smoke has parent Fire, Tampering and Fire have no parents. Smoke and Report are both grey (indicating "observed"), and Fire has a red border (indicating "queried").

A (Simplistic) Algorithm for Queries



$$P(F, T, A, L, S, R) = P(F)P(T)P(A \mid T, F)P(L \mid A)P(S \mid F)P(R \mid L)$$

Query: $P(F \mid S = 1, R = 1)$

1. **Condition:** $P(F, T, A, L, S = 1, R = 1) = P(F)P(T)P(A \mid T, F)P(S = 1 \mid F)P(L \mid A)P(R = 1 \mid L)$

2. **Normalize:** $P(F, T, A, L \mid S = 1, R = 1) = \frac{P(F, T, A, L, S = 1, R = 1)}{\sum_{\substack{f \in \text{dom}(F), \\ t \in \text{dom}(T), \\ a \in \text{dom}(A), \\ l \in \text{dom}(L)}} P(F = f, T = t, A = a, L = l, S = 1, R = 1)}$

3. **Marginalize:** $P(F \mid S = 1, R = 1) = \sum_{\substack{t \in \text{dom}(T), \\ a \in \text{dom}(A), \\ l \in \text{dom}(L)}} P(F, T = t, A = a, L = l \mid S = 1, R = 1)$

Factor Object

- The **Variable Elimination** algorithm exploits the **factorization** of a joint probability distribution encoded by a belief network in order to answer **queries**
- A **factor object** is a function $f(X_1, \dots, X_k)$ from **random variables** to a **real number**
- **Input:** factors representing the **conditional probability tables** from the belief network

$$P(L \mid A)P(S \mid F)P(A \mid T, F)P(T)P(F)$$

becomes factor objects

$$f_1(L, A)f_2(S, F)f_3(A, T, F)f_4(T)f_5(F)$$

- **Output:** A **new factor** encoding the target **posterior distribution**

E.g., $f_{12}(T)$.

Conditional Probabilities as Factor Objects

- A **conditional probability** $P(Y \mid X_1, \dots, X_n)$ is a factor object $f(Y, X_1, \dots, X_n)$ that obeys the **constraint**:

$$\forall v_1 \in \text{dom}(X_1), v_2 \in \text{dom}(X_2), \dots, v_n \in \text{dom}(X_n) : \left[\sum_{y \in \text{dom}(Y)} f(y, v_1, \dots, v_n) \right] = 1.$$

- Answer to a query is a factor object **constructed** by applying **operations** to the input factors
 - Operations on factor objects are *not* guaranteed to **maintain** this constraint!
 - Solution: **Don't sweat it!**
 - Operate on **unnormalized probabilities** during the computation
 - **Normalize** at the end of the algorithm to re-impose the constraint

Conditioning

Conditioning is an operation on a **single factor**

- Constructs a **new factor** that returns the values of the original factor with some of its inputs fixed

Definition:

For a factor $f_1(X_1, \dots, X_k)$, **conditioning on** $X_i = v_i$ yields a new factor

$$(f_1)_{X_i=v_i} = f_2(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_k)$$

such that for all values $v_1, \dots, v_{i-1}, v_{i+1}, \dots, v_k$ in the domain of $X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_k$,

$$f_2(v_1, \dots, v_{i-1}, v_{i+1}, \dots, v_k) = f_1(v_1, \dots, v_{i-1}, \mathbf{v_i}, v_{i+1}, \dots, v_k).$$

Conditioning Example

$$f_1(A, B, C)_{C=true} = f_2(A, B) = f_1(A, B, true)$$

f_1

A	B	C	value
F	F	F	0.1
F	F	T	0.88
F	T	F	0.12
F	T	T	0.45
T	F	F	0.7
T	F	T	0.66
T	T	F	0.1
T	T	T	0.25

f_2

A	B	value
F	F	0.88
F	T	0.45
T	F	0.66
T	T	0.25

Multiplication

Multiplication is an operation on **two factors**

- Constructs a new factor that returns the **product** of the rows selected from each factor by its arguments

