Uncertainty and knowledge

- All knowledge representation formalism and problem solving mechanisms that we have seen until now are based on the following assumptions:
  - All facts can be evaluated to true or false
  - All the facts needed to solve the problem are available
  - The decision mechanism, if applied, always obtains a conclusion that is true
- This only happens in a perfect world

Uncertainty and knowledge

- In practice we make decision without knowing all the facts and with incomplete/heuristic decision mechanisms
- Incomplete knowledge
  - It is impossible to include all the facts that represent a problem
  - Not all the decision mechanisms to solve problems are known
- Uncertain/Imprecise Knowledge
  - There is not absolute confidence on the veracity of the facts
  - Decision mechanisms are heuristic, the conclusion is not always true
  - Decision mechanisms are used on uncertain data, the conclusion is also uncertain

Reasoning with uncertainty

AI has developed different formalisms that allow to reason under uncertainty and incomplete knowledge, we will talk about two of them:

- Probabilistic models (**Bayesian networks**)
- Possibilistic models (**Fuzzy logic**)

Probabilistic models

- Probabilistic models are based on probability theory
- Probabilities are used to model our belief on the probability distribution of the values of a fact
- Each fact has associated a probability distribution (the model) that is used for reasoning
- The probability of a fact could be modified by our belief on the values of other related facts

Probabilistic Decisions

Will you take your umbrella tomorrow?

Probabilistic Decisions

Will you take your umbrella tomorrow?

It never rains in Barcelona (Yes: 10 %)

Probabilistic Decisions

Will you take your umbrella tomorrow?

It never rains in Barcelona (Yes: 10 %)

The weather forecast is cloudy skies

Probabilistic Decisions

Will you take your umbrella tomorrow?

It never rains in Barcelona (Yes: 10 %)

The weather forecast is cloudy skies

May be I will, may be I won't (Yes: 50 %)

Probabilistic Decisions

Will you take your umbrella tomorrow?

It never rains in Barcelona (Yes: 10 %)

The weather forecast is cloudy skies

May be I will, may be I won't (Yes: 50 %)

Today the streets are wet

Probabilistic Decisions

Will you take your umbrella tomorrow?

It never rains in Barcelona (Yes: 10 %)

The weather forecast is cloudy skies

May be I will, may be I won't (Yes: 50 %)

Today the streets are wet

Better if I take my umbrella (Yes: 95 %)

Probability theory

- The basic element of probability theory is the **random variable**
- A random variable has a domain of values, we have **boolean**, **discrete** o **continuous** random variables
- A **logic proposition** is defined as a formula in propositional or predicate calculus
- A logic proposition has associated a random variable that represents the degree of belief on its values

Probability theory

- A random variable has associated a **probability distribution**
- The expression of the probability distribution will depend on the kind of random variable (Discrete: Binomial, Multinomial, ..., Continuous: Normal, χ^2, ...)
- We will only talk about discrete random variables
- The union of random variables can be described by the **joint probability distribution**

Probability theory

- $P(a)$ is the probability that the proposition (random variable) A has the value a. For example, the proposition *Smoker* has two values $\{smokes, \neg smokes\}$, $P(\neg smokes)$ is the probability of the proposition $Smoker = \neg smokes$
- $P(A)$ is the **probability vector** of all the possible values of the proposition A

Probability theory

- The **a priori probability** of a proposition ($P(a)$) is defined as the degree of belief on the proposition if we have no other information
- The **a posteriori or conditional probability** ($P(a|b)$) is defined as the degree of belief on a propositions after observing other proposition associated to it
- A posteriori probability can be defined from a priori probabilities as:

$$P(a|b) = \frac{P(a \wedge b)}{P(b)}$$

- This formula can be transformed as we will see into the **product rule**:

$$P(a \wedge b) = P(a|b)P(b) = P(b|a)P(a)$$

Probability axioms

The probability axioms are the framework that gives the constraints to what we can believe and infer

- Probability value is in the interval $[0, 1]$

$$0 \leq P(a) \leq 1$$

- A *true* proposition has probability 1 and a *false* proposition has probability 0

$$P(true) = 1 \quad P(false) = 0$$

- The probability of the disjunction of two propositions is obtained by the formula

$$P(a \vee b) = P(a) + P(b) - P(a \wedge b)$$

Probabilistic inference

The laws of probability allow to have different kinds of inference

- **Marginalization:** Probability of an atomic proposition independently of the value of the rest of propositions

$$P(X) = \sum_z P(X, z)$$

- **Conditional probabilities:** Probability of a proposition given the values of some propositions and independent of the rest of propositions (from the product rule)

$$P(X|e) = \alpha \sum_y P(X, e, y)$$

The value α is a normalization term that corresponds to common factors and that makes the sum of the probabilities equal to 1

Probabilistic inference: example

Lets consider a problem with the propositions
$Smokes = \{smokes, \neg smokes\}$, $Gender = \{man, woman\}$,
$Emphysema = \{emphysema, \neg emphysema\}$

	emphysema		$\neg$*emphysema*	
	man	*woman*	*man*	*woman*
smokes	0.2	0.1	0.05	0.05
$\neg$*smokes*	0.02	0.02	0.23	0.33

Probabilistic inference: example

$$
\begin{aligned}
P(\mathit{emphysema} \wedge \mathit{man}) &= 0{,}2 + 0{,}02 \\
P(\mathit{smokes} \vee \mathit{woman}) &= 0{,}2 + 0{,}1 + 0{,}05 + 0{,}05 + 0{,}02 + 0{,}33 \\
P(\mathit{Smoker}|\mathit{emphysema}) &= \langle P(\mathit{smokes}, \mathit{emphysema}, \mathit{man}) \\
&\quad + P(\mathit{smokes}, \mathit{emphysema}, \mathit{woman}), \\
&\quad P(\neg \mathit{smokes}, \mathit{emphysema}, \mathit{man}) \\
&\quad + P(\neg \mathit{smokes}, \mathit{emphysema}, \mathit{woman}) \rangle \\
&= \alpha \langle 0{,}3, 0{,}04 \rangle \\
&= \langle 0{,}88, 0{,}12 \rangle
\end{aligned}
$$

Probabilistic inference: Problems

- To compute these inferences it is required to store and search the joint probability distribution of all the propositions
- Assuming binary propositions the cost in space and time is $O(2^n)$ being n the number of propositions
- For any real problem this is impracticable
- It is needed a mechanism that allows to reduce the inference computational cost

Probabilistic independence

- Usually not all the propositions from a problem are related to each other
- They have the property of **probabilistic independence**
- This means that some propositions do not have influence over others and their probabilities can be expressed as:

$$P(X|Y) = P(X); \quad P(Y|X) = P(Y); \quad P(X,Y) = P(X)P(Y)$$

- Because of this property the joint probability distribution can be expressed in a more compact way, reducing the computational complexity

