Uncertain Databases - Querying

Databases and Information Systems

Fabian Panse

panse@informatik.uni-hamburg.de

University of Hamburg

Possible Worlds Semantics

Semantics:

- Step 1: Separate Evaluation in each possible world
- Step 2: Combination of identical results

Result:

- A compactly represented probabilistic database (world schema with single table)

$\Rightarrow$ This semantics is compositional (it can be used to evaluate views)

Problem:

- Difficult to understand for the user

Possible Worlds Semantics - Example

W_1, Pr=0.5

WK	name	age
p1	J.Doe	27
p2	K.Smith	32
p3	J.Ho	28
p4	J.J.Doe	31

W_2, Pr=0.4

WK	name	age
p1	J.Doe	27
p2	K.Smith	32
p3	J.Ho	28
p4	J.J.Doe	41

W_3, Pr=0.1

WK	name	age
p1	J.Doe	27
p2	K.Smith	32
p4	J.J.Doe	41

Possible Worlds Semantics - Example

W_1, Pr=0.5

WK	name	age
p1	J.Doe	27
p2	K.Smith	32
p3	J.Ho	28
p4	J.J.Doe	31

W_2, Pr=0.4

WK	name	age
p1	J.Doe	27
p2	K.Smith	32
p3	J.Ho	28
p4	J.J.Doe	41

W_3, Pr=0.1

WK	name	age
p1	J.Doe	27
p2	K.Smith	32
p4	J.J.Doe	41

```
SELECT *
FROM Person
WHERE age<30
```

Possible Worlds Semantics - Example

Step 1: Evaluate the query in each world separately

W_1, Pr=0.5

WK	name	age
p1	J.Doe	27
p2	K.Smith	32
p3	J.Ho	28
p4	J.J.Doe	31

W_2, Pr=0.4

WK	name	age
p1	J.Doe	27
p2	K.Smith	32
p3	J.Ho	28
p4	J.J.Doe	41

W_3, Pr=0.1

WK	name	age
p1	J.Doe	27
p2	K.Smith	32
p4	J.J.Doe	41

```
SELECT *
FROM Person
WHERE age<30
```

Possible Worlds Semantics - Example

Step 1: Evaluate the query in each world separately

W'_1, Pr=0.5

WK	name	age
p1	J.Doe	27
p3	J.Ho	28

W'_2, Pr=0.4

WK	name	age
p1	J.Doe	27
p3	J.Ho	28

W'_3, Pr=0.1

WK	name	age
p1	J.Doe	27

```
SELECT *
FROM Person
WHERE age<30
```

Possible Worlds Semantics - Example

Step 1: Evaluate the query in each world separately

Step 2: Combine identical worlds

W'_1, Pr=0.9

WK	name	age
p1	J.Doe	27
p3	J.Ho	28

W'_3, Pr=0.1

WK	name	age
p1	J.Doe	27

```
SELECT *
FROM Person
WHERE age<30
```

Possible Answers Semantics

- Query result: List of tuple-probability pairs
- Query answer: Single tuple
- Answers are usually sorted by their probabilities
- No dependencies between tuples reflected

$\Rightarrow$ Not a valid probabilistic database

$\Rightarrow$ Cannot be used to evaluate views

Definition: Let $pdb = (\mathbf{W}, Pr)$ be a probabilistic database and let Q be a conventional database query, the result of posing Q to pdb under the *possible answers semantics* is a set of tuple-probability pairs $Q(pdb) = \{(t_1, p_1), \ldots, (t_k, p_k)\}$ where

$$\{t_1, \ldots, t_k\} = \{t \mid \exists W \in \mathbf{W} \colon t \in Q(W)\}$$

$$\text{and} \quad \forall i \in \{1, \ldots, k\} \colon p_i = \sum_{W \in \mathbf{W}, t_i \in Q(W)} Pr(W)$$

Possible Answers Semantics

- Each tuple of the set

$$Q_{poss}(pdb) = \{t \mid \exists W \in \mathbf{W}\colon\ t \in Q(W)\} = \{t \mid (t,p) \in Q(pdb)\}$$

is called a *possible answer* of query Q to database pdb

- Each tuple of the set

$$\begin{aligned} Q_{cert}(pdb) &= \{t \mid \forall W \in \mathbf{W}\colon\ t \in Q(W)\} \\ &= \{t \mid (t,p) \in Q(pdb), p = 1\} \subseteq Q_{poss}(pdb) \end{aligned}$$

is called a *certain answer* of query Q to database pdb

Possible Answers Semantics - Example 1

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

SELECT WK, name, age
FROM Person
WHERE t.age $<$ 29

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_4	4	p3	J.Doe	28	0.8

BID-database
(possible worlds semantics)

Person

	WK	name	age	p
t_4	p3	J.Doe	28	0.8
t_1	p1	J.Doe	27	0.6
t_2	p1	J.Doe	28	0.2

Tuple-probability pair list
(possible answers semantics)

Possible Answers Semantics - Example 2

Person

	WK	name		age		p
t_1	p1	J.Doe	:1.0	27 28	:0.8 :0.2	0.8
t_2	p2	K.Smith	:1.0	32	:1.0	1.0
t_3	p3	J.Doe J.Ho	:0.7 :0.3	28 29	:0.5 :0.5	1.0

SELECT WK, name, age
FROM Person
WHERE t.age $<$ 29

Person

	WK	name		age		p
t_1	p1	J.Doe	:1.0	27 28	:0.8 :0.2	0.8
t_3	p3	J.Doe J.Ho	:0.7 :0.3	28	:1.0	0.5

AOR?-database
(possible worlds semantics)

Person

	WK	name	age	p
t_A	p1	J.Doe	27	0.64
t_B	p3	J.Doe	28	0.35
t_C	p1	J.Doe	28	0.16
t_D	p3	J.Ho	28	0.15

Tuple-probability pair list
(possible answers semantics)

Lineage

- Logical formula that describes the origin of an output tuple

$\Rightarrow$ It describes in which way which input tuples are combined in order to produce a particular output tuple

- Uncertain databases contain maybe-tuples

$\Rightarrow$ Existence of a tuple t as an uncertain event ϵ_t (Boolean random variable)

- Example: $\Phi_t = \epsilon_{t_1} \vee (\epsilon_{t_2} \wedge \epsilon_{t_3})$

$\Rightarrow$ "Tuple t results from tuple t_1 and from combining the tuples t_2 and t_3"

$\Rightarrow$ "Tuple t a query answer if either the tuple t_1 or the tuples t_2 and t_3 belong to the input database"

Query Evaluation Approaches

Intensional Query Evaluation:

Query evaluation and probability computation as two separate processes

- Database engine computes all possible query answers along with their lineage formulas
- Algorithm to compute the probabilities of the possible query answers based on their lineage formulas

Extensional Query Evaluation:

Query evaluation and probability computation as a single process

- Probabilistic computation is integrated into query evaluation

$\Rightarrow$ Probabilities are directly calculated by the database engine

Intensional Query Evaluation

Step 1: Construction of the lineage formulas

- Construction depends on the semantics of the operator

$\Rightarrow$ Specific construction rule per operator

Example:

- The join operator can only join two tuples if these tuples exist in the joined tables

$\Rightarrow$ The tuple $t = t_r \bowtie t_s$ has the lineage $\Phi_t = \Phi_{t_r} \wedge \Phi_{t_s}$

Assumption for the following rules:

- The input tables do not contain duplicate entries, or
- The individual operators (except projection) do not remove duplicate entries

Lineage Construction Rules

Projection:

- Tuple t belongs to $\pi_{\mathcal{A}}(Q)$ if input table Q contains at least one tuple t_r that has the same value as t in every attribute in $\mathcal{A}$ (i.e. $\forall A \in \mathcal{A}$: $t_r[A] = t[A]$) and hence if $\pi_{\mathcal{A}}(t_r) = t$

$\Rightarrow$ Tuple t exists if at least one tuple $t_r \in Q$ with $\pi_{\mathcal{A}}(t_r) = t$ exist

$\Rightarrow$ Construction rule: $\Phi_t = \bigvee_{t_r \in Q, \pi_{\mathcal{A}}(t_r)=t} \Phi_{t_r}$

Selection:

- Tuple t belongs to $\sigma_c(Q)$ if it satisfies condition c and input table Q contains a tuple t_r that corresponds to t

$\Rightarrow$ Tuple t exists if a tuple $t_r \in Q$ with $t_r = t$ exists

$\Rightarrow$ Construction rule: $\Phi_t = \Phi_{t_r}$

Lineage Construction Rules

Identity:

- Tuple t belongs to table T if T contains a tuple t_r that corresponds to t

$\Rightarrow$ Tuple t exists if a tuple $t_r \in T$ with $t_r = t$ exists

$\Rightarrow$ Construction rule: $\Phi_t = \epsilon_r$

- (This rule is required if the tuples of the queried database do not have a lineage/condition (e.g. in BID-databases))

Cross Product:

- Tuple t belongs to $Q_r \times Q_s$ if the input tables Q_r and Q_s contain two tuples $t_r \in Q_r$ and $t_s \in Q_s$ whose cross product corresponds to t

$\Rightarrow$ Tuple t exists if two tuples $t_r \in Q_r$ and $t_s \in Q_s$ with $t_r \times t_s = t$ exist

$\Rightarrow$ Construction rule: $\Phi_t = \Phi_{t_r} \wedge \Phi_{t_s}$

Lineage Construction Rules

Set Union:

- Tuple t belongs to $Q_r \cup Q_s$ if at least one of the two input tables Q_r and Q_s contains a tuple that corresponds to t

$\Rightarrow$ Tuple t exists if a tuple $t_r \in Q_r$ with $t_r = t$ or a tuple $t_s \in Q_s$ with $t_s = t$ exists

$\Rightarrow$ Construction rule: $\Phi_t = \Phi_{t_r} \vee \Phi_{t_s}$

Set Difference:

- Tuple t belongs to $Q_r \setminus Q_s$ if input table Q_r contains a tuple that corresponds to t and input table Q_s does not contain a tuple that corresponds to t

$\Rightarrow$ Tuple t exists if a tuple $t_r \in Q_r$ with $t_r = t$ exists and no tuple $t_s \in Q_s$ with $t_s = t$ exists

$\Rightarrow$ Construction rule: $\Phi_t = \Phi_{t_r} \wedge \neg(\Phi_{t_s})$

Lineage Construction Rules - Overview

Relational Algebra Operator		Lineage Construction Rule
Projection:	$t \in \pi_{\mathcal{A}}(Q)$	$\Phi_t = \bigvee_{t_r \in Q, \pi_{\mathcal{A}}(t_r)=t} \Phi_{t_r}$
Selection:	$t \in \sigma_c(Q)$	$\Phi_t = \Phi_{t_r}$ where $t_r \in Q, t_r = t$
Cross Product:	$t \in Q_r \times Q_s$	$\Phi_t = \Phi_{t_r} \wedge \Phi_{t_s}$ where $t_r \in Q_r, t_s \in Q_s, t_r \times t_s = t$
Set Union:	$t \in Q_r \cup Q_s$	$\Phi_t = \Phi_{t_r} \vee \Phi_{t_s}$ where $t_r \in Q_r, t_s \in Q_s, t_r = t_s = t$
Set Difference:	$t \in Q_r - Q_s$	$\Phi_t = \Phi_{t_r} \wedge \neg(\Phi_{t_s})$ where $t_r \in Q_r, t_s \in Q_s, t_r = t_s = t$
Identity:	$t \in T$	$\Phi_t = \epsilon$ where ϵ is the event that $t \in T$

Lineage - Example

```
SELECT   h.hobby
FROM     Person p, Hobby h
WHERE    p.WK = h.WK
AND      p.age < 30
```

Person

	WK	name	age
t_1	p1	Alice	27
t_2	p2	Bob	32
t_3	p3	Charlie	45
t_4	p4	Doris	28

Hobby

	WK	hobby
t_5	p1	football
t_6	p1	reading
t_7	p3	dancing
t_8	p4	reading

Lineage - Example

```
SELECT   h.hobby
FROM     Person p, Hobby h
WHERE    p.WK = h.WK
AND      p.age < 30
```

$Person \bowtie_{WK=WK} Hobby$

	WK	name	age	hobby	lineage
t_9	p1	Alice	27	football	$(\epsilon_1 \wedge \epsilon_5)$
t_{10}	p1	Alice	27	reading	$(\epsilon_1 \wedge \epsilon_6)$
t_{11}	p3	Charlie	45	dancing	$(\epsilon_3 \wedge \epsilon_7)$
t_{12}	p4	Doris	28	reading	$(\epsilon_4 \wedge \epsilon_8)$

Lineage - Example

```
SELECT   h.hobby
FROM     Person p, Hobby h
WHERE    p.WK = h.WK
AND      p.age < 30
```

$\sigma_{age<30}(Person \bowtie_{WK=WK} Hobby)$

	WK	name	age	hobby	lineage
t_9	p1	Alice	27	football	$(\epsilon_1 \wedge \epsilon_5)$
t_{10}	p1	Alice	27	reading	$(\epsilon_1 \wedge \epsilon_6)$
t_{12}	p4	Doris	28	reading	$(\epsilon_4 \wedge \epsilon_8)$

Lineage - Example

```
SELECT   h.hobby
FROM     Person p, Hobby h
WHERE    p.WK = h.WK
AND      p.age < 30
```

$$\pi_{hobby}\,(\sigma_{age<30}\,(Person \bowtie_{WK=WK} Hobby))$$

	hobby	*lineage*
t_{13}	*football*	$(\epsilon_1 \wedge \epsilon_5)$
t_{14}	*reading*	$(\epsilon_1 \wedge \epsilon_6) \vee (\epsilon_4 \wedge \epsilon_8)$

Query Evaluation based on Lineage

- Result can be considered as a pc-database that has the queried database as world-table

$\Rightarrow$ Events correspond to atomic conditions

$\Rightarrow$ Input tuples correspond to Boolean variables (not all of these variables are independent!!)

$\Rightarrow$ Each possible world of the queried database corresponds to a variable assignment

- If queried database is a pc-database

$\Rightarrow$ Each tuple event can be another logical condition (e.g. $\epsilon_t \equiv (X = 1) \vee ((X = 3) \wedge (Y = 2))$)

Query Evaluation based on Lineage - Example

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

```
SELECT  t.name AS nameA, u.name AS nameB
FROM    Person t, Person u
WHERE   t.WK <> u.WK
AND     t.age < u.age
```

Query Evaluation based on Lineage - Example

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

```
SELECT  t.name AS nameA, u.name AS nameB
FROM    Person t, Person u
WHERE   t.WK <> u.WK
AND     t.age < u.age
```

IsYoungerThan

	nameA	nameB	lineage	p
t_6	J.Doe	K.Smith	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	J.Doe	J.Doe	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	J.Ho	K.Smith	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	J.Doe	J.Ho	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Person (World-Table)

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

Query Evaluation based on Lineage - Example

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

```
SELECT  t.name AS nameA, u.name AS nameB
FROM    Person t, Person u
WHERE   t.WK <> u.WK
AND     t.age < u.age
```

IsYoungerThan

	nameA	nameB	lineage	p
t_6	J.Doe	K.Smith	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	J.Doe	J.Doe	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	J.Ho	K.Smith	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	J.Doe	J.Ho	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Assignments

$W_1=\{t_1,t_3,t_4\}$	$Pr(W_1)=0.48$
$W_2=\{t_1,t_3,t_5\}$	$Pr(W_2)=0.12$
$W_3=\{t_2,t_3,t_4\}$	$Pr(W_3)=0.16$
$W_4=\{t_2,t_3,t_5\}$	$Pr(W_4)=0.04$
$W_5=\{t_3,t_4\}$	$Pr(W_5)=0.16$
$W_6=\{t_3,t_5\}$	$Pr(W_6)=0.04$

Intensional Query Evaluation

Step 2: Probability Computation

- Semantics: Tuple is a query answer if its lineage formula is satisfied by the possible world of the queried database

$\Rightarrow$ Marginal probability of a tuple results in the overall probability of all possible worlds of the queried database that satisfy its lineage

$\Rightarrow$ The problem of computing the marginal probability of a possible query answer can be reduced to the problem of computing the probability of a propositional formula

Probability Computation - Example

IsYoungerThan

	nameA	*nameB*	*lineage*	*p*
t_6	*J.Doe*	*K.Smith*	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	*J.Doe*	*J.Doe*	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	*J.Ho*	*K.Smith*	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	*J.Doe*	*J.Ho*	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Assignments

$W_1 = \{t_1, t_3, t_4\}$	$Pr(W_1) = 0.48$
$W_2 = \{t_1, t_3, t_5\}$	$Pr(W_2) = 0.12$
$W_3 = \{t_2, t_3, t_4\}$	$Pr(W_3) = 0.16$
$W_4 = \{t_2, t_3, t_5\}$	$Pr(W_4) = 0.04$
$W_5 = \{t_3, t_4\}$	$Pr(W_5) = 0.16$
$W_6 = \{t_3, t_5\}$	$Pr(W_6) = 0.04$

- t_6 is a query answer if t_1 and t_3, t_2 and t_3, or t_4 and t_3 exist

$\Rightarrow$ Its lineage formula is satisfied by all worlds except W_6

$\Rightarrow$ The probability of t_6 is $p(t_6) = 1 - 0.04 = 0.96$

Probability Computation - Example

IsYoungerThan

	nameA	*nameB*	*lineage*	*p*
t_6	*J.Doe*	*K.Smith*	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	*J.Doe*	*J.Doe*	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	*J.Ho*	*K.Smith*	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	*J.Doe*	*J.Ho*	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Assignments

$W_1 = \{t_1, t_3, t_4\}$	$Pr(W_1) = 0.48$
$W_2 = \{t_1, t_3, t_5\}$	$Pr(W_2) = 0.12$
$W_3 = \{t_2, t_3, t_4\}$	$Pr(W_3) = 0.16$
$W_4 = \{t_2, t_3, t_5\}$	$Pr(W_4) = 0.04$
$W_5 = \{t_3, t_4\}$	$Pr(W_5) = 0.16$
$W_6 = \{t_3, t_5\}$	$Pr(W_6) = 0.04$

- t_7 is a query answer if t_1 and t_4 exist

$\Rightarrow$ Its lineage formula is satisfied only by world W_1

$\Rightarrow$ The probability of t_7 is $p(t_7) = 0.48$

Probability Computation - Example

IsYoungerThan

	nameA	*nameB*	*lineage*	*p*
t_6	*J.Doe*	*K.Smith*	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	*J.Doe*	*J.Doe*	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	*J.Ho*	*K.Smith*	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	*J.Doe*	*J.Ho*	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Assignments

$W_1 = \{t_1, t_3, t_4\}$	$Pr(W_1) = 0.48$
$W_2 = \{t_1, t_3, t_5\}$	$Pr(W_2) = 0.12$
$W_3 = \{t_2, t_3, t_4\}$	$Pr(W_3) = 0.16$
$W_4 = \{t_2, t_3, t_5\}$	$Pr(W_4) = 0.04$
$W_5 = \{t_3, t_4\}$	$Pr(W_5) = 0.16$
$W_6 = \{t_3, t_5\}$	$Pr(W_6) = 0.04$

- t_8 is a query answer if t_5 and t_3 exist

$\Rightarrow$ Its lineage formula is satisfied by the worlds W_2, W_4, and W_6

$\Rightarrow$ The probability of t_8 is $p(t_8) = 0.2$

Probability Computation - Example

IsYoungerThan

	nameA	*nameB*	*lineage*	*p*
t_6	*J.Doe*	*K.Smith*	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	*J.Doe*	*J.Doe*	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	*J.Ho*	*K.Smith*	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	*J.Doe*	*J.Ho*	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Assignments

$W_1 = \{t_1, t_3, t_4\}$	$Pr(W_1) = 0.48$
$W_2 = \{t_1, t_3, t_5\}$	$Pr(W_2) = 0.12$
$W_3 = \{t_2, t_3, t_4\}$	$Pr(W_3) = 0.16$
$W_4 = \{t_2, t_3, t_5\}$	$Pr(W_4) = 0.04$
$W_5 = \{t_3, t_4\}$	$Pr(W_5) = 0.16$
$W_6 = \{t_3, t_5\}$	$Pr(W_6) = 0.04$

- t_9 is a query answer if t_1 and t_5, or t_2 and t_5 exist

$\Rightarrow$ Its lineage formula is satisfied by the worlds W_2 and W_4

$\Rightarrow$ The probability of t_9 is $0.12 + 0.04 = 0.16$

Probability Computation

Problem:

- An enumeration of all possible worlds is impractical

$\Rightarrow$ The probability of an output tuple has to be computed directly on the probabilities of the input tuples

Probability Computation - Example

IsYoungerThan

	nameA	*nameB*	*lineage*	*p*
t_6	*J.Doe*	*K.Smith*	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	*J.Doe*	*J.Doe*	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	*J.Ho*	*K.Smith*	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	*J.Doe*	*J.Ho*	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Assignments

$W_1=\{t_1,t_3,t_4\}$	$Pr(W_1)=0.48$
$W_2=\{t_1,t_3,t_5\}$	$Pr(W_2)=0.12$
$W_3=\{t_2,t_3,t_4\}$	$Pr(W_3)=0.16$
$W_4=\{t_2,t_3,t_5\}$	$Pr(W_4)=0.04$
$W_5=\{t_3,t_4\}$	$Pr(W_5)=0.16$
$W_6=\{t_3,t_5\}$	$Pr(W_6)=0.04$

- t_3 is certain $\Rightarrow$ lineage of t_6 can be simplified to:
  $(\epsilon_1 \vee \epsilon_2 \vee \epsilon_4) \wedge \epsilon_3 \equiv (\epsilon_1 \vee \epsilon_2 \vee \epsilon_4) \wedge \mathit{true} \equiv \epsilon_1 \vee \epsilon_2 \vee \epsilon_4$
- Let ϵ^* be the event that either t_1 or t_2 exists (i.e. $\epsilon^* = \epsilon_1 \vee \epsilon_2$)
- t_1 and t_2 are exclusive ($\Rightarrow p(\epsilon^*) = p(t_1) + p(t_2)$)
- t_4 is independent of t_1 and t_2 ($\Rightarrow \epsilon_4$ is independent of ϵ^*)

$\Rightarrow$ $p(t_6) = 1 - (1 - p(\epsilon^*)) \times (1 - p(t_4)) = 1 - (1 - (p(t_1) + p(t_2))) \times (1 - p(t_4)) = 1 - (1 - (0.6 + 0.2)) \times (1 - 0.8) = 1 - 0.2 \times 0.2 = 0.96$

Probability Computation - Example

IsYoungerThan

	nameA	*nameB*	*lineage*	*p*
t_6	*J.Doe*	*K.Smith*	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	*J.Doe*	*J.Doe*	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	*J.Ho*	*K.Smith*	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	*J.Doe*	*J.Ho*	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Assignments

$W_1=\{t_1,t_3,t_4\}$	$Pr(W_1)=0.48$
$W_2=\{t_1,t_3,t_5\}$	$Pr(W_2)=0.12$
$W_3=\{t_2,t_3,t_4\}$	$Pr(W_3)=0.16$
$W_4=\{t_2,t_3,t_5\}$	$Pr(W_4)=0.04$
$W_5=\{t_3,t_4\}$	$Pr(W_5)=0.16$
$W_6=\{t_3,t_5\}$	$Pr(W_6)=0.04$

- t_1 and t_4 are independent

$\Rightarrow\ p(t_7) = p(t_1) \times p(t_4) = 0.6 \times 0.8 = 0.48$

Probability Computation - Example

IsYoungerThan

	nameA	*nameB*	*lineage*	*p*
t_6	*J.Doe*	*K.Smith*	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	*J.Doe*	*J.Doe*	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	*J.Ho*	*K.Smith*	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	*J.Doe*	*J.Ho*	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Assignments

$W_1=\{t_1,t_3,t_4\}$	$Pr(W_1)=0.48$
$W_2=\{t_1,t_3,t_5\}$	$Pr(W_2)=0.12$
$W_3=\{t_2,t_3,t_4\}$	$Pr(W_3)=0.16$
$W_4=\{t_2,t_3,t_5\}$	$Pr(W_4)=0.04$
$W_5=\{t_3,t_4\}$	$Pr(W_5)=0.16$
$W_6=\{t_3,t_5\}$	$Pr(W_6)=0.04$

- t_3 is a certain-tuple

$\Rightarrow$ The lineage of t_8 can be simplified to: $\epsilon_5 \wedge \mathit{true} = \epsilon_5$

$\Rightarrow$ t_8 is a query answer if and only if t_5 exists

$\Rightarrow$ $p(t_8) = p(t_5) = 0.2$

Probability Computation - Example

IsYoungerThan

	nameA	*nameB*	*lineage*	*p*
t_6	*J.Doe*	*K.Smith*	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	*J.Doe*	*J.Doe*	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	*J.Ho*	*K.Smith*	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	*J.Doe*	*J.Ho*	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Assignments

$W_1=\{t_1,t_3,t_4\}$	$Pr(W_1)=0.48$
$W_2=\{t_1,t_3,t_5\}$	$Pr(W_2)=0.12$
$W_3=\{t_2,t_3,t_4\}$	$Pr(W_3)=0.16$
$W_4=\{t_2,t_3,t_5\}$	$Pr(W_4)=0.04$
$W_5=\{t_3,t_4\}$	$Pr(W_5)=0.16$
$W_6=\{t_3,t_5\}$	$Pr(W_6)=0.04$

- Lineage of t_9 can be transformed into $(\epsilon_1 \vee \epsilon_2) \wedge \epsilon_5 = \epsilon^* \wedge \epsilon_5$
- t_5 is independent of t_1 and t_2 ($\Rightarrow \epsilon_5$ is independent of ϵ^*)

$$\Rightarrow p(t_9) = p(\epsilon^*) \times p(t_5) = (p(t_1) + p(t_2)) \times p(t_5)$$
$$= (0.6 + 0.2) \times 0.2 = 0.16$$

Probabilistic Inference

Algorithms:

- Exact: Variable Elimination (general-purpose), Rule-based (problem-specific)
- Approximate: Monte-Carlo simulation

Problem:

- Probabilistic inference is NP-hard

$\Rightarrow$ Scales bad if the lineage formulas are long and/or complex

- One computation per possible query answer

$\Rightarrow$ Large computation overhead if the query result is large

$\Rightarrow$ Probability computation often only ad-hoc for particular answers on user request

Probabilistic Inference

- Computing the probability distributions of unobserved variables based on the probability distributions of observed variables
- Relationships (direct or indirect) between these variables are given (e.g. in form of conditional probabilities)
- Probability distributions of related variables can be unknown (but need to be inferable from the given relationships)

General-purpose Algorithms for Probabilistic Inference:

- Are based on probabilistic graphical models such as Bayesian networks or Markov networks

$\Rightarrow$ Problem of computing the probability of a tuple condition needs to be transformed into such a graphical model

Naive Transformation into Bayesian Network

Transformation Process:

- One node per database variable used in the tuple condition
- One node for the Boolean (query) variable representing the whole tuple condition
- An edge from each node of the database variables (i.e. the observed variables) to the node of the query variable

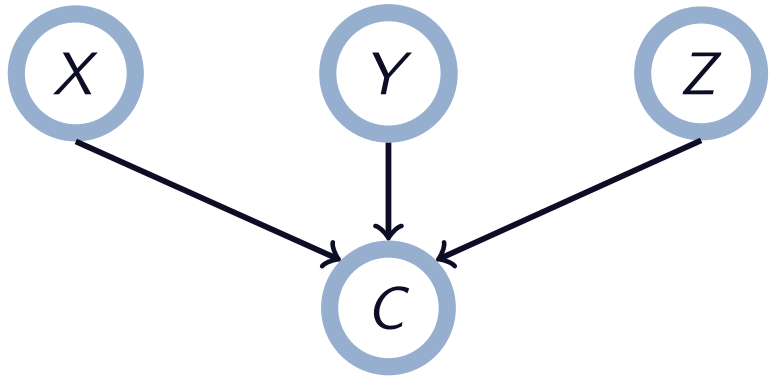
Characteristics:

- Conditional probabilities of the query variable are defined on the whole tuple condition

$\Rightarrow$ Inference will enumerate all possible combinations of the database variables (i.e. brute force)

Naive Transformation - Example

$$\underbrace{(X = 1 \wedge Y = 2) \quad \vee \quad (Y = 1 \wedge Z = 1)}_{C = true}$$



$Prob(C \mid X, Y, Z)$	*true*	*false*
$(X = 1 \wedge Y = 2) \vee (Y = 1 \wedge Z = 1)$	1.0	0.0
else	0.0	1.0

Hierarchical Transformation into Bayesian Network

Transformation Process:

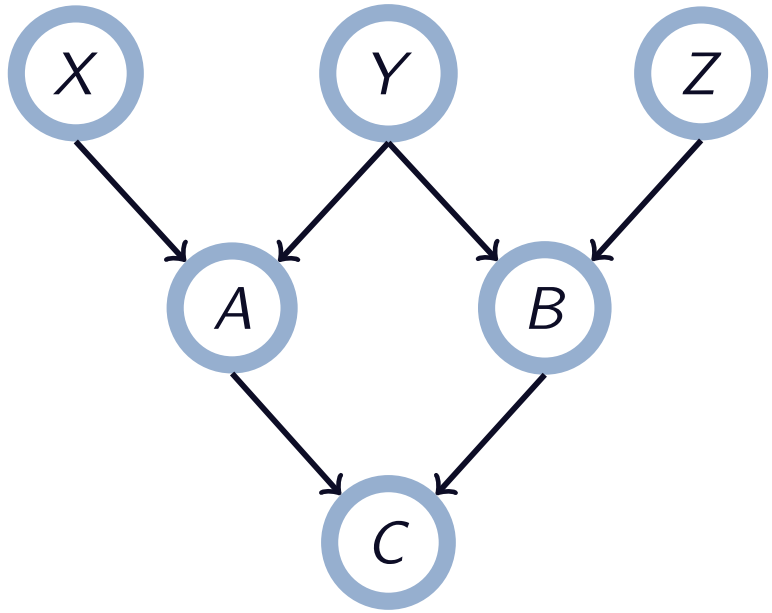
- Iterative construction of Boolean variables (one for each logical connection of variables of the previous iteration step)
- Boolean variable of last iteration step represents the whole tuple condition (i.e. the query variable)
- One node per variable (database or Boolean)
- Edge from X to Y if Y is constructed based on X

Characteristics:

- More variables (than in naive), but benefit increases with
  - Number of considered database variables
  - Number of possible values per considered database variable
  - Number of independent subconditions

Hierarchical Transformation - Example

$$\underbrace{(\underbrace{X = 1 \wedge Y = 2}_{A = true}) \vee (\underbrace{Y = 1 \wedge Z = 1}_{B = true})}_{C = true}$$



$Prob(A \mid X, Y)$	$true$	$false$
$X = 1 \wedge Y = 2$	1.0	0.0
else	0.0	1.0

$Prob(B \mid Y, Z)$	$true$	$false$
$Y = 1 \wedge Z = 1$	1.0	0.0
else	0.0	1.0

$Prob(C \mid A, B)$	$true$	$false$
$A = true \vee B = true$	1.0	0.0
else	0.0	1.0

Variable Elimination

Summing out variables:

- Summing a variable X_1 out from a factor $f_1(X_1, X_2, \ldots, X_n)$ results in a factor $f_2(X_2, \ldots, X_n)$ with

$$f_2(X_2, \ldots, X_n) = \sum_{X_1} f_1(X_1 = x, X_2, \ldots, X_n)$$

Efficient sum out from a product $f_1 \times \ldots \times f_k$ of factors:

- Partition the set of factors into
  - those that do not contain X_1, say $f_1, \ldots, f_i$
  - those that contain X_1, say $f_{i+1}, \ldots, f_k$
- Generate the new factor $f' = \sum_{X_1}(f_{i+1} \times \ldots \times f_k)$
- Discard $f_{i+1}, \ldots, f_k$ and store f' explicitly