Definition:

For two factors $f_1(X_1, \dots, X_j, Y_1, \dots, Y_k)$ and $f_2(Y_1, \dots, Y_k, Z_1, \dots, Z_\ell)$,

multiplication of f_1 and f_2 yields a new factor

$$(f_1 \times f_2) = f_3(X_1, \dots, X_j, Y_1, \dots, Y_k, Z_1, \dots, Z_\ell)$$

such that for all values $x_1, \dots, x_j, y_1, \dots, y_k, z_1, \dots, z_\ell$,

$$f_3(x_1, \dots, x_j, y_1, \dots, y_k, z_1, \dots, z_\ell) = f_1(x_1, \dots, x_j, y_1, \dots, y_k) f_2(y_1, \dots, y_k, z_1, \dots, z_\ell).$$

Multiplication Example

$$f_3(A, B, C) = f_1(A, B) \times f_2(B, C)$$

f_1

A	B	value
F	F	0.1
F	T	0.2
T	F	0.3
T	T	0.4

f_2

B	C	value
F	F	1.0
F	T	0
T	F	0.5
T	T	0.25

f_3

A	B	C	value
F	F	F	0.1
F	F	T	0
F	T	F	0.1
F	T	T	0.05
T	F	F	0.3
T	F	T	0
T	T	F	0.2
T	T	T	0.1

Summing Out

Summing out is an operation on a **single factor**

- Constructs a new factor that returns the **sum over all values** of a random variable of the original factor

Definition:

For a factor $f_1(X_1, \dots, X_k)$, summing out a variable X_i yields a new factor

$$\left(\sum_{X_i} f_1 \right) = f_2(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_k)$$

such that for all values $v_1, \dots, v_{i-1}, v_{i+1}, \dots, v_k$ in the domain of $X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_k$

$$f_2(v_1, \dots, v_{i-1}, v_{i+1}, \dots, v_k) = \sum_{\mathbf{v}_i \in \text{dom}(X_i)} f_1(v_1, \dots, v_{i-1}, \mathbf{v}_i, v_{i+1}, \dots, v_k).$$

Summing Out Example

$$f_2(B) = \sum_A f_1(A, B)$$

f_1

A	B	value
F	F	0.1
F	T	0.2
T	F	0.3
T	T	0.4

f_2

B	value
F	0.4
T	0.6

Variable Elimination

- Given **observations** $Y_1 = v_1, \dots, Y_k = v_k$ and query variable Q , we want

$$P(Q \mid Y_1 = v_1, \dots, Y_k = v_k) = \frac{P(Q, Y_1 = v_1, \dots, Y_k = v_k)}{\sum_{q \in \text{dom}(Q)} P(Q = q, Y_1 = v_1, \dots, Y_k = v_k)}.$$

- Basic idea of variable elimination:
 1. Condition on observations by **conditioning**
 2. Construct joint distribution factor by **multiplication**
 3. Remove unwanted variables (neither query nor observed) by **summing out**
 4. **Normalize** at the end
- Doing these steps in order is **correct** but not **efficient**
- Efficiency comes from **interleaving** the order of operations

Sums of Products

2. Construct joint distribution factor by **multiplication**
3. Remove unwanted variables (neither query nor observed) by **summing out**

The computationally intensive part of variable elimination is computing **sums** of **products**

Example: multiply factors $f_1(Q, A, B, C), f_2(C, D, E)$; sum out A, E

1. $f_3(Q, A, B, C, D, E) = f_1(Q, A, B, C) \times f_2(C, D, E) : 2^6$ multiplications

2. $f_4(Q, B, C, D) = \sum_{A, E} f_3(Q, A, B, C, D, E) : 3 \times 16$ additions

Total: **112** arithmetic operations

(*) For all numerical examples,
we assume binary domains

Efficient Sums of Products

We can reduce the number of computations required by changing their **order**.