Bayes rule

- The product rule can been expressed as:

$$P(X, Y) = P(X|Y)P(Y) = P(Y|X)P(X)$$

- This derives to the **Bayes rule**

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

- This rule and the probabilistic independence property is the basis of probabilistic reasoning and it will allow us to propagate the probabilities of propositions to others

Bayes rule + Independence

- Assuming that we can exhaustively estimate all the probabilities that are related to the values of the variable Y the Bayes rules can be rewritten as:

$$P(Y|X) = \alpha P(X|Y)P(Y)$$

- Assuming conditional independence between two variables we have that:

$$P(X, Y|Z) = P(X|Z)P(Y|Z)$$

- so:

$$P(Z|X, Y) = \alpha P(X, Y|Z)P(Z) = \alpha P(X|Z)P(Y|Z)P(Z)$$

- This means that we can decompose the computation of joint probabilities on independent computations

Bayesian networks

- If we know the independence relations among variables we can simplify the computation of their combination and also their representation
- **Bayesian networks** is a formalism for representing these independence relations
- A bayesian network is an **acyclic directed graph** that stores probabilistic information in its nodes that represents the influence of the ancestors of a node ($P(X_i|parent(X_i))$) on its probability distribution
- The intuitive meaning of an edge between two nodes X and Y is that the variable X has influence over Y probability
- The probabilities represented by the network describe the joint probability distribution of all the variables

Bayesian networks: example (1)

- We want to find out the probability of having a heart attack of a person
- We know that this probability is determined by four variables: the practice of exercise, adequate diet, blood pressure and smoking
- We also know that blood pressure depends directly of exercising and diet, that are independent variables, and smoking is independent of the rest of variables
- This knowledge allows us to create a dependency network among the variables
- Our knowledge about the domain allows us to estimate the probabilities of each independent variable and their influence to the dependent variables

Bayesian networks: example (2)

Exercise	P(E)
yes	0.1
no	0.9

Diet	P(D)
equilibrated	0.4
not equlibrated	0.6

Diet	Exer	P(P=high)	P(P=normal)
eq	yes	0.01	0.99
no eq	yes	0.2	0.8
eq	no	0.25	0.75
no eq	no	0.7	0.3

Smoker	P(S)
yes	0.4
no	0.6

Blod P	Smok	P(A=yes)	P(A=no)
high	yes	0.8	0.2
normal	yes	0.6	0.4
high	no	0.7	0.3
normal	no	0.3	0.7

Bayesian networks - Joint probability distribution

- Each node in the network has the probability distribution of the node given its parents
- This allows to factorize the joint probability distribution transforming its expression to a product of independent conditional probabilities

$$P(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | parents(x_i))$$

Bayesian networks - Joint probability distribution - example

$$
\begin{aligned}
&P(Attack = yes \wedge Pressure = high \wedge Smoker = yes \\
&\wedge\ Exercise = yes \wedge Diet = equil) \\
&= \\
&P(Attack = yes | Pressure = high, Smoker = yes) \\
&P(Pressure = high | Exercise = yes, Diet = equil) \\
&P(Smoker = yes) P(Exercise = yes) P(Diet = equil) \\
&= 0{,}8 \times 0{,}01 \times 0{,}4 \times 0{,}1 \times 0{,}4 \\
&= 0{,}000128
\end{aligned}
$$

Design of bayesian networks

- The properties of bayesian networks give some ideas about how to build them. Considering that (by the product rule):

$$P(x_1, x_2, \ldots, x_n) = P(x_n|x_{n-1}, \ldots, x_1)P(x_{n-1}, \ldots, x_1)$$

- Iterating the process we have that:

$$\begin{aligned} P(x_1, \ldots, x_n) &= P(x_n|x_{n-1}, \ldots, x_1)P(x_{n-1}|x_{n-2}, \ldots, x_1) \\ &\quad \cdots P(x_2|x_1)P(x_1) \\ &= \prod_{i=1}^{n} P(x_i|x_{i-1}, \ldots, x_1) \end{aligned}$$

- This is the **chain rule**

Design of bayesian networks

- Given that properties we can say that if $parents(X_i) \subseteq \{X_{i-1}, \ldots, X_1\}$, then:

$$P(X_i|X_{i-1}, \ldots, X_1) = P(X_i|parents(X_i))$$

- This means that a bayesian network is a correct representation of a domain only if each node is conditional independent of its ancestors given its parents
- The parents of a variable X_i must be the variables $X_1, \ldots X_{i-1}$ that have a direct influence over X_i

Cost of the representation

- The cost of representing a joint probability distribution of n binary variables is $O(2^n)$
- Bayesian networks allow a more compact representation because of the factorization of the joint probability distribution
- Assuming that each node has no more that k parents ($k \ll n$), a node needs 2^k to represent the influence of its parents, then the space necessary is $O(n2^k)$.
- For example, with 10 variables and assuming 3 parents we have 80 instead of 1024, with 100 variables and assuming 5 parents we have 3200 instead of approximately 10^{30}

Inference in Bayesian networks

- The goal of probabilistic inference is to compute the a posteriori probability distribution of a set of variables given the observation of an event (observed values for a subset of variables)
- X is the variable we want to know its probability distribution
- $\mathbf{E}$ is the set of variables which their value is known $E_1, \ldots, E_n$
- $\mathbf{Y}$ is the set of variables not observed $Y_1, \ldots, Y_n$ (hidden variables)
- $\mathbf{X} = \{X\} \cup \mathbf{E} \cup \mathbf{Y}$ is the set of all variables
- We want to compute $P(X|\mathbf{e})$ (e = observed values for E)

Exact inference

- **Inference by enumeration**: Any conditional probability can be calculated as the sum of all possible cases from the joint probability distribution

$$P(X|\mathbf{e}) = \alpha P(X, \mathbf{e}) = \alpha \sum_{y} P(X, \mathbf{e}, \mathbf{y})$$

- The bayesian network allows to factorize the probability distribution and obtain an expression that can be evaluated in a simpler way
- For instance we can compute using the bayesian network from the example the probability of being a smoker if we have had a heart attack and do not exercise

$$P(Smoker|Attack = yes, Exercise = no)$$

Exact inference: Example

The joint probability distribution of the network is:

$$P(E, D, S, P, A) = P(A|P, S)P(S)P(P|E, D)P(E)P(D)$$

We have to compute $P(S|A = yes, E = no)$, so

$$\begin{aligned} P(S|A = y, E = n) &= \alpha P(S, A = y, E = n) \\ &= \alpha \sum_{D \in \{e, \neg e\}} \sum_{P \in \{h, n\}} P(E = n, D, P, S, A = y) \\ &= \alpha P(E = n)P(S) \sum_{D \in \{e, \neg e\}} P(D) \sum_{P \in \{h, n\}} P(P|E = n, D)P(A = y|P, S) \end{aligned}$$