In summary, we compute:

$$\begin{aligned} \sum_{X_1}(f_1 \times \ldots \times f_k) &= (f_1 \times \ldots \times f_i) \times \left(\sum_{X_1}(f_{i+1} \times \ldots \times f_k)\right) \\ &= f_1 \times \ldots \times f_i \times f' \end{aligned}$$

Variable Elimination - General Case

Given:

- Query variable Q (variable of any node)
- Observed variables $Y_1, \ldots, Y_m$
- Remaining variables $Z_1, \ldots, Z_n$

Computation of $Prob(Q \mid Y_1 = y_1, \ldots, Y_m = y_m)$:

- Construct a factor for each (cond.) probability distribution
- Set the observed variables to their observed values
- Sum out each of the other variables $Z_i \in \{Z_1, \ldots, Z_n\}$
- Multiply the remaining factors
  (result $f(Q) = Prob(Q, Y_1 = y_1, \ldots, Y_m = y_m)$)
- Normalize $f(Q)$ by dividing it by $\sum_Q f(Q)$

Variable Elimination - Our Case

Given:

- Query variable Q (variable of the single leaf node)
- ~~Observed variables $Y_1, \ldots, Y_m$~~
- Remaining variables $Z_1, \ldots, Z_n$

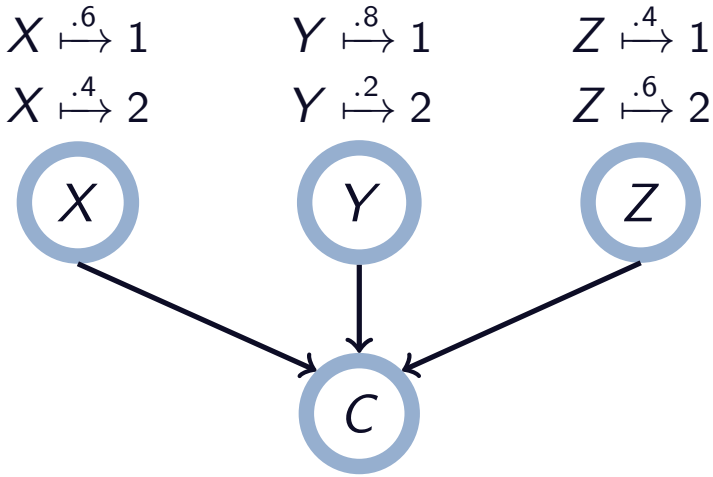
Computation of $Prob(Q \mid Y_1 = y_1, \ldots, Y_m = y_m)$:

- Construct a factor for each (cond.) probability distribution
- ~~Set the observed variables to their observed values~~
- Sum out each of the other variables $Z_i \in \{Z_1, \ldots, Z_n\}$
- ~~Multiply the remaining factors (result: $f(Q) = Prob(Q, Y_1 = y_1, \ldots, Y_m = y_m)$)~~
- ~~Normalize $f(Q)$ by dividing it by $\sum_Q f(Q)$~~

Variable Elimination - Example 1

- Given factors:

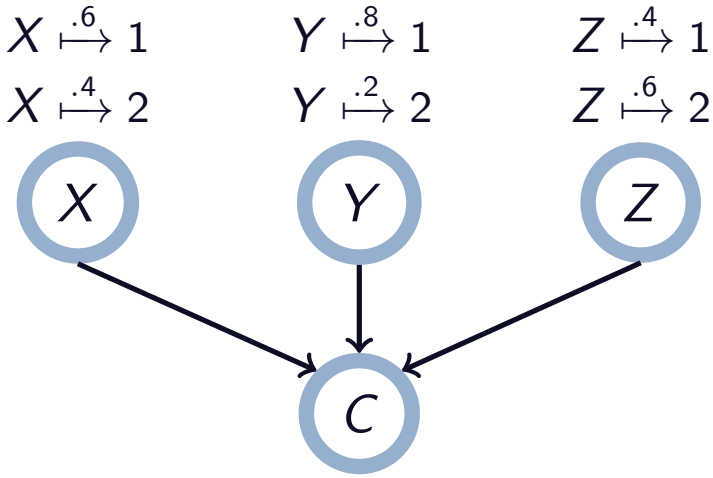
  $Prob(X)$, $Prob(Y)$, $Prob(Z)$,
  $Prob(C \mid X, Y, Z)$



Variable Elimination - Example 1

- Given factors:

  $Prob(X)$, $Prob(Y)$, $Prob(Z)$,
  $Prob(C \mid X, Y, Z)$



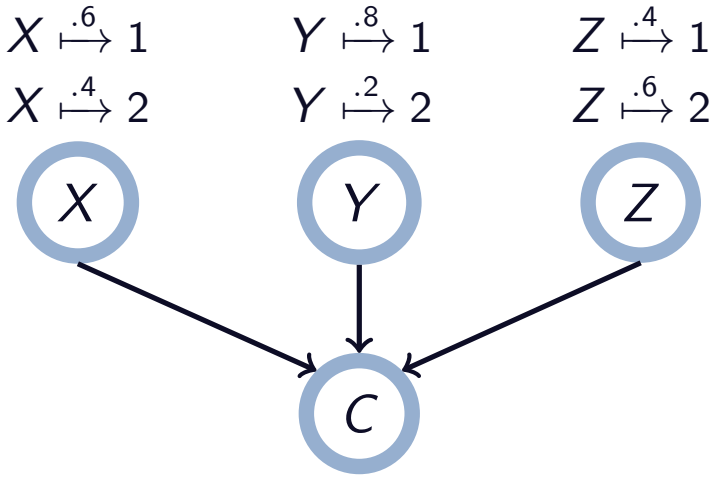
1. Elimination of variable X by generating factor:

$$f_1(C, Y, Z) = \sum_{x \in \{1,2\}} Prob(X = x) \times Prob(C \mid X = x, Y, Z)$$
$$= Prob(C \mid Y, Z)$$

Variable Elimination - Example 1

- Given factors:

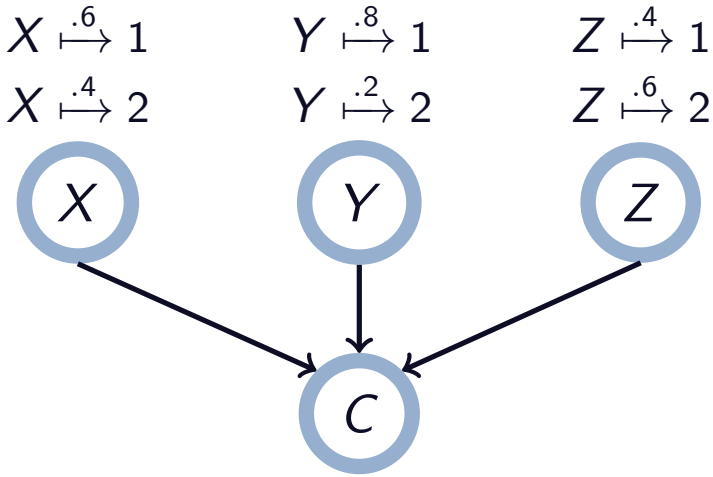
  ~~$Prob(X)$~~, $Prob(Y)$, $Prob(Z)$,
  ~~$Prob(C \mid X, Y, Z)$~~, $f_1(C, Y, Z)$



Variable Elimination - Example 1

- Given factors:

  ~~$Prob(X)$~~, $Prob(Y)$, $Prob(Z)$,
  ~~$Prob(C \mid X, Y, Z)$~~, $f_1(C, Y, Z)$



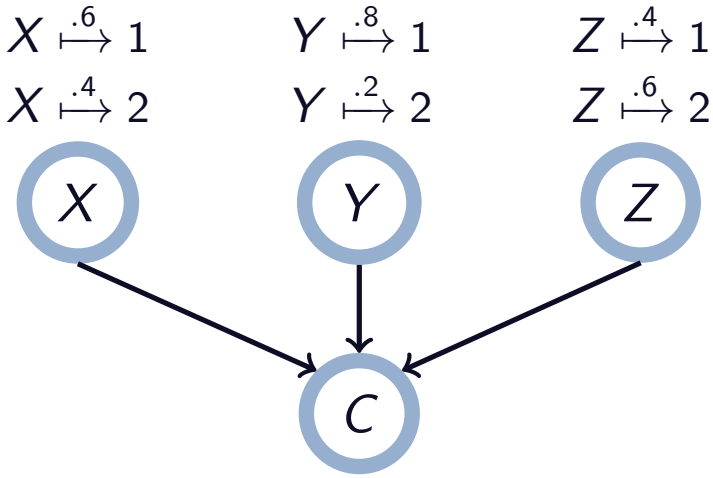
2. Elimination of variable Z by generating factor:

$$f_2(C, Y) = \sum_{z \in \{1,2\}} Prob(Z = z) \times f_1(C, Y, Z = z)$$
$$= Prob(C \mid Y)$$

Variable Elimination - Example 1

- Given factors:

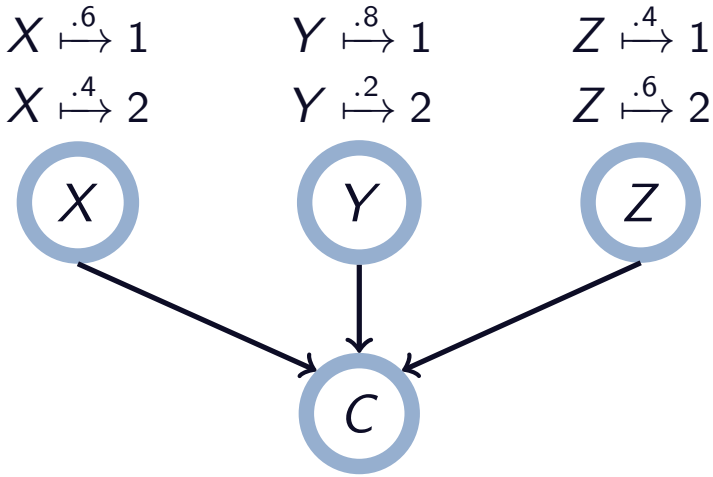
  ~~$Prob(X)$~~, $Prob(Y)$, ~~$Prob(Z)$~~,
  ~~$Prob(C \mid X, Y, Z)$~~, ~~$f_1(C, Y, Z)$~~,
  $f_2(C, Y)$



Variable Elimination - Example 1

- Given factors:

  ~~$Prob(X)$~~, $Prob(Y)$, ~~$Prob(Z)$~~,
  ~~$Prob(C \mid X, Y, Z)$~~, ~~$f_1(C, Y, Z)$~~,
  $f_2(C, Y)$



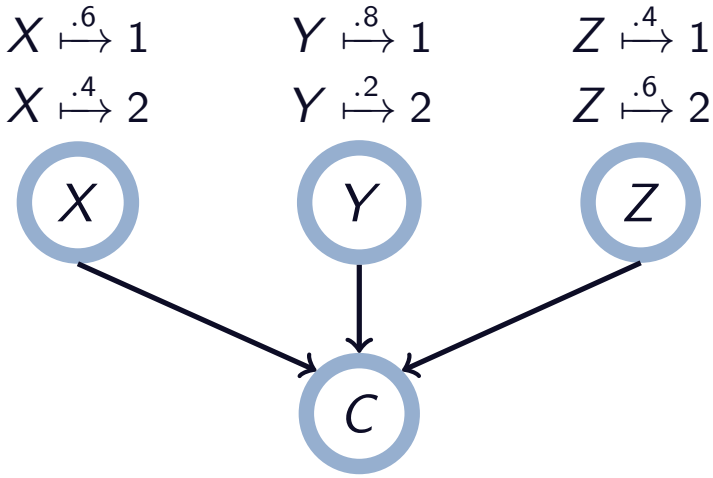
3. Elimination of variable Y by generating factor:

$$f_3(C) = \sum_{y \in \{1,2\}} Prob(Y = y) \times f_2(C, Y = y)$$
$$= Prob(C)$$

Variable Elimination - Example 1

- Given factors:

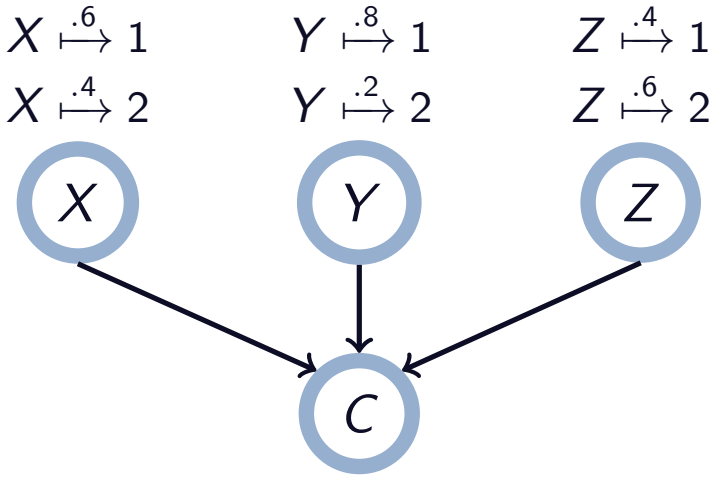
  ~~$Prob(X)$~~, ~~$Prob(Y)$~~, ~~$Prob(Z)$~~,
  ~~$Prob(C \mid X, Y, Z)$~~, ~~$f_1(C, Y, Z)$~~,
  ~~$f_2(C, Y)$~~, $f_3(C)$



Variable Elimination - Example 1

- Given factors:

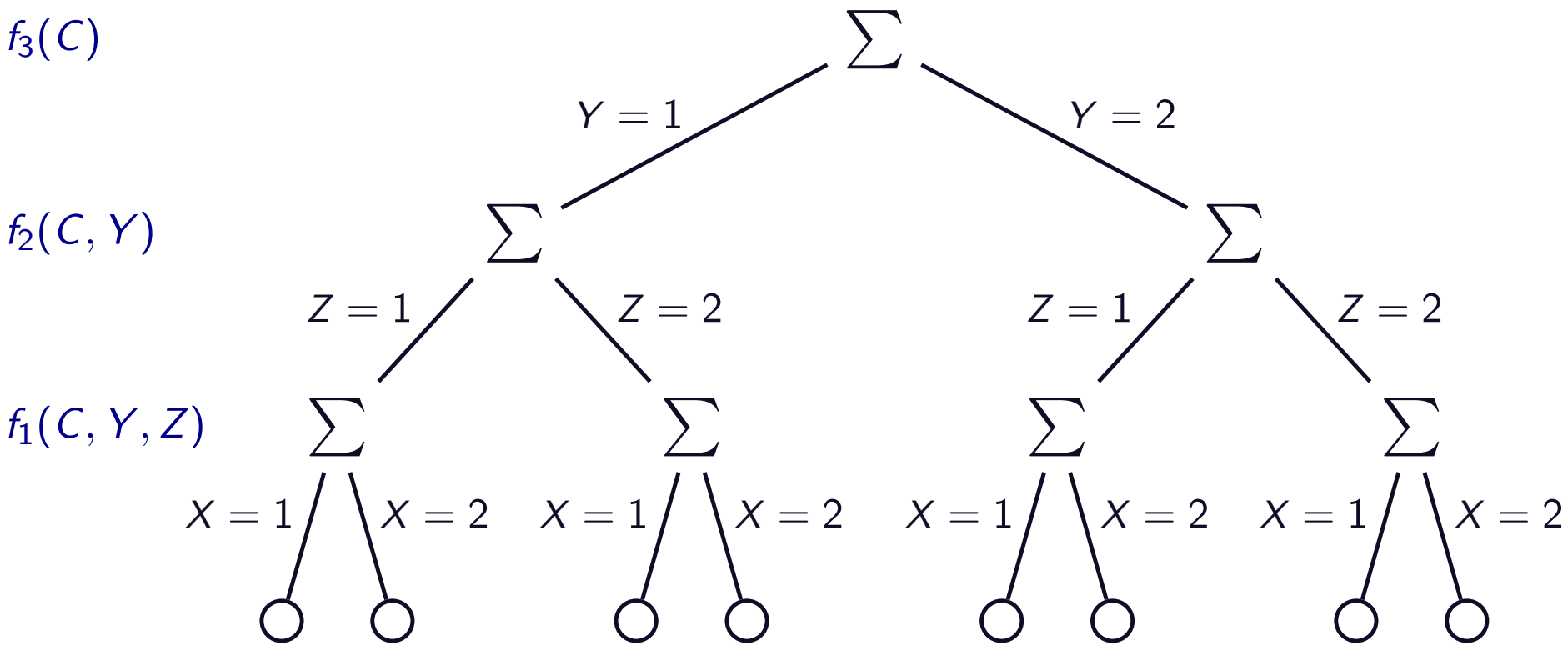
~~$Prob(X)$, $Prob(Y)$, $Prob(Z)$,~~
~~$Prob(C \mid X, Y, Z)$, $f_1(C, Y, Z)$,~~
~~$f_2(C, Y)$,~~ $f_3(C)$



Final Result:

$$f_3(C) = \sum_{y \in \{1,2\}} Prob(Y = y) \times \left(\sum_{z \in \{1,2\}} Prob(Z = z) \times \left(\sum_{x \in \{1,2\}} Prob(X = x) \times Prob(C \mid X = x, Y = y, Z = z) \right) \right)$$

Variable Elimination - Example 1 (Operator Tree)



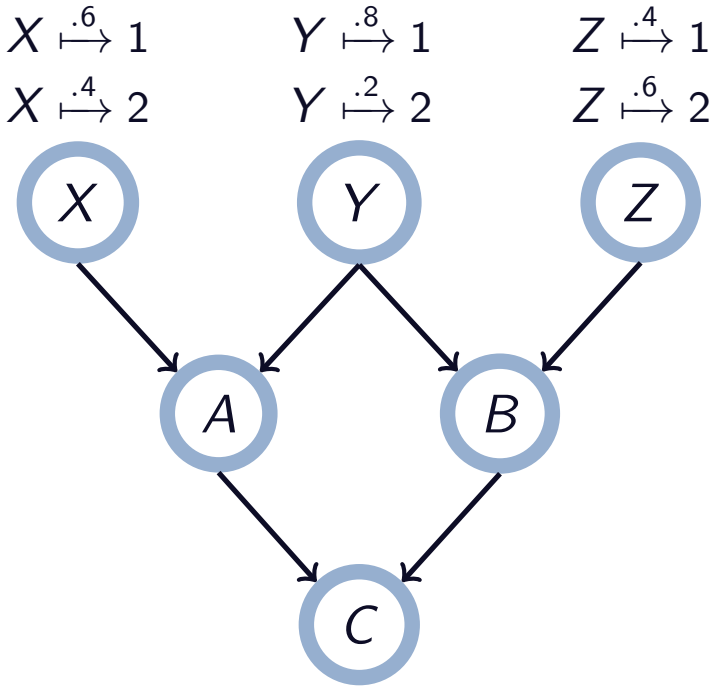
- Eight paths from root to leaf

$\Rightarrow$ Enumeration of all possible combinations

Variable Elimination - Example 2

- Given factors:

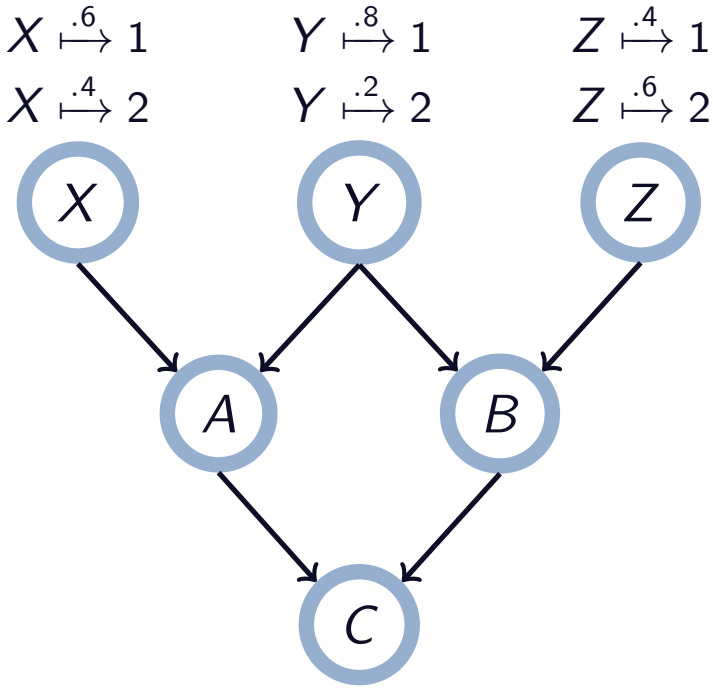
  $Prob(X)$, $Prob(Y)$, $Prob(Z)$,
  $Prob(A \mid X, Y)$, $Prob(B \mid Y, Z)$,
  $Prob(C \mid A, B)$



Variable Elimination - Example 2

- Given factors:

  $Prob(X)$, $Prob(Y)$, $Prob(Z)$,
  $Prob(A \mid X, Y)$, $Prob(B \mid Y, Z)$,
  $Prob(C \mid A, B)$



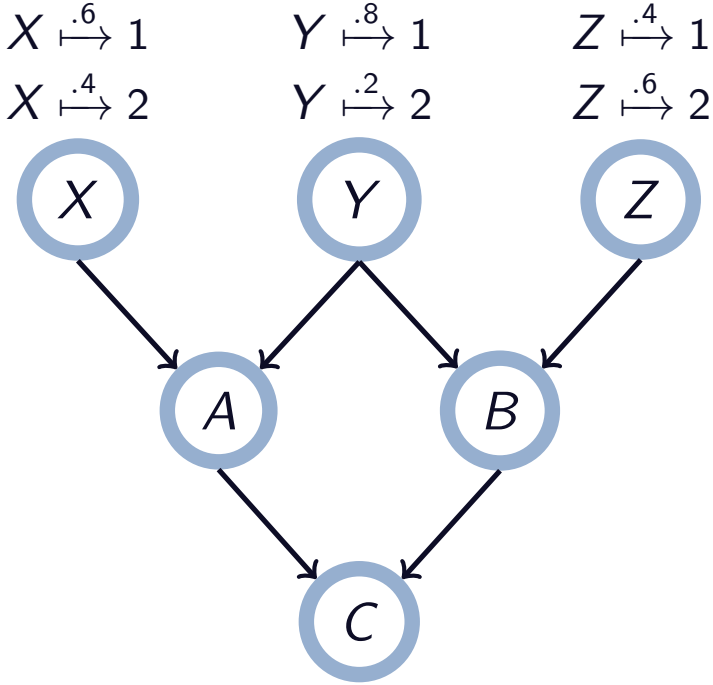
1. Elimination of variable X by generating factor:

   $f_1(A, Y) = \sum_{x \in \{1,2\}} Prob(X = x) \times Prob(A \mid X = x, Y)$

Variable Elimination - Example 2

- Given factors:

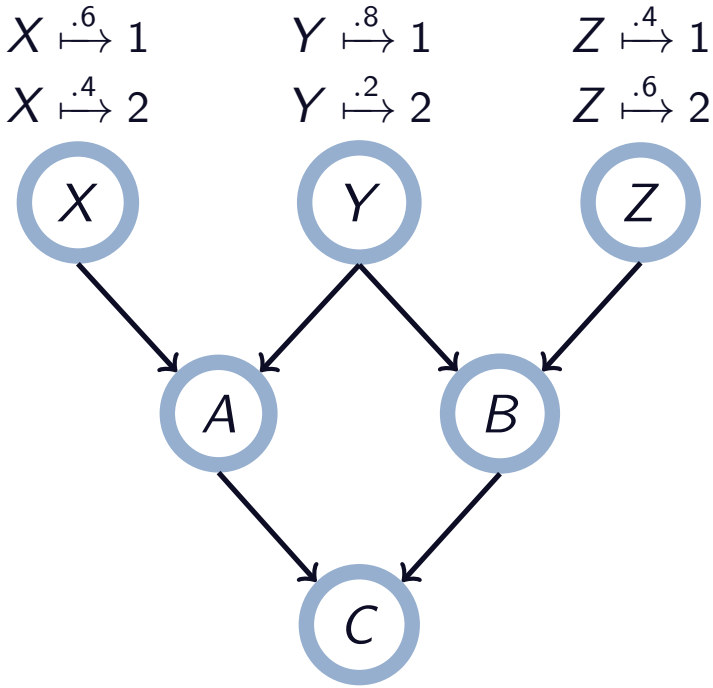
  ~~$Prob(X)$~~, $Prob(Y)$, $Prob(Z)$,
  ~~$Prob(A \mid X, Y)$~~, $Prob(B \mid Y, Z)$,
  $Prob(C \mid A, B)$, $f_1(A, Y)$



Variable Elimination - Example 2

- Given factors:

  ~~$Prob(X)$~~, $Prob(Y)$, $Prob(Z)$,
  ~~$Prob(A \mid X, Y)$~~, $Prob(B \mid Y, Z)$,
  $Prob(C \mid A, B)$, $f_1(A, Y)$



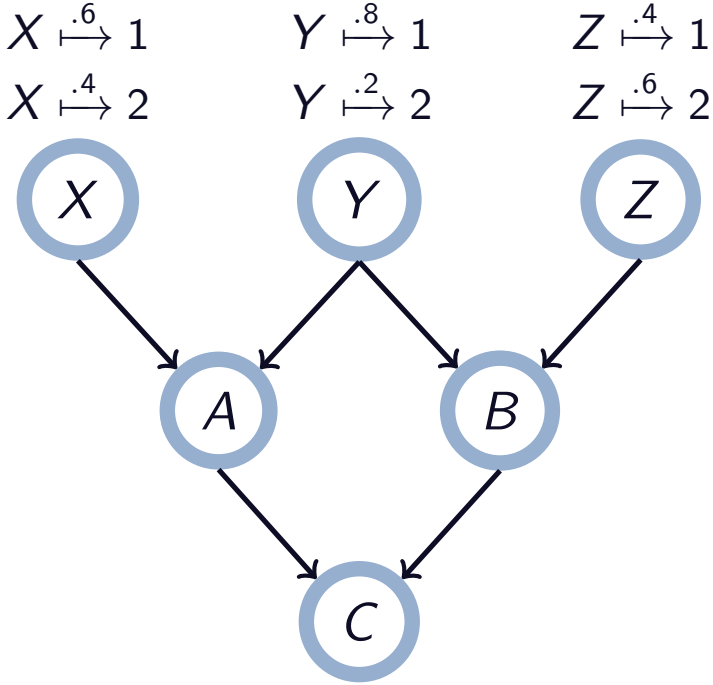
2. Elimination of variable Z by generating factor:

   $f_2(B, Y) = \sum_{z \in \{1,2\}} Prob(Z = z) \times Prob(B \mid Y, Z = z)$

Variable Elimination - Example 2

- Given factors:

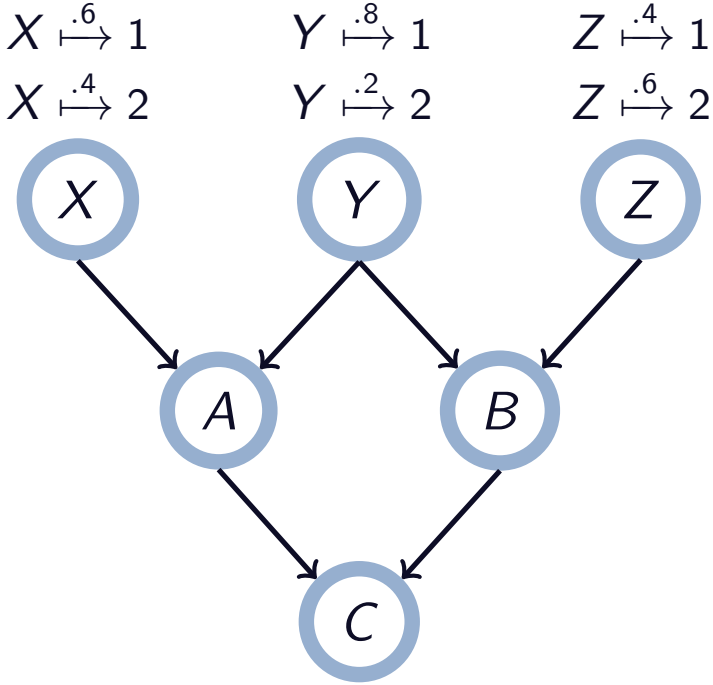
  ~~$Prob(X)$~~, $Prob(Y)$, ~~$Prob(Z)$~~,
  ~~$Prob(A \mid X, Y)$~~, ~~$Prob(B \mid Y, Z)$~~,
  $Prob(C \mid A, B)$, $f_1(A, Y)$, $f_2(B, Y)$



Variable Elimination - Example 2

- Given factors:

  ~~$Prob(X)$~~, $Prob(Y)$, ~~$Prob(Z)$~~,
  ~~$Prob(A \mid X, Y)$~~, ~~$Prob(B \mid Y, Z)$~~,
  $Prob(C \mid A, B)$, $f_1(A, Y)$, $f_2(B, Y)$



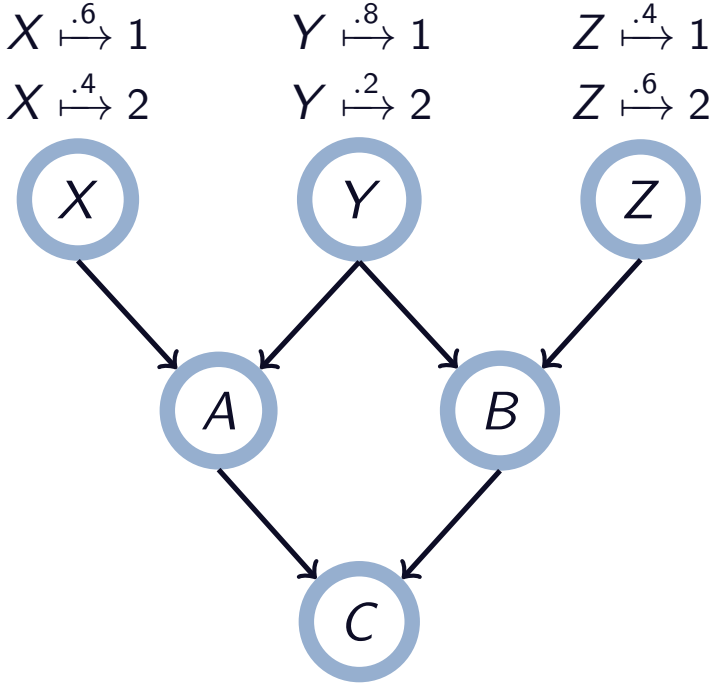
3. Elimination of variable A by generating factor:

   $f_3(B, C, Y) = \sum_{a \in \{true, false\}} Prob(C \mid A = a, B) \times f_1(A = a, Y)$

Variable Elimination - Example 2

- Given factors:

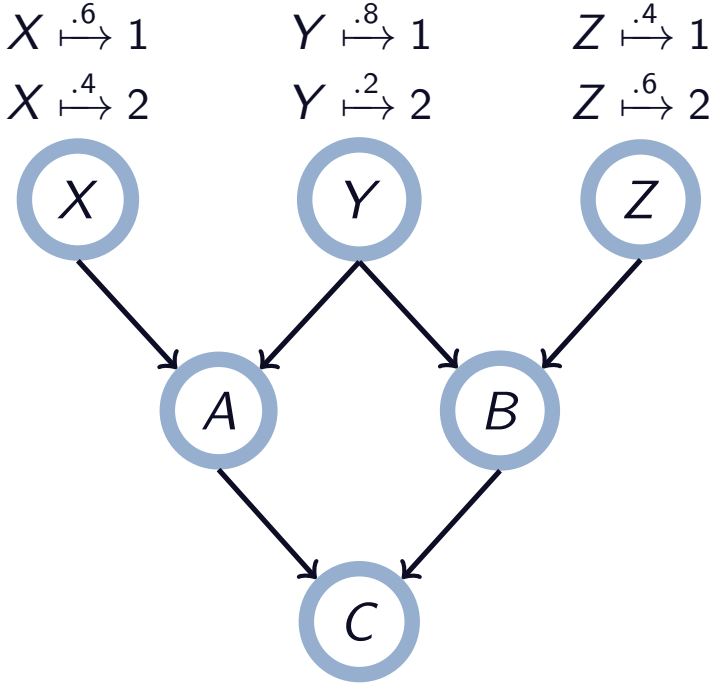
  ~~$Prob(X)$~~, $Prob(Y)$, ~~$Prob(Z)$~~,
  ~~$Prob(A \mid X, Y)$~~, ~~$Prob(B \mid Y, Z)$~~,
  ~~$Prob(C \mid A, B)$~~, ~~$f_1(A, Y)$~~, $f_2(B, Y)$,
  $f_3(B, C, Y)$



Variable Elimination - Example 2

- Given factors:

  ~~$Prob(X)$~~, $Prob(Y)$, ~~$Prob(Z)$~~,
  ~~$Prob(A \mid X, Y)$~~, ~~$Prob(B \mid Y, Z)$~~,
  ~~$Prob(C \mid A, B)$~~, ~~$f_1(A, Y)$~~, $f_2(B, Y)$,
  $f_3(B, C, Y)$



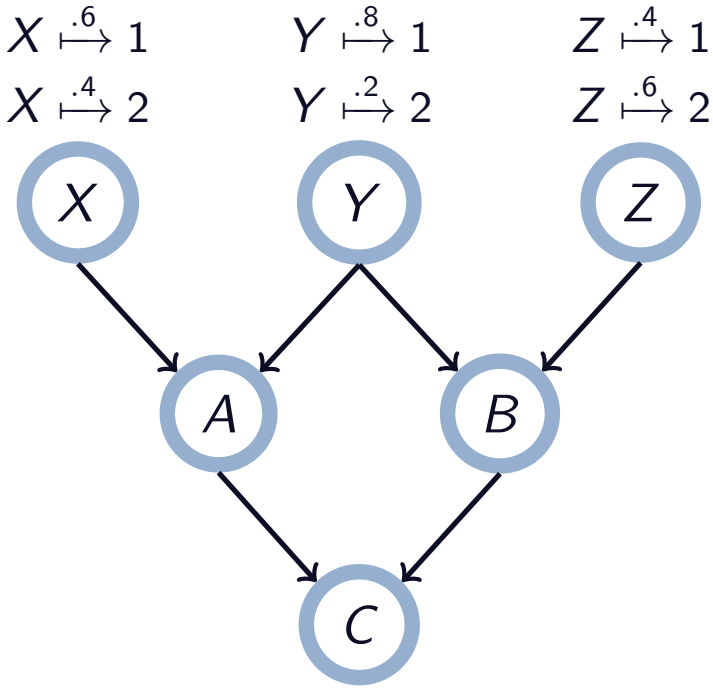
4. Elimination of variable B by generating factor:

   $f_4(C, Y) = \sum_{b \in \{true, false\}} f_2(B = b, Y) \times f_3(B = b, C, Y)$

Variable Elimination - Example 2

- Given factors:

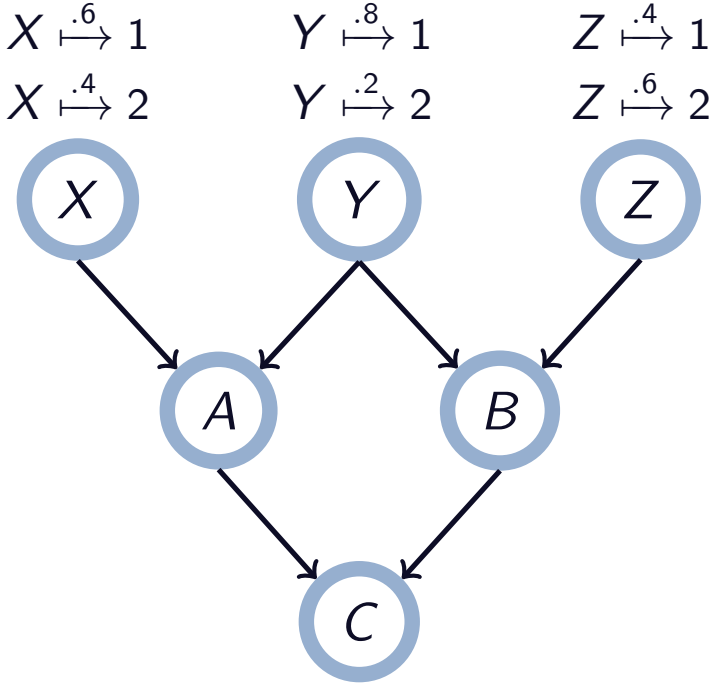
~~$Prob(X)$~~, $Prob(Y)$, ~~$Prob(Z)$~~,
~~$Prob(A \mid X, Y)$~~, ~~$Prob(B \mid Y, Z)$~~,
~~$Prob(C \mid A, B)$~~, ~~$f_1(A, Y)$~~, ~~$f_2(B, Y)$~~,
~~$f_3(B, C, Y)$~~, $f_4(C, Y)$



Variable Elimination - Example 2

- Given factors:

  ~~$Prob(X)$~~, $Prob(Y)$, ~~$Prob(Z)$~~,
  ~~$Prob(A \mid X, Y)$~~, ~~$Prob(B \mid Y, Z)$~~,
  ~~$Prob(C \mid A, B)$~~, ~~$f_1(A, Y)$~~, ~~$f_2(B, Y)$~~,
  ~~$f_3(B, C, Y)$~~, $f_4(C, Y)$



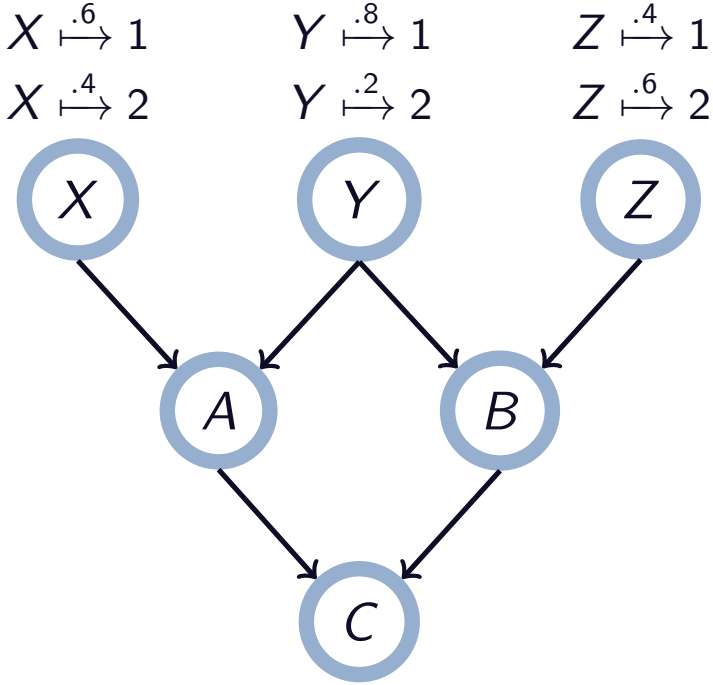
5. Elimination of variable Y by generating factor:

   $f_5(C) = Prob(C) = \sum_{y \in \{1,2\}} Prob(Y = y) \times f_4(C, Y = y)$

Variable Elimination - Example 2

- Given factors:

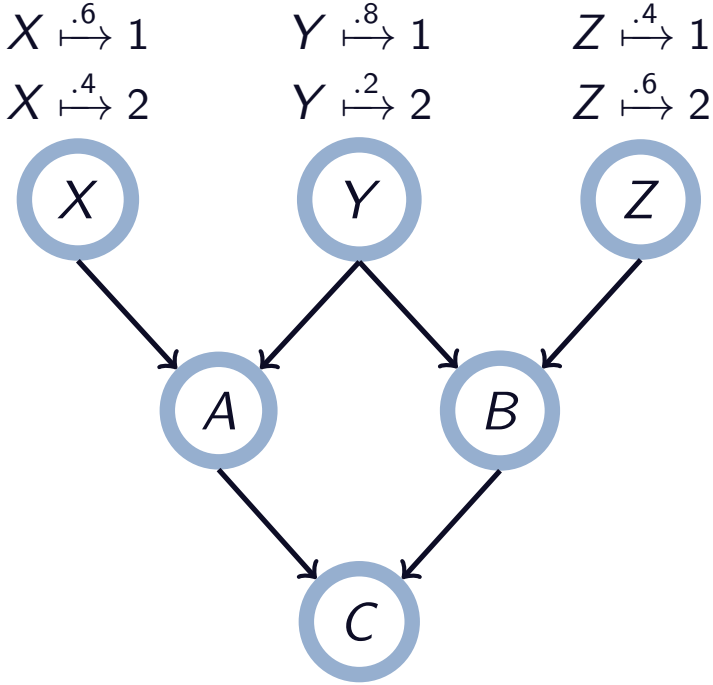
~~$Prob(X)$~~, ~~$Prob(Y)$~~, ~~$Prob(Z)$~~,
~~$Prob(A \mid X, Y)$~~, ~~$Prob(B \mid Y, Z)$~~,
~~$Prob(C \mid A, B)$~~, ~~$f_1(A, Y)$~~, ~~$f_2(B, Y)$~~,
~~$f_3(B, C, Y)$~~, ~~$f_4(C, Y)$~~, $f_5(C)$



Variable Elimination - Example 2

- Given factors:

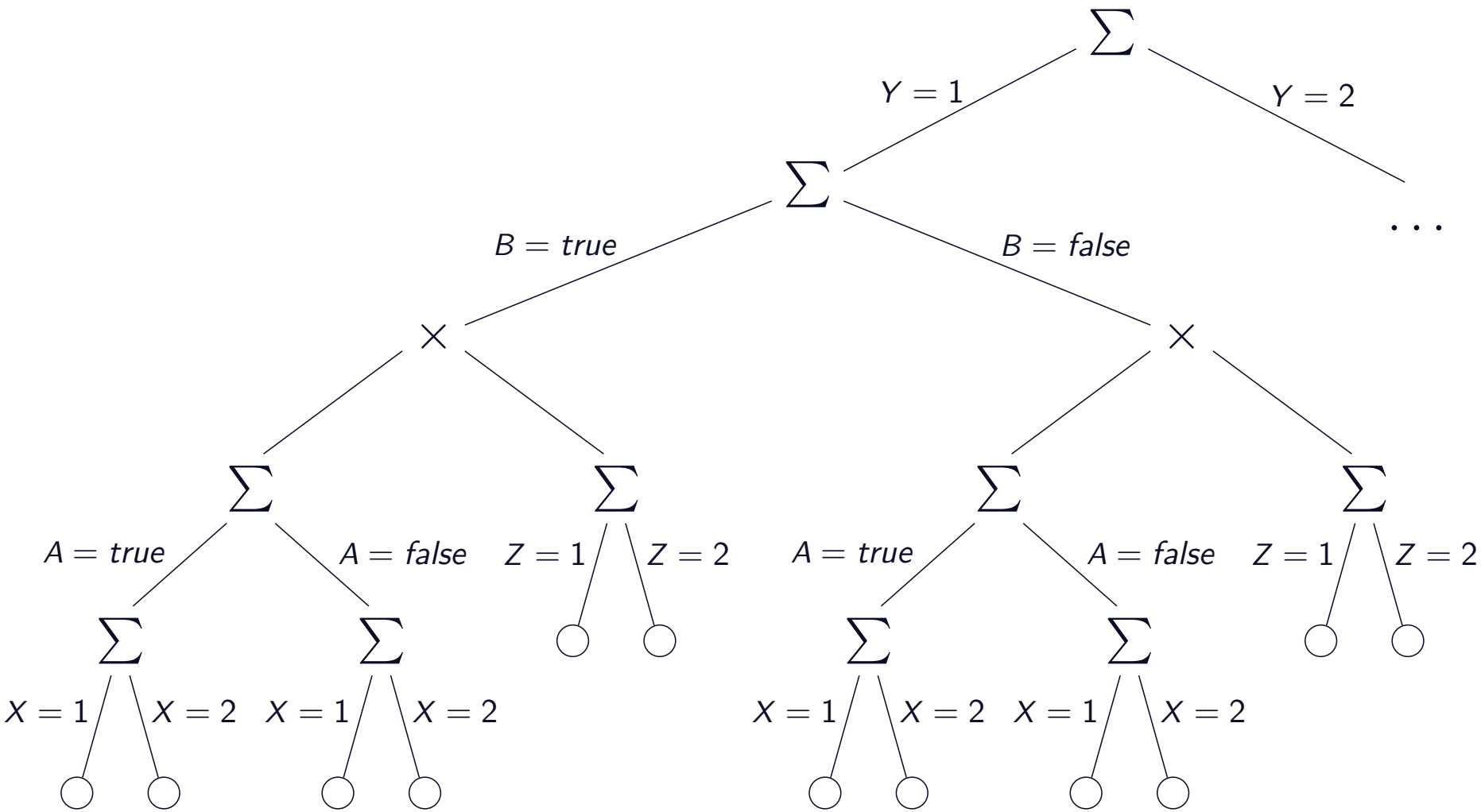
~~$Prob(X)$~~, ~~$Prob(Y)$~~, ~~$Prob(Z)$~~,
~~$Prob(A \mid X, Y)$~~, ~~$Prob(B \mid Y, Z)$~~,
~~$Prob(C \mid A, B)$~~, ~~$f_1(A, Y)$~~, ~~$f_2(B, Y)$~~,
~~$f_3(B, C, Y)$~~, ~~$f_4(C, Y)$~~, $f_5(C)$



Final Result:

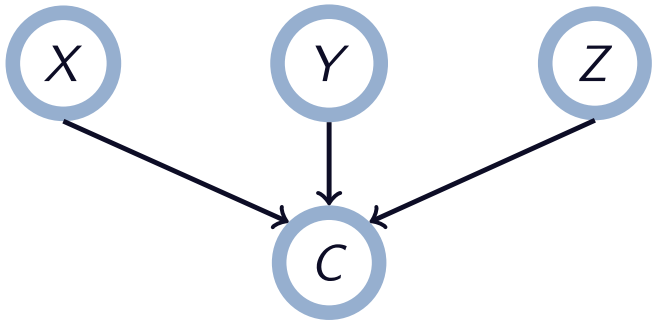
$$f_5(C) = \sum_{y \in \{1,2\}} Prob(Y = y) \times$$
$$\Big(\sum_{b \in \{t,f\}} \Big(\sum_{z \in \{1,2\}} Prob(Z = z) \times Prob(B = b \mid Y = y, Z = z) \Big) \times$$
$$\Big(\sum_{a \in \{t,f\}} \Big(\sum_{x \in \{1,2\}} Prob(X = x) \times Prob(A = a \mid X = x, Y = y) \Big) \Big) \Big)$$

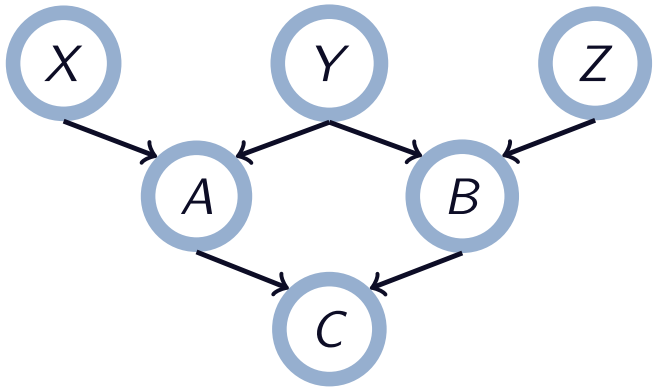
Variable Elimination - Example 2 (Operator Tree)



Variable Elimination - Approach Comparison

Example: $(X = 1 \wedge Y = 2) \vee (Y = 1 \wedge Z = 1)$

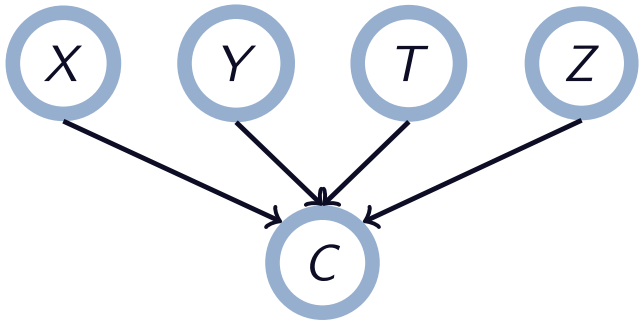


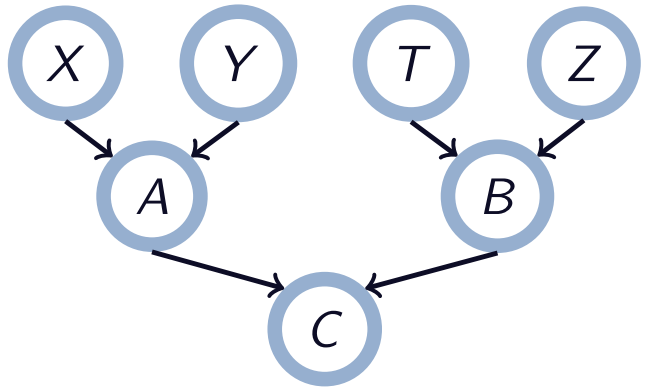


Sizes $\lvert X\rvert = \lvert Y\rvert = \lvert Z\rvert$	**Naive Approach** $\lvert X\rvert \times \lvert Y\rvert \times \lvert Z\rvert$	**Hierarchical Approach** $4\lvert X\rvert\lvert Y\rvert + 2\lvert Y\rvert\lvert Z\rvert$
2	**8**	**24**
3	27	54
5	125	150
10	1000	600

Variable Elimination - Approach Comparison

Example: $(X = 1 \wedge Y = 2) \vee (T = 1 \wedge Z = 1)$





Sizes $\lvert X\rvert = \lvert Y\rvert = \lvert T\rvert = \lvert Z\rvert$	**Naive Approach** $\lvert X\rvert \times \lvert Y\rvert \times \lvert T\rvert \times \lvert Z\rvert$	**Hierarchical Approach** $4\lvert X\rvert\lvert Y\rvert + 2\lvert T\rvert\lvert Z\rvert$
2	16	24
3	81	54
5	625	150
10	10,000	600

Problem-specific Inference Algorithms

Given Situation:

- General-purpose algorithms for probabilistic inference can be used to compute any probability distribution (joint or conditional) on any subset of variables
- We always need to compute the probability of the Boolean variable of the single leaf node
- Several potentials for complexity reduction are unused
- Probabilistic database systems often use problem-specific inference algorithms

Lineage: Normalforms

Disjunctive Normalform (DNF):

- Disjunction of conjunctions, i.e. $\bigvee_i \bigwedge_j (\neg) x_{ij}$
- Example: $(\epsilon_1 \wedge \epsilon_3 \wedge \epsilon_4) \vee (\epsilon_1 \wedge \epsilon_2) \vee (\neg\epsilon_1 \wedge \epsilon_5)$

Conjunctive Normalform (CNF):

- Conjunction of disjunctions, i.e. $\bigwedge_i \bigvee_j (\neg) x_{ij}$
- Example: $(\epsilon_1 \vee \epsilon_3 \vee \epsilon_4) \wedge (\epsilon_1 \vee \epsilon_2) \wedge (\neg\epsilon_1 \vee \epsilon_5)$

Once-occurrence Form (1OF):

- Every tuple event occurs at most once
- Example: $(\epsilon_1 \vee \epsilon_3 \wedge \epsilon_4) \vee \neg\epsilon_5 \wedge \epsilon_2$
- Also known as read-once formula

1OF for TI-databases

- Input database is tuple-independent
- Lineage formula Φ is in 1OF (i.e. each variable occurs once)

$\Rightarrow$ All atomic conditions in Φ are mutually independent

$\Rightarrow$ Probability of Φ can be computed in linear time of its size

Computation Rules:

- Identity ($\Phi \equiv \epsilon_i$): $p(\Phi) = p(\epsilon_i)$
- AND ($\Phi \equiv \Phi_i \wedge \Phi_j$): $p(\Phi) = p(\Phi_i) \times p(\Phi_j)$
- OR ($\Phi \equiv \Phi_i \vee \Phi_j$): $p(\Phi) = p(\Phi_i) + p(\Phi_j) - p(\Phi_i) \times p(\Phi_j)$
  $= 1 - (1 - p(\Phi_i)) \times (1 - p(\Phi_j))$
- Negation ($\Phi \equiv \neg\Phi_i$): $p(\Phi) = 1 - p(\Phi_i)$

Shannon Expansion

- Decomposition of a logical formula into exclusive subformulas
- Rule: $\Phi \equiv (\epsilon_i \wedge \Phi_{\langle \epsilon_i | true \rangle}) \oplus (\neg \epsilon_i \wedge \Phi_{\langle \epsilon_i | false \rangle})$ where
  - $\oplus$ is the logical XOR operator (mutual exclusion)
  - $\Phi_{\langle \epsilon | true \rangle}$ (or $\Phi_{\langle \epsilon | false \rangle}$) means that every occurrence of event ϵ in formula Φ is replaced with the logical constant *true* (or *false*)
- Example:
  $(\epsilon_2 \vee (\epsilon_1 \wedge \epsilon_4)) \wedge (\epsilon_1 \vee \epsilon_3) \equiv (\epsilon_1 \wedge (\epsilon_2 \vee \epsilon_4)) \oplus (\neg \epsilon_1 \wedge (\epsilon_2 \wedge \epsilon_3))$
- Probability: $p(\Phi) = p(\epsilon_i \wedge \Phi_{\langle \epsilon_i | true \rangle}) + p(\neg \epsilon_i \wedge \Phi_{\langle \epsilon_i | false \rangle})$
- Subsequent application until every subformula is in 1OF
- Problem: Formula size doubles in each expansion step

$\Rightarrow$ Computation of $p(\Phi)$ can require exponential time in the size of the initial formula

1OF for pc-databases

pc-database:

- Every tuple event corresponds to a condition defined on variable assignments

$\Rightarrow$ Different tuple events are not necessarily independent

Once-occurrence Form in pc-databases:

- 1OF does not guarantee that all atomic conditions are independent ($\Rightarrow$ rules from Slide 37 cannot be used)

$\Rightarrow$ Redefinition: Formula in 1OF only if every variable occurs at most once

- Example 1: $((X = 1) \vee (Y = 2) \wedge (Z = 1))$ is in 1OF
- Example 2: $((X = 1) \vee (Y = 2) \wedge (X = 3))$ is not in 1OF
- New Identity Rule: $\Phi \equiv (X = i) \Rightarrow p(\Phi) = Prob(X = i)$

Shannon Expansion for pc-databases

- One exclusive subformula per possible assignment of expanded variable X

$\Rightarrow$ Rule: $\Phi \equiv \bigoplus_{i=1}^{n}((X = i) \wedge \Phi_{\langle X|i \rangle})$

Example:

- $\Phi = ((X = 1) \vee ((X = 2) \wedge (Y = 1))) \wedge ((Z = 1) \vee (Y = 2))$
- Not in 1OF because X and Y occur more than once

Shannon Expansion for pc-databases

- One exclusive subformula per possible assignment of expanded variable X

$\Rightarrow$ Rule: $\Phi \equiv \bigoplus_{i=1}^{n}((X = i) \wedge \Phi_{\langle X|i \rangle})$

Example:

- $\Phi = ((X = 1) \vee ((X = 2) \wedge (Y = 1))) \wedge ((Z = 1) \vee (Y = 2))$
- Not in 1OF because X and Y occur more than once

Expansion on X:

$$\begin{aligned} \Phi_{\langle X|1 \rangle} &= (\mathit{true} \vee (\mathit{false} \wedge (Y = 1))) \wedge ((Z = 1) \vee (Y = 2)) \\ &\equiv (Z = 1) \vee (Y = 2) && \Rightarrow \text{1OF} \end{aligned}$$

$$\begin{aligned} \Phi_{\langle X|2 \rangle} &= (\mathit{false} \vee (\mathit{true} \wedge (Y = 1))) \wedge ((Z = 1) \vee (Y = 2)) \\ (\equiv \Phi^*) &\equiv (Y = 1) \wedge ((Z = 1) \vee (Y = 2)) && \Rightarrow \neg\text{1OF} \end{aligned}$$

Shannon Expansion for pc-databases

- One exclusive subformula per possible assignment of expanded variable X

$\Rightarrow$ Rule: $\Phi \equiv \bigoplus_{i=1}^{n}((X = i) \wedge \Phi_{\langle X|i \rangle})$

Example:

- $\Phi = ((X = 1) \vee ((X = 2) \wedge (Y = 1))) \wedge ((Z = 1) \vee (Y = 2))$
- Not in 1OF because X and Y occur more than once

Expansion on Y:

$$\begin{aligned} \Phi^*_{\langle Y|1\rangle} &= (\mathit{true} \wedge ((Z = 1) \vee \mathit{false})) \\ &\equiv (Z = 1) && \Rightarrow \text{1OF} \end{aligned}$$

$$\begin{aligned} \Phi^*_{\langle Y|2\rangle} &= (\mathit{false} \wedge ((Z = 1) \vee \mathit{true})) \\ &\equiv \mathit{false} && \Rightarrow \text{1OF} \end{aligned}$$

Shannon Expansion for pc-databases

- One exclusive subformula per possible assignment of expanded variable X

$\Rightarrow$ Rule: $\Phi \equiv \bigoplus_{i=1}^{n}((X = i) \wedge \Phi_{\langle X|i \rangle})$

Example:

- $\Phi = ((X = 1) \vee ((X = 2) \wedge (Y = 1))) \wedge ((Z = 1) \vee (Y = 2))$
- Not in 1OF because X and Y occur more than once

$$
\begin{aligned}
\Phi &\equiv (X = 1) \wedge \Phi_{\langle X|1 \rangle} \oplus (X = 2) \wedge \Phi_{\langle X|2 \rangle} \\
&\equiv (X = 1) \wedge \Phi_{\langle X|1 \rangle} \oplus (X = 2) \\
&\quad \wedge ((Y = 1) \wedge \Phi^{*}_{\langle Y|1 \rangle} \oplus (Y = 2) \wedge \Phi^{*}_{\langle Y|2 \rangle}) \\
&\equiv (X = 1) \wedge ((Z = 1) \vee (Y = 2)) \oplus (X = 2) \wedge (Y = 1) \wedge (Z = 1)
\end{aligned}
$$

Inference using Shannon Expansion and 1OF

- Probability computation in two steps
- Step 1: Create an expanded lineage formula which is in 1OF

$$Shannon(\Phi) = \begin{cases} \Phi, & \text{if } \Phi \text{ is in 1OF,} \\ \bigoplus_{i=1}^{n} (X = i) \wedge Shannon(\Phi_{\langle X|i \rangle}), & \text{else.} \end{cases}$$

- Step 2: Compute probabilities according to rules from Slide 37 and the following rule for exclusive-ORs

$$\text{XOR } (\Phi = \bigoplus_{i=1}^{n} \Phi_i): \qquad p(\Phi) = \sum_{i=1}^{n} p(\Phi_i)$$

- Problem: Does not exploit independencies between subconditions

Inference using Shannon Expansion and 1OF - Example 1

- $\Phi = (X = 1 \wedge Y = 2) \vee (Y = 1 \wedge Z = 1)$
- Not in 1OF because Y occurs more than once

$\Rightarrow$ Expansion of Y

$$\begin{aligned} \Phi &\equiv (Y = 1) \wedge \Phi_{\langle Y|1 \rangle} \oplus (Y = 2) \wedge \Phi_{\langle Y|1 \rangle} \\ &\equiv (Y = 1) \wedge (Z = 1) \oplus (Y = 2) \wedge (X = 1) \end{aligned}$$

- Each subcondition is now in 1OF
- Condition actually corresponds to the original one, but now we know that the OR is exclusive
- Probability computation:

$$p(\Phi) = Prob(Y = 1) \times Prob(Z = 1) + Prob(Y = 2) \times Prob(X = 1)$$

- Much more efficient than Variable Elimination

Inference using Shannon Expansion and 1OF - Example 2

- $\Phi = (Y = 2 \wedge X = 1) \vee (Y = 2 \wedge Z = 1) \vee (Y = 1 \wedge Z = 2)$
  $\vee (Y = 2 \wedge T = 1) \vee (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)$
- Condition is in DNF but not in 1OF

$$
\begin{aligned}
\Phi \equiv \quad & (Y = 1) \wedge (Z = 2 \vee (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)) \\
& \oplus (Y = 2) \wedge (X = 1 \vee Z = 1 \vee T = 1 \vee (U = 1 \wedge V = 1) \\
& \qquad \vee (U = 2 \wedge W = 2)) \\
\equiv \quad & (Y = 1) \wedge ((U = 1) \wedge (Z = 2 \vee V = 1) \\
& \qquad \oplus (U = 2) \wedge (Z = 2 \vee W = 2)) \\
& \oplus (Y = 2) \wedge ((U = 1) \wedge (X = 1 \vee Z = 1 \vee T = 1 \vee V = 1) \\
& \qquad \oplus (U = 2) \wedge (X = 1 \vee Z = 1 \vee T = 1 \vee W = 2))
\end{aligned}
$$

- Unnecessarily complex since several subconditions are independent

Inference using Shannon Expansion and 1OF - Example 2

- $\Phi = (Y = 2 \wedge X = 1) \vee (Y = 2 \wedge Z = 1) \vee (Y = 1 \wedge Z = 2)$
  $\vee (Y = 2 \wedge T = 1) \vee (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)$

- Better: Split Φ into independent subconditions

  $\Phi_1 \equiv (Y = 2 \wedge X = 1) \vee (Y = 2 \wedge Z = 1) \vee (Y = 1 \wedge Z = 2)$
  $\vee (Y = 2 \wedge T = 1)$

  $\Phi_2 \equiv (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)$

- Then process each of them separately, i.e.:

  $\Phi_1 \equiv (Y = 1) \wedge (Z = 2) \oplus (Y = 2) \wedge (X = 1 \vee Z = 1 \vee T = 1)$

  $\Phi_2 \equiv (U = 1) \wedge (V = 1) \oplus (U = 2) \wedge (W = 2)$

  and combine them with an independent-OR

Simple Rule-based Approach (Used in MayBMS system)

Phase 1: Top-down Tree Construction

- Alternate execution of two rules until all variables are eliminated
  1. Decomposition of each condition into (syntactically) independent sub-conditions (i.e. conditions which do not share variables)
  2. Elimination of one variable per sub-condition

Phase 2: Bottom-up Probability Computation

- Given: Node N and its child nodes $\{N_1, \ldots, N_k\}$
  1. Leaf nodes represent the logical constant *true* and have the probability 1.0
  2. Decomposition ($\otimes$-node): $1 - \prod_{i=1}^{k}(1 - Prob(N_i))$
  3. Elimination of X ($\oplus$-node): $\sum_{i=1}^{k} Prob(X = i) \times Prob(N_i)$