$$\begin{aligned} \sum_A \sum_E f_1(Q, A, B, C) \times f_2(C, D, E) &= \left(\sum_A f_1(Q, A, B, C) \right) \times \left(\sum_E f_2(C, D, E) \right) \\ &= \sum_A f_1(Q, A, B, C) \left(\sum_E f_2(C, D, E) \right) \\ &= \left(\sum_E f_2(C, D, E) \right) \sum_A f_1(Q, A, B, C) \end{aligned}$$

1. $f_3(C, D) = \sum_E f_2(C, D, E) : 2^2$ additions
2. $f_4(Q, B, C) = \sum_A f_1(Q, A, B, C) : 2^3$ additions
3. $f_5(Q, B, C, D) = f_3(Q, B, C) \times f_4(B, C, D) : 2^4$ multiplications

Total: **28** arithmetic operations

Variable Elimination Algorithm

Input: query variable Q ; set of variables Vs ; observations O ; factors Ps representing conditional probability tables

$Fs := Ps$

for each X in $Vs \setminus \{Q\}$ according to some **elimination ordering**:

$R_s := \{F \in Fs \mid F \text{ involves } X\}$

if $X \in O$:

 for each $F \in R_s$:

$F' := F$ **conditioned** on observed value of X

$Fs := (Fs \setminus \{F\}) \cup \{F'\}$

else:

$T :=$ **product** of factors in R_s

$N :=$ **sum** X out of T

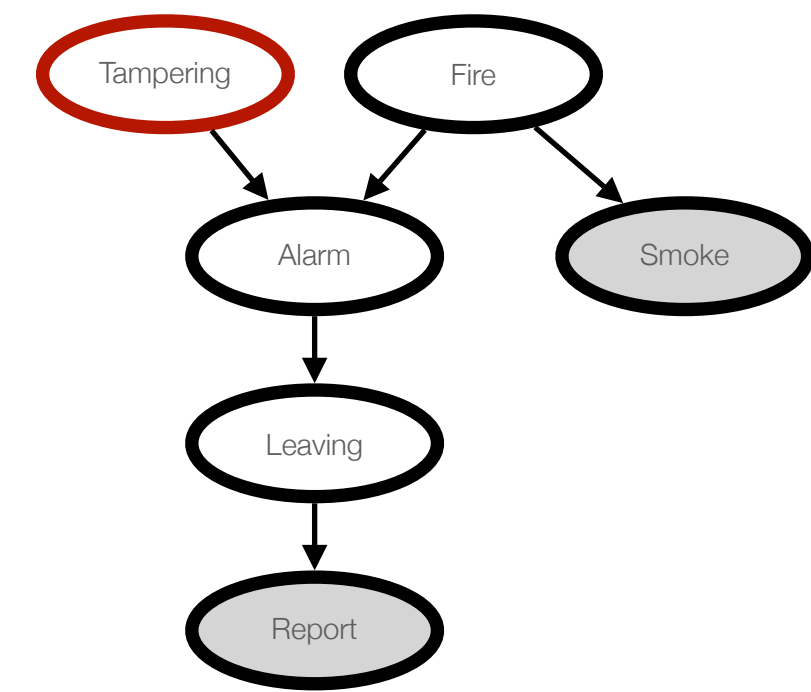
$Fs := (Fs \setminus R_s) \cup \{N\}$

$T :=$ **product** of factors in Fs

$N :=$ **sum** Q out of T (returns a scalar number)

return T/N (i.e., normalize T)

Variable Elimination Example: Conditioning



Bayes Net: Report has parent Leaving, Leaving has parent Alarm, Alarm has parents Tampering and Fire, Smoke has parent Fire, Tampering and Fire have no parents. Smoke and Report are both grey (indicating "observed"), and Tampering has a red border (indicating "queried").