Exact inference: Example

If we enumerate all the possibilities and sum them up using the joint probability distribution we have that:

$$
\begin{aligned}
&P(Smoker|Attack = yes, Exercise = no) \\
&= \alpha\langle 0{,}9 \cdot 0{,}4 \cdot (0{,}4 \cdot (0{,}25 \cdot 0{,}8 + 0{,}75 \cdot 0{,}6) + 0{,}6 \cdot (0{,}7 \cdot 0{,}8 + 0{,}3 \cdot 0{,}6)) \\
&\qquad 0{,}9 \cdot 0{,}6 \cdot (0{,}4 \cdot (0{,}25 \cdot 0{,}7 + 0{,}75 \cdot 0{,}3) + 0{,}6 \cdot (0{,}7 \cdot 0{,}7 + 0{,}3 \cdot 0{,}3))\rangle \\
&= \alpha\langle 0{,}253, 0{,}274\rangle \\
&= \langle 0{,}48, 0{,}52\rangle
\end{aligned}
$$

Variable elimination algorithm

- The **variable elimination algorithm** tries to solve the duplication of computations that makes the inference by enumeration
- The algorithm uses methods from dynamic programming storing intermediate computations for each variable to reuse them (*factors*)
- The computation of the probabilities of a query is performed by evaluating the expression of the joint probability distribution from left to right
- The *factors* of each variable are summed up when needed
- The advantage of this algorithm is that the variables not relevant to the computation are constant factors and are eliminated

Variable elimination algorithm

Function: Variable Elimination(X, e, rb)

factors $\leftarrow$ []
vars $\leftarrow$ REVERSE(VARS(rb))
foreach *var* $\in$ *vars* **do**
    factors $\leftarrow$ concatenate(factors,COMPUTE-FACTOR(var,e))
    if *var is hidden variable* **then**
        factors $\leftarrow$ PRODUCT-AND-SUM(var,factors)
    end
end
return *NORMALIZE(PRODUCT(factors))*

- COMPUTE-FACTOR generates the factor corresponding to a variable in the joint probability distribution
- PRODUCT-AND-SUM multiplies the factors and sums a hidden variable
- PRODUCT multiplies a set of factors

Variable elimination algorithm - Factors

- A factor corresponds to the joint probability distribution of a set of variables given the hidden variables
- It is represented by a table where each combination of hidden variables is associated to the probabilities of the variables of the factor

$f_X(Y, Z) =$

Y	Z	
T	T	0.2
T	F	0.4
F	T	0.8
F	F	0.6

- The factors have two operators: sum and product

Sum of factors

- The sum is applied to a factor over a hidden variable of the factor. The result is a reduced matrix where the rows with the same values have been accumulated

$$f_{X\overline{Z}}(Y) = \sum_Z f_X(Y,Z) =$$

Y	
T	0.6
F	1.4

- It is the same as an column aggregation operation in a database

Product of factors

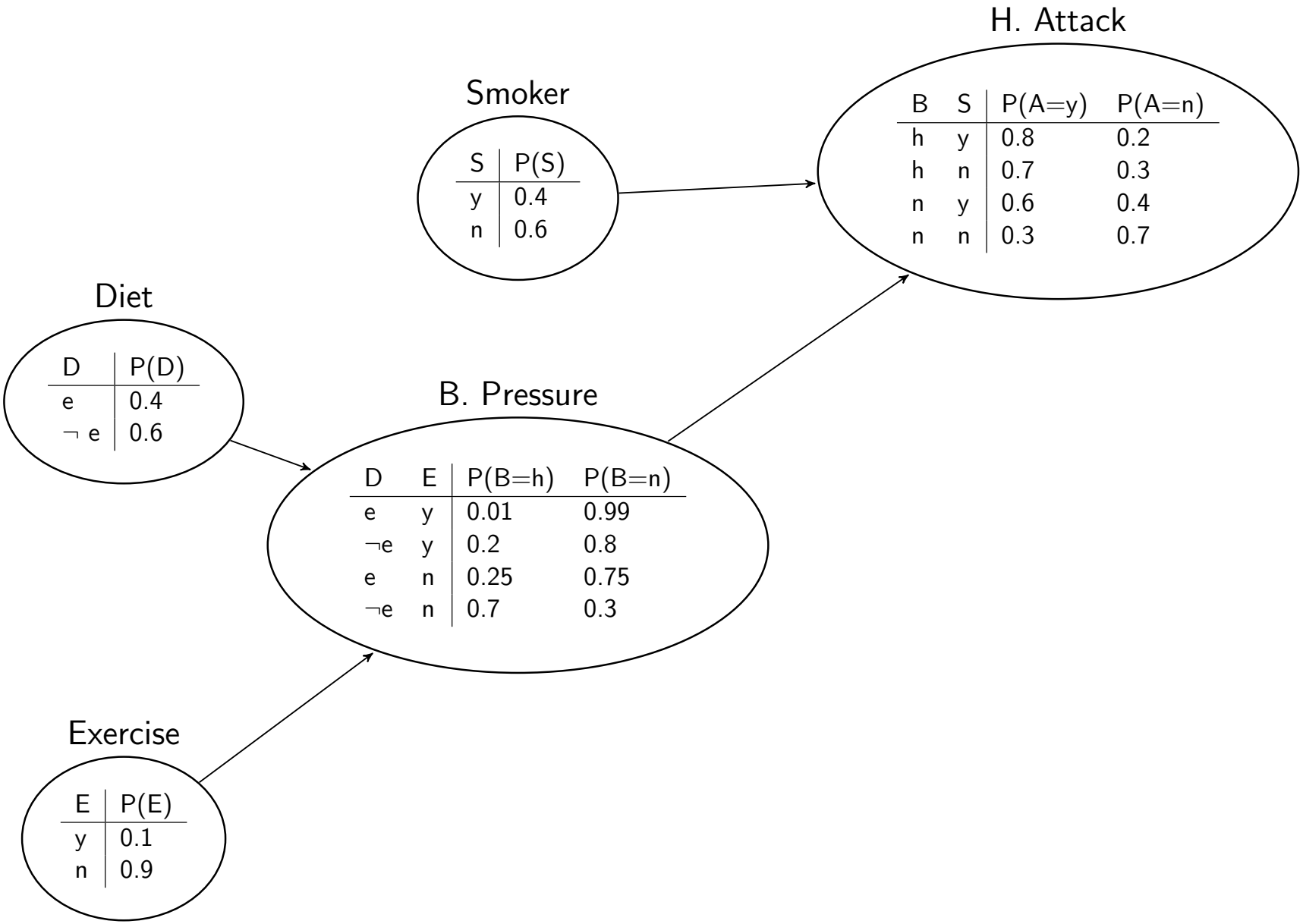
- The product of factors allows to join factors using the common hidden variables