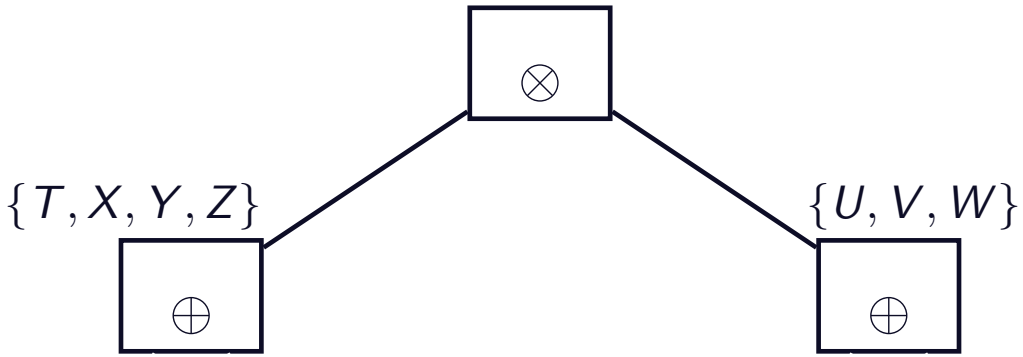
Limited to conditions in disjunctive normal form (DNF)

Simple Rule-based Approach - Example 1

$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$

Simple Rule-based Approach - Example 1

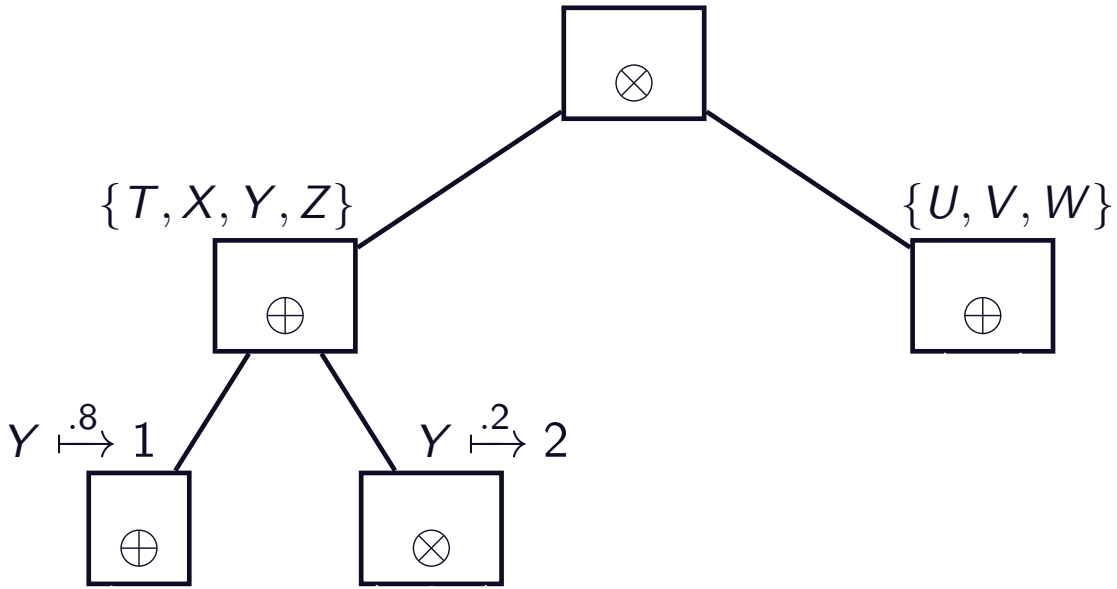
$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



Decomposition

Simple Rule-based Approach - Example 1

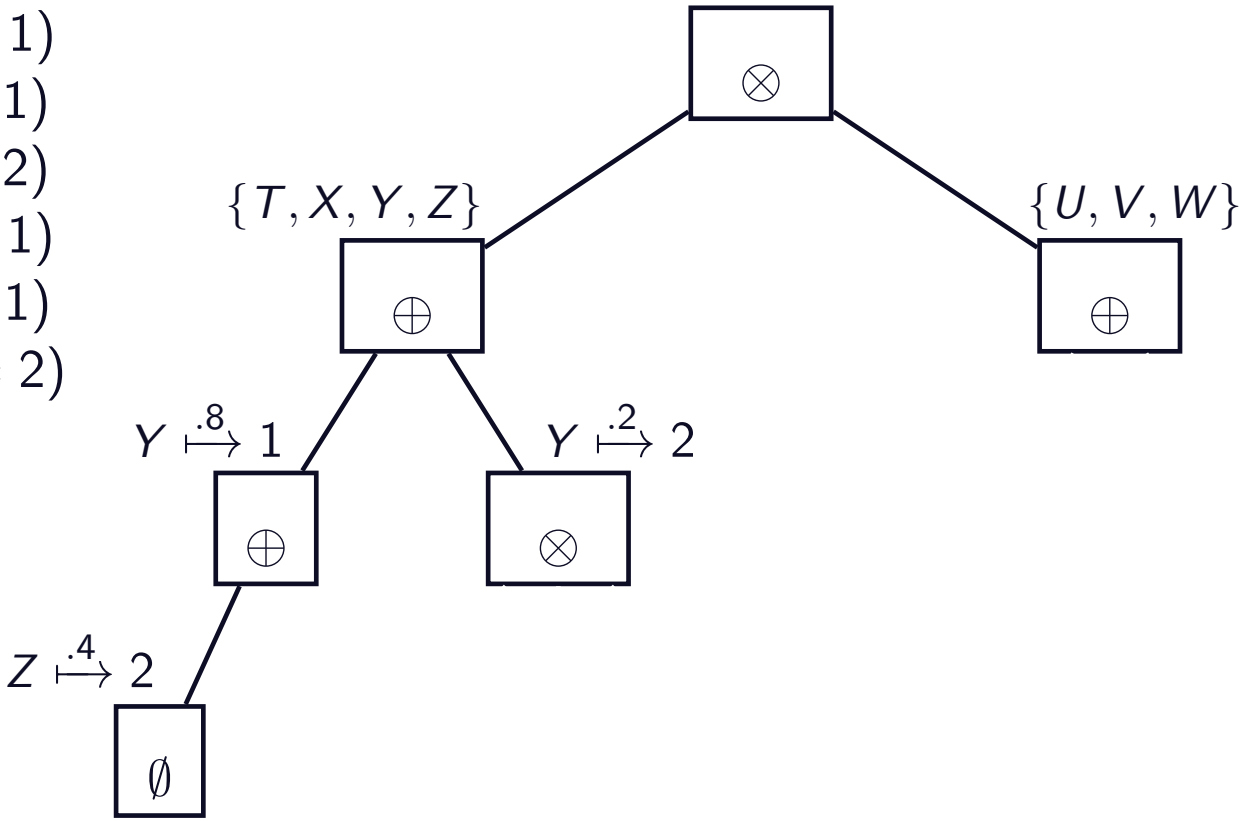
$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



Elimination of Y

Simple Rule-based Approach - Example 1

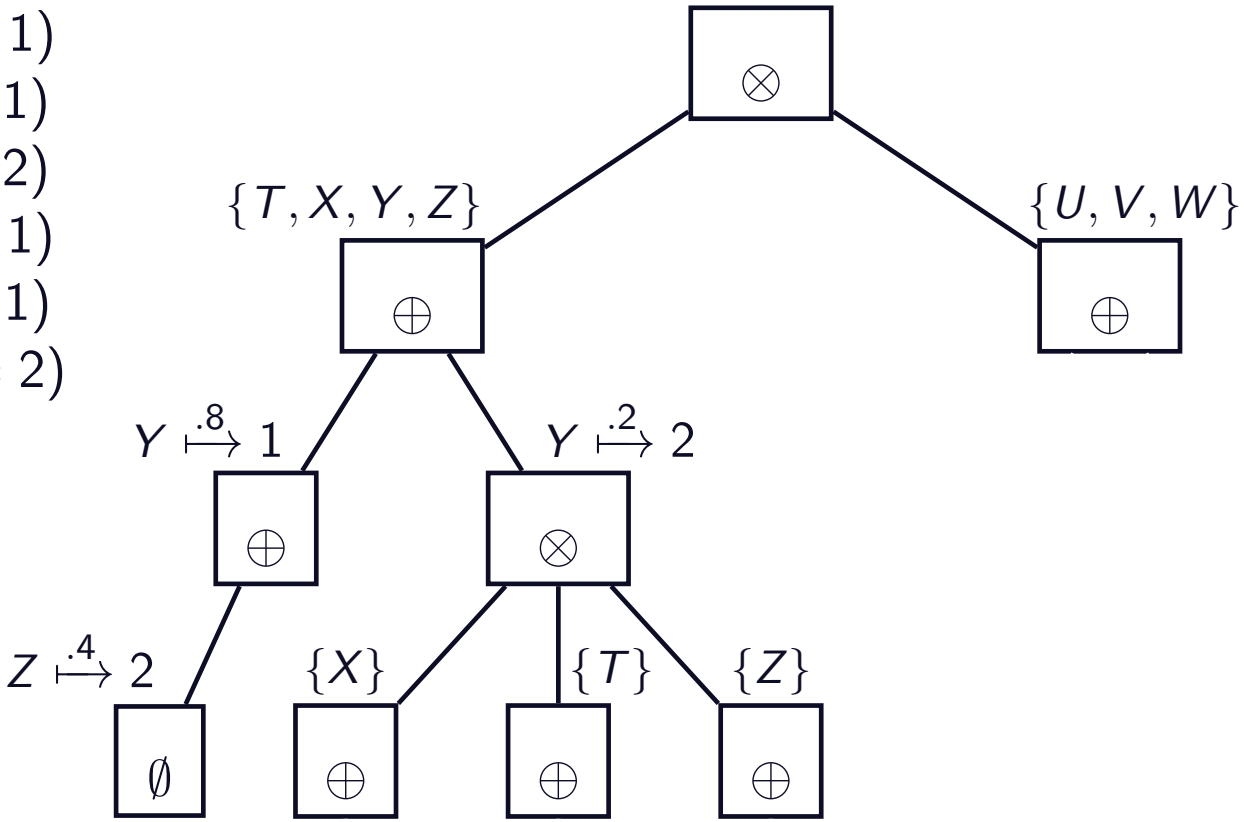
$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



Decomposition (not shown), Elimination of Z

Simple Rule-based Approach - Example 1

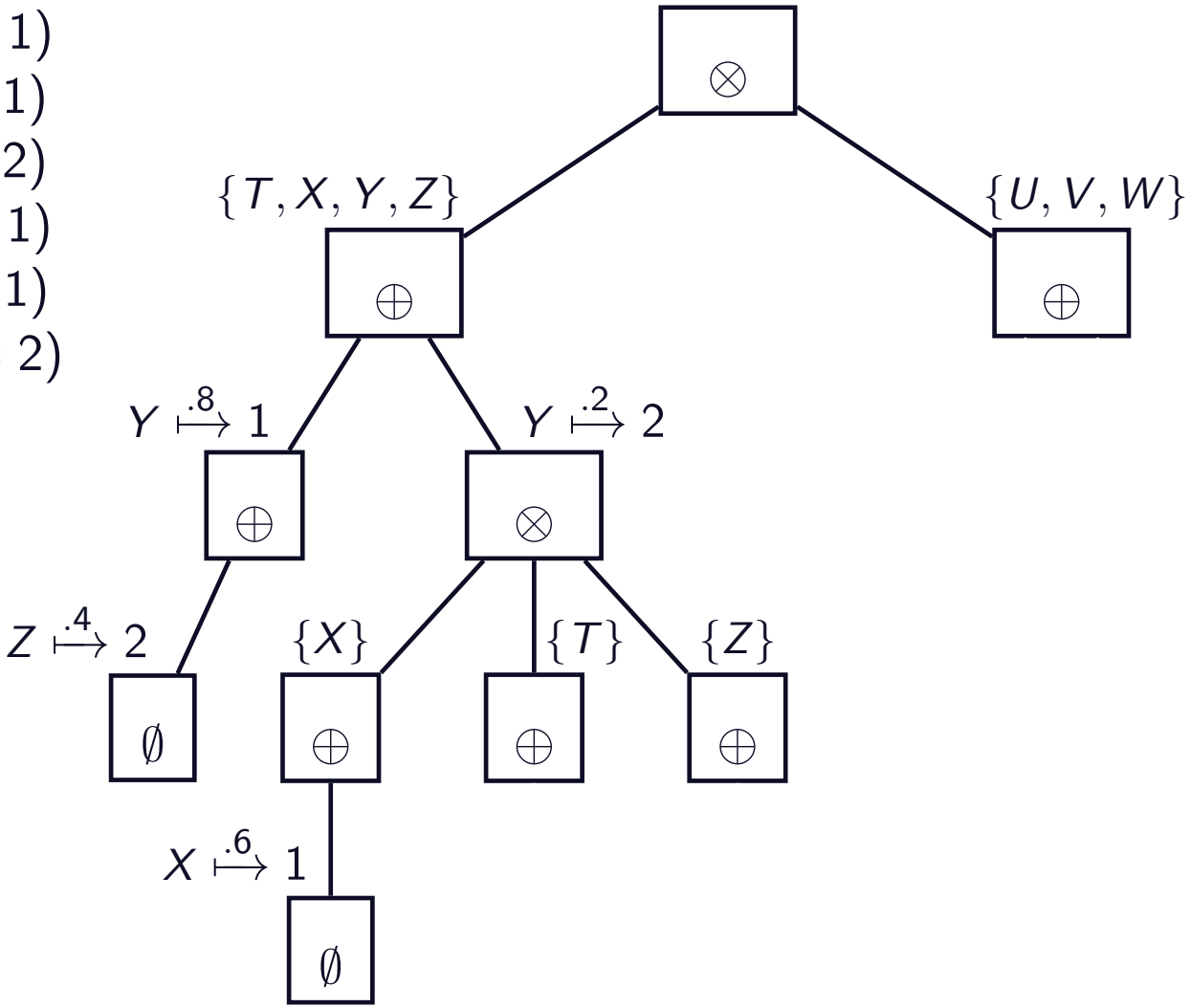
$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



Decomposition

Simple Rule-based Approach - Example 1

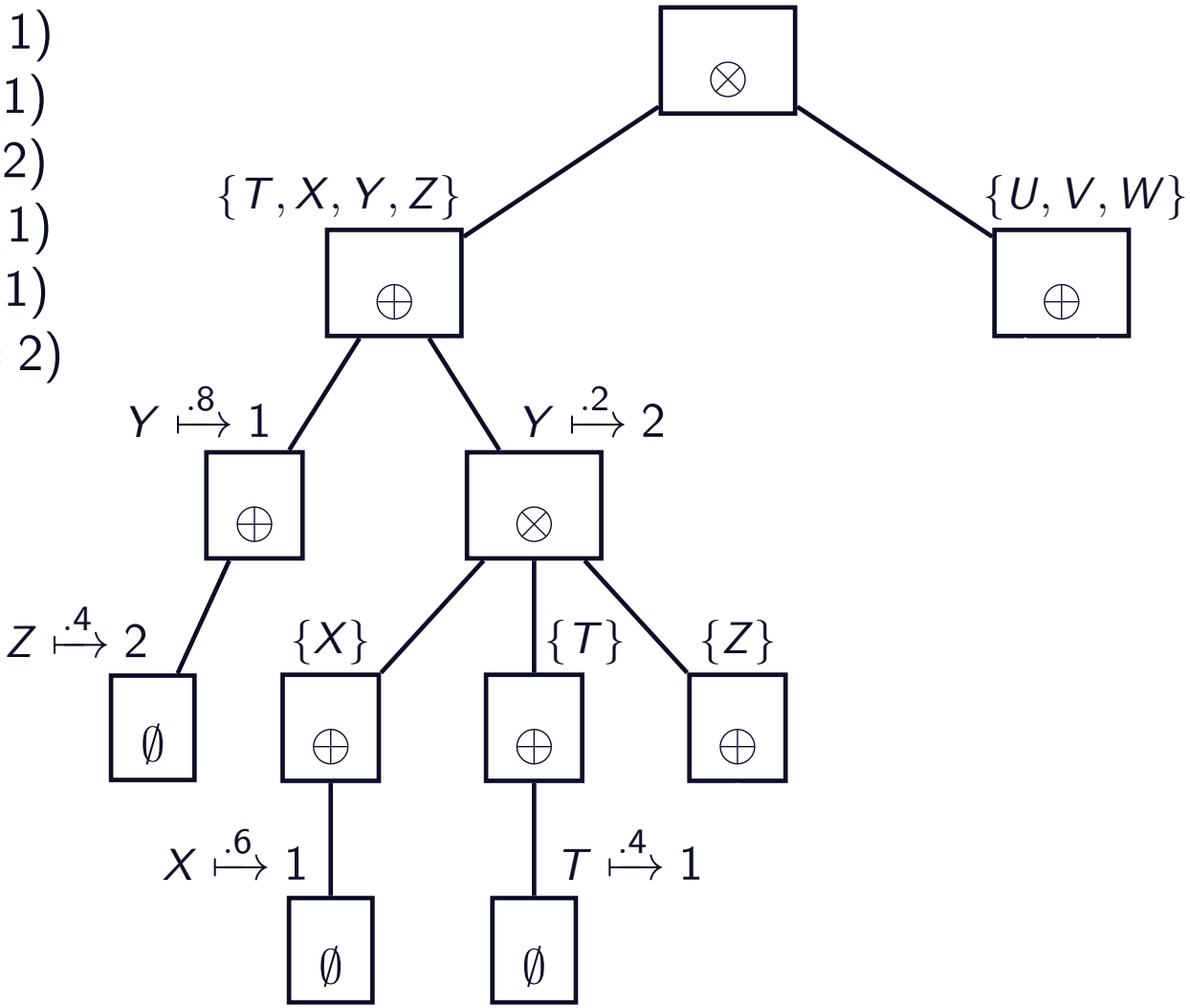
$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



Decomposition (not shown), Elimination of X

Simple Rule-based Approach - Example 1

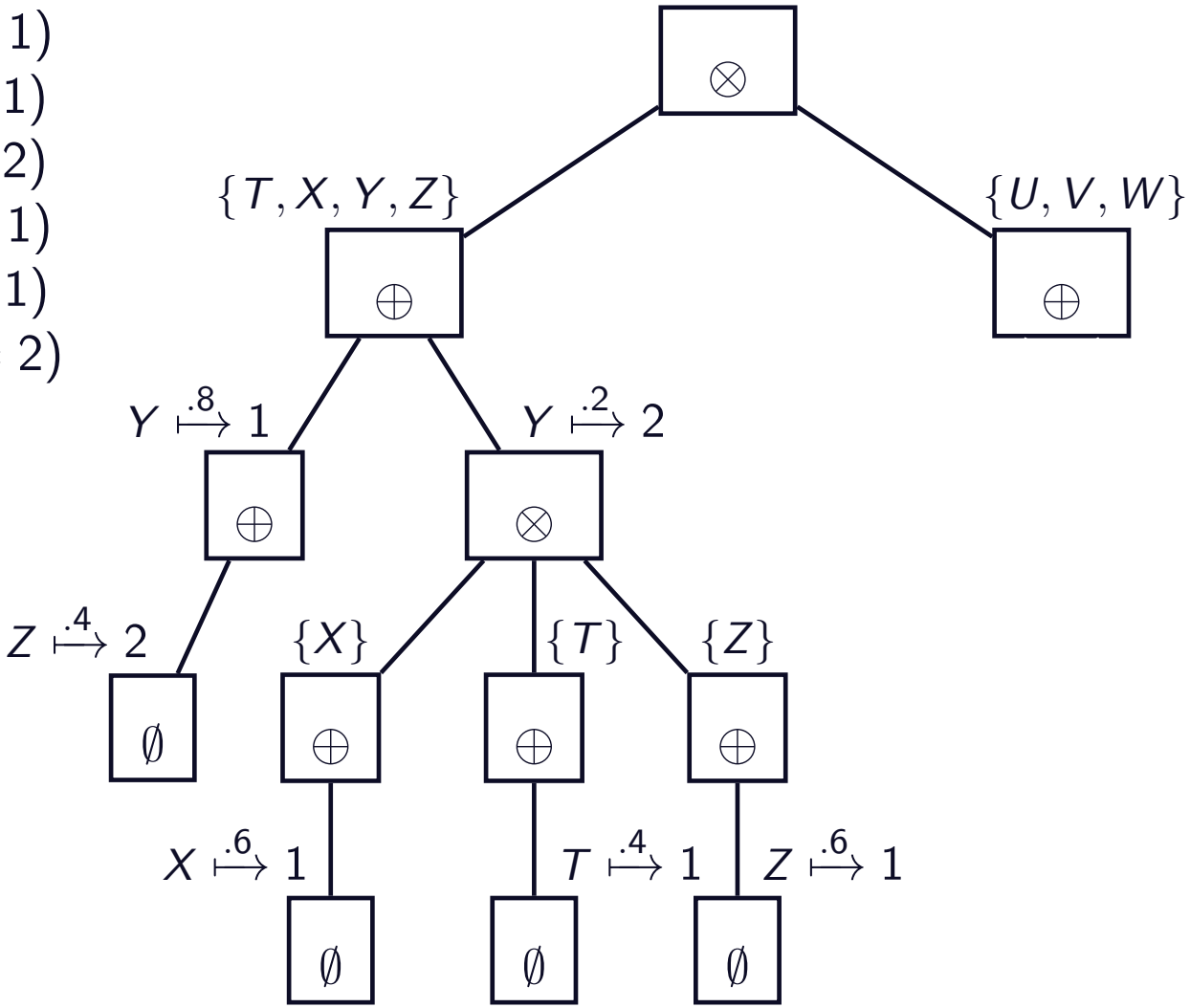
$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



Decomposition (not shown), Elimination of T

Simple Rule-based Approach - Example 1

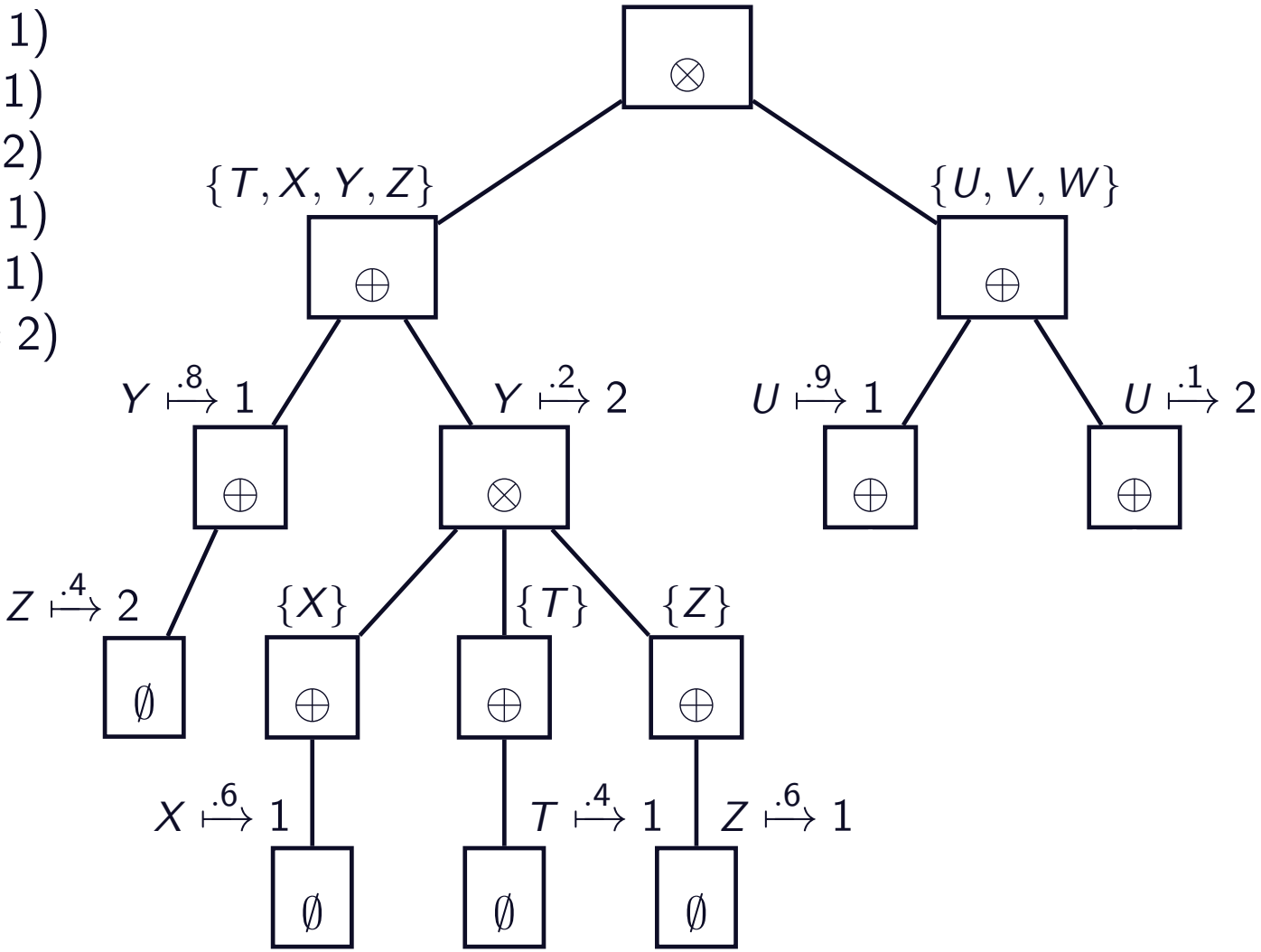
$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



Decomposition (not shown), Elimination of Z

Simple Rule-based Approach - Example 1

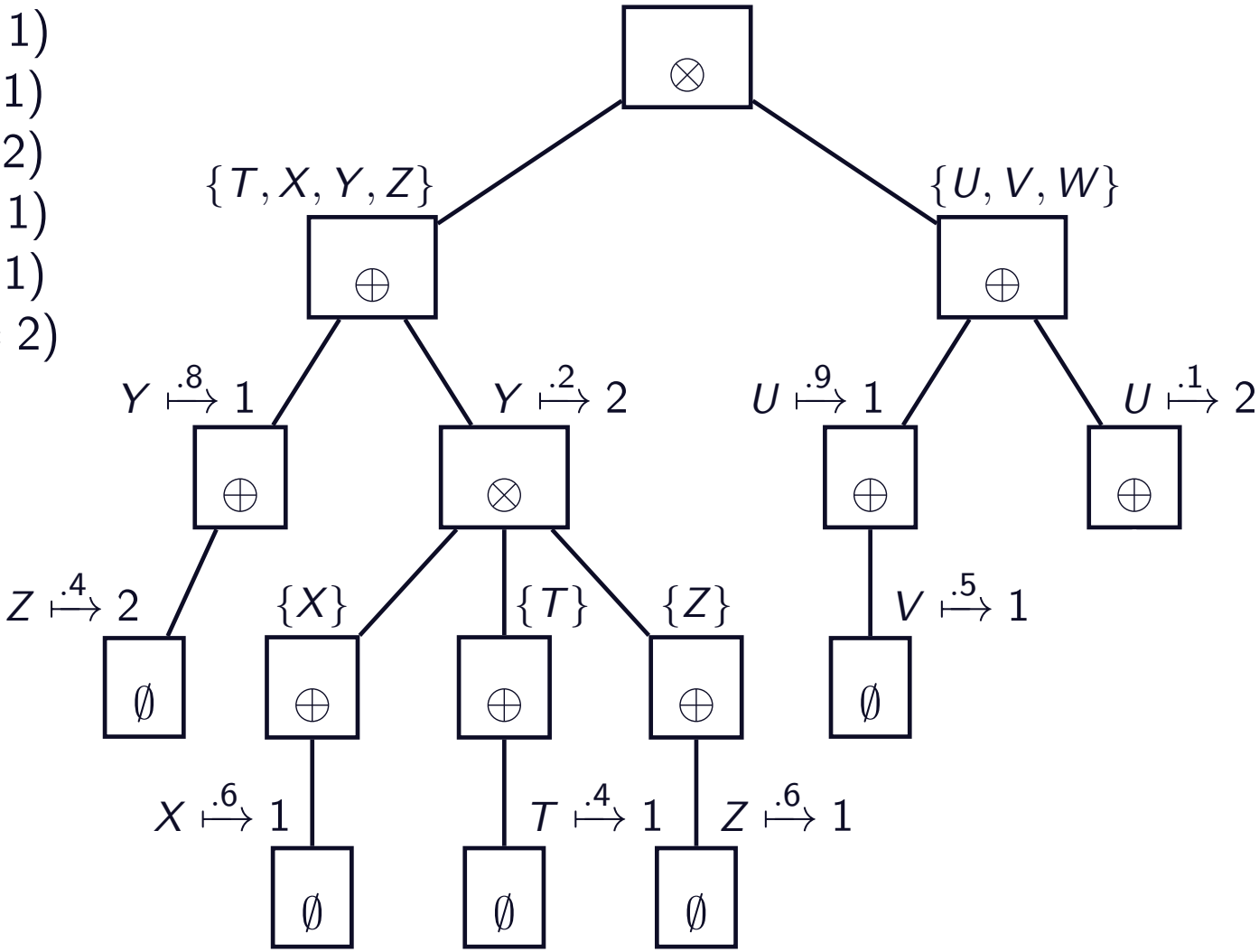
$$\Phi = (Y = 2 \wedge X = 1) \vee (Y = 2 \wedge Z = 1) \vee (Y = 1 \wedge Z = 2) \vee (Y = 2 \wedge T = 1) \vee (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)$$



Elimination of U

Simple Rule-based Approach - Example 1

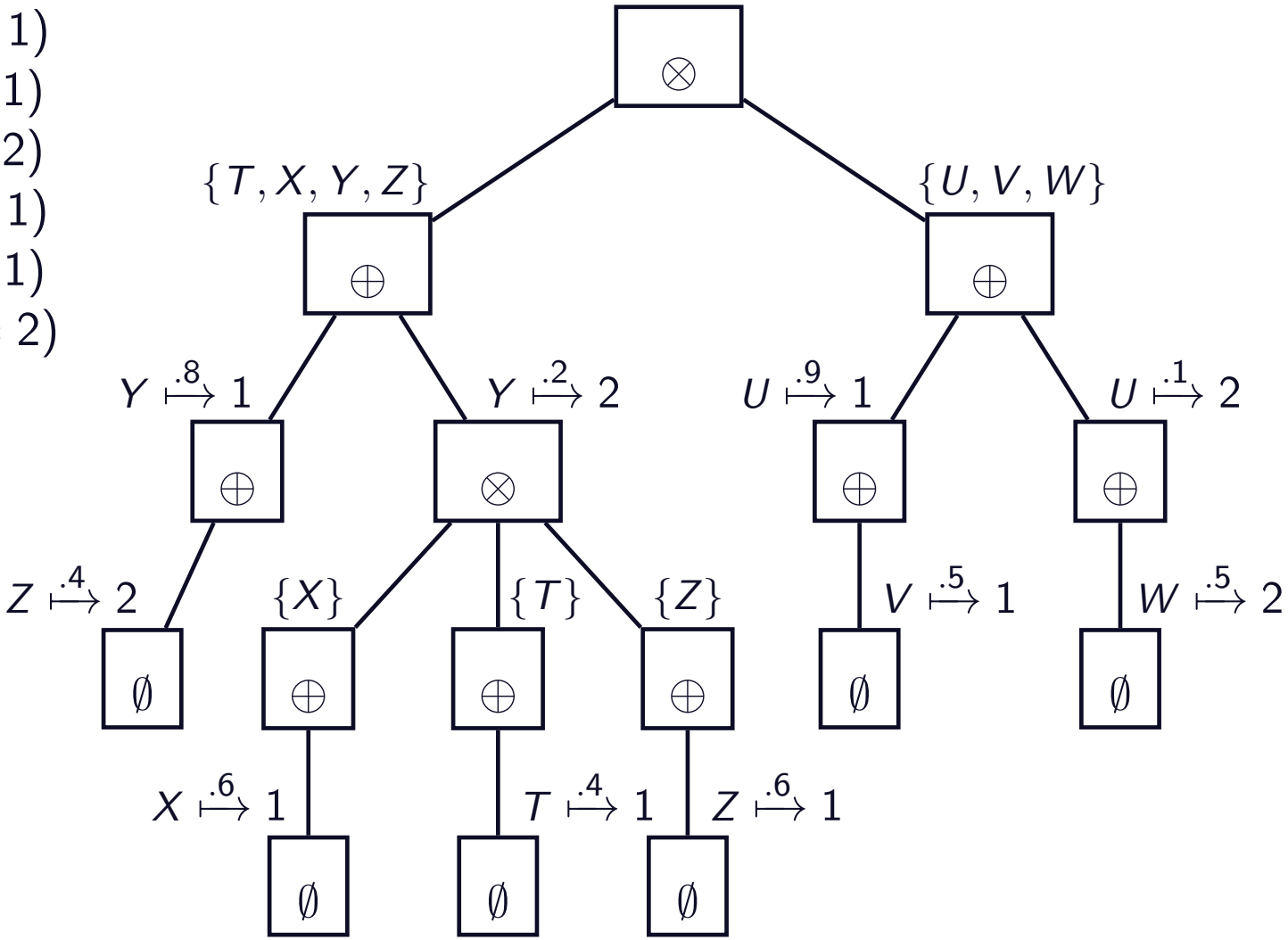
$$\Phi = (Y = 2 \wedge X = 1) \vee (Y = 2 \wedge Z = 1) \vee (Y = 1 \wedge Z = 2) \vee (Y = 2 \wedge T = 1) \vee (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)$$