Query: $P(T | S = 1, R = 1)$

Variable ordering: S, R, F, A, L

$$P(T, F, A, S, L, R) = P(T)P(F)P(A | T, F)P(S | F)P(L | A)P(R | L)$$

Construct **factors** for each table:

$$\{f_0(T), f_1(F), f_2(T, A, F), f_3(S, F), f_4(L, A), f_5(R, L)\}$$

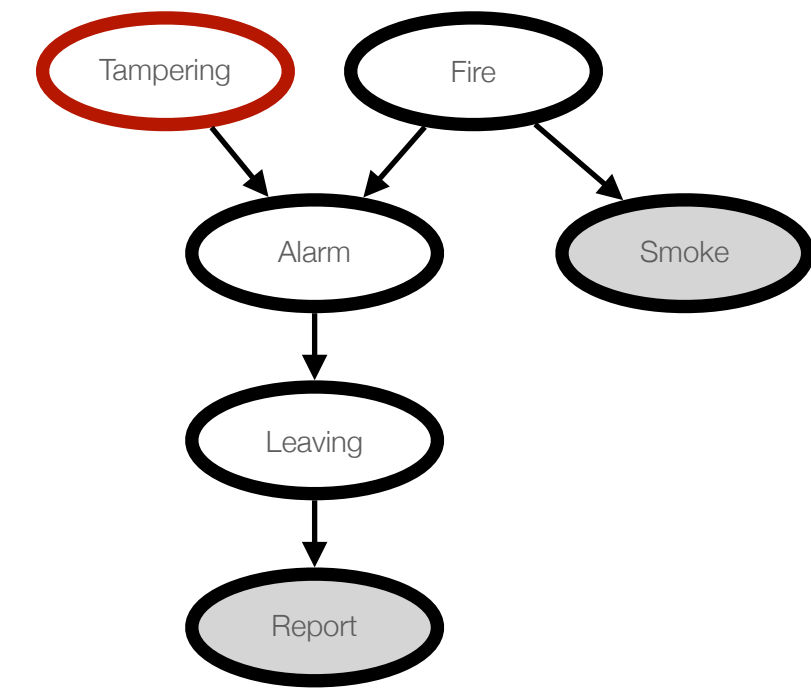
Condition on S : $f_6 = (f_3)_{S=1}$

$$\{f_0(T), f_1(F), f_2(T, A, F), f_6(F), f_4(L, A), f_5(R, L)\}$$

Condition on R : $f_7 = (f_5)_{R=1}$

$$\{f_0(T), f_1(F), f_2(T, A, F), f_6(F), f_4(L, A), f_7(L)\}$$

Variable Elimination Example: Elimination



Bayes Net: Report has parent Leaving, Leaving has parent Alarm, Alarm has parents Tampering and Fire, Smoke has parent Fire, Tampering and Fire have no parents. Smoke and Report are both grey (indicating "observed"), and Tampering has a red border (indicating "queried").

Query: $P(T | S = 1, R = 1)$

Variable ordering: ~~S, R, F~~ , A, L

$\{f_0(T), f_1(F), f_2(T, A, F), f_6(F), f_4(L, A), f_7(L)\}$

Sum out F from **product** of f_1, f_2, f_6 : $f_8 = \sum_F (f_1 \times f_2 \times f_6)$

