Uncertain Data Management
Probabilistic Query Evaluation

Antoine Amarilli[1], Silviu Maniu[2]

December 5th, 2017

[1]Télécom ParisTech

[2]LRI

Table of contents

Reminder: TID

Remember tuple-independent databases (TID):

U

date	prof	
04	S	0.8
04	A	0.2

Reminder: TID

Remember tuple-independent databases (TID):

U

date	prof	
04	S	0.8
04	A	0.2

Remember that they stand for a probabilistic database:

U

date	prof
04	S
04	A

Reminder: TID

Remember tuple-independent databases (TID):

U

date	prof	
04	S	0.8
04	A	0.2

Remember that they stand for a probabilistic database:

U

date	prof
04	S
04	A

U

date	prof
04	S
04	A

Reminder: TID

Remember tuple-independent databases (TID):

U

date	prof	
04	S	0.8
04	A	0.2

Remember that they stand for a probabilistic database:

U

date	prof
04	S
04	A

U

date	prof
04	S
04	A

U

date	prof
04	S
04	A

Reminder: TID

Remember tuple-independent databases (TID):

U

date	prof	
04	S	0.8
04	A	0.2

Remember that they stand for a probabilistic database:

U

date	prof
04	S
04	A

U

date	prof
04	S
04	A

U

date	prof
04	S
04	A

U

date	prof
04	S
04	A

Reminder: TID

Remember tuple-independent databases (TID):

U

date	prof	
04	S	0.8
04	A	0.2

Remember that they stand for a probabilistic database:

U

date	prof
04	S
04	A

0.8×0.2

U

date	prof
04	S
04	A

U

date	prof
04	S
04	A

U

date	prof
04	S
04	A

Reminder: TID

Remember tuple-independent databases (TID):

U

date	prof	
04	S	0.8
04	A	0.2

Remember that they stand for a probabilistic database:

U

date	prof
04	S
04	A

0.8×0.2

U

date	prof
04	S
04	A

$(1 - 0.8) \times 0.2$

U

date	prof
04	S
04	A

U

date	prof
04	S
04	A

Reminder: TID

Remember tuple-independent databases (TID):

U

date	prof	
04	S	0.8
04	A	0.2

Remember that they stand for a probabilistic database:

U

date	prof
04	S
04	A

0.8×0.2

U

date	prof
04	S
04	A

$(1 - 0.8) \times 0.2$

U

date	prof
04	S
04	A

$0.8 \times (1 - 0.2)$

U

date	prof
04	S
04	A

Reminder: TID

Remember tuple-independent databases (TID):

U

date	prof	
04	S	0.8
04	A	0.2

Remember that they stand for a probabilistic database:

U

date	prof
04	S
04	A

0.8×0.2

U

date	prof
04	S
04	A

$(1 - 0.8) \times 0.2$

U

date	prof
04	S
04	A

$0.8 \times (1 - 0.2)$

U

date	prof
04	S
04	A

$(1 - 0.8) \times (1 - 0.2)$

Reminder: Relational algebra on probabilistic databases

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

U_2

11	A.	C017

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

$\bigcup$

U_2

11	A.	C017

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

V_1

$\bigcup$

U_2

11	A.	C017

V_2

11	A.	C017

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

$\pi(U_1) = 0.8$

U_2

11	A.	C017

$\pi(U_2) = 0.2$

$\bigcup$

V_1

$\pi(V_1) = 0.9$

V_2

11	A.	C017

$\pi(V_2) = 0.1$

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

$\pi(U_1) = 0.8$

U_2

11	A.	C017

$\pi(U_2) = 0.2$

$\cup$

V_1

$\pi(V_1) = 0.9$

V_2

11	A.	C017

$\pi(V_2) = 0.1$

$=$

Reminder: Relational algebra on probabilistic databases

W_1

04	S.	C017
11	S.	C47

U_1

04	S.	C017
11	S.	C47

$\pi(U_1) = 0.8$

U_2

11	A.	C017

$\pi(U_2) = 0.2$

$\cup$

V_1

$\pi(V_1) = 0.9$

V_2

11	A.	C017

$\pi(V_2) = 0.1$

$=$

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

$\pi(U_1) = 0.8$

U_2

11	A.	C017

$\pi(U_2) = 0.2$

$\cup$

V_1

$\pi(V_1) = 0.9$

V_2

11	A.	C017

$\pi(V_2) = 0.1$

$=$

W_1

04	S.	C017
11	S.	C47

W_2

04	S.	C017
11	S.	C47
11	A.	C017

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

$\pi(U_1) = 0.8$

U_2

11	A.	C017

$\pi(U_2) = 0.2$

$\cup$

V_1

$\pi(V_1) = 0.9$

V_2

11	A.	C017

$\pi(V_2) = 0.1$

$=$

W_1

04	S.	C017
11	S.	C47

W_2

04	S.	C017
11	S.	C47
11	A.	C017

W_3

11	A.	C017

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

$\pi(U_1) = 0.8$

U_2

11	A.	C017

$\pi(U_2) = 0.2$

$\cup$

V_1

$\pi(V_1) = 0.9$

V_2

11	A.	C017

$\pi(V_2) = 0.1$

$=$

W_1

04	S.	C017
11	S.	C47

$\pi(W_1) = 0.8 \times 0.9$

W_2

04	S.	C017
11	S.	C47
11	A.	C017

W_3

11	A.	C017

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

$\pi(U_1) = 0.8$

U_2

11	A.	C017

$\pi(U_2) = 0.2$

$\cup$

V_1

$\pi(V_1) = 0.9$

V_2

11	A.	C017

$\pi(V_2) = 0.1$

$=$

W_1

04	S.	C017
11	S.	C47

$\pi(W_1) = 0.8 \times 0.9$

W_2

04	S.	C017
11	S.	C47
11	A.	C017

$\pi(W_1) = 0.8 \times 0.1$

W_3

11	A.	C017

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

$\pi(U_1) = 0.8$

U_2

11	A.	C017

$\pi(U_2) = 0.2$

$\cup$

V_1

$\pi(V_1) = 0.9$

V_2

11	A.	C017

$\pi(V_2) = 0.1$

$=$

W_1

04	S.	C017
11	S.	C47

$\pi(W_1) = 0.8 \times 0.9$

W_2

04	S.	C017
11	S.	C47
11	A.	C017

$\pi(W_1) = 0.8 \times 0.1$

W_3

11	A.	C017

$\pi(W_1) = 0.2 \times 0.9$

Reminder: Relational algebra on probabilistic databases

U_1

04	S.	C017
11	S.	C47

$\pi(U_1) = 0.8$

U_2

11	A.	C017

$\pi(U_2) = 0.2$

$\cup$

V_1

$\pi(V_1) = 0.9$

V_2

11	A.	C017

$\pi(V_2) = 0.1$

$=$

W_1

04	S.	C017
11	S.	C47

$\pi(W_1) = 0.8 \times 0.9$

W_2

04	S.	C017
11	S.	C47
11	A.	C017

$\pi(W_1) = 0.8 \times 0.1$

W_3

11	A.	C017

$$\pi(W_1) = 0.2 \times 0.9 + 0.2 \times 0.1$$

Reminder: strong representation system

- TID are not a strong representation system

Reminder: strong representation system

- TID are not a strong representation system
- The result of a relational algebra query on a TID database is generally not representable as a TID database

Reminder: strong representation system

- TID are not a strong representation system
- The result of a relational algebra query on a TID database is generally not representable as a TID database

→ Often, we don't want the entire result

Reminder: strong representation system

- TID are not a strong representation system
- The result of a relational algebra query on a TID database is generally not representable as a TID database

$\rightarrow$ Often, we don't want the entire result

$\rightarrow$ We just want to know the probability of each output tuple

Probabilistic query evaluation

- Inputs:
  - a database D of TID instances
  - a relational algebra query Q
  - a result tuple $\vec{t}$

Probabilistic query evaluation

- Inputs:
  - a database D of TID instances
  - a relational algebra query Q
  - a result tuple $\vec{t}$
- Output : what is the probability that $\vec{t}$ is in $Q(D)$?

Probabilistic query evaluation

- Inputs:
  - a database D of TID instances
  - a relational algebra query Q
  - a result tuple $\vec{t}$
- Output : what is the probability that $\vec{t}$ is in $Q(D)$?

$\rightarrow$ What is the marginal probability of obtaining $\vec{t}$ as a result?