Decomposition (not shown), Elimination of V

Simple Rule-based Approach - Example 1

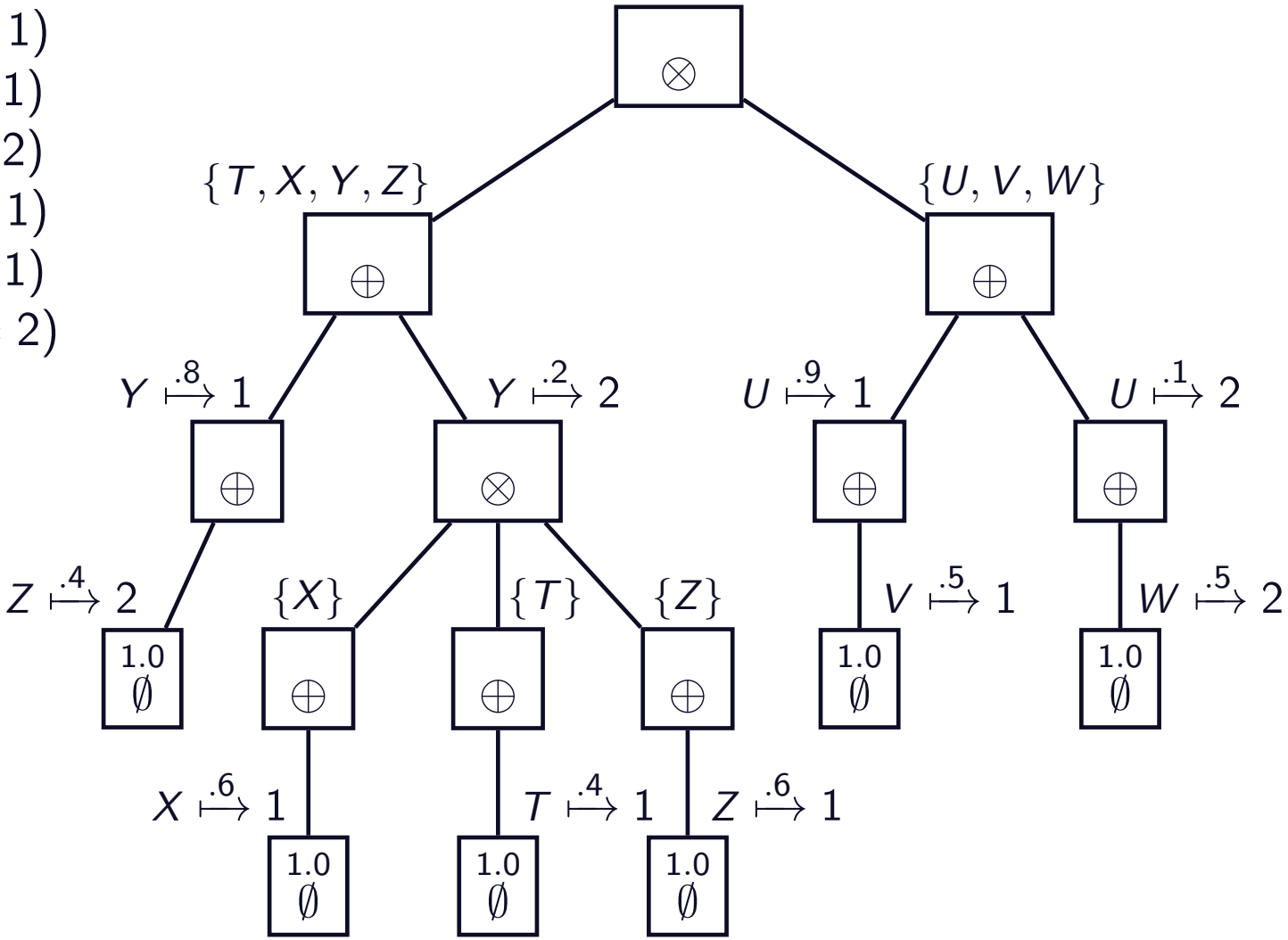
$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



Decomposition (not shown), Elimination of W

Simple Rule-based Approach - Example 1

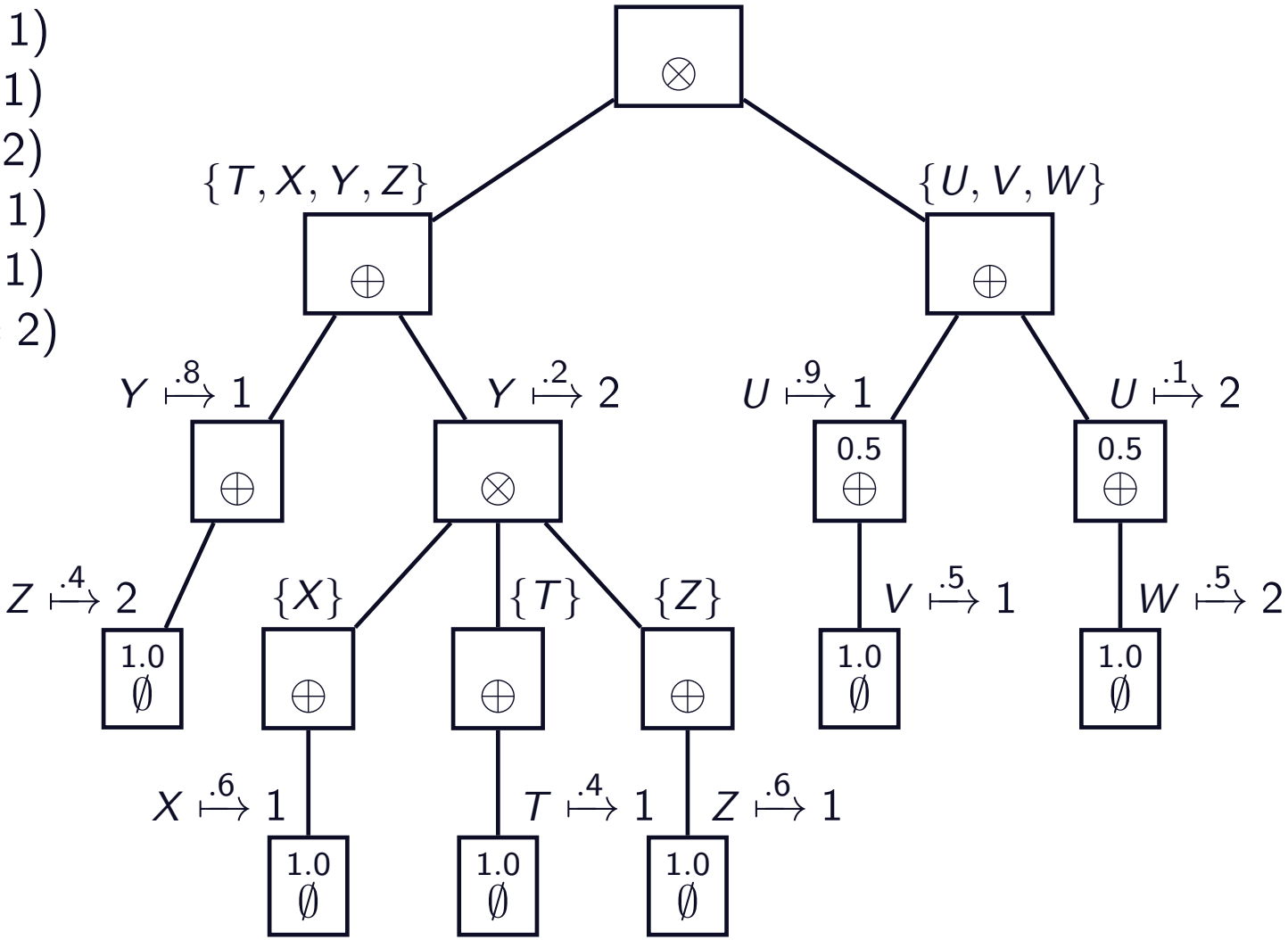
$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



Leaf nodes have probability 1.0

Simple Rule-based Approach - Example 1

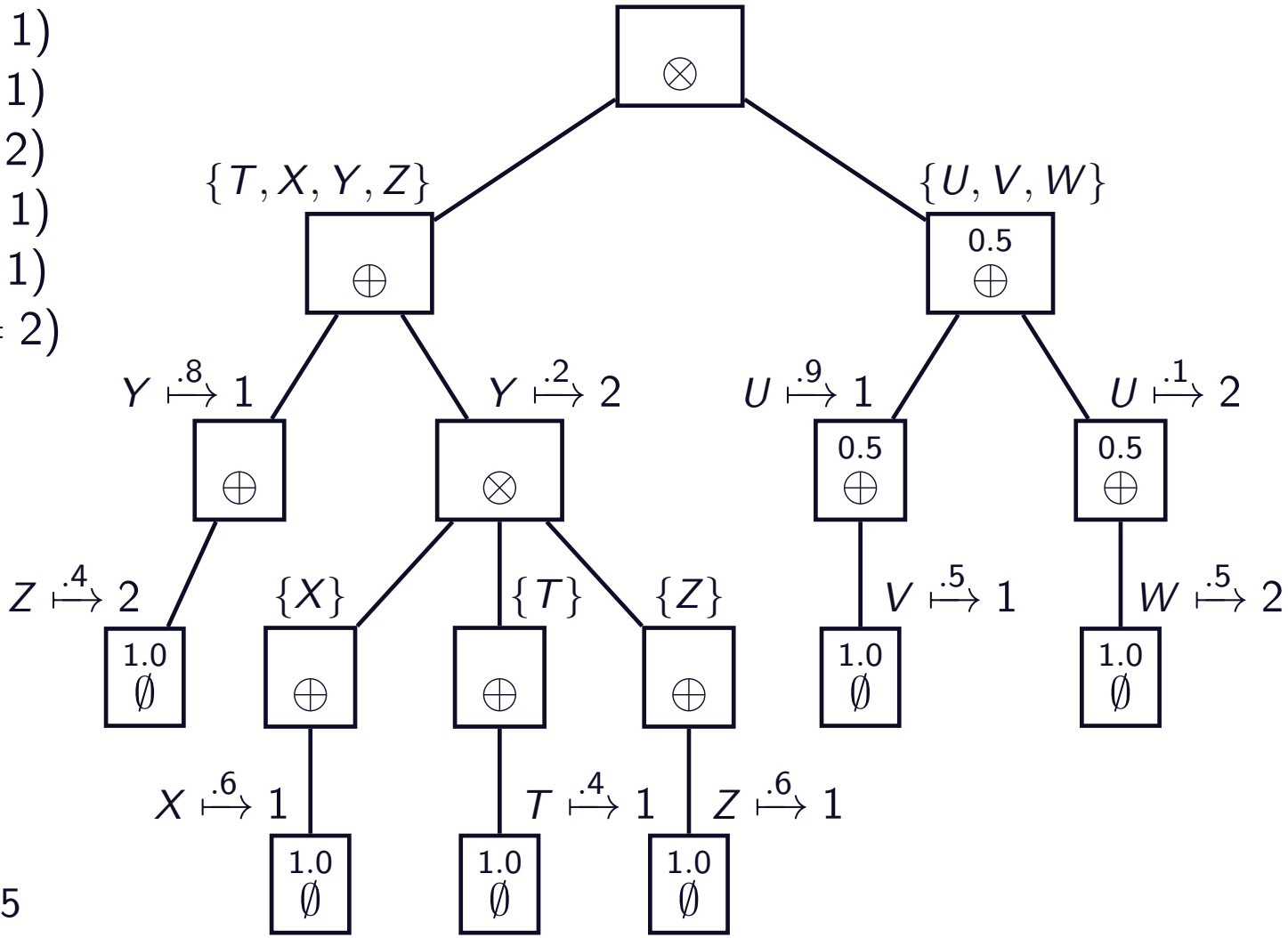
$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



Variable Elimination:
$Prob(N) = 0.5 \times 1.0$

Simple Rule-based Approach - Example 1

$$\Phi = (Y = 2 \wedge X = 1) \vee (Y = 2 \wedge Z = 1) \vee (Y = 1 \wedge Z = 2) \vee (Y = 2 \wedge T = 1) \vee (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)$$

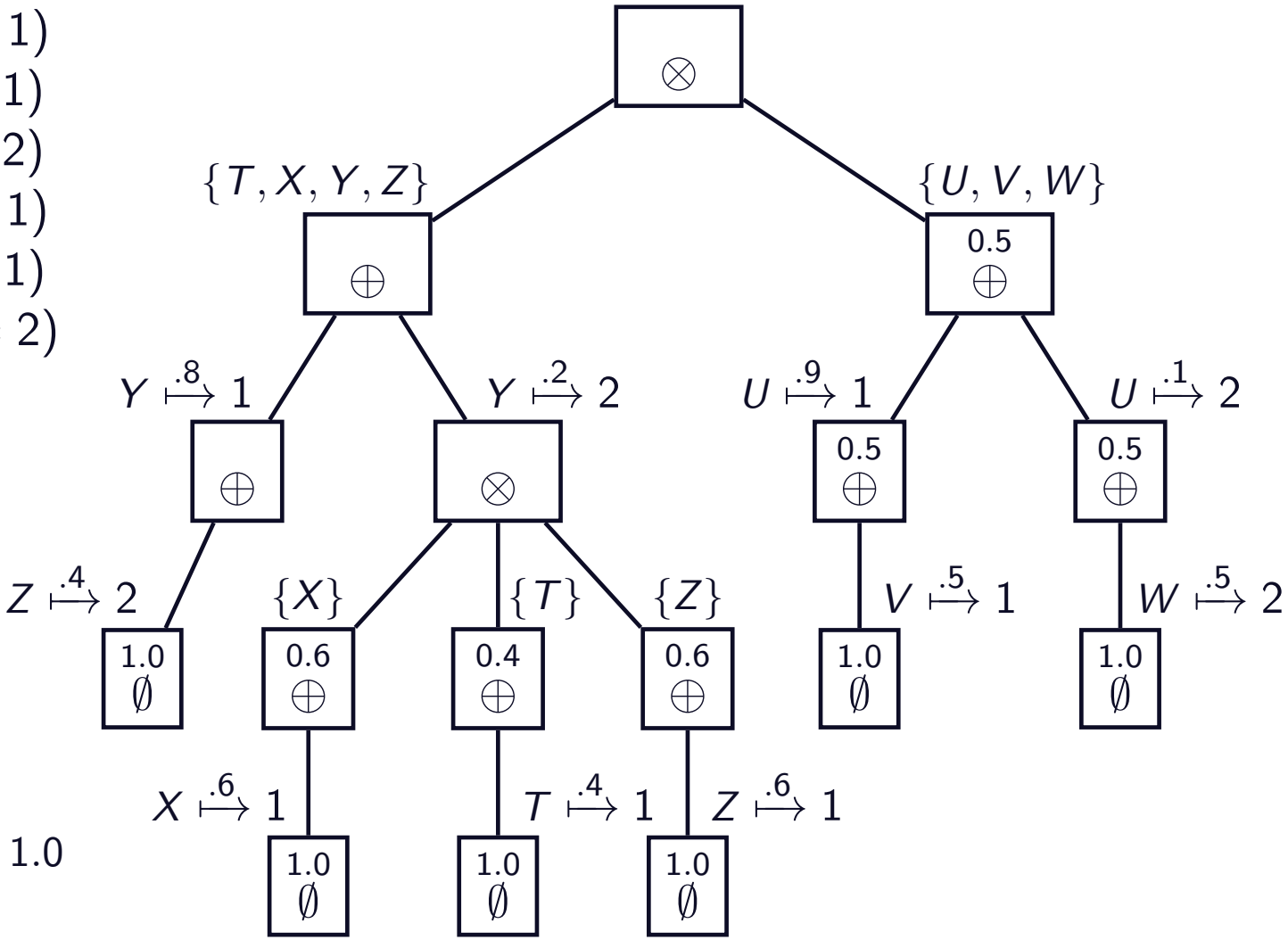


Variable Elimination:

$Prob(N) = 0.9 \times 0.5 + 0.1 \times 0.5$

Simple Rule-based Approach - Example 1

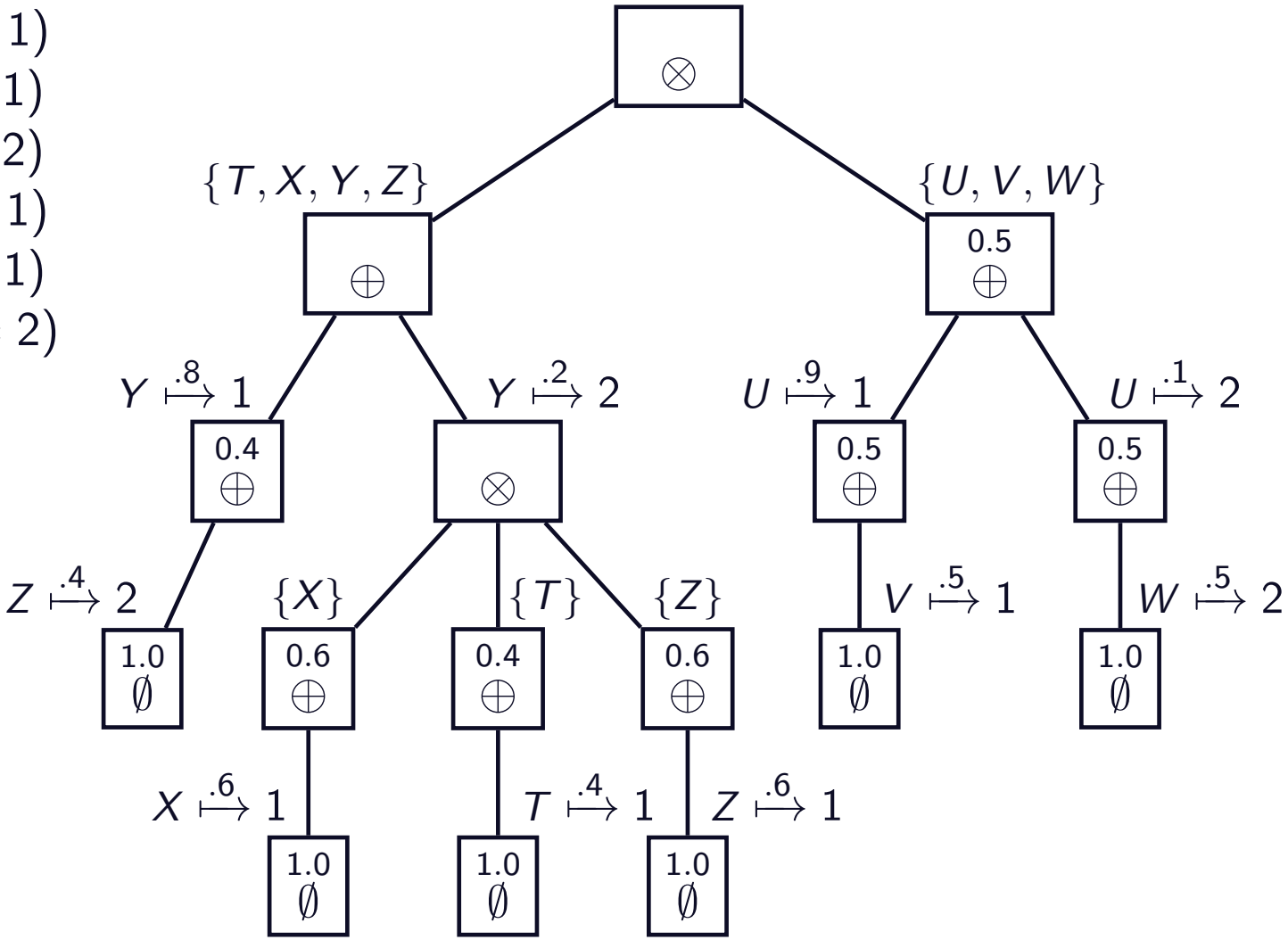
$$\Phi = (Y = 2 \wedge X = 1) \vee (Y = 2 \wedge Z = 1) \vee (Y = 1 \wedge Z = 2) \vee (Y = 2 \wedge T = 1) \vee (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)$$



Variable Elimination:
e.g. $Prob(N) = 0.6 \times 1.0$

Simple Rule-based Approach - Example 1

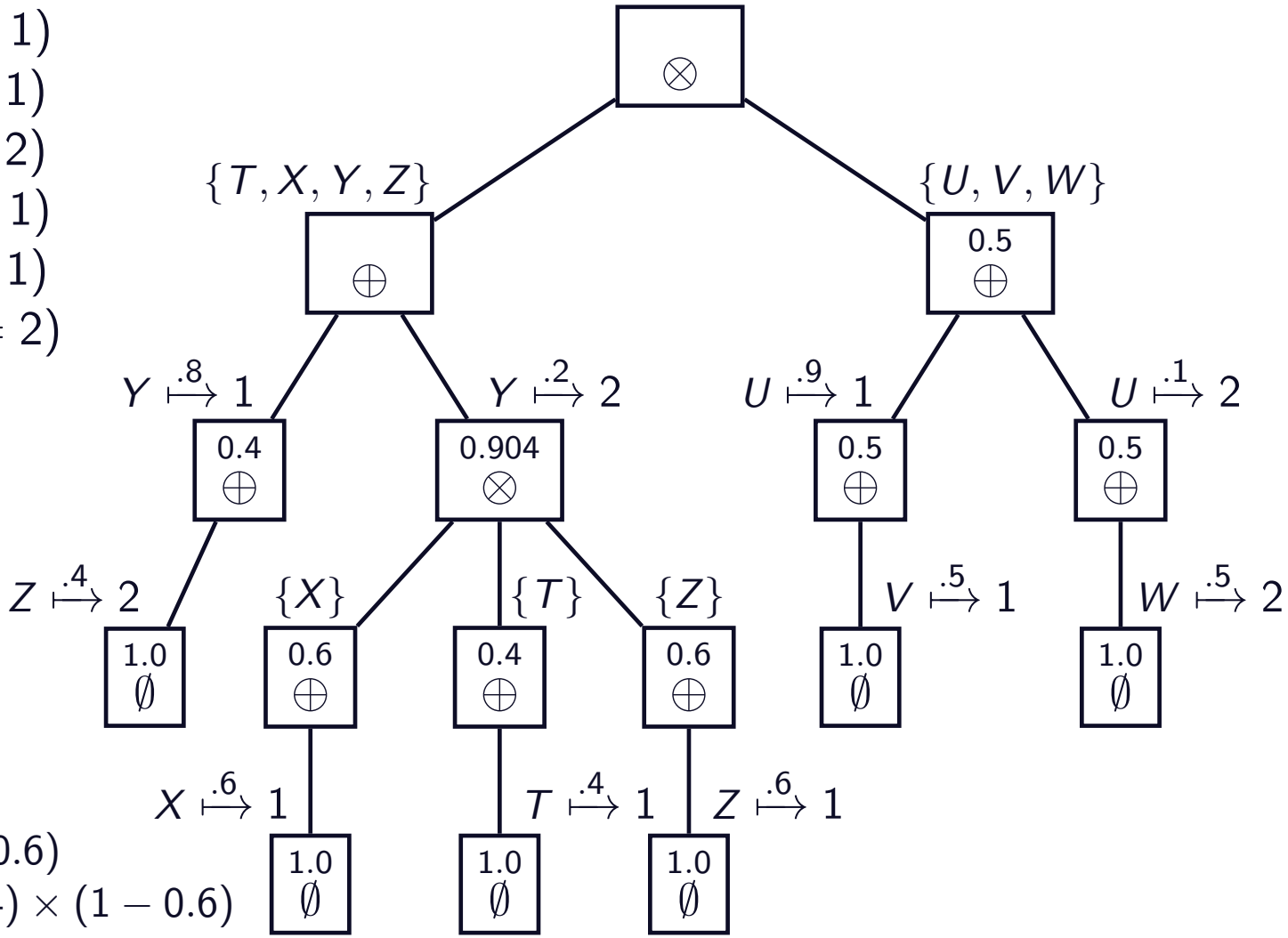
$$\Phi = (Y = 2 \wedge X = 1) \vee (Y = 2 \wedge Z = 1) \vee (Y = 1 \wedge Z = 2) \vee (Y = 2 \wedge T = 1) \vee (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)$$



Variable Elimination:
$Prob(N) = 0.4 \times 1.0$

Simple Rule-based Approach - Example 1

$$\Phi = (Y = 2 \wedge X = 1) \vee (Y = 2 \wedge Z = 1) \vee (Y = 1 \wedge Z = 2) \vee (Y = 2 \wedge T = 1) \vee (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)$$

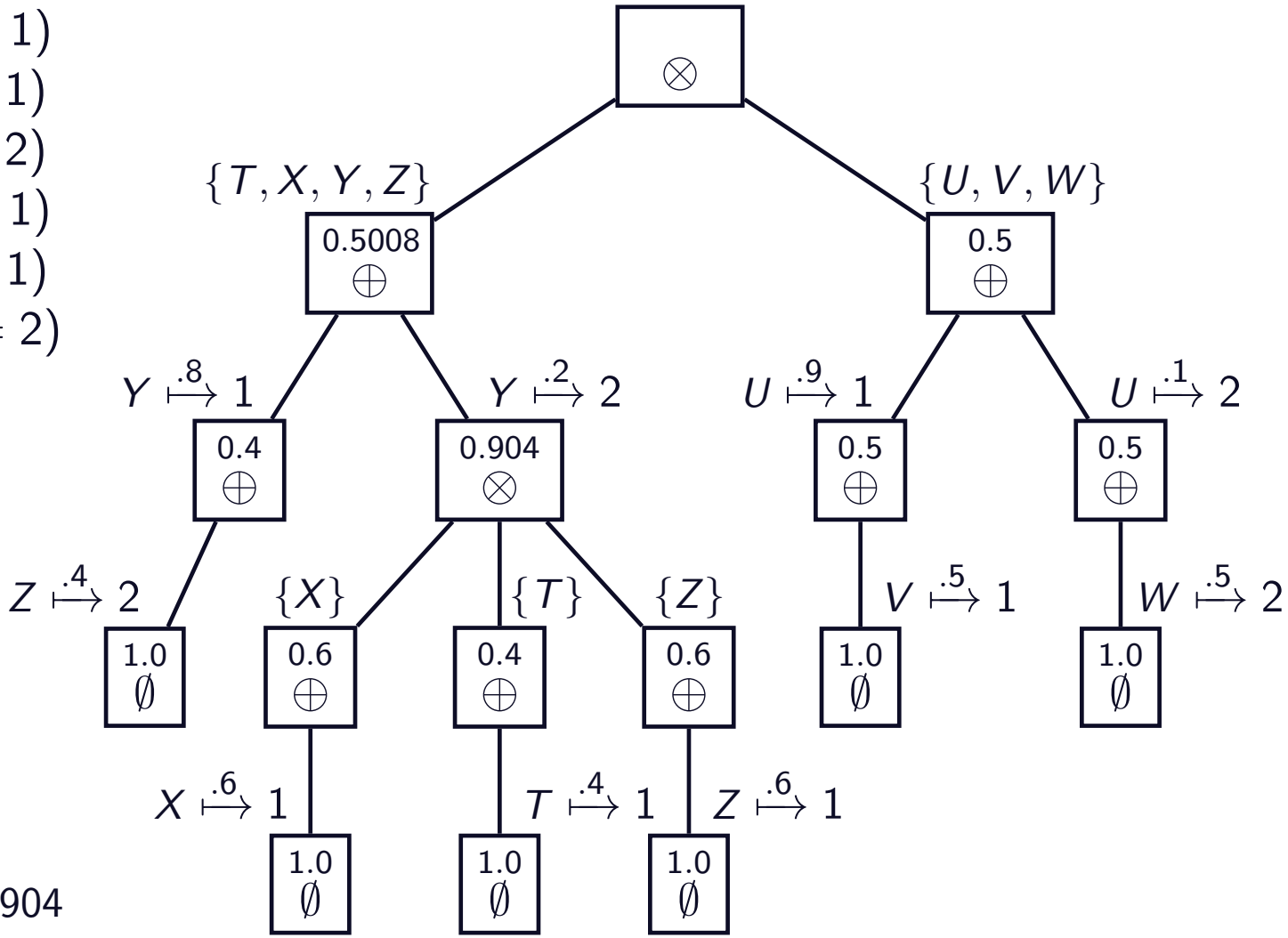


Decomposition:

$$Prob(N) = 1 - (1 - 0.6) \times (1 - 0.4) \times (1 - 0.6)$$

Simple Rule-based Approach - Example 1

$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$

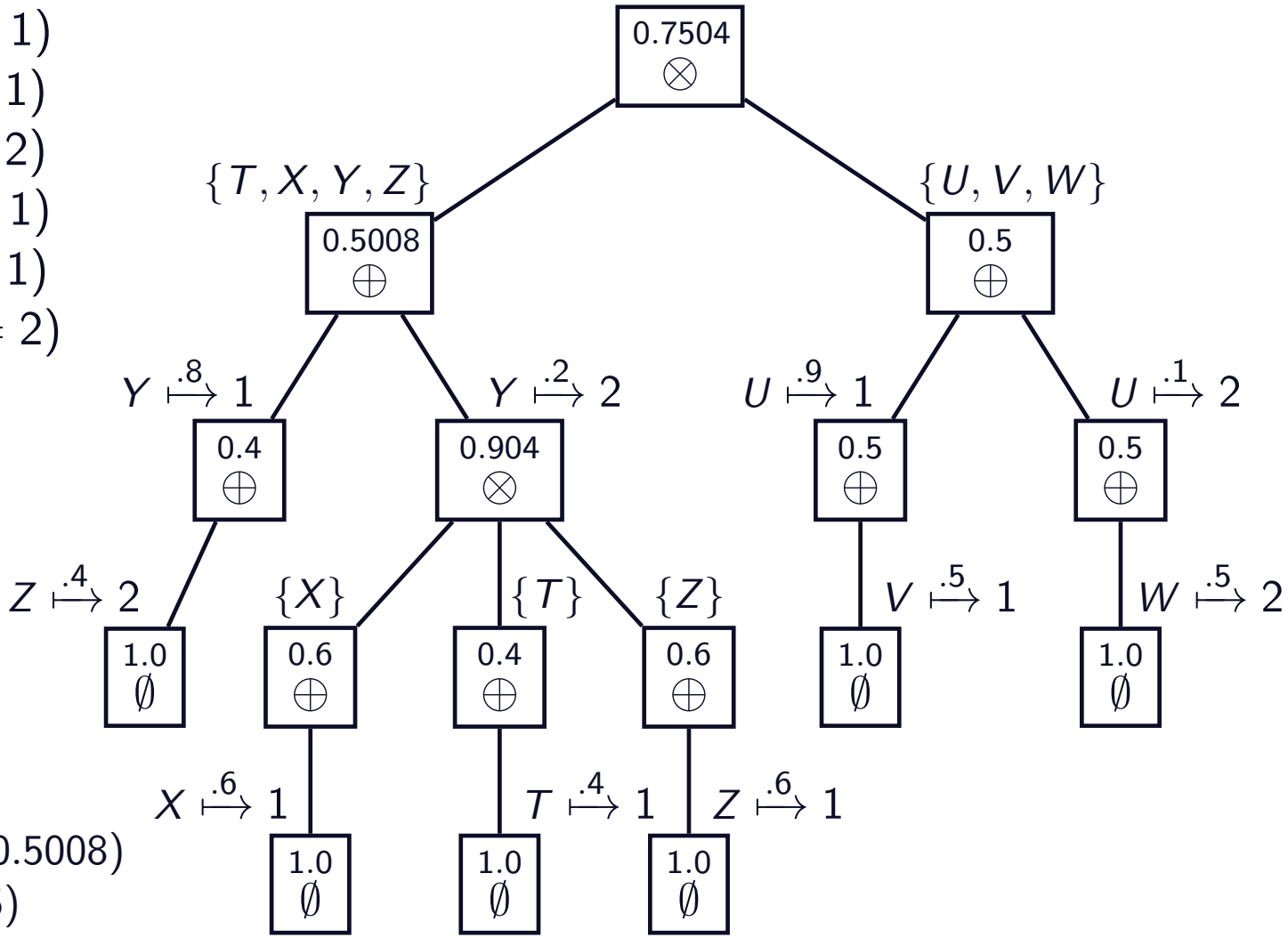


Variable Elimination:

$$\begin{aligned}Prob(N) = &0.8 \times 0.4\\ &+0.2 \times 0.904\end{aligned}$$

Simple Rule-based Approach - Example 1

$$\Phi = (Y = 2 \wedge X = 1) \vee (Y = 2 \wedge Z = 1) \vee (Y = 1 \wedge Z = 2) \vee (Y = 2 \wedge T = 1) \vee (U = 1 \wedge V = 1) \vee (U = 2 \wedge W = 2)$$