$\{f_0(T), f_8(T, A), f_4(L, A), f_7(L)\}$

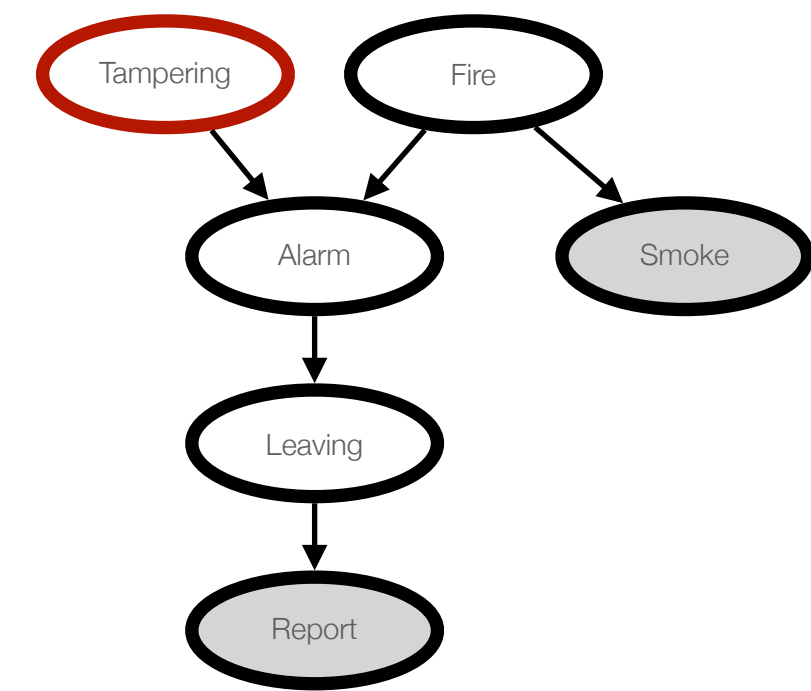
Sum out A from **product** of f_8, f_4 : $f_9 = \sum_A (f_8 \times f_4)$

$f_0(T), f_9(T, L), f_7(L)$

Sum out L from **product** of f_9, f_7 : $f_{10} = \sum_L (f_9 \times f_7)$

$\{f_0(T), f_{10}(T)\}$

Variable Elimination Example: Normalization



Bayes Net: Report has parent Leaving, Leaving has parent Alarm, Alarm has parents Tampering and Fire, Smoke has parent Fire, Tampering and Fire have no parents. Smoke and Report are both grey (indicating "observed"), and Tampering has a red border (indicating "queried").

Query: $P(T | S = 1, R = 1)$

Variable ordering: ~~S, R, F, A, L~~

$\{f_0(T), f_{10}(T)\}$

Product of remaining factors: $f_{11} = f_0 \times f_{10}$

$\{f_{11}(T)\}$

Normalize by division:

$$f_{12}(T) = \frac{f_{11}(T)}{\sum_T f_{11}(T)}$$

Optimizing Elimination Order

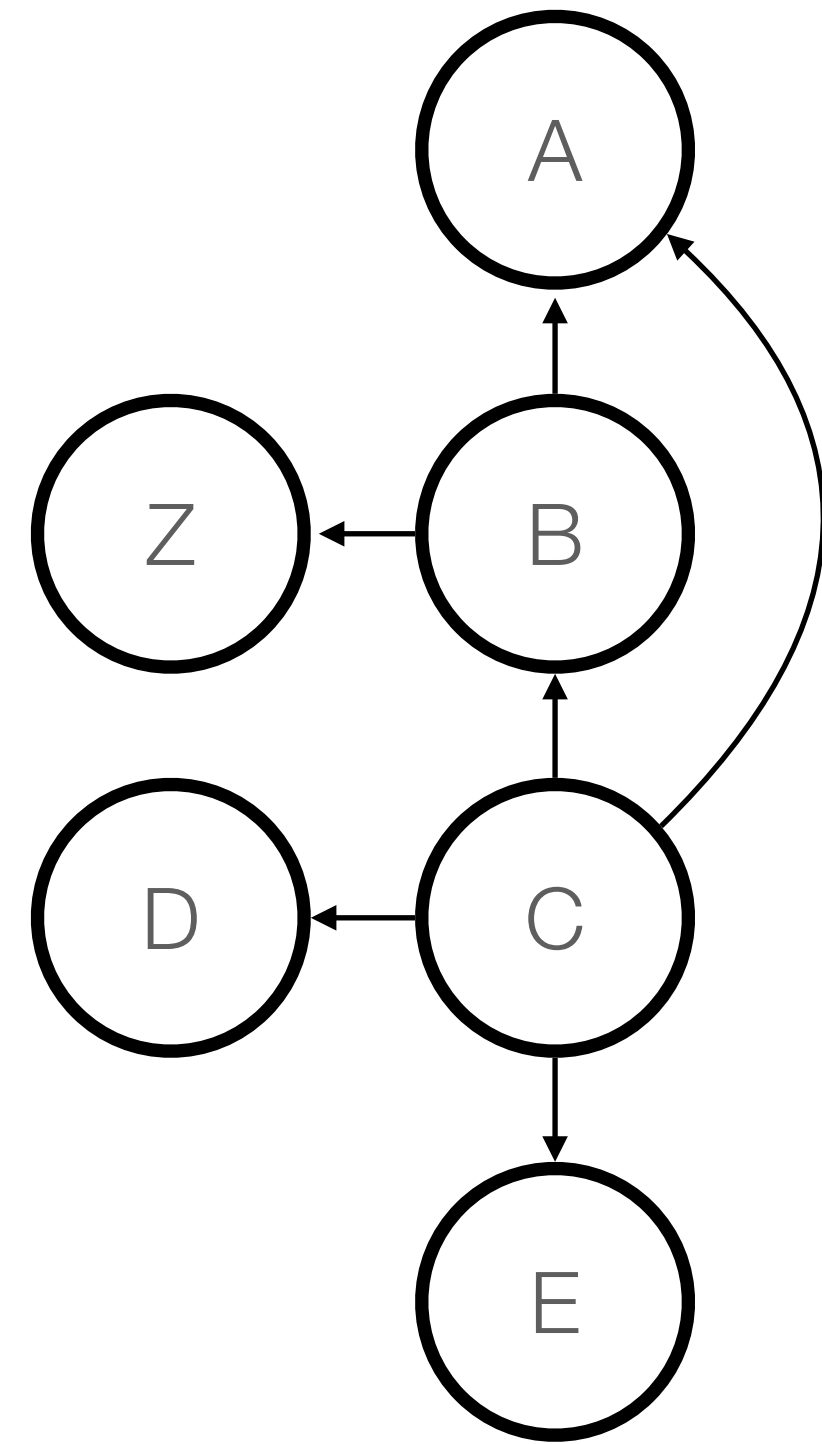
- Variable elimination exploits **efficient sums of products** on a **factored joint distribution**
- The **elimination order** of the variables affects the **efficiency** of the algorithm
- Finding an **optimal** elimination ordering is **NP-hard**
- Rules of thumb for good orderings:
 - **Observations first:** Condition on all of the observed variables first
 - **Min-factor:** At every stage, select the variable that constructs the **smallest new factor**
 - Problem-specific heuristics

Min-Factor Example

Factors:

$\{f_1(Z, B), f_2(B, C), f_3(C), f_4(D, C), f_5(A, B, C), f_6(E, C)\}$

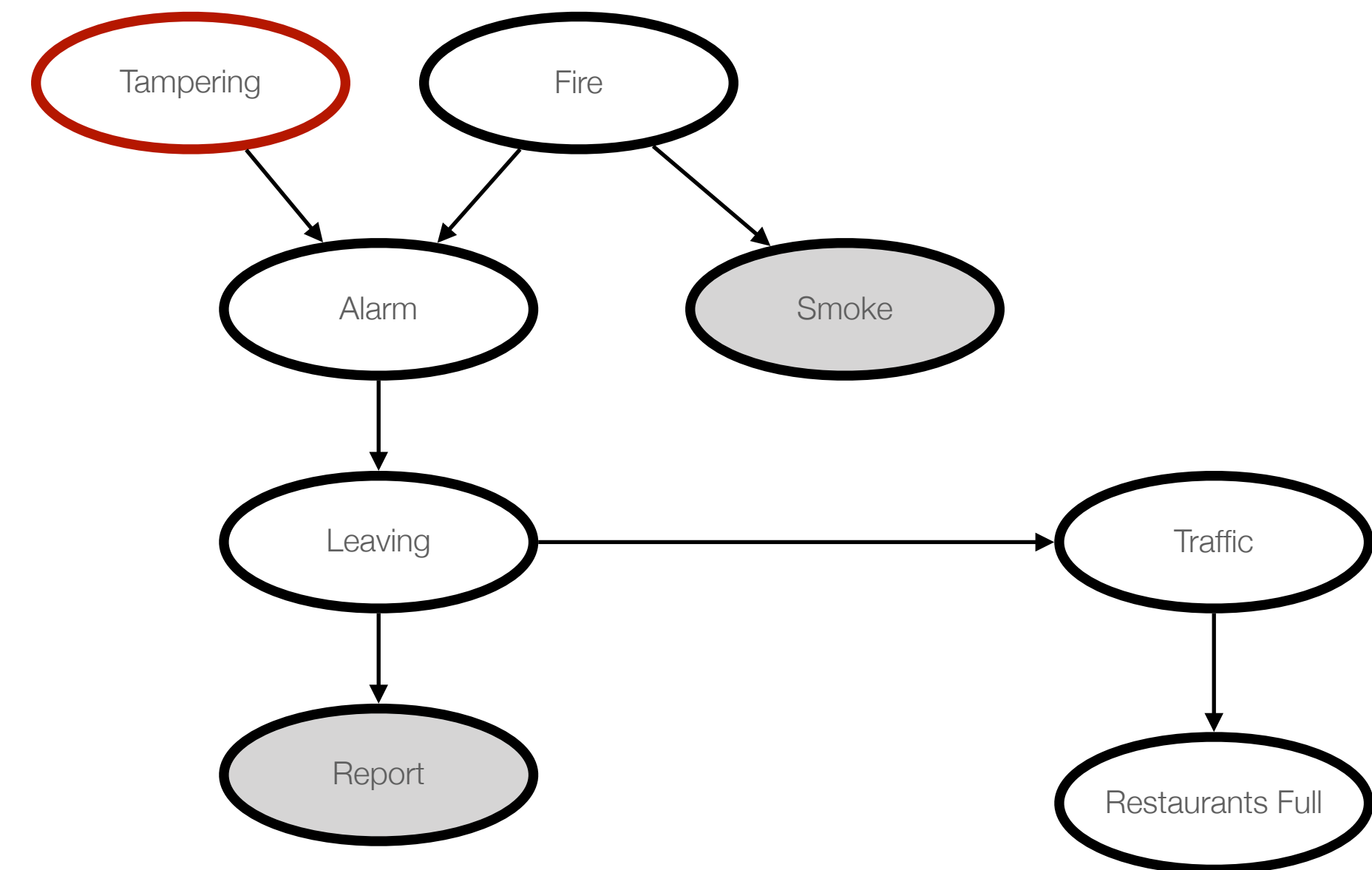
- Which variable creates the **largest** new factor when it is eliminated?
 - **C**: Remove $f_2(B, C), f_3(C), f_4(D, C), f_5(A, B, C), f_6(E, C)$,
Add $f_7(A, B, D, E)$
- Which variable creates the **smallest** new factor when it is eliminated?
 - **Z**: Remove $f_1(Z, B)$, add $f_7(B)$
 - (**E** would also work)
 - Number of **rows** is what matters, not number of arguments
 - (But if every variable has the same size domain, then these are equivalent)



Bayes Net: E has parent C, D has parent C, B has parent C, Z has parent B, A has parents B and C

Optimization: Pruning

- The structure of the graph can allow us to **drop leaf nodes** that are **neither observed nor queried**
 - Summing them out for **free**
- We can **repeat** this process:



Optimization: Preprocessing

Finally, if we know that we are always going to be observing and/or querying the same variables, we can **preprocess** our graph; e.g.:

1. **Precompute** the **joint distribution** of all the variables we will observe and/or query
2. **Precompute conditional distributions** for our exact queries

Summary

- **Variable elimination** is an algorithm for answering **queries** based on a **belief network**
- Operates by using three **operations** on **factors** to reduce graph to a single posterior distribution
 1. Conditioning
 2. Multiplication
 3. Summing out
 4. (Once only): Normalization
- **Distributes** operations more efficiently than taking full product and then summing out
 - **Optimal** order of operations is **NP-hard** to compute
- Additional **optimization** techniques: heuristic ordering, pruning, precomputation