Probabilistic query evaluation example

TID instance U

date	**prof**	
04	S	0.8
04	A	0.2

Query Q

$\pi_{\textbf{prof}}(U)$

Tuple $\vec{t}$

S

Probabilistic query evaluation example

TID instance U | Query Q | Tuple $\vec{t}$

date	prof	
04	S	0.8
04	A	0.2

$\pi_{\textbf{prof}}(U)$

S

$\rightarrow$ The marginal probability is 0.8

Marginal probabilities vs TID representations

Here's another example:

TID instance U'

date	**prof**	
04	S	1
04	A	1

TID instance V

date	
04	0.5

Query Q

$\pi_{\textbf{prof}}(U' \bowtie V)$

Tuple $\vec{t}$

S
and
A

Marginal probabilities vs TID representations

Here's another example:

TID instance U'

date	prof	
04	S	1
04	A	1

TID instance V

date	
04	0.5

Query Q

$\pi_{\mathbf{prof}}(U' \bowtie V)$

Tuple $\vec{t}$

S
and
A

$\rightarrow$ The marginal probability of S is 0.5

$\rightarrow$ The marginal probability of A is also 0.5

Marginal probabilities vs TID representations

Here's another example:

TID instance U'

date	prof	
04	S	1
04	A	1

TID instance V

date	
04	0.5

Query Q

$\pi_{\mathbf{prof}}(U' \bowtie V)$

Tuple $\vec{t}$

S
and
A

- $\rightarrow$ The marginal probability of S is 0.5
- $\rightarrow$ The marginal probability of A is also 0.5
- $\rightarrow$ Caution: It does not mean that the result is the TID instance at the right!

prof	
S	0.5
A	0.5

Motivation for probabilistic query evaluation

- Answers the intuitive question "what is the probability of this"?
- Often more interesting than the correlations between worlds

Motivation for probabilistic query evaluation

- Answers the intuitive question "what is the probability of this"?
- Often more interesting than the correlations between worlds

$\rightarrow$ How to compute these probabilities?

Table of contents

Naive probabilistic query evaluation

- Compute the probabilistic instance represented by the input
  - $\rightarrow$ Finite number of possible worlds

Naive probabilistic query evaluation

- Compute the probabilistic instance represented by the input
  - $\rightarrow$ Finite number of possible worlds
- Run the query over each possible world
  - $\rightarrow$ Check if the result tuple is in the output

Naive probabilistic query evaluation

- Compute the probabilistic instance represented by the input
  - $\rightarrow$ Finite number of possible worlds
- Run the query over each possible world
  - $\rightarrow$ Check if the result tuple is in the output
- Sum the probabilities of all worlds that contain the output tuple

Naive probabilistic query evaluation example

TID instance U

date	**prof**	
04	S	0.8
04	A	0.2

Query Q

$\pi_{\textbf{prof}}(U)$

Tuple $\vec{t}$

S

Naive probabilistic query evaluation example

TID instance U

date	prof	
04	S	0.8
04	A	0.2

Query Q

$\pi_{\textbf{prof}}(U)$

Tuple $\vec{t}$

S

Probabilistic relation $Q(U)$:

prof
S
A

0.8×0.2

prof
S
A

$(1 - 0.8) \times 0.2$

prof
S
A

$0.8 \times (1 - 0.2)$

prof
S
A

$(1 - 0.8) \times (1 - 0.2)$

Naive probabilistic query evaluation example

TID instance U

date	prof	
04	S	0.8
04	A	0.2

Query Q

$\pi_{\textbf{prof}}(U)$

Tuple $\vec{t}$

S

Probabilistic relation $Q(U)$:

prof
S
A

0.8×0.2

prof
S
A

$(1 - 0.8) \times 0.2$

prof
S
A

$0.8 \times (1 - 0.2)$

prof
S
A

$(1 - 0.8) \times (1 - 0.2)$

Total probability that $\vec{t}$ is in $Q(U)$: $0.8 \times 0.2 + 0.8 \times (1 - 0.2) = 0.8$

Naive evaluation advantages and drawbacks

- Naive evaluation is always possible

Naive evaluation advantages and drawbacks

- Naive evaluation is always possible
- However, it takes exponential time in general
  - $\rightarrow$ Even if the query output has few possible worlds!
  - $\rightarrow$ Feasible if the input has few possible worlds (few tuples)