$$f_{X_1X_2}(Y,W,Z) = f_{X_1}(Y,Z) \times f_{X_2}(Z,W) =$$

Y	Z		Z	W		Y	Z	W	
T	T	0.2	T	T	0.3	T	T	T	$0{,}2 \times 0{,}3$
T	F	0.8	T	F	0.7	T	T	F	$0{,}2 \times 0{,}7$
F	T	0.4	F	T	0.1	T	F	T	$0{,}8 \times 0{,}1$
F	F	0.6	F	F	0.9	T	F	F	$0{,}8 \times 0{,}9$
						F	T	T	$0{,}4 \times 0{,}3$
						F	T	F	$0{,}4 \times 0{,}7$
						F	F	T	$0{,}6 \times 0{,}1$
						F	F	F	$0{,}6 \times 0{,}9$

- It is the same that a join operation in databases but multiplying the values of the columns

Variable elimination algorithm - example



S	P(S)
y	0.4
n	0.6

B	S	P(A=y)	P(A=n)
h	y	0.8	0.2
h	n	0.7	0.3
n	y	0.6	0.4
n	n	0.3	0.7

D	P(D)
e	0.4
¬ e	0.6

D	E	P(B=h)	P(B=n)
e	y	0.01	0.99
¬e	y	0.2	0.8
e	n	0.25	0.75
¬e	n	0.7	0.3

E	P(E)
y	0.1
n	0.9

Variable elimination algorithm - example

Lets compute again $P(Smoker|Attack = yes, Exercise = no)$ from the joint probability distribution:

$$P(E, D, S, P, A) = P(A|B, S)P(S)P(B|E, D)P(E)P(D)$$

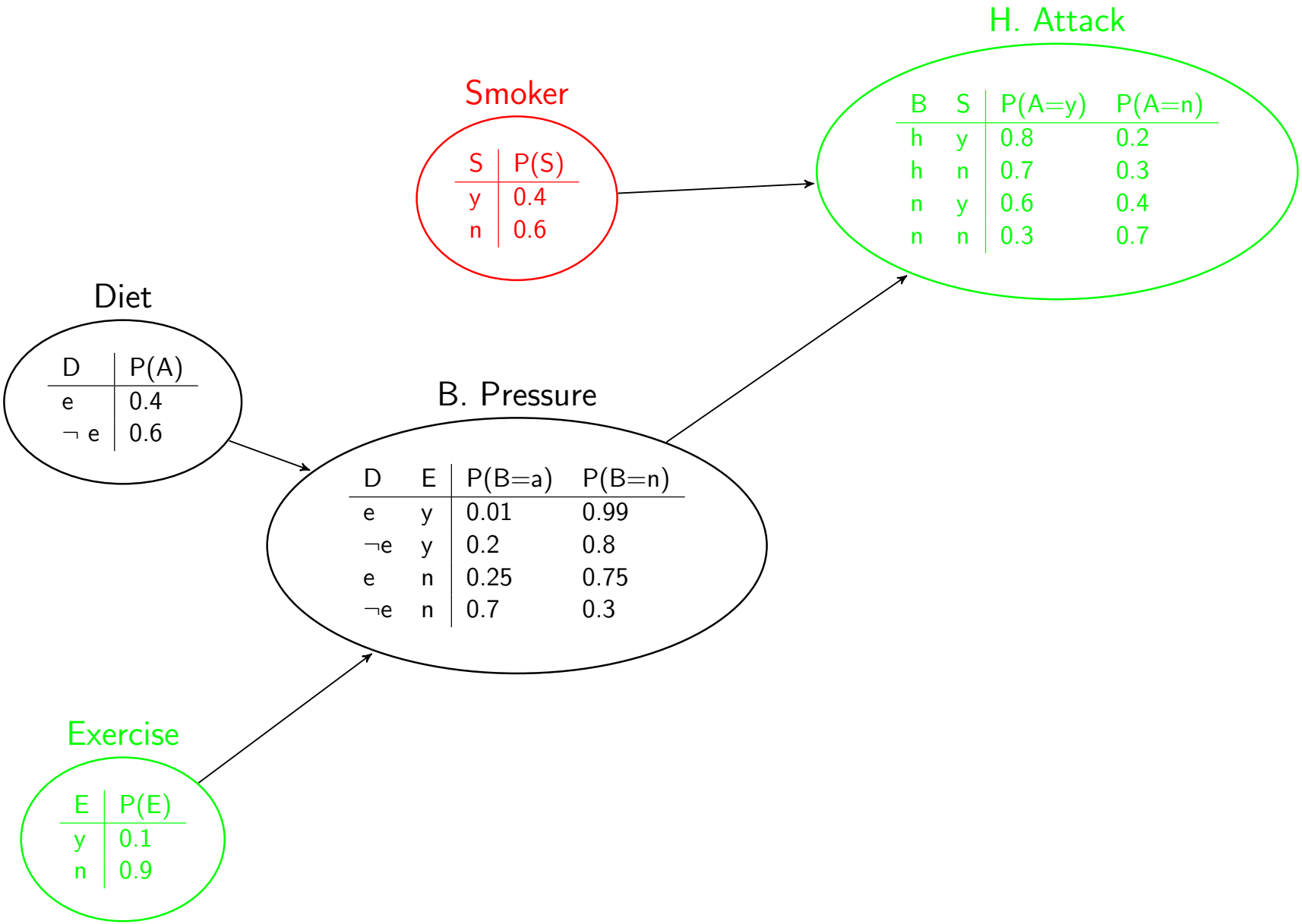
We have to compute $P(S|A = yes, E = no)$, so we have

$$\begin{aligned} P(S|H = y, E = n) &= \alpha P(A = y, S, E = n) \\ &= \alpha \sum_{D \in \{e, \neg e\}} \sum_{B \in \{h, n\}} P(E = n, D, B, S, A = y) \end{aligned}$$

This time we will not take common factors to follow the algorithm

$$\alpha P(E = n) \sum_{D \in \{e, \neg e\}} P(D) \sum_{B \in \{h, n\}} P(B|E = n, D)P(S)P(A = y|B, S)$$