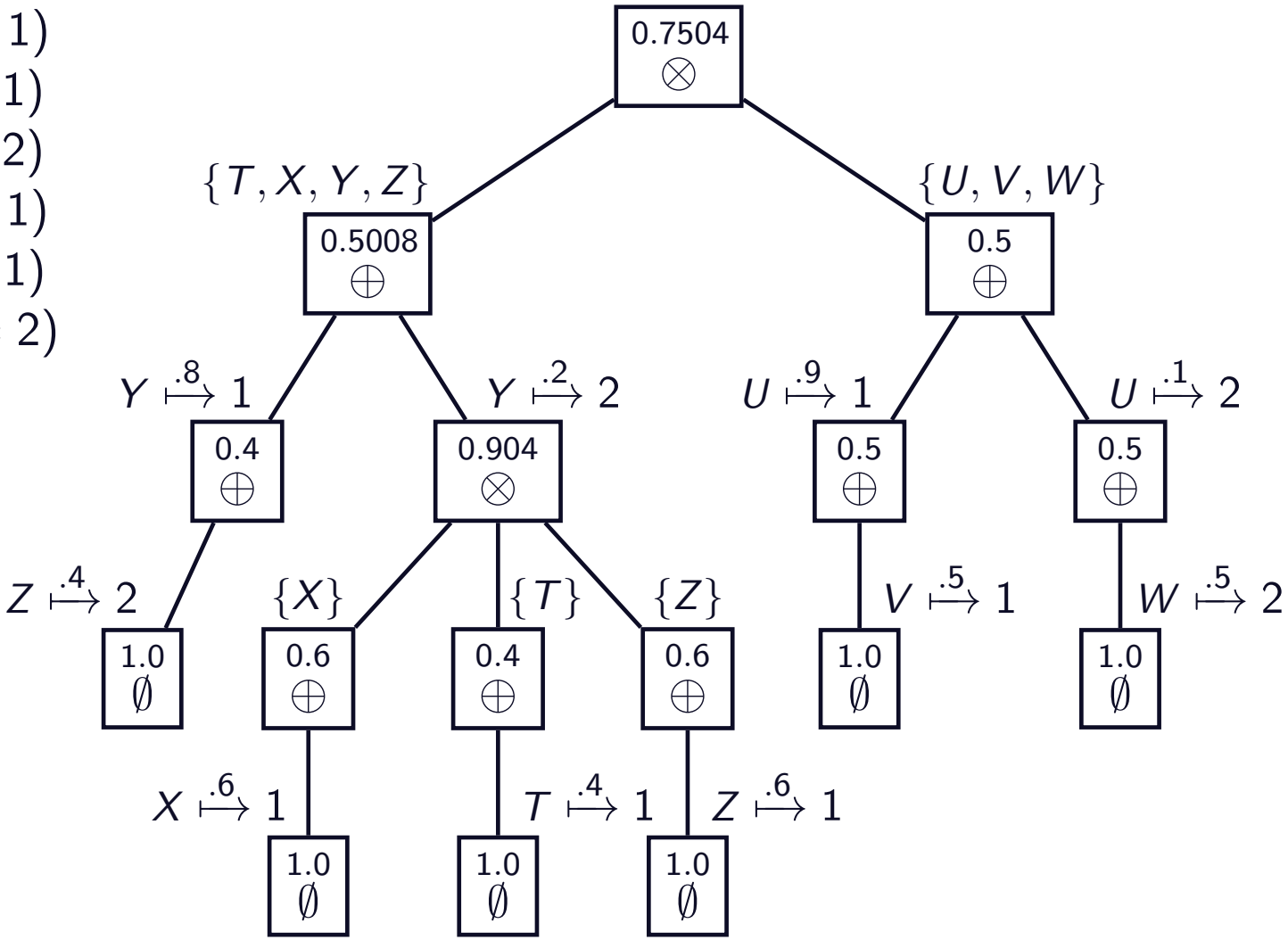


Decomposition:

$$Prob(N) = 1 - (1 - 0.5008) \times (1 - 0.5)$$

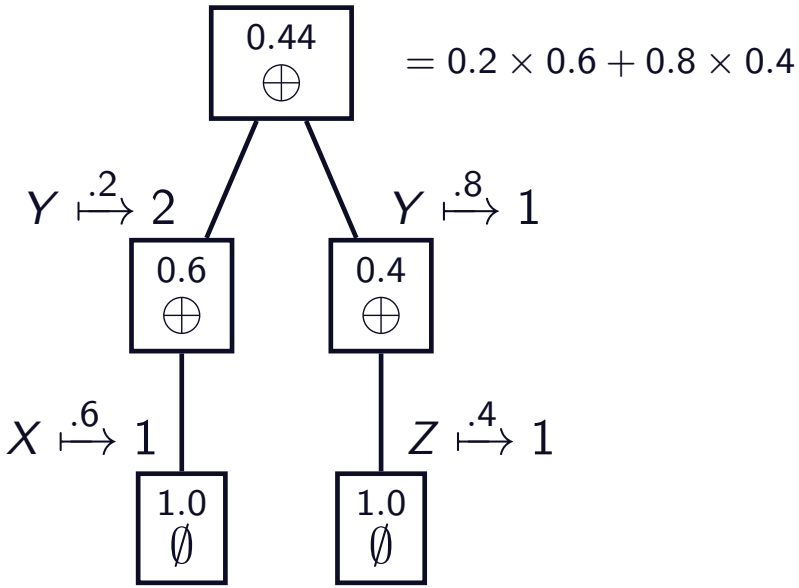
Simple Rule-based Approach - Example 1

$$\begin{aligned}\Phi = &(Y = 2 \wedge X = 1)\\ \vee &(Y = 2 \wedge Z = 1)\\ \vee &(Y = 1 \wedge Z = 2)\\ \vee &(Y = 2 \wedge T = 1)\\ \vee &(U = 1 \wedge V = 1)\\ \vee &(U = 2 \wedge W = 2)\end{aligned}$$



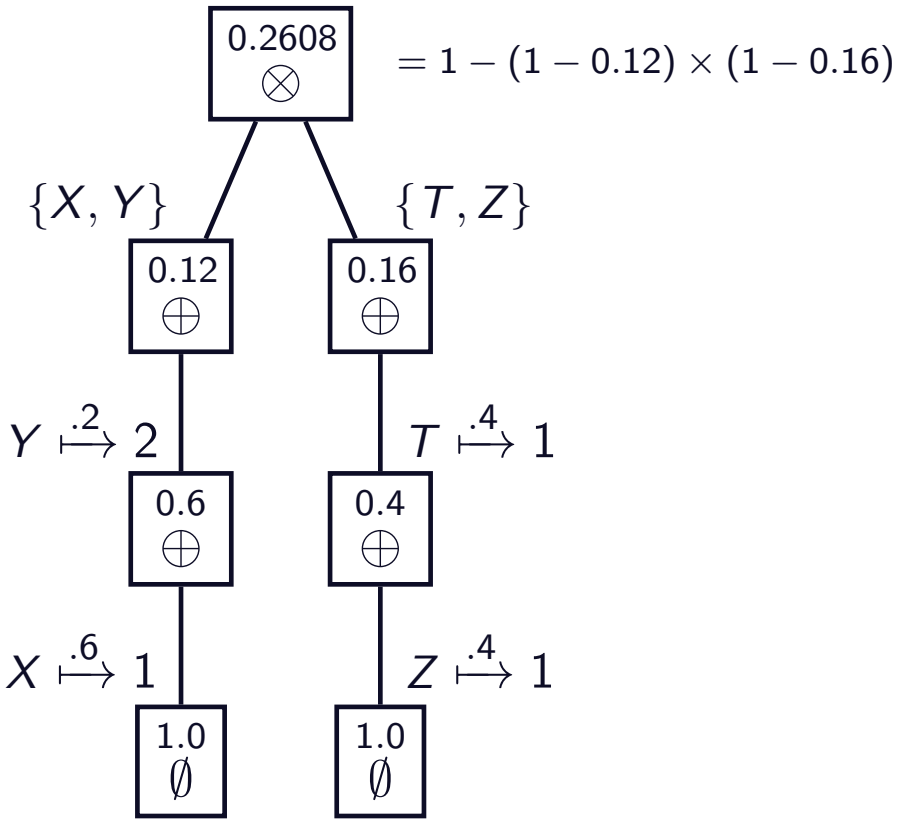
Simple Rule-based Approach - Example 2

$$\Phi = (X = 1 \wedge Y = 2) \vee (Y = 1 \wedge Z = 1)$$



Simple Rule-based Approach - Example 3

$$\Phi = (X = 1 \wedge Y = 2) \vee (T = 1 \wedge Z = 1)$$



Monte-Carlo Simulation

Underlying idea:

- Approximation by evaluating the query in N sample worlds
- Probability of a possible query answer as the proportion of sample worlds in which this tuple is a query answer

Formal:

- Let $pdb = (\mathbf{W}, Pr)$ be a probabilistic database
- Let Q be a conventional database query
- Let $\mathbf{W}_N \subseteq \mathbf{W}$ be a multiset of N sample worlds
- The probability that tuple t is query answer of $Q(pdb)$ is approximated as:

$$p(t) = \frac{|\{W \in \mathbf{W}_N \mid t \in Q(W)\}|}{|\mathbf{W}_N|}$$

Monte-Carlo Simulation

Sampling concept:

- A possible world is selected randomly
- The higher the probability of a world, the greater the chance that this world is selected (maybe more than once)
- The more sample worlds are computed, the more accurate is the approximation result

Sampling method:

- Depends on the considered representation system
  - TI-databases: Select some of the maybe-tuples randomly
  - AOR-database: Select an alternative value per A-tuple randomly
  - BID-databases: Select a tuple (or no tuple) per block randomly
  - pc-databases: Select a variable assignment randomly

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Person

	RK	WK	name	age	condition
t_1	1	p1	J.Doe	27	X=1
t_2	2	p1	J.Doe	28	X=2
t_3	3	p2	K.Smith	32	Y=1
t_4	4	p2	S.Kmith	32	Y=2
t_5	5	p3	J.Doe	28	X=1 v X=3
t_6	6	p3	J.Ho	29	X=2

World-Table

var	value	Prob
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Before iteration starts:

Sample worlds: $\emptyset$

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Person

	RK	WK	name	age	condition
t_1	1	p1	J.Doe	27	X=1
t_2	2	p1	J.Doe	28	X=2
t_3	3	p2	K.Smith	32	Y=1
t_4	4	p2	S.Kmith	32	Y=2
t_5	5	p3	J.Doe	28	X=1 v X=3
t_6	6	p3	J.Ho	29	X=2

World-Table

var	value	Prob
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Iteration step 1: Result of the random selection: $X = 2$ and $Y = 1$

Sample worlds: $W_{S1} = \{t_3, t_6\}$

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Person

	RK	WK	name	age	condition
t_1	1	p1	J.Doe	27	X=1
t_2	2	p1	J.Doe	28	X=2
t_3	3	p2	K.Smith	32	Y=1
t_4	4	p2	S.Kmith	32	Y=2
t_5	5	p3	J.Doe	28	X=1 v X=3
t_6	6	p3	J.Ho	29	X=2

World-Table

var	value	Prob
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Iteration step 2: Result of the random selection: $X = 1$ and $Y = 1$

Sample worlds: $W_{S1} = \{t_3, t_6\}$, $W_{S2} = \{t_1, t_3, t_5\}$

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Person

	RK	WK	name	age	condition
t_1	1	p1	J.Doe	27	X=1
t_2	2	p1	J.Doe	28	X=2
t_3	3	p2	K.Smith	32	Y=1
t_4	4	p2	S.Kmith	32	Y=2
t_5	5	p3	J.Doe	28	X=1 v X=3
t_6	6	p3	J.Ho	29	X=2

World-Table

var	value	Prob
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Iteration step 3: Result of the random selection: $X = 1$ and $Y = 2$

Sample worlds: $W_{S1} = \{t_3, t_6\}$, $W_{S2} = \{t_1, t_3, t_5\}$, $W_{S3} = \{t_1, t_4, t_5\}$

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Person

	RK	WK	name	age	condition
t_1	1	p1	J.Doe	27	X=1
t_2	2	p1	J.Doe	28	X=2
t_3	3	p2	K.Smith	32	Y=1
t_4	4	p2	S.Kmith	32	Y=2
t_5	5	p3	J.Doe	28	X=1 v X=3
t_6	6	p3	J.Ho	29	X=2

World-Table

var	value	Prob
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Iteration step 4: Result of the random selection: $X = 1$ and $Y = 1$

Sample worlds: $W_{S1} = \{t_3, t_6\}$, $W_{S2} = \{t_1, t_3, t_5\}$, $W_{S3} = \{t_1, t_4, t_5\}$, $W_{S4} = \{t_1, t_3, t_5\}$

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Person

	RK	WK	name	age	condition
t_1	1	p1	J.Doe	27	X=1
t_2	2	p1	J.Doe	28	X=2
t_3	3	p2	K.Smith	32	Y=1
t_4	4	p2	S.Kmith	32	Y=2
t_5	5	p3	J.Doe	28	X=1 v X=3
t_6	6	p3	J.Ho	29	X=2

World-Table

var	value	Prob
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Iteration step 5: Result of the random selection: $X = 1$ and $Y = 1$

Sample worlds: $W_{S1} = \{t_3, t_6\}$, $W_{S2} = \{t_1, t_3, t_5\}$, $W_{S3} = \{t_1, t_4, t_5\}$,
$W_{S4} = \{t_1, t_3, t_5\}$, $W_{S5} = \{t_1, t_3, t_5\}$

Monte-Carlo Simulation

Problem:

- Querytime increases linear with the number of sample worlds

$\Rightarrow$ Number of sample worlds has to be small

- Redundant computation of the same query answers

Solution:

- Compute all possible query answers along with their lineage

$\Rightarrow$ Each answer tuple is computed only once

- Compute a set of sample worlds
- Evaluate the lineage formulas of the answer tuples for each of the sample worlds

Monte-Carlo Simulation - Example

Person

	RK	WK	name	age	condition
t_1	1	p1	J.Doe	27	X=1
t_2	2	p1	J.Doe	28	X=2
t_3	3	p2	K.Smith	32	Y=1
t_4	4	p2	S.Kmith	32	Y=2
t_5	5	p3	J.Doe	28	X=1 v X=3
t_6	6	p3	J.Ho	29	X=2

World-Table

var	value	Prob
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

```
SELECT  name, age
FROM    Person
WHERE   age < 30
```

Query result

	name	age	condition
t_7	J.Doe	27	X=1
t_8	J.Doe	28	X=1 v X=2 v X=3
t_9	J.Ho	32	X=2

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Query result

	name	*age*	*condition*
t_7	*J.Doe*	*27*	X=1
t_8	*J.Doe*	*28*	X=1 ∨ X=2 ∨ X=3
t_9	*J.Ho*	*32*	X=2

World-Table

var	*value*	*Prob*
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Before iteration starts:

Number of worlds in which t_7 exists: 0
Number of worlds in which t_8 exists: 0
Number of worlds in which t_9 exists: 0

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Query result

	name	*age*	*condition*
t_7	*J.Doe*	*27*	X=1
t_8	*J.Doe*	*28*	X=1 v X=2 v X=3
t_9	*J.Ho*	*32*	X=2

World-Table

var	*value*	*Prob*
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Iteration step 1: Result of the random selection: $X = 2$ and $Y = 1$

Number of worlds in which t_7 exists: 0
Number of worlds in which t_8 exists: 1
Number of worlds in which t_9 exists: 1

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Query result

	name	*age*	*condition*
t_7	*J.Doe*	*27*	X=1
t_8	*J.Doe*	*28*	X=1 v X=2 v X=3
t_9	*J.Ho*	*32*	X=2

World-Table

var	*value*	*Prob*
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Iteration step 2: Result of the random selection: $X = 1$ and $Y = 1$

Number of worlds in which t_7 exists: 1
Number of worlds in which t_8 exists: 2
Number of worlds in which t_9 exists: 1

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Query result

	name	*age*	*condition*
t_7	*J.Doe*	*27*	X=1
t_8	*J.Doe*	*28*	X=1 v X=2 v X=3
t_9	*J.Ho*	*32*	X=2

World-Table

var	*value*	*Prob*
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Iteration step 3: Result of the random selection: $X = 1$ and $Y = 2$

Number of worlds in which t_7 exists: 2
Number of worlds in which t_8 exists: 3
Number of worlds in which t_9 exists: 1

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Query result

	name	*age*	*condition*
t_7	*J.Doe*	*27*	X=1
t_8	*J.Doe*	*28*	X=1 v X=2 v X=3
t_9	*J.Ho*	*32*	X=2

World-Table

var	*value*	*Prob*
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Iteration step 4: Result of the random selection: $X = 1$ and $Y = 1$

Number of worlds in which t_7 exists: 3
Number of worlds in which t_8 exists: 4
Number of worlds in which t_9 exists: 1

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Query result

	name	*age*	*condition*
t_7	*J.Doe*	*27*	X=1
t_8	*J.Doe*	*28*	X=1 v X=2 v X=3
t_9	*J.Ho*	*32*	X=2

World-Table

var	*value*	*Prob*
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Iteration step 5: Result of the random selection: $X = 1$ and $Y = 1$

Number of worlds in which t_7 exists: 4
Number of worlds in which t_8 exists: 5
Number of worlds in which t_9 exists: 1

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Query result

	name	*age*	*condition*
t_7	*J.Doe*	*27*	X=1
t_8	*J.Doe*	*28*	X=1 ∨ X=2 ∨ X=3
t_9	*J.Ho*	*32*	X=2

World-Table

var	*value*	*Prob*
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Probability computation:

$p(t_7)$: $4/5 = 0.8$
$p(t_8)$: $5/5 = 1.0$
$p(t_9)$: $1/5 = 0.2$

Monte-Carlo Simulation - Example

pc-database: (5 sample worlds)

Query result

	name	*age*	*condition*
t_7	*J.Doe*	*27*	X=1
t_8	*J.Doe*	*28*	X=1 v X=2 v X=3
t_9	*J.Ho*	*32*	X=2

World-Table

var	*value*	*Prob*
X	1	0.6
X	2	0.2
X	3	0.2
Y	1	0.8
Y	2	0.2

Approximated probabilities:

$p(t_7)$: $4/5 = 0.8$
$p(t_8)$: $5/5 = 1.0$
$p(t_9)$: $1/5 = 0.2$

Correct probabilities:

$p(t_7)$: 0.6
$p(t_8)$: 1.0
$p(t_9)$: 0.2

Extensional Query Evaluation

Underlying idea:

- Integrating probability computation into the database engine
- Each algebraic operator computes the probabilities of its output tuples based on the probabilities of the input tuples

$\Rightarrow$ Probability computation hard-coded in the implementations of the algebraic operators

- Operator has no context information except the tuples and their probabilities

$\Rightarrow$ Operator implementation has to assume a specific dependency for all its input tuples

- Such an assumption is only required if multiple tuples need to be combined

$\Rightarrow$ The selection operator does not need to compute a probability

Computation Rules - Projection

- Let t be the tuple that results from projecting any of the tuples $t_1, \ldots, t_k$ on the attribute set $\mathcal{A}$
- Tuple t exists if one of the tuples $t_1, \ldots, t_k$ exists

$\Rightarrow$ $p(t) = p_\vee(\{t_1, \ldots, t_k\})$

$\Rightarrow$ Depending on the assumed dependency, one of the following four rules is used

$$p(t) = \begin{cases} 1 - \prod_{i=1}^{k}(1 - p(t_i)), & \text{if } t_1, \ldots, t_k \text{ are mutually independent,} \\ p(t_r), & \text{if } t_r \in \{t_1, \ldots, t_k\} \text{ is (positively) implicated by each of the tuples } t_1, \ldots, t_k, \\ 1.0, & \text{if } t_r \in \{t_1, \ldots, t_k\} \text{ is negatively implicated by at least one of the tuples } t_1, \ldots, t_k, \\ \sum_{i=1}^{k} p(t_i), & \text{if } t_1, \ldots, t_k \text{ are mutually exclusive,} \end{cases}$$

Computation Rules - Join/Cross Product

- Let t be the tuple that results from joining the two tuples t_r and t_s
- Tuple t exists if t_r and t_s exist

$\Rightarrow$ $p(t) = p_{\wedge}(\{t_r, t_s\})$

$\Rightarrow$ Depending on the assumed dependency, one of the following four rules is used

$$p(t) = \begin{cases} p(t_r) \times p(t_s), & \text{if } t_r \text{ and } t_s \text{ are independent,} \\ p(t_s), & \text{if } t_r \text{ is (positively) implicated by } t_s, \\ p(t_r) + p(t_s) - 1, & \text{if } t_r \text{ is negatively implicated by } t_s, \\ 0, & \text{if } t_r \text{ and } t_s \text{ are exclusive,} \end{cases}$$

Computation Rules - Set Union

- Let $t \in Q_r \cup Q_s$ be the tuple that results from combining the two tuples $t_r \in Q_r$ and $t_s \in Q_s$
- Tuple t exists if t_r or t_s exist

$\Rightarrow$ $p(t) = p_{\vee}(\{t_1, \ldots, t_k\})$

$\Rightarrow$ Depending on the assumed dependency, one of the following four rules is used

$$p(t) = \begin{cases} 1 - (1 - p(t_r)) \times (1 - p(t_s)), & \text{if } t_r \text{ and } t_s \text{ are independent,} \\ p(t_r), & \text{if } t_r \text{ is (positively) implicated by } t_s, \\ 1, & \text{if } t_r \text{ is negatively implicated by } t_s, \\ p(t_r) + p(t_s), & \text{if } t_r \text{ and } t_s \text{ are exclusive,} \end{cases}$$

Computation Rules - Set Difference

- Let $t \in Q_r - Q_s$ be the tuple that results from combining the two tuples $t_r \in Q_r$ and $t_s \in Q_s$
- Tuple t exists if t_r exists and t_s does not exist

$\Rightarrow$ Depending on the assumed dependency, one of the following four rules is used

$$p(t) = \begin{cases} p(t_r) \times (1 - p(t_s)), & \text{if } t_r \text{ and } t_s \text{ are independent,} \\ p(t_r) - p(t_s), & \text{if } t_r \text{ is (positively) implicated by } t_s, \\ 1 - p(t_s), & \text{if } t_r \text{ is negatively implicated by } t_s, \\ p(t_r), & \text{if } t_r \text{ and } t_s \text{ are exclusive,} \end{cases}$$

Extensional Query Evaluation

Dependency assumption:

- An implementation of an algebraic operator has to assume the same dependency for all its input tuples

$\Rightarrow$ A separate implementation for each combination of operator and dependency

- Number of possible dependencies is infinite

$\Rightarrow$ An implementation only for the most typical dependencies

Implication:

- In the case of implication (positive and negative), the operator needs to know which tuple is implicated by which other tuple
- Such information is most often not available

$\Rightarrow$ Implementations typically assume independence or exclusion

Extensional Query Evaluation

Query plan & algebraic optimization in conventional databases:

- For most queries several algebraic equivalent query plans can be constructed
- Each plan produces the correct result

$\Rightarrow$ The cheapest of these plans is executed

Query plan & algebraic optimization in probabilistic databases:

- Algebraic equivalent query plans can produce incorrect probabilities
- Safe Plan: A plan that computes correct probabilities
- Unsafe Plan: A plan that computes incorrect probabilities

$\Rightarrow$ The cheapest of all safe plans is executed

Extensional Query Evaluation - Example 1

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

Read

	WK	person	title	year	p
t_6	r1	p1	The Stranger	1998	0.6
t_7	r2	p2	Pale Fire	2003	1.0
t_8	r3	p2	The Stranger	1997	0.4
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

- 'Person' is a BID-table (tuples of the same block are exclusive, tuples of different blocks are independent)
- 'Read' is a TI-table (all tuples are mutually independent)

Extensional Query Evaluation - Example 1

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

Read

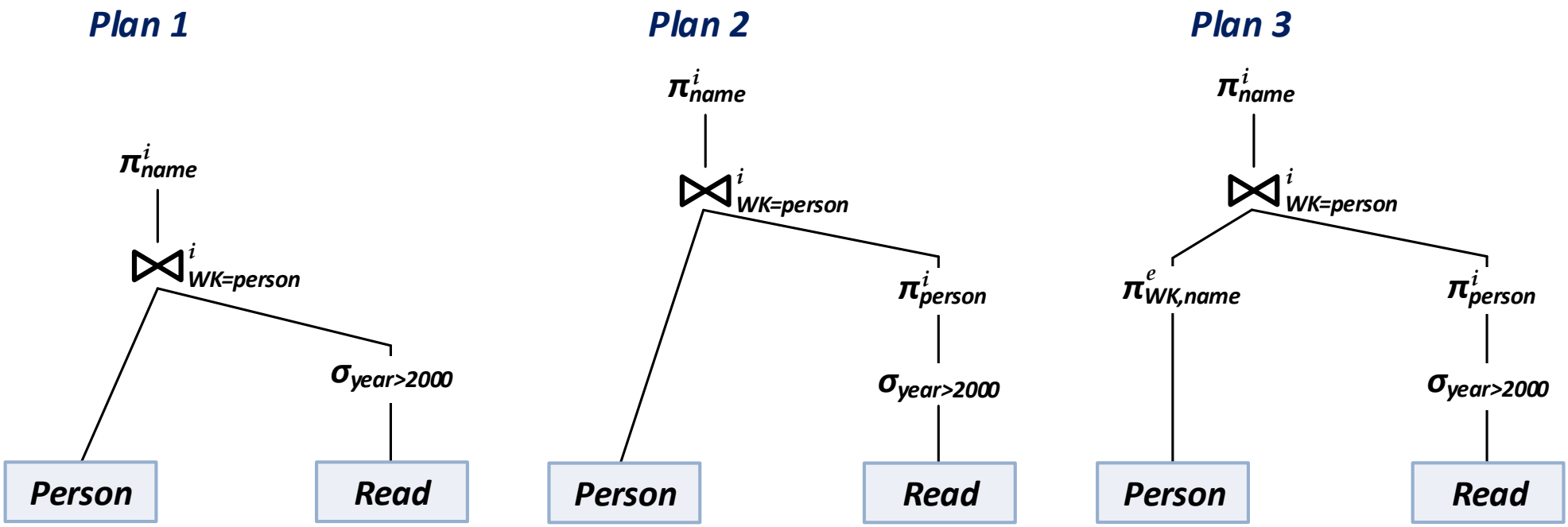
	WK	person	title	year	p
t_6	r1	p1	The Stranger	1998	0.6
t_7	r2	p2	Pale Fire	2003	1.0
t_8	r3	p2	The Stranger	1997	0.4
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

```
SELECT  p.name
FROM    Person p, Read r
WHERE   p.WK = r.person
AND     r.year > 2000
```

	name	lineage	p
t_{11}	K.Smith	$(\epsilon_3 \wedge \epsilon_7)$	1.0
t_{12}	J.Doe	$(\epsilon_4 \wedge \epsilon_9) \vee (\epsilon_4 \wedge \epsilon_{10})$	0.8
t_{13}	J.Ho	$(\epsilon_5 \wedge \epsilon_9) \vee (\epsilon_5 \wedge \epsilon_{10})$	0.2

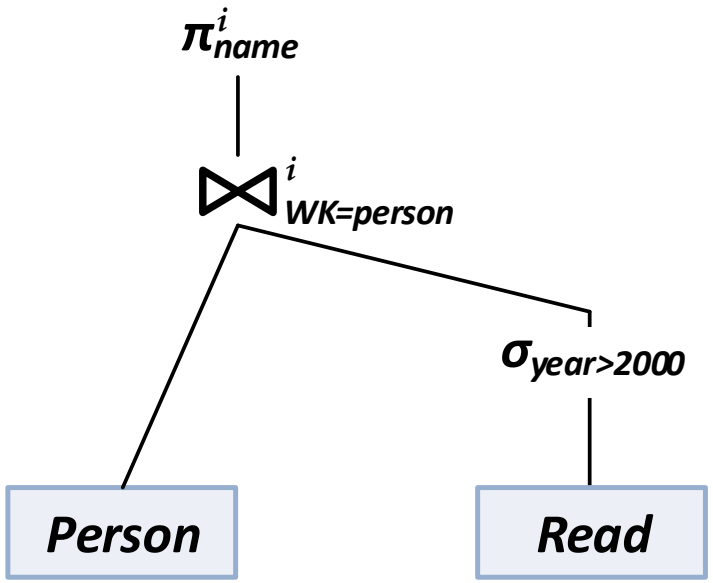
Extensional Query Evaluation - Example 1

SELECT p.name
FROM Person p, Read r
WHERE p.WK = r.person
AND r.year > 2000



Extensional Query Evaluation - Example 1

Plan 1:



Extensional Query Evaluation - Example 1

Plan 1:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

Read

	WK	person	title	year	p
t_6	r1	p1	The Stranger	1998	0.6
t_7	r2	p2	Pale Fire	2003	1.0
t_8	r3	p2	The Stranger	1997	0.4
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

Extensional Query Evaluation - Example 1

Plan 1:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

$\sigma_{year>2001}(Read)$

	WK	person	title	year	p
t_7	r2	p2	Pale Fire	2003	1.0
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

No probability computation required

Extensional Query Evaluation - Example 1

Plan 1:

$Person \bowtie_{WK=person} \sigma_{year>2001}(Read)$

	RK	WK	name	age	person	title	year	p
t_{11}	3	p2	K.Smith	32	p2	Pale Fire	2003	1.0
t_{12}	4	p3	J.Doe	28	p3	White Nights	2002	0.64
t_{13}	4	p3	J.Doe	28	p3	Homo Faber	2001	0.8
t_{14}	5	p3	J.Ho	29	p3	White Nights	2002	0.16
t_{15}	5	p3	J.Ho	29	p3	Homo Faber	2001	0.2

$$p(t_{11}) = p(t_3) \times p(t_7) = 1.0 \times 1.0 = 1.0$$
$$p(t_{12}) = p(t_4) \times p(t_9) = 0.8 \times 0.8 = 0.64$$
$$p(t_{13}) = p(t_4) \times p(t_{10}) = 0.8 \times 1.0 = 0.8$$
$$p(t_{14}) = p(t_5) \times p(t_9) = 0.2 \times 0.8 = 0.16$$
$$p(t_{15}) = p(t_5) \times p(t_{10}) = 0.2 \times 1.0 = 0.2$$

Extensional Query Evaluation - Example 1

Plan 1:

$\pi_{name}(Person \bowtie_{WK=person} \sigma_{year>2001}(Read))$

	name	p
t_{16}	K.Smith	1.0
t_{17}	J.Doe	0.928
t_{18}	J.Ho	0.328

$$p(t_{16}) = p(t_{11}) = 1.0$$
$$p(t_{17}) = 1 - (1 - p(t_{12})) \times (1 - p(t_{13})) = 0.928$$
$$p(t_{18}) = 1 - (1 - p(t_{14})) \times (1 - p(t_{15})) = 0.328$$

Extensional Query Evaluation - Example 1

Plan 1:

$\pi_{name}(Person \bowtie_{WK=person} \sigma_{year>2001}(Read))$

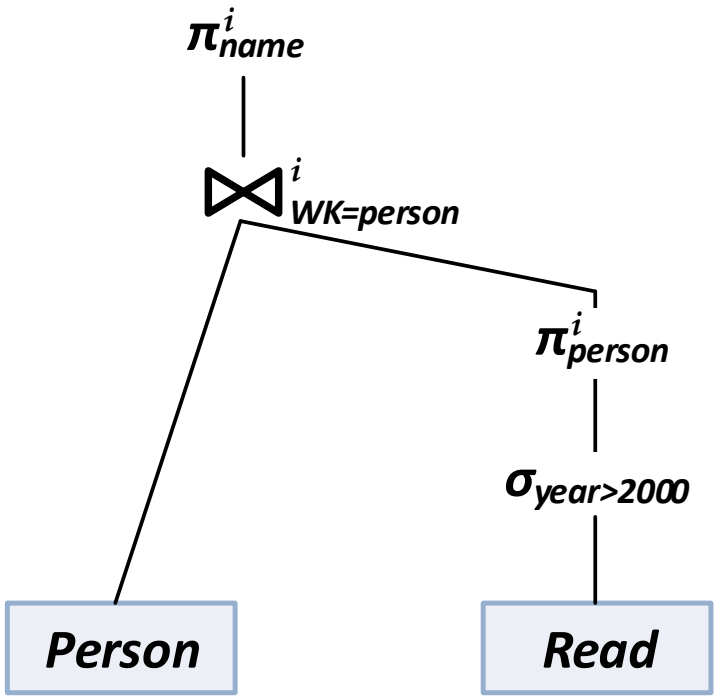
	name	p
t_{16}	K.Smith	1.0
t_{17}	J.Doe	0.928
t_{18}	J.Ho	0.328

	name	lineage	p
t_{16}	K.Smith	$(\epsilon_3 \wedge \epsilon_7)$	1.0
t_{17}	J.Doe	$(\epsilon_4 \wedge \epsilon_9) \vee (\epsilon_4 \wedge \epsilon_{10})$	0.8
t_{18}	J.Ho	$(\epsilon_5 \wedge \epsilon_9) \vee (\epsilon_5 \wedge \epsilon_{10})$	0.2

The probabilities of t_{17} and t_{18} are incorrect!