Naive evaluation advantages and drawbacks

- Naive evaluation is always possible
- However, it takes exponential time in general
  - $\rightarrow$ Even if the query output has few possible worlds!
  - $\rightarrow$ Feasible if the input has few possible worlds (few tuples)
- Probabilistic query evaluation is computationally intractable so it is unlikely that we can beat naive evaluation in general

Naive evaluation advantages and drawbacks

- Naive evaluation is always possible
- However, it takes exponential time in general
  - $\rightarrow$ Even if the query output has few possible worlds!
  - $\rightarrow$ Feasible if the input has few possible worlds (few tuples)
- Probabilistic query evaluation is computationally intractable so it is unlikely that we can beat naive evaluation in general
  - $\rightarrow$ More efficient methods for special cases

Table of contents

Extensional evaluation idea

- Sometimes we can compute the probabilities at each step:

Extensional evaluation idea

- Sometimes we can compute the probabilities at each step:

U

date	prof	
04	S	0.8
04	A	0.2

Extensional evaluation idea

- Sometimes we can compute the probabilities at each step:

U

date	prof	
04	S	0.8
04	A	0.2

V

student	
1	0.4
2	0.6

Extensional evaluation idea

- Sometimes we can compute the probabilities at each step:

U

date	prof	
04	S	0.8
04	A	0.2

V

student	
1	0.4
2	0.6

$U \times V$

date	prof	student
04	S	1
04	S	2
04	A	1
04	A	2

Extensional evaluation idea

- Sometimes we can compute the probabilities at each step:

U

date	prof	
04	S	0.8
04	A	0.2

V

student	
1	0.4
2	0.6

$U \times V$

date	prof	student	
04	S	1	0.8×0.4
04	S	2	
04	A	1	
04	A	2	

Extensional evaluation idea

- Sometimes we can compute the probabilities at each step:

U

date	prof	
04	S	0.8
04	A	0.2

V

student	
1	0.4
2	0.6

$U \times V$

date	prof	student	
04	S	1	0.8×0.4
04	S	2	0.8×0.6
04	A	1	
04	A	2	

Extensional evaluation idea

- Sometimes we can compute the probabilities at each step:

U

date	prof	
04	S	0.8
04	A	0.2

V

student	
1	0.4
2	0.6

$U \times V$

date	prof	student	
04	S	1	0.8×0.4
04	S	2	0.8×0.6
04	A	1	0.2×0.4
04	A	2	

Extensional evaluation idea

- Sometimes we can compute the probabilities at each step:

U

date	prof	
04	S	0.8
04	A	0.2

V

student	
1	0.4
2	0.6

$U \times V$

date	prof	student	
04	S	1	0.8×0.4
04	S	2	0.8×0.6
04	A	1	0.2×0.4
04	A	2	0.2×0.6

Query independence

- We say that queries Q and Q' are syntactically independent if no relation is used in both Q and Q'
  - $\rightarrow$ Example: $Q = R \bowtie S$ and $Q' = \pi_{\mathbf{a}}(T \times U)$
  - $\rightarrow$ Intuition: the tuples in Q and Q' are independent

Query independence

- We say that queries Q and Q' are syntactically independent if no relation is used in both Q and Q'
  - → Example: $Q = R \bowtie S$ and $Q' = \pi_{\mathbf{a}}(T \times U)$
  - → Intuition: the tuples in Q and Q' are independent

- Independent join: if Q and Q' are syntactically independent then we can compute $Q \bowtie Q'$ and $Q \times Q'$: multiply the probabilities

Query independence

- We say that queries Q and Q' are syntactically independent if no relation is used in both Q and Q'
  - $\rightarrow$ Example: $Q = R \bowtie S$ and $Q' = \pi_{\mathbf{a}}(T \times U)$
  - $\rightarrow$ Intuition: the tuples in Q and Q' are independent

- Independent join: if Q and Q' are syntactically independent then we can compute $Q \bowtie Q'$ and $Q \times Q'$: multiply the probabilities

U

date	**pr**	
04	S	0.8
04	A	0.2

Query independence

- We say that queries Q and Q' are syntactically independent if no relation is used in both Q and Q'
  - → Example: $Q = R \bowtie S$ and $Q' = \pi_{\mathbf{a}}(T \times U)$
  - → Intuition: the tuples in Q and Q' are independent

- Independent join: if Q and Q' are syntactically independent then we can compute $Q \bowtie Q'$ and $Q \times Q'$: multiply the probabilities