Variable elimination algorithm - example



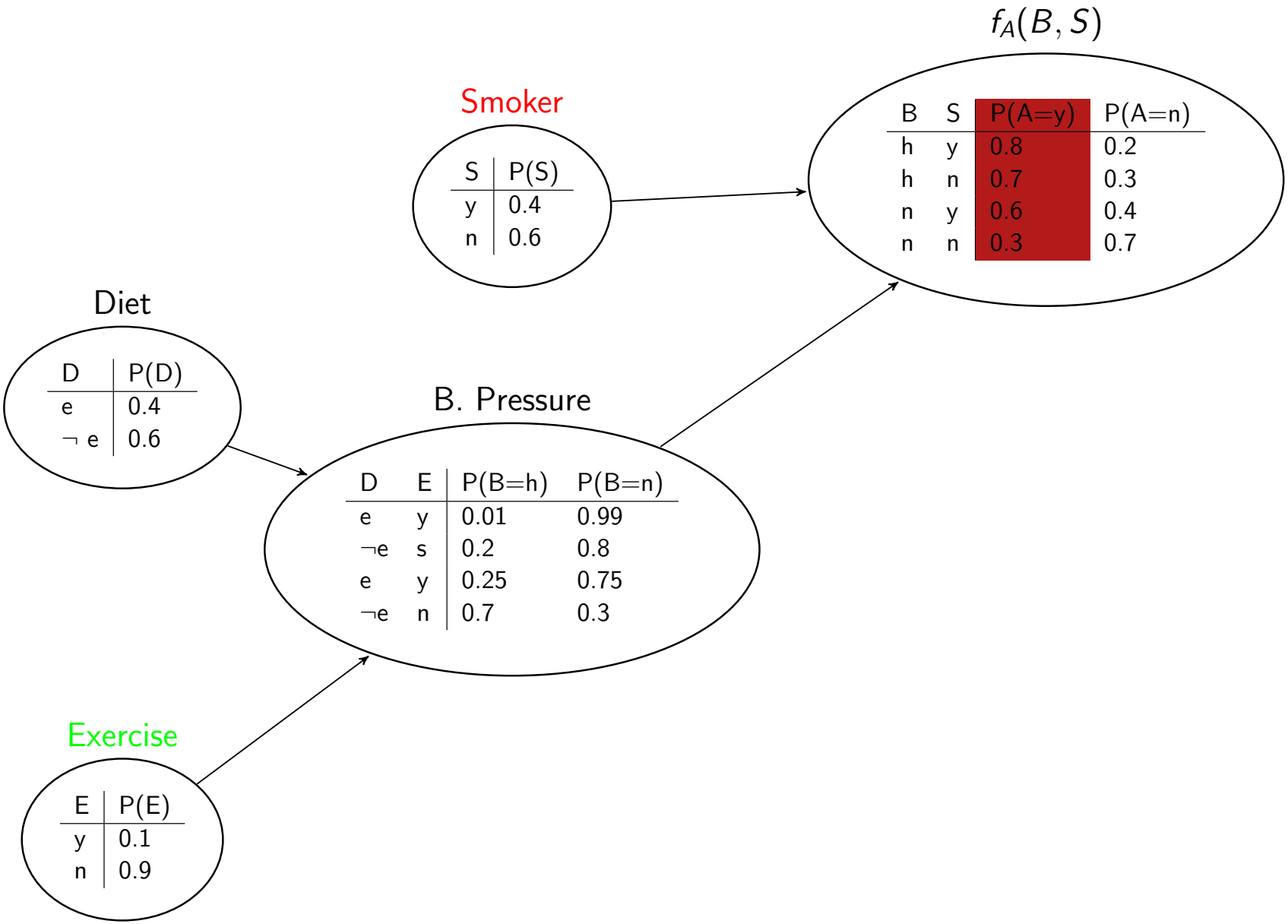
Variable elimination algorithm - example

The algorithm begins computing the factor for the variable *Attack* ($P(A = y|B, S)$), this variable has the value yes, depends on the variables *Blood Pressure* and *Smoker*

$f_A(B, S)=$

B	S	
h	y	0.8
h	n	0.7
n	y	0.6
n	n	0.3

Variable elimination algorithm - example



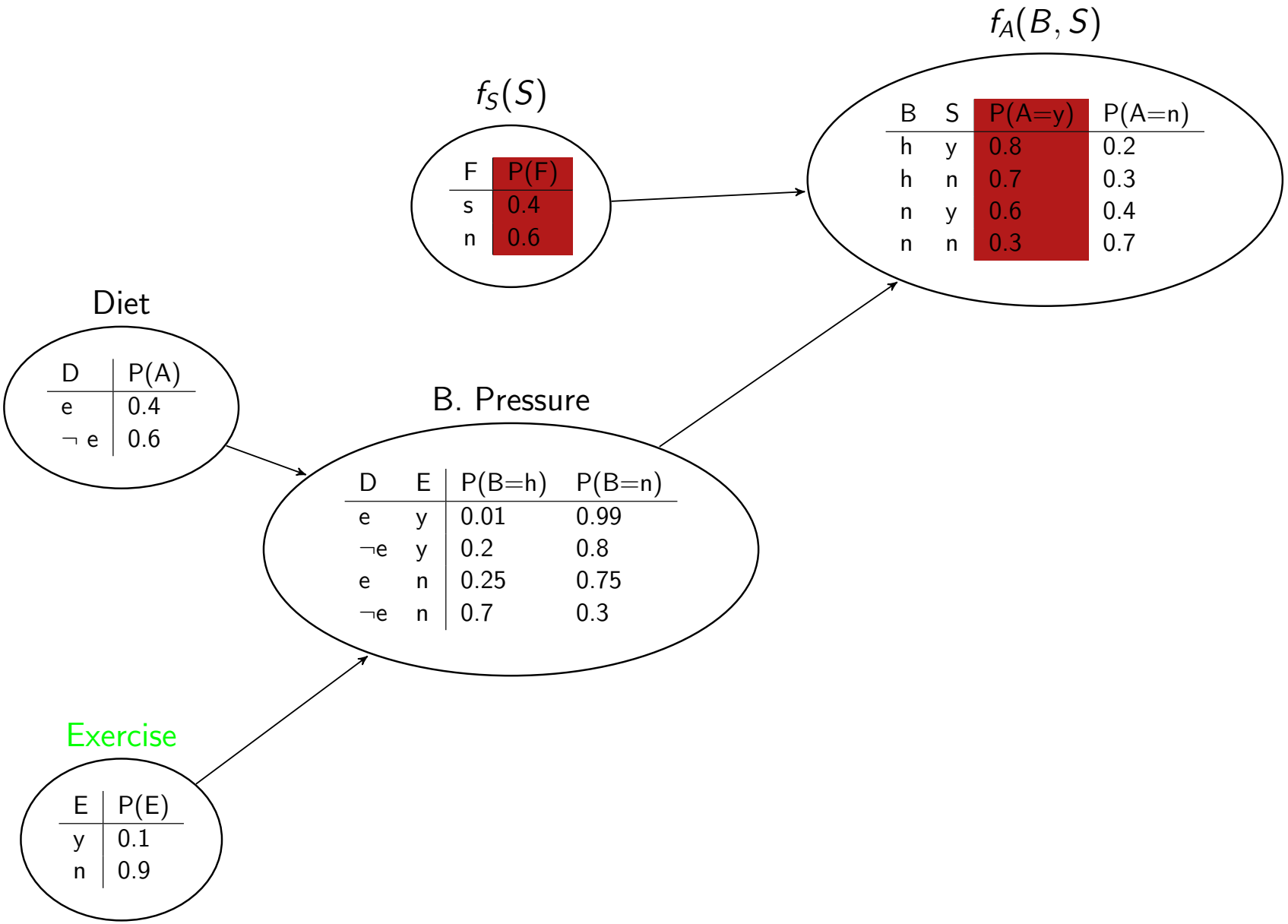
Variable elimination algorithm - example