Extensional Query Evaluation - Example 1

Plan 2:



Extensional Query Evaluation - Example 1

Plan 2:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

Read

	WK	person	title	year	p
t_6	r1	p1	The Stranger	1998	0.6
t_7	r2	p2	Pale Fire	2003	1.0
t_8	r3	p2	The Stranger	1997	0.4
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

Extensional Query Evaluation - Example 1

Plan 2:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

$\sigma_{year>2001}(Read)$

	WK	person	title	year	p
t_7	r2	p2	Pale Fire	2003	1.0
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

No probability computation required

Extensional Query Evaluation - Example 1

Plan 2:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

$\pi_{person}(\sigma_{year>2001}(Read))$

	person	p
t_{11}	p2	1.0
t_{12}	p3	1.0

$$p(t_{11}) = p(t_7) = 1.0$$
$$p(t_{12}) = 1 - (1 - p(t_9)) \times (1 - p(t_{10})) = 1.0$$

Extensional Query Evaluation - Example 1

Plan 2:

$Person \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read))$

	RK	WK	name	age	person	p
t_{13}	3	p2	K.Smith	32	p2	1.0
t_{14}	4	p3	J.Doe	28	p3	0.8
t_{15}	5	p3	J.Ho	29	p3	0.2

$$p(t_{13}) = p(t_3) \times p(t_{11}) = 1.0$$
$$p(t_{14}) = p(t_4) \times p(t_{12}) = 0.8$$
$$p(t_{15}) = p(t_5) \times p(t_{12}) = 0.2$$

Extensional Query Evaluation - Example 1

Plan 2:

$$\pi_{name}(Person \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read)))$$

	name	*p*
t_{16}	*K.Smith*	1.0
t_{17}	*J.Doe*	0.8
t_{18}	*J.Ho*	0.2

$$p(t_{16}) = p(t_{13}) = 1.0$$
$$p(t_{17}) = p(t_{14}) = 0.8$$
$$p(t_{18}) = p(t_{15}) = 0.2$$

Extensional Query Evaluation - Example 1

Plan 2:

$$\pi_{name}(Person \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read)))$$

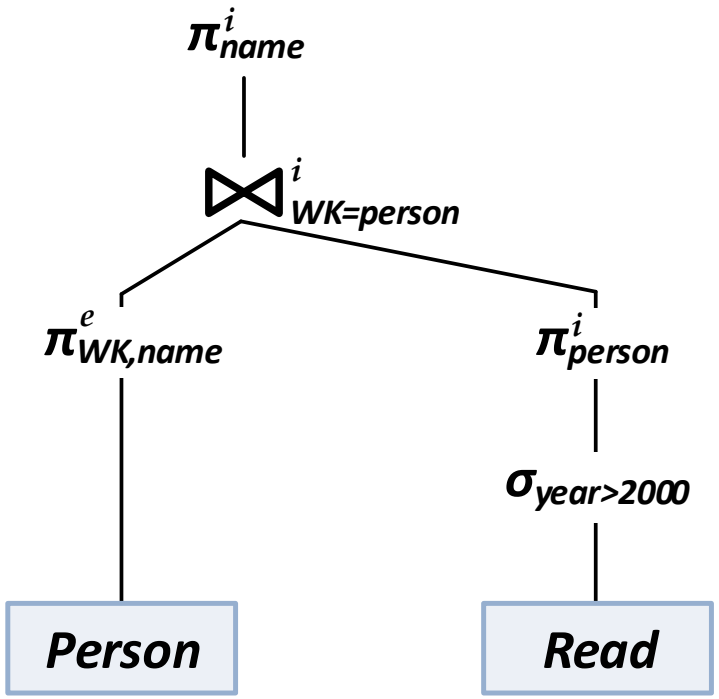
	name	p
t_{16}	K.Smith	1.0
t_{17}	J.Doe	0.8
t_{18}	J.Ho	0.2

	name	lineage	p
t_{16}	K.Smith	$(\epsilon_3 \wedge \epsilon_7)$	1.0
t_{17}	J.Doe	$(\epsilon_4 \wedge \epsilon_9) \vee (\epsilon_4 \wedge \epsilon_{10})$	0.8
t_{18}	J.Ho	$(\epsilon_5 \wedge \epsilon_9) \vee (\epsilon_5 \wedge \epsilon_{10})$	0.2

All probabilities are correct!

Extensional Query Evaluation - Example 1

Plan 3:



Extensional Query Evaluation - Example 1

Plan 3:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

Read

	WK	person	title	year	p
t_6	r1	p1	The Stranger	1998	0.6
t_7	r2	p2	Pale Fire	2003	1.0
t_8	r3	p2	The Stranger	1997	0.4
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

Extensional Query Evaluation - Example 1

Plan 3:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

$\sigma_{year>2001}(Read)$

	WK	person	title	year	p
t_7	r2	p2	Pale Fire	2003	1.0
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

No probability computation required

Extensional Query Evaluation - Example 1

Plan 3:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

$\pi_{person}(\sigma_{year>2001}(Read))$

	person	p
t_{11}	p2	1.0
t_{12}	p3	1.0

$$p(t_{11}) = p(t_7) = 1.0$$
$$p(t_{12}) = 1 - (1 - p(t_9)) \times (1 - p(t_{10})) = 1.0$$

Extensional Query Evaluation - Example 1

Plan 3:

$\pi_{WK,name}(Person)$

	WK	name	p
t_{1b}	p1	J.Doe	0.8
t_{3b}	p2	K.Smith	1.0
t_{4b}	p3	J.Doe	0.8
t_{5b}	p3	J.Ho	0.2

$\pi_{person}(\sigma_{year>2001}(Read))$

	person	p
t_{11}	p2	1.0
t_{12}	p3	1.0

$$p(t_{1b}) = p(t_1) + p(t_2) = 0.8$$
$$p(t_{3b}) = p(t_3) = 1.0$$
$$p(t_{4b}) = p(t_4) = 0.8$$
$$p(t_{5b}) = p(t_5) = 0.2$$

Extensional Query Evaluation - Example 1

Plan 3:

$\pi_{WK,name}(Person) \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read))$

	WK	name	person	p
t_{13}	p2	K.Smith	p2	1.0
t_{14}	p3	J.Doe	p3	0.8
t_{15}	p3	J.Ho	p3	0.2

$$p(t_{13}) = p(t_{3b}) \times p(t_{11}) = 1.0$$
$$p(t_{14}) = p(t_{4b}) \times p(t_{12}) = 0.8$$
$$p(t_{15}) = p(t_{5b}) \times p(t_{12}) = 0.2$$

Extensional Query Evaluation - Example 1

Plan 3:

$$\pi_{name}(\pi_{WK,name}(Person) \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read)))$$

	name	*p*
t_{16}	*K.Smith*	1.0
t_{17}	*J.Doe*	0.8
t_{18}	*J.Ho*	0.2

$$p(t_{16}) = p(t_{13}) = 1.0$$
$$p(t_{17}) = p(t_{14}) = 0.8$$
$$p(t_{18}) = p(t_{15}) = 0.2$$

Extensional Query Evaluation - Example 1

Plan 3:

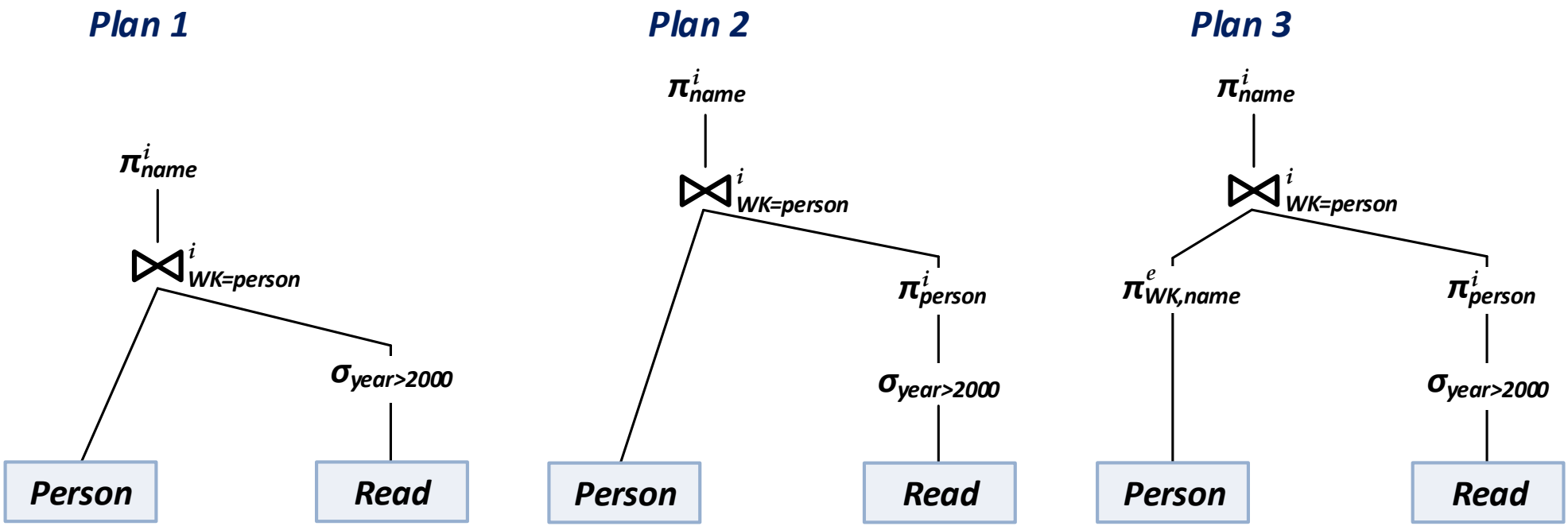
$$\pi_{name}(\pi_{WK,name}(Person) \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read)))$$

	name	p
t_{16}	K.Smith	1.0
t_{17}	J.Doe	0.8
t_{18}	J.Ho	0.2

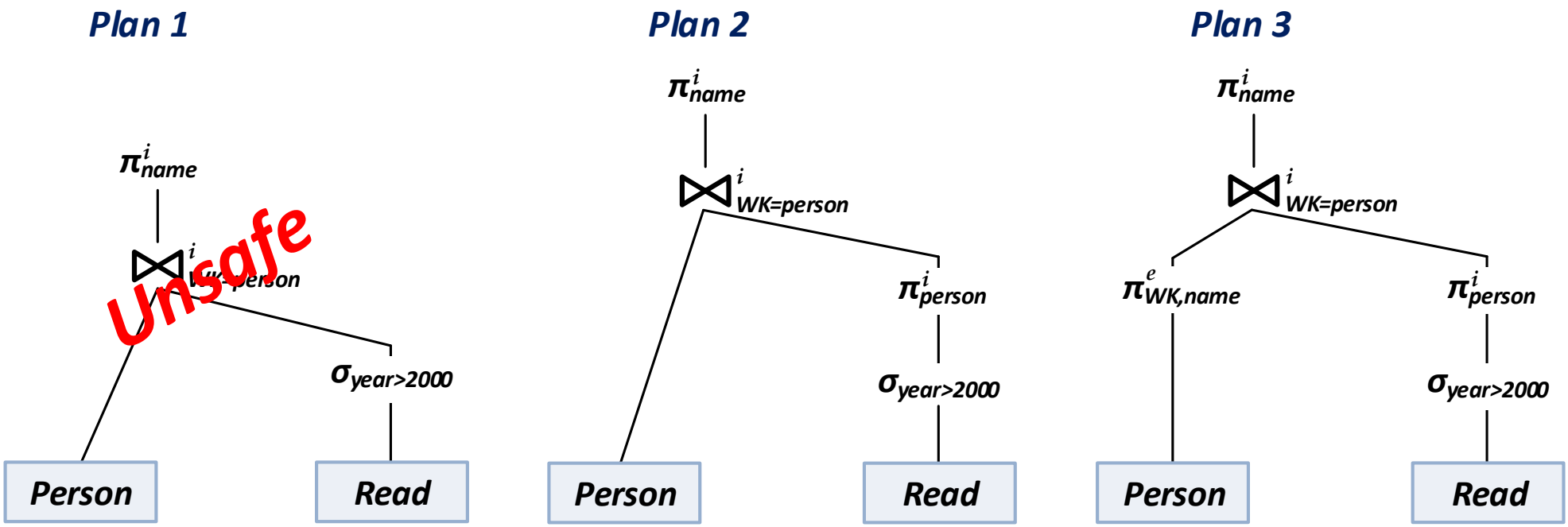
	name	lineage	p
t_{16}	K.Smith	$(\epsilon_3 \wedge \epsilon_7)$	1.0
t_{17}	J.Doe	$(\epsilon_4 \wedge \epsilon_9) \vee (\epsilon_4 \wedge \epsilon_{10})$	0.8
t_{18}	J.Ho	$(\epsilon_5 \wedge \epsilon_9) \vee (\epsilon_5 \wedge \epsilon_{10})$	0.2

All probabilities are correct!

Extensional Query Evaluation - Example 1



Extensional Query Evaluation - Example 1



Extensional Query Evaluation - Example 2

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Doe	29	0.2

Read

	WK	person	title	year	p
t_6	r1	p1	The Stranger	1998	0.6
t_7	r2	p2	Pale Fire	2003	1.0
t_8	r3	p2	The Stranger	1997	0.4
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

```
SELECT  p.name
FROM    Person p, Read r
WHERE   p.WK = r.person
AND     r.year > 2000
```

	name	lineage	p
t_{11}	K.Smith	$(\epsilon_3 \wedge \epsilon_7)$	1.0
t_{12}	J.Doe	$(\epsilon_4 \wedge \epsilon_9) \vee (\epsilon_4 \wedge \epsilon_{10}) \vee (\epsilon_5 \wedge \epsilon_9) \vee (\epsilon_5 \wedge \epsilon_{10})$	1.0

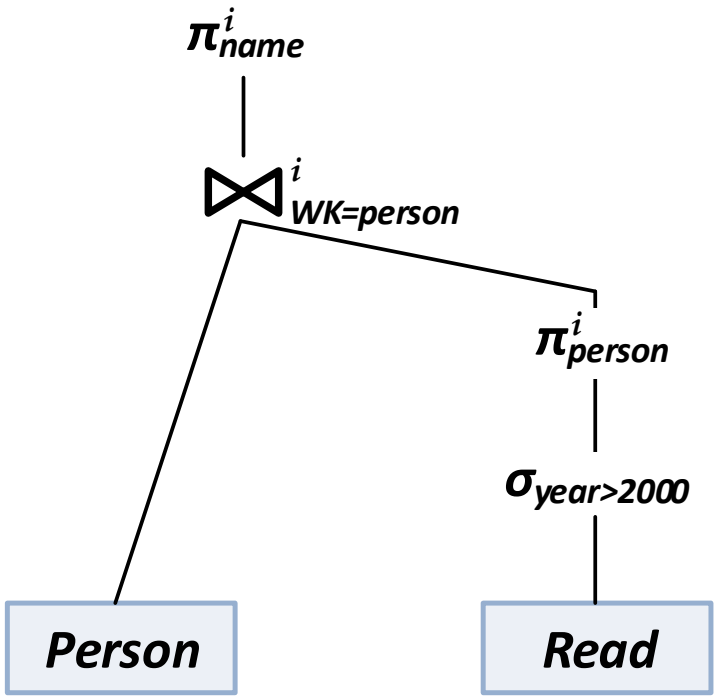
Extensional Query Evaluation - Example 2

SELECT p.name
FROM Person p, Read r
WHERE p.WK = r.person
AND r.year > 2000

	name	*lineage*	*p*
t_{11}	*K.Smith*	$(\epsilon_3 \wedge \epsilon_7)$	1.0
t_{12}	*J.Doe*	$((\epsilon_4 \vee \epsilon_5) \wedge \epsilon_9) \vee ((\epsilon_4 \vee \epsilon_5) \wedge \epsilon_{10})$	1.0

Extensional Query Evaluation - Example 2

Plan 2:



Extensional Query Evaluation - Example 2

Plan 2:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Doe	29	0.2

Read

	WK	person	title	year	p
t_6	r1	p1	The Stranger	1998	0.6
t_7	r2	p2	Pale Fire	2003	1.0
t_8	r3	p2	The Stranger	1997	0.4
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

Extensional Query Evaluation - Example 2

Plan 2:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Doe	29	0.2

$\sigma_{year>2001}(Read)$

	WK	person	title	year	p
t_7	r2	p2	Pale Fire	2003	1.0
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

No probability computation required

Extensional Query Evaluation - Example 2

Plan 2:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Doe	29	0.2

$\pi_{person}(\sigma_{year>2001}(Read))$

	person	p
t_{11}	p2	1.0
t_{12}	p3	1.0

$$p(t_{11}) = p(t_7) = 1.0$$
$$p(t_{12}) = 1 - (1 - p(t_9)) \times (1 - p(t_{10})) = 1.0$$

Extensional Query Evaluation - Example 2

Plan 2:

$Person \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read))$

	RK	WK	name	age	person	p
t_{13}	3	p2	K.Smith	32	p2	1.0
t_{14}	4	p3	J.Doe	28	p3	0.8
t_{15}	5	p3	J.Doe	29	p3	0.2

$$p(t_{13}) = p(t_3) \times p(t_{11}) = 1.0$$
$$p(t_{14}) = p(t_4) \times p(t_{12}) = 0.8$$
$$p(t_{15}) = p(t_5) \times p(t_{12}) = 0.2$$

Extensional Query Evaluation - Example 2

Plan 2:

$$\pi_{name}(Person \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read)))$$

	name	p
t_{16}	K.Smith	1.0
t_{17}	J.Doe	0.84

$$p(t_{16}) = p(t_{13}) = 1.0$$
$$p(t_{17}) = 1 - (1 - p(t_{14})) \times (1 - p(t_{15})) = 0.84$$

Extensional Query Evaluation - Example 2

Plan 2:

$\pi_{name}(Person \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read)))$

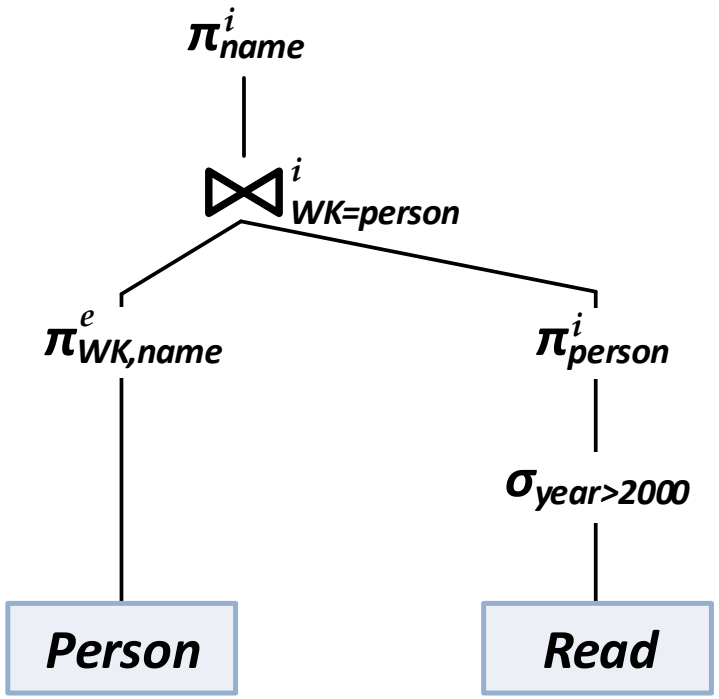
	name	p
t_{16}	K.Smith	1.0
t_{17}	J.Doe	0.84

	name	lineage	p
t_{16}	K.Smith	$(\epsilon_3 \wedge \epsilon_7)$	1.0
t_{17}	J.Doe	$((\epsilon_4 \vee \epsilon_5) \wedge \epsilon_9) \vee ((\epsilon_4 \vee \epsilon_5) \wedge \epsilon_{10})$	1.0

The probability of t_{17} is incorrect!

Extensional Query Evaluation - Example 2

Plan 3:



Extensional Query Evaluation - Example 2

Plan 3:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Doe	29	0.2

Read

	WK	person	title	year	p
t_6	r1	p1	The Stranger	1998	0.6
t_7	r2	p2	Pale Fire	2003	1.0
t_8	r3	p2	The Stranger	1997	0.4
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

Extensional Query Evaluation - Example 2

Plan 3:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Doe	29	0.2

$\sigma_{year>2001}(Read)$

	WK	person	title	year	p
t_7	r2	p2	Pale Fire	2003	1.0
t_9	r4	p3	White Nights	2002	0.8
t_{10}	r5	p3	Homo Faber	2001	1.0

No probability computation required

Extensional Query Evaluation - Example 2

Plan 3:

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Doe	29	0.2

$\pi_{person}(\sigma_{year>2001}(Read))$

	person	p
t_{11}	p2	1.0
t_{12}	p3	1.0

$$p(t_{11}) = p(t_7) = 1.0$$
$$p(t_{12}) = 1 - (1 - p(t_9)) \times (1 - p(t_{10})) = 1.0$$

Extensional Query Evaluation - Example 2

Plan 3:

$\pi_{WK,name}(Person)$

	WK	name	p
t_{1b}	p1	J.Doe	0.8
t_{3b}	p2	K.Smith	1.0
t_{4b}	p3	J.Doe	1.0

$\pi_{person}(\sigma_{year>2001}(Read))$

	person	p
t_{11}	p2	1.0
t_{12}	p3	1.0

$$p(t_{1b}) = p(t_1) + p(t_2) = 0.8$$
$$p(t_{3b}) = p(t_3) = 1.0$$
$$p(t_{4b}) = p(t_4) + p(t_5) = 1.0$$

Extensional Query Evaluation - Example 2

Plan 3:

$\pi_{WK,name}(Person) \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read))$

	WK	name	person	p
t_{12}	p2	K.Smith	p2	1.0
t_{13}	p3	J.Doe	p3	1.0

$$p(t_{13}) = p(t_{3b}) \times p(t_{11}) = 1.0$$
$$p(t_{14}) = p(t_{4b}) \times p(t_{12}) = 1.0$$

Extensional Query Evaluation - Example 2

Plan 3:

$$\pi_{name}(\pi_{WK,name}(Person) \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read)))$$

	name	*p*
t_{16}	*K.Smith*	1.0
t_{17}	*J.Doe*	1.0

$$p(t_{16}) = p(t_{13}) = 1.0$$
$$p(t_{17}) = p(t_{14}) = 1.0$$

Extensional Query Evaluation - Example 2

Plan 3:

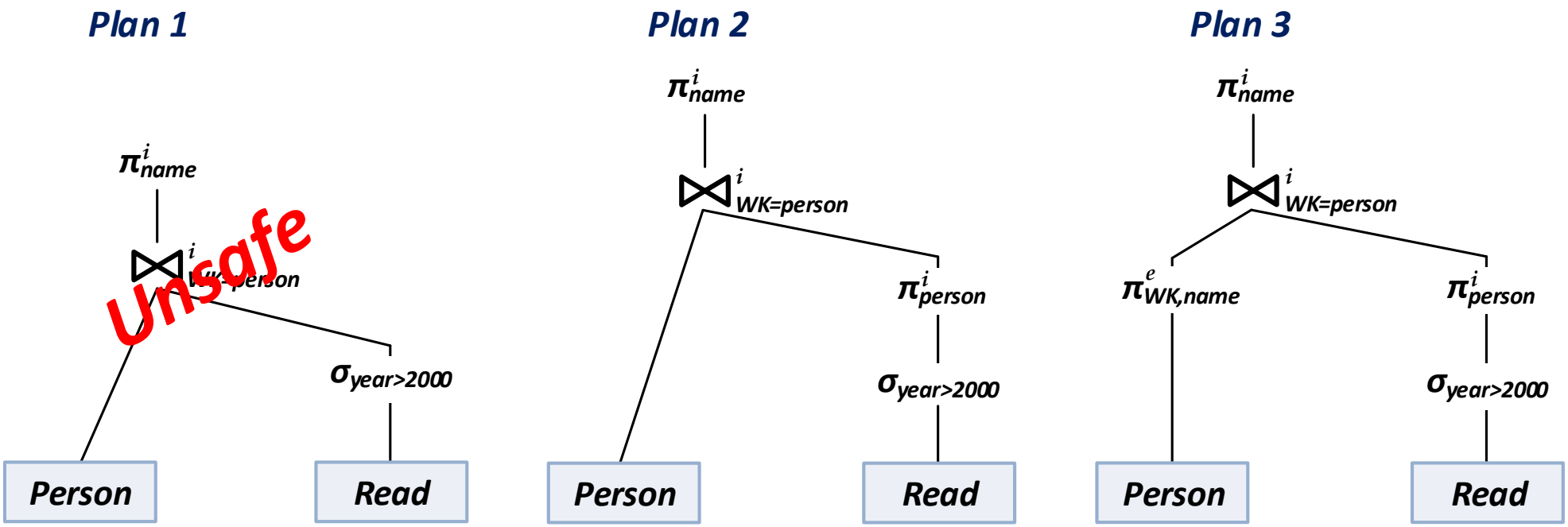
$$\pi_{name}(\pi_{WK,name}(Person) \bowtie_{WK=person} \pi_{person}(\sigma_{year>2001}(Read)))$$

	name	p
t_{16}	K.Smith	1.0
t_{17}	J.Doe	1.0

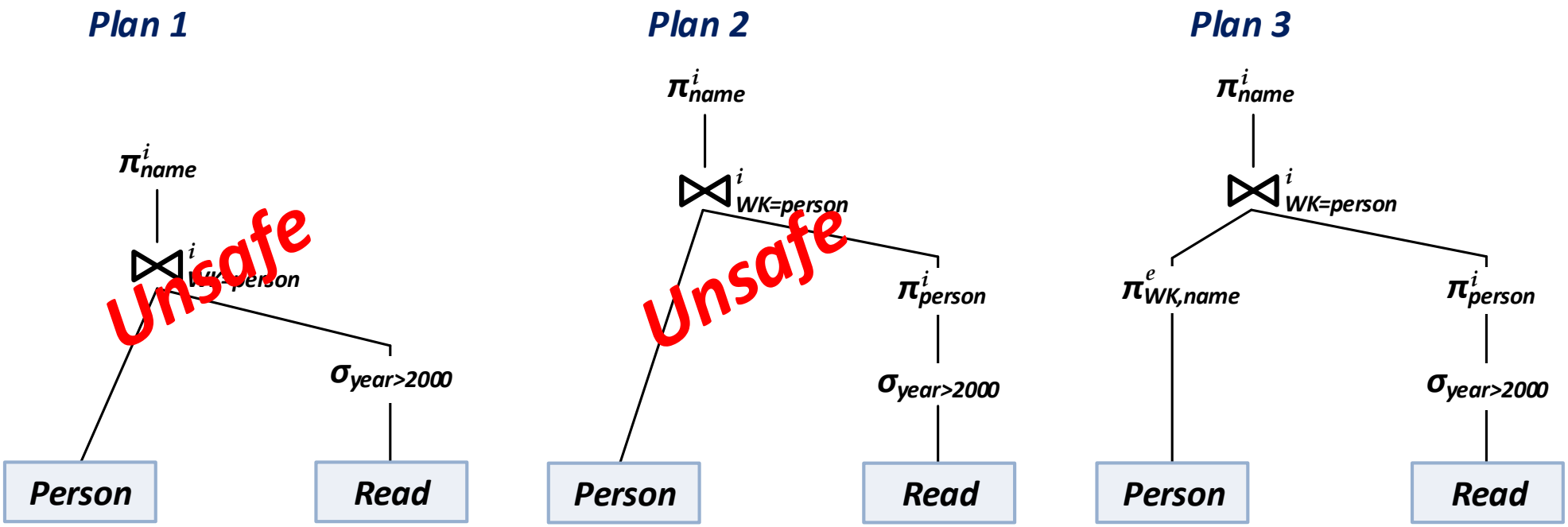
	name	lineage	p
t_{16}	K.Smith	$(\epsilon_3 \wedge \epsilon_7)$	1.0
t_{17}	J.Doe	$((\epsilon_4 \vee \epsilon_5) \wedge \epsilon_9) \vee ((\epsilon_4 \vee \epsilon_5) \wedge \epsilon_{10})$	1.0

All probabilities are correct!