U

date	pr	
04	S	0.8
04	A	0.2

V

pr	st	
A	1	0.4
S	2	0.6

Query independence

- We say that queries Q and Q' are syntactically independent if no relation is used in both Q and Q'
  - → Example: $Q = R \bowtie S$ and $Q' = \pi_{\mathbf{a}}(T \times U)$
  - → Intuition: the tuples in Q and Q' are independent

- Independent join: if Q and Q' are syntactically independent then we can compute $Q \bowtie Q'$ and $Q \times Q'$: multiply the probabilities

U

date	**pr**	
04	S	0.8
04	A	0.2

V

pr	**st**	
A	1	0.4
S	2	0.6

$U \bowtie V$

date	**pr**	**st**
04	S	2
04	A	1

Query independence

- We say that queries Q and Q' are syntactically independent if no relation is used in both Q and Q'
  - → Example: $Q = R \bowtie S$ and $Q' = \pi_{\mathbf{a}}(T \times U)$
  - → Intuition: the tuples in Q and Q' are independent

- Independent join: if Q and Q' are syntactically independent then we can compute $Q \bowtie Q'$ and $Q \times Q'$: multiply the probabilities

U

date	pr	
04	S	0.8
04	A	0.2

V

pr	st	
A	1	0.4
S	2	0.6

$U \bowtie V$

date	pr	st	
04	S	2	0.8×0.6
04	A	1	

Query independence

- We say that queries Q and Q' are syntactically independent if no relation is used in both Q and Q'
  - → Example: $Q = R \bowtie S$ and $Q' = \pi_{\mathbf{a}}(T \times U)$
  - → Intuition: the tuples in Q and Q' are independent

- Independent join: if Q and Q' are syntactically independent then we can compute $Q \bowtie Q'$ and $Q \times Q'$: multiply the probabilities

U

date	pr	
04	S	0.8
04	A	0.2

V

pr	st	
A	1	0.4
S	2	0.6

$U \bowtie V$

date	pr	st	
04	S	2	0.8×0.6
04	A	1	0.2×0.4

More query independence

- Independent union: for syntactically independent Q and Q' we can compute $Q \cup Q'$ using the rule for independent OR

More query independence

- Independent union: for syntactically independent Q and Q' we can compute $Q \cup Q'$ using the rule for independent OR

U

date	prof	
04	S	0.8
04	A	0.2

More query independence

- Independent union: for syntactically independent Q and Q' we can compute $Q \cup Q'$ using the rule for independent OR

U

date	prof	
04	S	0.8
04	A	0.2

V

date	prof	
04	A	0.4
11	A	0.2

More query independence

- Independent union: for syntactically independent Q and Q' we can compute $Q \cup Q'$ using the rule for independent OR

U

date	prof	
04	S	0.8
04	A	0.2

V

date	prof	
04	A	0.4
11	A	0.2

$U \cup V$

date	prof
04	S
04	A
11	A

More query independence

- Independent union: for syntactically independent Q and Q' we can compute $Q \cup Q'$ using the rule for independent OR

U

date	prof	
04	S	0.8
04	A	0.2

V

date	prof	
04	A	0.4
11	A	0.2

$U \cup V$

date	prof	
04	S	0.8
04	A	
11	A	

More query independence

- Independent union: for syntactically independent Q and Q' we can compute $Q \cup Q'$ using the rule for independent OR

U

date	prof	
04	S	0.8
04	A	0.2

V

date	prof	
04	A	0.4
11	A	0.2

$U \cup V$

date	prof	
04	S	0.8
04	A	$1 - (1 - 0.2) \times (1 - 0.4)$
11	A	

More query independence

- Independent union: for syntactically independent Q and Q' we can compute $Q \cup Q'$ using the rule for independent OR

U

date	prof	
04	S	0.8
04	A	0.2

V

date	prof	
04	A	0.4
11	A	0.2

$U \cup V$

date	prof	
04	S	0.8
04	A	$1 - (1 - 0.2) \times (1 - 0.4)$
11	A	0.2

Selection

Selection can just be applied in the straightforward way:

Selection

Selection can just be applied in the straightforward way:

U

date	prof	
04	S	0.8
04	A	0.2

Selection

Selection can just be applied in the straightforward way:

U

date	prof	
04	S	0.8
04	A	0.2

$\sigma_{\textbf{prof}=\text{“S”}}(U)$

date	prof

Selection

Selection can just be applied in the straightforward way:

U

date	prof	
04	S	0.8
04	A	0.2

$\sigma_{\textbf{prof}=\text{“S”}}(U)$

date	prof	
04	S	0.8

Independent projection

- Self-join-free conjunctive query: a join ($\bowtie$) of projections (π) that does not use the same relation name twice:
  - $\rightarrow$ Example: $Q = R \bowtie S \bowtie \pi_{\mathbf{a}}(T)$

Independent projection

- Self-join-free conjunctive query: a join ($\bowtie$) of projections (π) that does not use the same relation name twice:
  - $\rightarrow$ Example: $Q = R \bowtie S \bowtie \pi_{\mathbf{a}}(T)$
- A separator is an attribute that occurs in all tables of the join:
  - $\rightarrow$ Example: if $R(\mathbf{a}, \mathbf{b}), S(\mathbf{a}), T(\mathbf{a}, \mathbf{c})$ then $\mathbf{a}$ is a separator of Q

Independent projection

- Self-join-free conjunctive query: a join ($\bowtie$) of projections (π) that does not use the same relation name twice:
  - $\rightarrow$ Example: $Q = R \bowtie S \bowtie \pi_{\mathbf{a}}(T)$
- A separator is an attribute that occurs in all tables of the join:
  - $\rightarrow$ Example: if $R(\mathbf{a}, \mathbf{b}), S(\mathbf{a}), T(\mathbf{a}, \mathbf{c})$ then $\mathbf{a}$ is a separator of Q
- If Q is a self-join-free conjunctive query and $\mathbf{a}$ is a separator then $\pi_{-\mathbf{a}}$ (projecting away the attribute $\mathbf{a}$) can be computed using independent OR

Independent projection example

U

date	prof	
04	S	0.8
04	A	0.2
11	S	0.4
11	A	0.6

Independent projection example

U

date	prof	
04	S	0.8
04	A	0.2
11	S	0.4
11	A	0.6

$\pi_{\mathbf{date}}(U)$

date
04
11

Independent projection example

U

date	prof	
04	S	0.8
04	A	0.2
11	S	0.4
11	A	0.6

$\pi_{\textbf{date}}(U)$

date	
04	$1 - (1 - 0.8) \times (1 - 0.2)$
11	

Independent projection example

U

date	prof	
04	S	0.8
04	A	0.2
11	S	0.4
11	A	0.6

$\pi_{\textbf{date}}(U)$

date	
04	$1-(1-0.8)\times(1-0.2)$
11	$1-(1-0.4)\times(1-0.6)$

Another independent projection example

Consider the two tables:

U

date	prof	
04	S	$1/2$

Another independent projection example

Consider the two tables:

U

date	prof	
04	S	1/2

V

prof	cause	
S	illness	1/2
S	bahamas	1/2

Another independent projection example

Consider the two tables:

U

date	prof	
04	S	1/2

V

prof	cause	
S	illness	1/2
S	bahamas	1/2

- Query: $Q(U, V) = \pi_{\textbf{date},\textbf{prof}}(U \bowtie V)$
- Can be rewritten as: $U \bowtie \pi_{-\textbf{cause}}(V)$

Another independent projection example

Consider the two tables:

U

date	prof	
04	S	1/2

V

prof	cause	
S	illness	1/2
S	bahamas	1/2

- Query: $Q(U, V) = \pi_{\textbf{date},\textbf{prof}}(U \bowtie V)$
- Can be rewritten as: $U \bowtie \pi_{-\textbf{cause}}(V)$

$\pi_{-\textbf{cause}}(V)$

prof	
S	3/4

$U \bowtie \pi_{-\textbf{cause}}(V)$

date	prof	
04	S	3/8

The choice of plan matters!

- Query: $Q(U, V) = \pi_{\textbf{date},\textbf{prof}}(U \bowtie V)$
- Can be rewritten as $\pi_{-\textbf{cause}}(U \bowtie V)$ instead of $U \bowtie \pi_{-\textbf{cause}}(V)$

The choice of plan matters!

- Query: $Q(U, V) = \pi_{\textbf{date},\textbf{prof}}(U \bowtie V)$
- Can be rewritten as $\pi_{-\textbf{cause}}(U \bowtie V)$ instead of $U \bowtie \pi_{-\textbf{cause}}(V)$

U

date	**prof**	
04	S	1/2

The choice of plan matters!