The variable smokes ($P(S)$) has no dependencies, because it is the variable we are querying the factor includes all its values

$f_S(S)=$

S	
y	0.4
n	0.6

Variable elimination algorithm - example



F	P(F)
s	0.4
n	0.6

B	S	P(A=y)	P(A=n)
h	y	0.8	0.2
h	n	0.7	0.3
n	y	0.6	0.4
n	n	0.3	0.7

D	P(A)
e	0.4
¬ e	0.6

D	E	P(B=h)	P(B=n)
e	y	0.01	0.99
¬e	y	0.2	0.8
e	n	0.25	0.75
¬e	n	0.7	0.3

E	P(E)
y	0.1
n	0.9

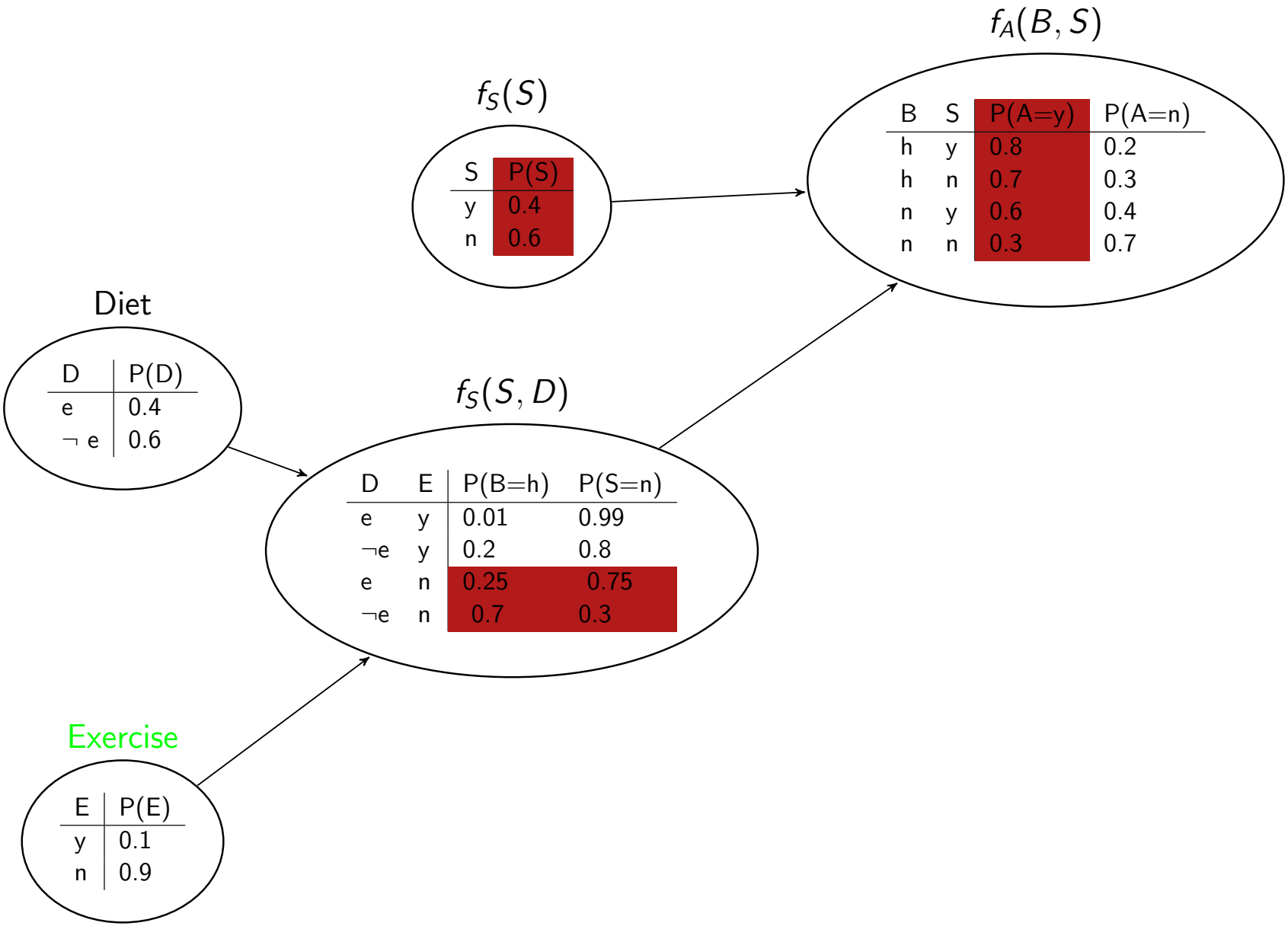
Variable elimination algorithm - example

The variable *Blood pressure* ($P(B|E = n, D)$), depends on the variable *Exercise* that has a value of `no` and *Diet*. This is a hidden variable, so it has to be computed for all its values