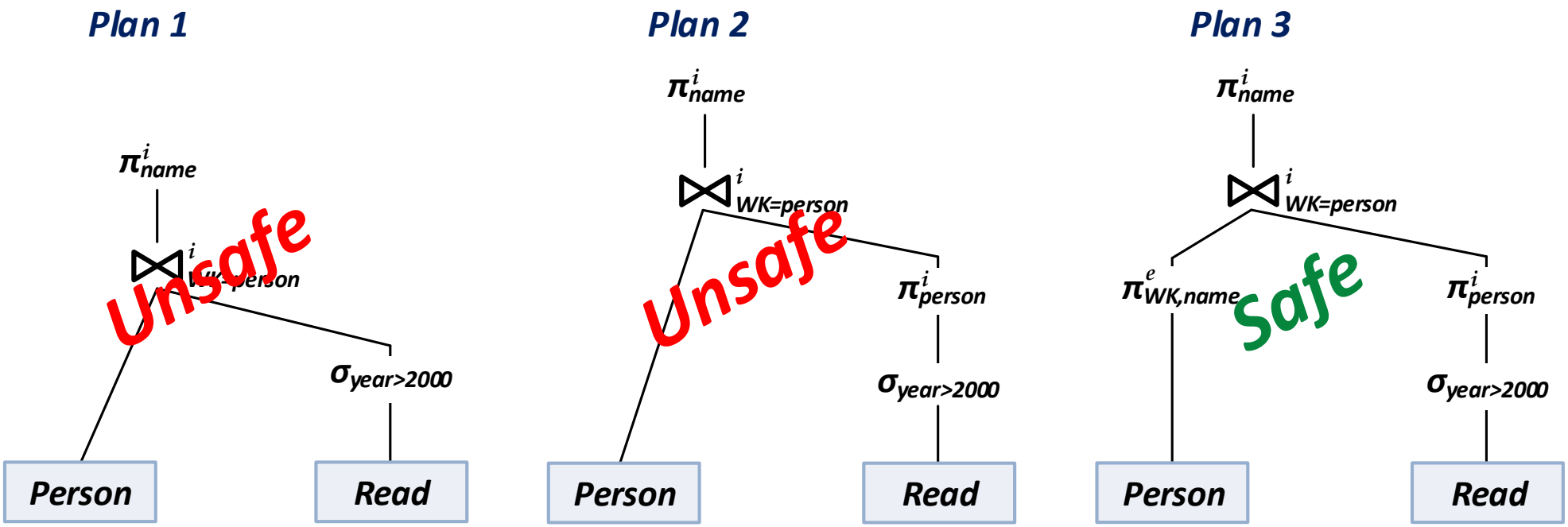
Extensional Query Evaluation - Example 2



Extensional Query Evaluation - Example 2

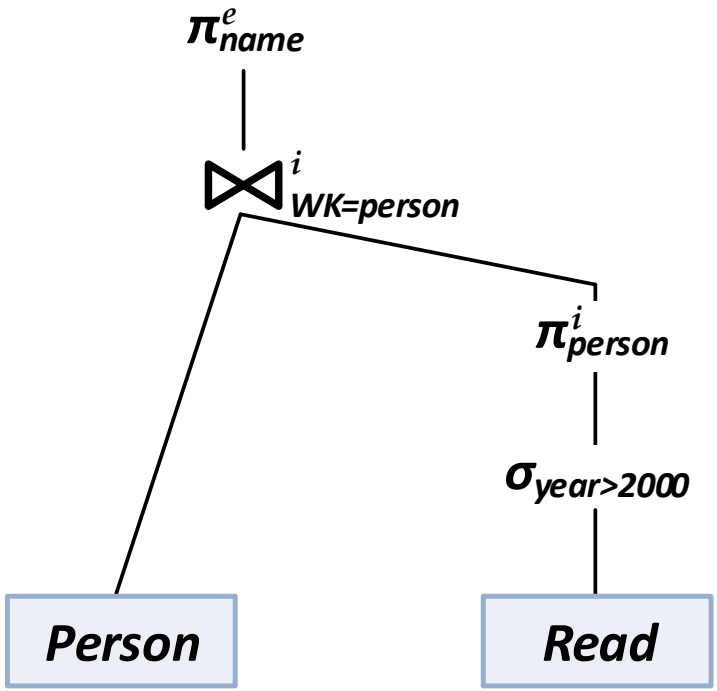


Extensional Query Evaluation - Example 2



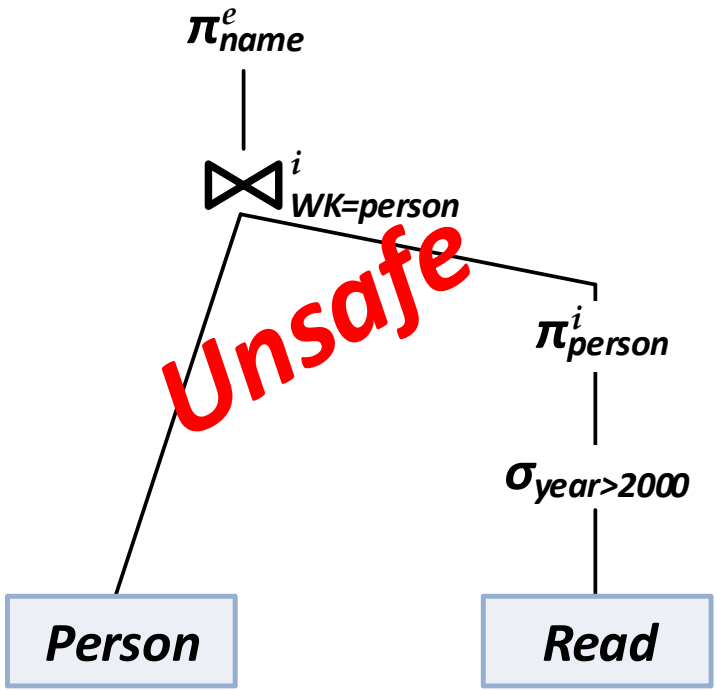
Extensional Query Evaluation - Example

Is this query plan safe?



Extensional Query Evaluation - Example

Is this query plan safe?



If person p_1 would have read a book since 2000, the final projection would also combine independent tuples

Extensional Query Evaluation

Benefits:

- We can leverage standard optimization techniques of conventional database systems
- Tuples with zero probability can be instantly discarded
- If safe plan is known, same complexity as evaluating a query on a conventional database

Drawbacks:

- Not every query has a safe plan with respect to a particular representation system
- Safe Query: A query that has a safe plan
- Unsafe Query: A query that does not have a safe plan

$\Rightarrow$ Extensional query evaluation cannot be used to compute correct probabilities for all combinations of representation systems and queries

Safe vs. Unsafe Queries - Example

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

```
SELECT  t.name AS nameA, u.name AS nameB
FROM    Person t, Person u
WHERE   t.WK <> u.WK
AND     t.age < u.age
```

	nameA	nameB	lineage	p
t_6	J.Doe	K.Smith	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3) \vee (\epsilon_4 \wedge \epsilon_3)$	0.96
t_7	J.Doe	J.Doe	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_8	J.Ho	K.Smith	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_9	J.Doe	J.Ho	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

- t_6 results from joining one of the tuples t_1, t_2, or t_4 with t_3
- t_1 and t_2 are exclusive, but t_4 is independent of t_1 and t_2

$\Rightarrow$ No dependency assumption of the projection operator is valid for all tuples of the join result

$\Rightarrow$ This query is **unsafe**

Safe vs. Unsafe Queries - Example

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

SELECT t.WK, t.name, u.WK, u.name
FROM Person t, Person u
WHERE t.WK $<>$ u.WK
AND t.age $<$ u.age

Safe vs. Unsafe Queries - Example

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

```
SELECT  t.WK, t.name, u.WK, u.name
FROM    Person t, Person u
WHERE   t.WK <> u.WK
AND     t.age < u.age
```

	t.WK	t.name	u.WK	u.name	lineage	p
t_6	p1	J.Doe	p2	K.Smith	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3)$	0.8
t_7	p3	J.Doe	p2	K.Smith	$(\epsilon_4 \wedge \epsilon_3)$	0.8
t_8	p1	J.Doe	p3	J.Doe	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_9	p3	J.Ho	p2	K.Smith	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_{10}	p1	J.Doe	p3	J.Ho	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

Safe vs. Unsafe Queries - Example

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

SELECT t.WK, t.name, u.WK, u.name
FROM Person t, Person u
WHERE t.WK <> u.WK
AND t.age < u.age

	t.WK	t.name	u.WK	u.name	lineage	p
t_6	p1	J.Doe	p2	K.Smith	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3)$	0.8
t_7	p3	J.Doe	p2	K.Smith	$(\epsilon_4 \wedge \epsilon_3)$	0.8
t_8	p1	J.Doe	p3	J.Doe	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_9	p3	J.Ho	p2	K.Smith	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_{10}	p1	J.Doe	p3	J.Ho	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16

- The join operator joins only independent tuples
- The projection operator combines only exclusive tuples

$\Rightarrow$ This query is **safe**

Safe vs. Unsafe Queries - Example

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

SELECT t.WK, t.name, u.WK, u.name
FROM Person t, Person u
WHERE t.age $<$ u.age

Safe vs. Unsafe Queries - Example

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

```
SELECT  t.WK, t.name, u.WK, u.name
FROM    Person t, Person u
WHERE   t.age < u.age
```

	t.WK	t.name	u.WK	u.name	lineage	p
t_6	p1	J.Doe	p1	J.Doe	$(\epsilon_1 \wedge \epsilon_2)$	0.0
t_7	p1	J.Doe	p2	K.Smith	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3)$	0.8
t_8	p1	J.Doe	p3	J.Doe	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_9	p1	J.Doe	p3	J.Ho	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16
t_{10}	p3	J.Doe	p2	K.Smith	$(\epsilon_4 \wedge \epsilon_3)$	0.8
t_{11}	p3	J.Ho	p2	K.Smith	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_{12}	p3	J.Doe	p3	J.Ho	$(\epsilon_4 \wedge \epsilon_5)$	0.0

Safe vs. Unsafe Queries - Example

Person

	RK	WK	name	age	p
t_1	1	p1	J.Doe	27	0.6
t_2	2	p1	J.Doe	28	0.2
t_3	3	p2	K.Smith	32	1.0
t_4	4	p3	J.Doe	28	0.8
t_5	5	p3	J.Ho	29	0.2

SELECT t.WK, t.name, u.WK, u.name
FROM Person t, Person u
WHERE t.age $<$ u.age

	t.WK	t.name	u.WK	u.name	lineage	p
t_6	p1	J.Doe	p1	J.Doe	$(\epsilon_1 \wedge \epsilon_2)$	0.0
t_7	p1	J.Doe	p2	K.Smith	$(\epsilon_1 \wedge \epsilon_3) \vee (\epsilon_2 \wedge \epsilon_3)$	0.8
t_8	p1	J.Doe	p3	J.Doe	$(\epsilon_1 \wedge \epsilon_4)$	0.48
t_9	p1	J.Doe	p3	J.Ho	$(\epsilon_1 \wedge \epsilon_5) \vee (\epsilon_2 \wedge \epsilon_5)$	0.16
t_{10}	p3	J.Doe	p2	K.Smith	$(\epsilon_4 \wedge \epsilon_3)$	0.8
t_{11}	p3	J.Ho	p2	K.Smith	$(\epsilon_5 \wedge \epsilon_3)$	0.2
t_{12}	p3	J.Doe	p3	J.Ho	$(\epsilon_4 \wedge \epsilon_5)$	0.0

- t_6 results from joining the two exclusive tuples t_1 and t_2
- t_8 results from joining the two independent tuples t_1 and t_4

$\Rightarrow$ No dependency assumption of the join operator is valid for all tuple pairs

$\Rightarrow$ This query is **unsafe**

Extensional Query Evaluation

Evaluation of unsafe queries:

- Using intensional query evaluation instead of extensional
- Extensional query evaluation can sometimes be used to compute probability bounds
- Using of materialized views to avoid an intensional recomputation (the results of frequently used unsafe subqueries are persisted in the database)
- Decomposing the query plan into sub-plans and to execute as many of these sub-plans by using extensional query evaluation as possible
- Using functional dependencies, e.g. keys, to transform an unsafe query into a semantically equivalent safe query

Aggregate Queries

Simple aggregate queries:

- One aggregate function, one group of tuples
- Result: a single value

Aggregate queries with multiple functions:

- k aggregate functions, one group of tuples
- Result: a single tuple with k values

Aggregate queries with grouping:

- One aggregate function, k groups of tuples
- Result: k tuples each with one aggregate value

Aggregate queries with grouping and multiple functions:

- k aggregate functions, n groups of tuples
- Result: n tuples each with k aggregate values

Aggregate Queries

Simple aggregate queries:

- One aggregate function, one group of tuples
- Result: a single value

Aggregate queries with multiple functions:

- k aggregate functions, one group of tuples
- Result: a single tuple with k values

Aggregate queries with grouping:

- One aggregate function, k groups of tuples
- Result: k tuples each with one aggregate value

Aggregate queries with grouping and multiple functions:

- k aggregate functions, n groups of tuples
- Result: n tuples each with k aggregate values

Aggregate Queries - Example

Person

	RK	***WK***	***name***	***age***	***p***
t_1	**1**	**1**	*A*	*20*	0.6
t_2	**2**	**1**	*A*	*50*	0.4
t_3	**3**	**2**	*B*	*30*	0.5
t_4	**4**	**2**	*A*	*40*	0.3
t_5	**5**	**3**	*B*	*40*	1.0

```
SELECT  AVG(age)
FROM    Person
```

Simple aggregate query:

- One aggregate function, one group of tuples
- Result: a single value

Aggregate Queries - Example

Person

	RK	*WK*	*name*	*age*	*p*
t_1	**1**	**1**	*A*	*20*	0.6
t_2	**2**	**1**	*A*	*50*	0.4
t_3	**3**	**2**	*B*	*30*	0.5
t_4	**4**	**2**	*A*	*40*	0.3
t_5	**5**	**3**	*B*	*40*	1.0

SELECT AVG(age)
FROM Person

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ $Pr=0.3$	W_2 $\{t_2, t_3, t_5\}$ $Pr=0.2$	W_3 $\{t_1, t_4, t_5\}$ $Pr=0.18$	W_4 $\{t_2, t_4, t_5\}$ $Pr=0.12$	W_5 $\{t_1, t_5\}$ $Pr=0.12$	W_6 $\{t_2, t_5\}$ $Pr=0.08$
30	40	33.3	43.3	30	45

Aggregate Queries - Example

Person

	RK	*WK*	*name*	*age*	*p*
t_1	**1**	**1**	*A*	*20*	0.6
t_2	**2**	**1**	*A*	*50*	0.4
t_3	**3**	**2**	*B*	*30*	0.5
t_4	**4**	**2**	*A*	*40*	0.3
t_5	**5**	**3**	*B*	*40*	1.0

SELECT MIN(age), MAX(age)
FROM Person

Complex aggregate query:

- Two aggregate functions, one group of tuples
- Result: a single tuple with two values

Aggregate Queries - Example

Person

	RK	WK	name	age	p
t_1	1	1	A	20	0.6
t_2	2	1	A	50	0.4
t_3	3	2	B	30	0.5
t_4	4	2	A	40	0.3
t_5	5	3	B	40	1.0

SELECT MIN(age), MAX(age)
FROM Person

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
(20,40)	(30,50)	(20,40)	(40,50)	(20,40)	(40,50)

Aggregate Queries - Example

Person

	RK	WK	name	age	p
t_1	1	1	A	20	0.6
t_2	2	1	A	50	0.4
t_3	3	2	B	30	0.5
t_4	4	2	A	40	0.3
t_5	5	3	B	40	1.0

```
SELECT    name, AVG(age)
FROM      Person
GROUP BY  name
```

Complex aggregate query:

- One aggregate functions, two groups of tuples
- Result: two tuples each with two values (one aggregate value)

Aggregate Queries - Example

Person

	RK	WK	name	age	p
t_1	1	1	A	20	0.6
t_2	2	1	A	50	0.4
t_3	3	2	B	30	0.5
t_4	4	2	A	40	0.3
t_5	5	3	B	40	1.0

```
SELECT    name, AVG(age)
FROM      Person
GROUP BY  name
```

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
(A,20) (B,35)	(A,50) (B,35)	(A,30) (B,40)	(A,45) (B,40)	(A,20) (B,40)	(A,50) (B,40)

Aggregate Queries - Challenges

Representation:

- It is possible that every world as another result

$\Rightarrow$ Large number of possible aggregation results

$\Rightarrow$ Cannot be enumerated individually

$\Rightarrow$ Compact representation required

Computation:

- Result cannot be computed by querying all tuples collectively

$\Rightarrow$ No intensional or extensional computation possible

- Enumeration of possible worlds not practical

$\Rightarrow$ Efficient computation methods required

Aggregate Queries - Representation

Simple aggregate queries:

- Three aggregation semantics
- Different compactness, different expressiveness

1. Distribution semantics:

   Probability distribution over the set of all possible aggregation results (corresponds to possible worlds semantics)

2. Range semantics:

   Minimal and maximal possible aggregation result

3. Expected value semantics:

   Expected aggregation result

Aggregate Queries - Example

Person

	RK	WK	name	age	p
t_1	1	1	A	20	0.6
t_2	2	1	A	50	0.4
t_3	3	2	B	30	0.5
t_4	4	2	A	40	0.3
t_5	5	3	B	40	1.0

SELECT AVG(age)
FROM Person

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
30	40	33.3	43.3	30	45

Aggregate Queries - Example

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
30	40	33.3	43.3	30	45

Distribution: $\{(30, 0.42), (33.3, 0.18), (40, 0.2), (43.3, 0.12), (45, 0.08)\}$

(result-probability pairs $\Rightarrow$ aggregation result 30 has probability 0.42)

Range: $(30, 45)$ ($\Rightarrow$ MIN=30 and MAX=45)

Expected value: 35.39

Aggregate Queries - Representation

Complex aggregate queries:

- Collective representation of all possible aggregation results impractical
- Separate representation per aggregate function

$\Rightarrow$ Loss of correlations between the aggregation results of the individual functions

- Separate representation per group

$\Rightarrow$ Loss of correlations between the aggregation results of the individual groups

- Usage of different aggregation semantics per combination of aggregate function and group (e.g. range semantics for average age and expected value semantics for average weight)

Aggregate Queries - Example

Person

	RK	WK	name	age	p
t_1	1	1	A	20	0.6
t_2	2	1	A	50	0.4
t_3	3	2	B	30	0.5
t_4	4	2	A	40	0.3
t_5	5	3	B	40	1.0

SELECT MIN(age), MAX(age)
FROM Person

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
(20,40)	(30,50)	(20,40)	(40,50)	(20,40)	(40,50)

Aggregate Queries - Example

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
(20,40)	(30,50)	(20,40)	(40,50)	(20,40)	(40,50)

Distribution over possible aggregation results:

Possible aggregation result: $\{(20, 40)\}$ Probability: 0.6

Possible aggregation result: $\{(30, 50)\}$ Probability: 0.2

Possible aggregation result: $\{(40, 50)\}$ Probability: 0.2

Aggregate Queries - Example

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
(20,40)	(30,50)	(20,40)	(40,50)	(20,40)	(40,50)

	MIN(age)	MAX(age)
Distribution:	$\{(20, 0.6), (30, 0.2), (40, 0.2)\}$	$\{(40, 0.6), (50, 0.4)\}$
Range:	$(20, 40)$	$(40, 50)$
Expected value:	26	44

Aggregate Queries - Example

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
(20,40)	(30,50)	(20,40)	(40,50)	(20,40)	(40,50)

	MIN(age)	MAX(age)
Distribution:		$0, 0.6), (50, 0.4)\}$
Range:		50)
Expected value:	26	44

Loss of correlations between the results of both functions

Aggregate Queries - Example

Person

	RK	WK	name	age	p
t_1	1	1	A	20	0.6
t_2	2	1	A	50	0.4
t_3	3	2	B	30	0.5
t_4	4	2	A	40	0.3
t_5	5	3	B	40	1.0

```
SELECT    name, AVG(age)
FROM      Person
GROUP BY  name
```

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
(A,20) (B,35)	(A,50) (B,35)	(A,30) (B,40)	(A,45) (B,40)	(A,20) (B,40)	(A,50) (B,40)

Aggregate Queries - Example

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
(A,20) (B,35)	(A,50) (B,35)	(A,30) (B,40)	(A,45) (B,40)	(A,20) (B,40)	(A,50) (B,40)

Each possible world produces another aggregation result
(distribution over possible aggregation results is represented above)

	name: A	name: B
Distribution:	$\{(20, 0.42), (30, 0.18), (45, 0.12), (50, 0.28)\}$	$\{(35, 0.5), (40, 0.5)\}$
Range:	$(20, 50)$	$(35, 40)$
Expected value:	33.2	37.5

Aggregate Queries - Example

Possible aggregation results:

W_1 $\{t_1, t_3, t_5\}$ Pr=0.3	W_2 $\{t_2, t_3, t_5\}$ Pr=0.2	W_3 $\{t_1, t_4, t_5\}$ Pr=0.18	W_4 $\{t_2, t_4, t_5\}$ Pr=0.12	W_5 $\{t_1, t_5\}$ Pr=0.12	W_6 $\{t_2, t_5\}$ Pr=0.08
(A,20) (B,35)	(A,50) (B,35)	(A,30) (B,40)	(A,45) (B,40)	(A,20) (B,40)	(A,50) (B,40)

	name: A	name: B
Distribution:	$\{(20, 0.3), (50, 0.2), (30, 0.18), (45, 0.12), (20, 0.12), (50, 0.08)\}$	$\{(35, 0.5), (40, 0.5)\}$
Range:	$(20, 50)$	$(35, 40)$
Expected value:	33.2	37.5

Loss of correlations between the results of both groups

Aggregate Queries - Example

Person

	RK	***WK***	***name***	***age***	***p***
t_1	**1**	**1**	*A*	*20*	0.6
t_2	**2**	**1**	*A*	*50*	0.4
t_3	**3**	**2**	*B*	*30*	0.5
t_4	**4**	**2**	*A*	*40*	0.3
t_5	**5**	**3**	*B*	*40*	1.0

```
SELECT      name,
            RANGE_MIN(age),
            EXP_MAX(age),
            DISTRIB_AVG(age)
FROM        Person
GROUP BY    name
```

Query result:

	name	MIN(*age*)	MAX(*age*)	AVG(*age*)
t_6	*A*	(*20*,*50*)	*35.6*	{(*20*,0.42),(*30*,0.18),(*45*,0.12),(*50*,0.28)}
t_7	*B*	(*30*,*40*)	*40*	{(*35*,0.5),(*40*,0.5)}

Aggregate Queries - Computation

Computation depends on:

- Aggregation semantics
- Aggregate function
- Representation system

Aggregation semantics:

- Distribution semantics is more complex than the range or the expected value semantics

Aggregate functions:

- COUNT is less complex than MIN, MAX, AVG, or SUM

Representation system:

- The more powerful the system, the more expensive the computation

Aggregate Queries - Computation Example

Distribution semantics, COUNT function, BID-databases:

- In this case, the number of possible aggregation results is limited by the number of blocks
- Let $\mathcal{B} = \{B_1, \ldots, B_k\}$ be the set of all blocks
- Dynamic programming algorithm that fills a $|\mathcal{B}| \times |\mathcal{B}+1|$ matrix M by the following computation rules:

$$
\begin{aligned}
M_{1,1} &= 1 - p(B_1) \\
M_{1,2} &= p(B_1) \\
M_{1,j} &= 0 && \text{for every } j \in \{3, \ldots, |\mathcal{B}+1|\} \\
M_{i,1} &= M_{i-1,1} \times (1 - p(B_i)) && \text{for every } i \in \{2, \ldots, |\mathcal{B}|\} \\
M_{i,j} &= M_{i-1,j-1} \times p(B_i) + M_{i-1,j} \times (1 - p(B_i)) && \text{for every } i, j > 1
\end{aligned}
$$

$\Rightarrow$ Computation in polynomial time

Aggregate Queries - Computation Example

Person

	RK	*WK*	*name*	*age*	*p*
t_1	**1**	**1**	*A*	*20*	0.6
t_2	**2**	**1**	*A*	*50*	0.3
t_3	**3**	**2**	*B*	*30*	0.5
t_4	**4**	**2**	*A*	*40*	0.3
t_5	**5**	**3**	*B*	*40*	0.4

```
SELECT  DISTRIB_COUNT(*)
FROM    Person
```

Result computation:

Number of persons

WK	0	1	2	3
1	-	-	-	-
2	-	-	-	-
3	-	-	-	-

The three blocks are added to the computation process one after another

Aggregate Queries - Computation Example

Person

	RK	WK	name	age	p
t_1	1	1	A	20	0.6
t_2	2	1	A	50	0.3
t_3	3	2	B	30	0.5
t_4	4	2	A	40	0.3
t_5	5	3	B	40	0.4

```
SELECT  DISTRIB_COUNT(*)
FROM    Person
```

Result computation:

Number of persons

WK	0	1	2	3
1	0.1	0.9	0	0
2	-	-	-	-
3	-	-	-	-

Add the first block to the computation process

Aggregate Queries - Computation Example

Person

	RK	WK	name	age	p
t_1	1	1	A	20	0.6
t_2	2	1	A	50	0.3
t_3	3	2	B	30	0.5
t_4	4	2	A	40	0.3
t_5	5	3	B	40	0.4

```
SELECT  DISTRIB_COUNT(*)
FROM    Person
```

Result computation:

Number of persons

WK	0	1	2	3
1	0.1	0.9	0	0
2	$0.1 \times 0.2 = 0.02$	$0.1 \times 0.8 + 0.9 \times 0.2 = 0.26$	$0.9 \times 0.8 = 0.72$	0
3	-	-	-	-

Add the second block to the computation process

Aggregate Queries - Computation Example

Person

	RK	*WK*	*name*	*age*	*p*
t_1	**1**	**1**	*A*	*20*	0.6
t_2	**2**	**1**	*A*	*50*	0.3
t_3	**3**	**2**	*B*	*30*	0.5
t_4	**4**	**2**	*A*	*40*	0.3
t_5	**5**	**3**	*B*	*40*	0.4

```
SELECT  DISTRIB_COUNT(*)
FROM    Person
```

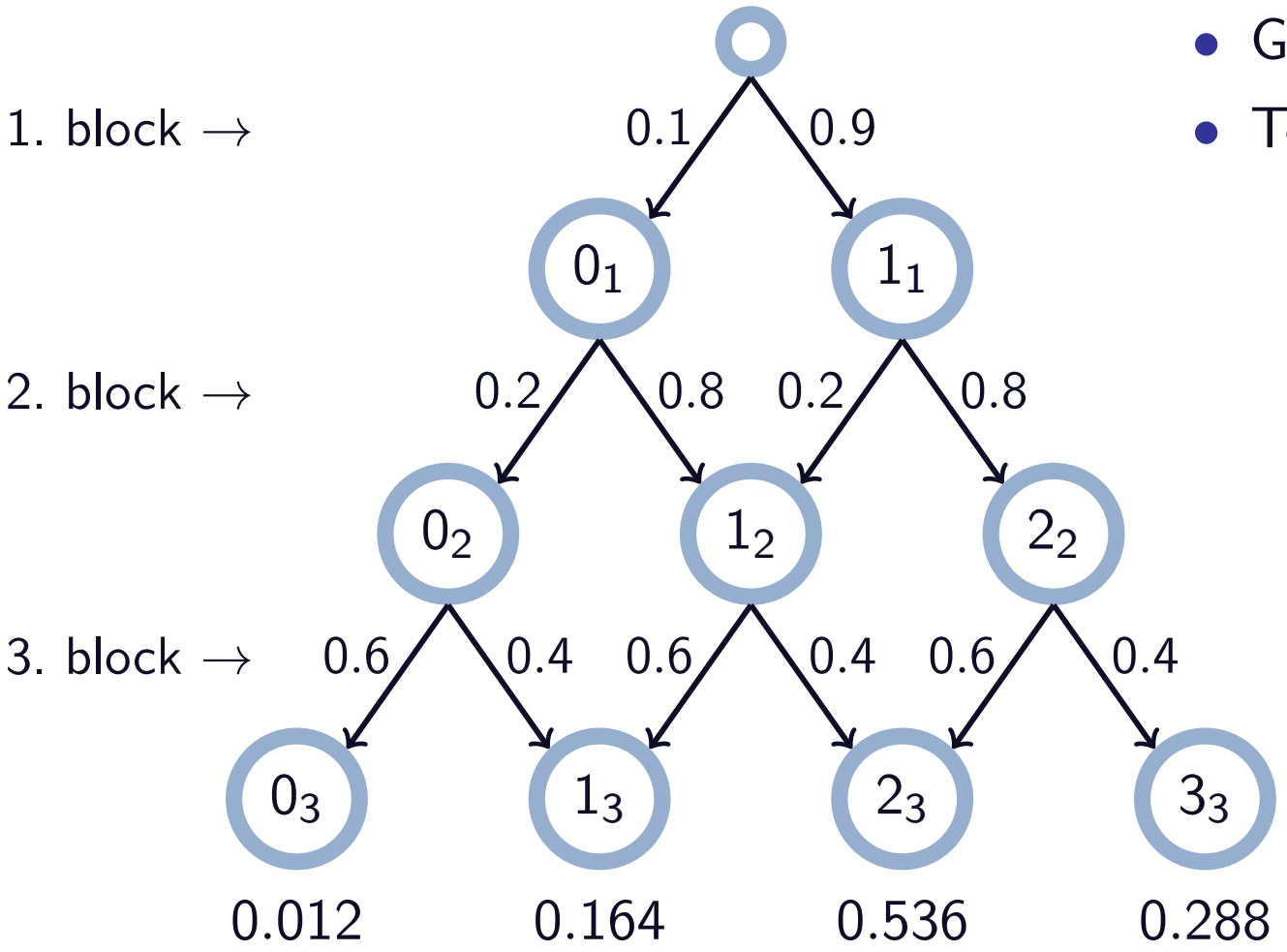
Result computation:

Number of persons

WK	0	1	2	3
1	0.1	0.9	0	0
2	$0.1 \times 0.2 = 0.02$	$0.1 \times 0.8 + 0.9 \times 0.2 = 0.26$	$0.9 \times 0.8 = 0.72$	0
3	$0.02 \times 0.6 = 0.012$	$0.02 \times 0.4 + 0.26 \times 0.6 = 0.164$	$0.26 \times 0.4 + 0.72 \times 0.6 = 0.536$	$0.72 \times 0.4 = 0.288$

Add the third block to the computation process

Aggregate Queries - Computation Example



- Given: n blocks
- Total complexity:

$$2 \times \sum_{i=1}^{n} i = n^2 + n$$

Aggregate Queries - Computation Example

More algorithms can be found in:

- Avigdor Gal, Maria Vanina Martinez, Gerardo I. Simari, and V. S. Subrahmanian. Aggregate Query Answering under Uncertain Schema Mappings. In ICDE, pages 940-951, 2009.
- Raghotham Murthy, Robert Ikeda, and Jennifer Widom. Making Aggregation Work in Uncertain and Probabilistic Databases. IEEE Trans. Knowl. Data Eng., 23(8):1261-1273, 2011.
- Robert B. Ross, V. S. Subrahmanian, and John Grant. Aggregate operators in probabilistic databases. J. ACM, 52(1):54-101, 2005.
- Robert Fink, Larisa Han, and Dan Olteanu. Aggregation in Probabilistic Databases via Knowledge Compilation. PVLDB, 5(5):490-501, 2012.