- Query: $Q(U, V) = \pi_{\textbf{date},\textbf{prof}}(U \bowtie V)$
- Can be rewritten as $\pi_{-\textbf{cause}}(U \bowtie V)$ instead of $U \bowtie \pi_{-\textbf{cause}}(V)$

U

date	**prof**	
04	S	1/2

V

prof	**cause**	
S	illness	1/2
S	bahamas	1/2

The choice of plan matters!

- Query: $Q(U, V) = \pi_{\textbf{date},\textbf{prof}}(U \bowtie V)$
- Can be rewritten as $\pi_{-\textbf{cause}}(U \bowtie V)$ instead of $U \bowtie \pi_{-\textbf{cause}}(V)$

U

date	prof	
04	S	1/2

V

prof	cause	
S	illness	1/2
S	bahamas	1/2

$U \bowtie V$

date	prof	cause
04	S	illness
04	S	bahamas

The choice of plan matters!

- Query: $Q(U, V) = \pi_{\textbf{date},\textbf{prof}}(U \bowtie V)$
- Can be rewritten as $\pi_{-\textbf{cause}}(U \bowtie V)$ instead of $U \bowtie \pi_{-\textbf{cause}}(V)$

U

date	**prof**	
04	S	1/2

V

prof	**cause**	
S	illness	1/2
S	bahamas	1/2

$U \bowtie V$

date	**prof**	**cause**	
04	S	illness	1/4
04	S	bahamas	

The choice of plan matters!

- Query: $Q(U, V) = \pi_{\textbf{date},\textbf{prof}}(U \bowtie V)$
- Can be rewritten as $\pi_{-\textbf{cause}}(U \bowtie V)$ instead of $U \bowtie \pi_{-\textbf{cause}}(V)$

U

date	**prof**	
04	S	1/2

V

prof	**cause**	
S	illness	1/2
S	bahamas	1/2

$U \bowtie V$

date	**prof**	**cause**	
04	S	illness	1/4
04	S	bahamas	1/4

The choice of plan matters!

- Query: $Q(U, V) = \pi_{\textbf{date},\textbf{prof}}(U \bowtie V)$
- Can be rewritten as $\pi_{-\textbf{cause}}(U \bowtie V)$ instead of $U \bowtie \pi_{-\textbf{cause}}(V)$

U

date	**prof**	
04	S	1/2

V

prof	**cause**	
S	illness	1/2
S	bahamas	1/2

$U \bowtie V$

date	**prof**	**cause**	
04	S	illness	1/4
04	S	bahamas	1/4

$\pi_{-\textbf{cause}}(U \bowtie V)$

date	**prof**
04	S

The choice of plan matters!

- Query: $Q(U, V) = \pi_{\textbf{date},\textbf{prof}}(U \bowtie V)$
- Can be rewritten as $\pi_{-\textbf{cause}}(U \bowtie V)$ instead of $U \bowtie \pi_{-\textbf{cause}}(V)$

U

date	prof	
04	S	1/2

V

prof	cause	
S	illness	1/2
S	bahamas	1/2

$U \bowtie V$

date	prof	cause	
04	S	illness	1/4
04	S	bahamas	1/4

$\pi_{-\textbf{cause}}(U \bowtie V)$

date	prof	
04	S	7/16 ??

The choice of plan matters!

- Query: $Q(U,V) = \pi_{\mathbf{date},\mathbf{prof}}(U \bowtie V)$
- Can be rewritten as $\pi_{-\mathbf{cause}}(U \bowtie V)$ instead of $U \bowtie \pi_{-\mathbf{cause}}(V)$

U

date	prof	
04	S	1/2

V

prof	cause	
S	illness	1/2
S	bahamas	1/2

$U \bowtie V$

date	prof	cause	
04	S	illness	1/4
04	S	bahamas	1/4

$\pi_{-\mathbf{cause}}(U \bowtie V)$

date	prof	
04	S	7/16 ??

→ The last projection is not independent, so incorrect result!