$f_B(B, D)=$

B	D	
h	e	0.25
h	¬e	0.7
n	e	0.75
n	¬ e	0.3

Variable elimination algorithm - example



Variable elimination algorithm - example

Because the variable *Blood Pressure* is a hidden variable we have to accumulate all the factors that we have computed

$$f_B(B, D) \times f_S(S) \times f_A(B, S)$$

$$f_{BA}(B, S) = f_S(S) \times f_A(B, S) =$$

B	S	
h	y	0.8×0.4
h	n	0.7×0.6
n	y	0.6×0.4
n	n	0.3×0.6

Variable elimination algorithm - example

$$f_{SAB}(B,S,D) = f_{SA}(B,S) \times f_B(B,D) =$$

B	S	D	
h	y	e	$0.8 \times 0.4 \times 0.25$
h	y	$\neg$e	$0.8 \times 0.4 \times 0.7$
h	n	e	$0.7 \times 0.6 \times 0.25$
h	n	$\neg$e	$0.7 \times 0.6 \times 0.7$
n	y	e	$0.6 \times 0.4 \times 0.75$
n	y	$\neg$e	$0.6 \times 0.4 \times 0.3$
n	n	e	$0.3 \times 0.6 \times 0.75$
n	n	$\neg$e	$0.3 \times 0.6 \times 0.3$

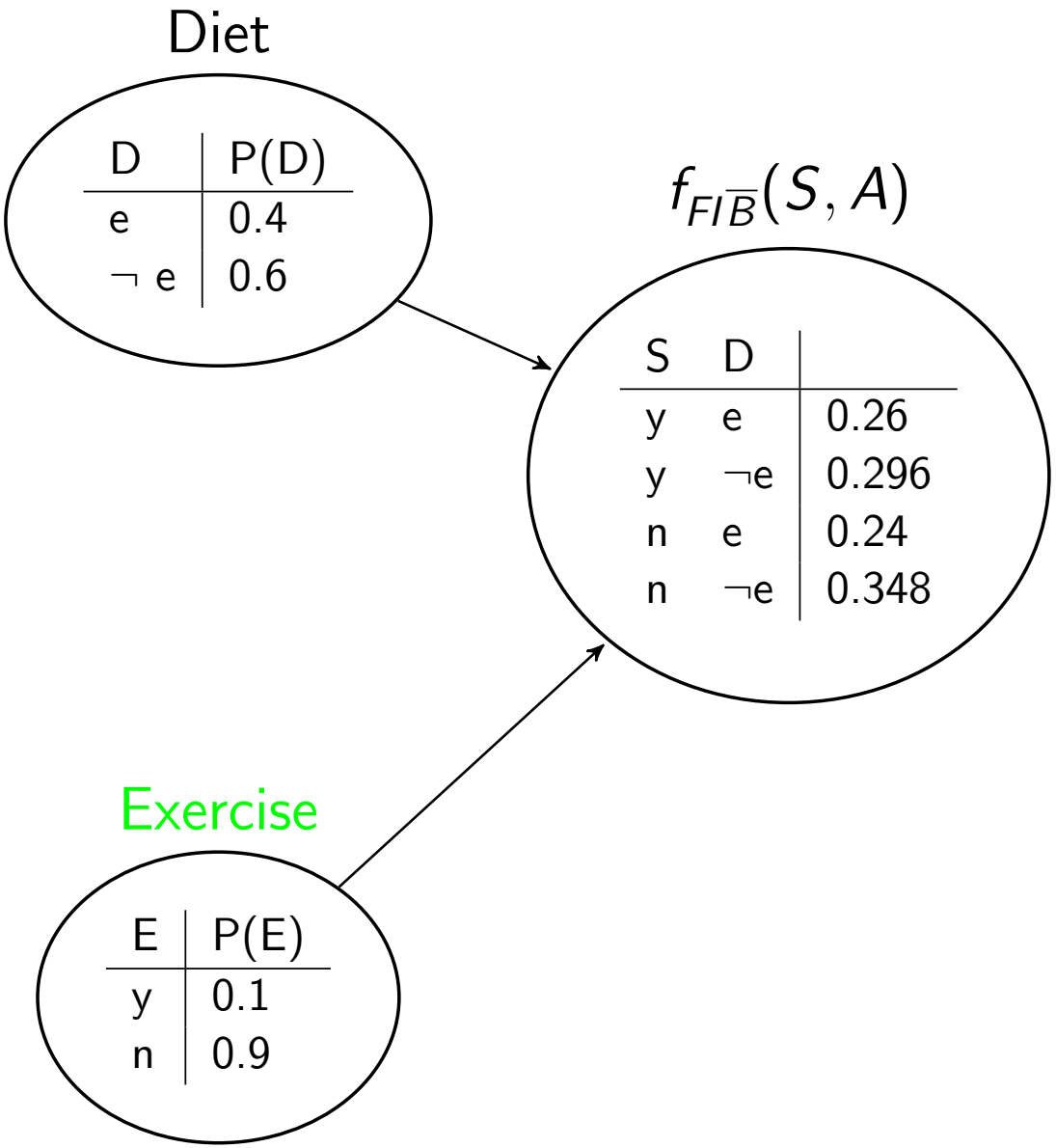
Variable elimination algorithm - example

And now we sum up over all the values of the variable B to obtain the factor corresponding to the variable *Blood Pressure*

$$f_{SA\overline{B}}(S, D) = \sum_{B \in \{h,n\}} f_{SAB}(B, S, D) =$$

S	D	
y	e	$0.8 \times 0.4 \times 0.25 + 0.6 \times 0.4 \times 0.75 = 0.26$
y	$\neg$e	$0.8 \times 0.4 \times 0.7 + 0.6 \times 0.4 \times 0.3 = 0.296$
n	e	$0.7 \times 0.6 \times 0.25 + 0.3 \times 0.6 \times 0.75 = 0.24$
n	$\neg$e	$0.7 \times 0.6 \times 0.7 + 0.3 \times 0.6 \times 0.3 = 0.348$

Variable elimination algorithm - example



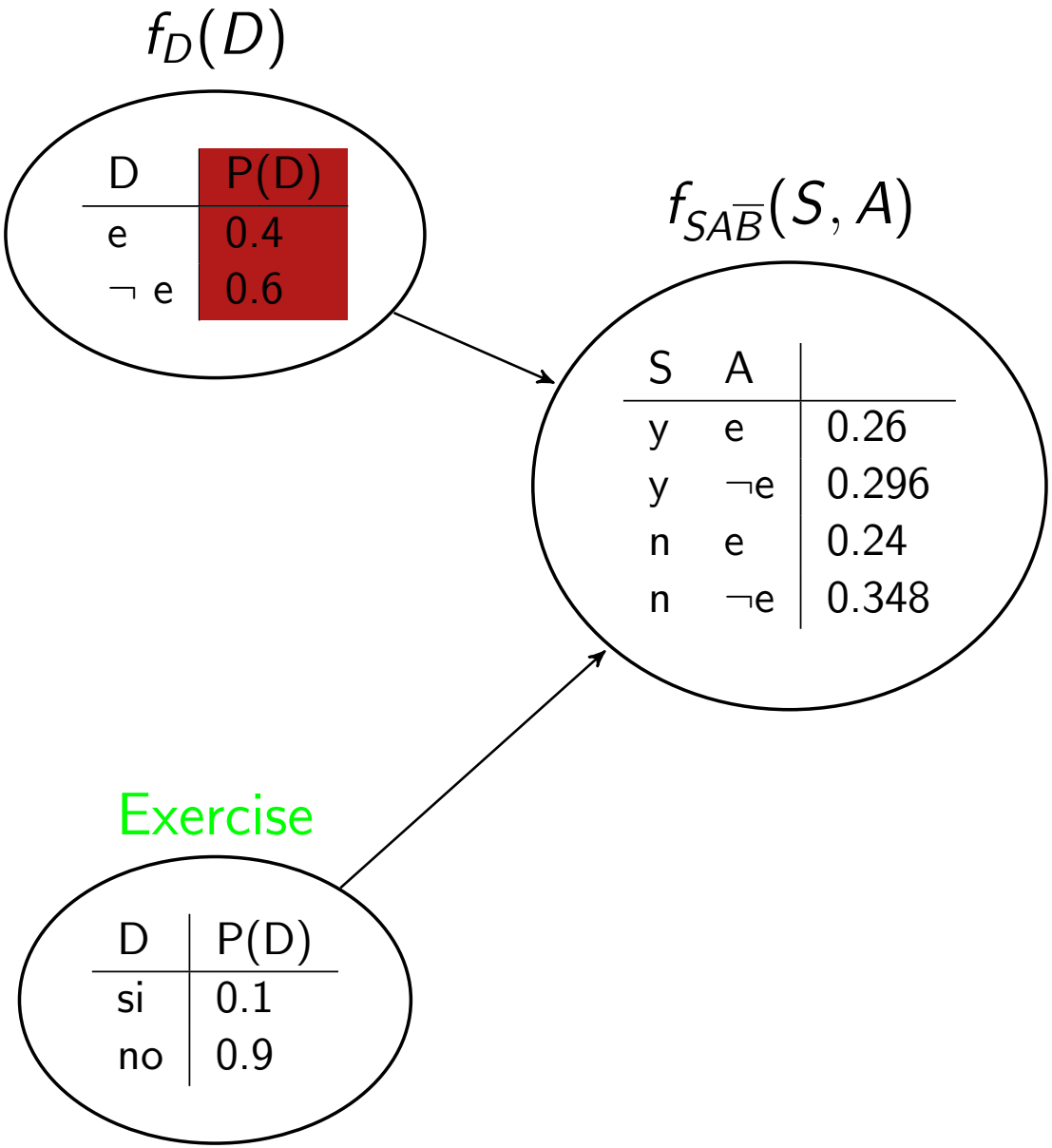
Variable elimination algorithm - example

The factor of the variable *Diet* ($P(D)$) is independent, because it is a hidden variable. We use all the possibilities

$f_D(D)=$

D	
e	0.4
$\neg$e	0.6

Variable elimination algorithm - example



Variable elimination algorithm - example

Now we accumulate all the computed factors

$$f_{DSA\overline{B}}(S,D) = f_D(D) \times f_{SA\overline{P}}(S,D) =$$

S	D	
y	e	$0.26 \times 0.4 = 0.104$
y	$\neg$e	$0.296 \times 0.6 = 0.177$
n	e	$0.24 \times 0.4 = 0.096$
n	$\neg$e	$0.348 \times 0.6 = 0.208$

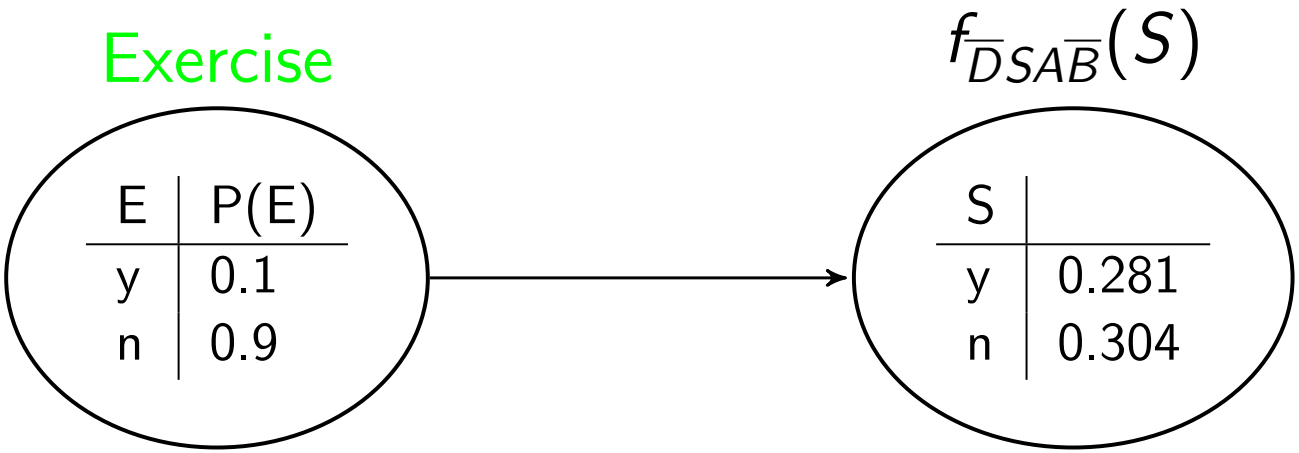
Variable elimination algorithm - example

And now we sum up all the values of the variable D to obtain the factor corresponding to the variable *Diet*

$$f_{\overline{D}SA\overline{B}}(S) = \sum_{D\in\{e,\neg e\}} f_{DSA\overline{P}}(S,D) =$$

S	
y	$0.104 + 0.177 = 0.281$
n	$0.096 + 0.208 = 0.304$

Variable elimination algorithm - example



Variable elimination algorithm - example

And finally the variable *Exercise* ($P(E = n)$) has the value *no* and given that does not depend of the variable *smoker* it can be eliminated, because is a constant factor.
Now, if we normalize to 1:

$$P(S|A = s, E = n) = \begin{array}{c|c} \text{S} & \\ \hline \text{y} & 0.48 \\ \text{n} & 0.52 \end{array}$$

Complexity of exact inference

- The computational complexity of the variable elimination algorithm depends on the size of the bigger factor, that depends on the order of evaluation of the variables and the network topology
- The order of evaluation that we will use is the topological order given the graph
- The worst case complexity of exact inference is NP-hard
- If the bayesian network holds that for each pair of nodes there is only one non directed path (**polytree**) then can be computed on polynomial time
- To compute inference in the general case approximate algorithms based on sampling are used