Safe plans

- A safe plan for a query Q is a way to implement Q using the extensional operators:

Safe plans

- A safe plan for a query Q is a way to implement Q using the extensional operators:
  - It must use them correctly, e.g., respecting independence

Safe plans

- A safe plan for a query Q is a way to implement Q using the extensional operators:
  - It must use them correctly, e.g., respecting independence
  - It must be equivalent to the desired query Q

Safe plans

- A **safe plan** for a query Q is a way to implement Q using the extensional operators:
  - It must use them **correctly**, e.g., respecting independence
  - It must be **equivalent** to the desired query Q

$\rightarrow$ With a **safe plan**, we can compute the marginal probability of all query results

Do all queries have a safe plan?

- Relations $R(\mathbf{a}), S(\mathbf{a}, \mathbf{b}), T(\mathbf{b})$

Do all queries have a safe plan?

- Relations $R(\mathbf{a}), S(\mathbf{a}, \mathbf{b}), T(\mathbf{b})$
- Query $Q = \pi_{-\mathbf{a}}(\pi_{-\mathbf{b}}(R \bowtie S \bowtie T))$

Do all queries have a safe plan?

- Relations $R(\mathbf{a}), S(\mathbf{a}, \mathbf{b}), T(\mathbf{b})$
- Query $Q = \pi_{-\mathbf{a}}(\pi_{-\mathbf{b}}(R \bowtie S \bowtie T))$
- Does Q have a safe plan?

Do all queries have a safe plan?

- Relations $R(\mathbf{a}), S(\mathbf{a}, \mathbf{b}), T(\mathbf{b})$
- Query $Q = \pi_{-\mathbf{a}}(\pi_{-\mathbf{b}}(R \bowtie S \bowtie T))$
- Does Q have a safe plan?
    - If we do the joins first then no projection is independent

Do all queries have a safe plan?

- Relations $R(\mathbf{a}), S(\mathbf{a}, \mathbf{b}), T(\mathbf{b})$
- Query $Q = \pi_{-\mathbf{a}}(\pi_{-\mathbf{b}}(R \bowtie S \bowtie T))$
- Does Q have a safe plan?
    - If we do the joins first then no projection is independent
    - If we write Q as $\pi_{-\mathbf{a}}(R \bowtie \pi_{-\mathbf{b}}(S \bowtie T)))$ then the projection is not safe

Do all queries have a safe plan?

- Relations $R(\mathbf{a}), S(\mathbf{a}, \mathbf{b}), T(\mathbf{b})$
- Query $Q = \pi_{-\mathbf{a}}(\pi_{-\mathbf{b}}(R \bowtie S \bowtie T))$
- Does Q have a safe plan?
  - If we do the joins first then no projection is independent
  - If we write Q as $\pi_{-\mathbf{a}}(R \bowtie \pi_{-\mathbf{b}}(S \bowtie T)))$
    then the projection is not safe
  - Same problem for $\pi_{-\mathbf{b}}(\pi_{-\mathbf{a}}(R \bowtie S) \bowtie T)$

Do all queries have a safe plan?

- Relations $R(\mathbf{a}), S(\mathbf{a}, \mathbf{b}), T(\mathbf{b})$
- Query $Q = \pi_{-\mathbf{a}}(\pi_{-\mathbf{b}}(R \bowtie S \bowtie T))$
- Does Q have a safe plan?
  - If we do the joins first then no projection is independent
  - If we write Q as $\pi_{-\mathbf{a}}(R \bowtie \pi_{-\mathbf{b}}(S \bowtie T)))$ then the projection is not safe
  - Same problem for $\pi_{-\mathbf{b}}(\pi_{-\mathbf{a}}(R \bowtie S) \bowtie T)$

$\rightarrow$ In fact Q is intractable and it has no safe plan

Extensional query evaluation summary

- Extensional query evaluation:
  - Express the query as a safe plan with the extensional operators
  - Compute the query results and their probabilities via the plan
  - The probabilities are correct because the plan is safe

Extensional query evaluation summary

- Extensional query evaluation:
  - Express the query as a safe plan with the extensional operators
  - Compute the query results and their probabilities via the plan
  - The probabilities are correct because the plan is safe
- Summary of safe operators:
  - Product and join of syntactically independent queries
    $\rightarrow$ Product of independent probabilities
  - Union of syntactically independent queries
    $\rightarrow$ Independent OR of the probabilities
  - Projecting away a separator attribute
    $\rightarrow$ Independent OR because the tuples in each group are independent
  - Applying selection in the straightforward way
  - Also other rules: negation, inclusion-exclusion, etc.

Extensional query evaluation summary

- Extensional query evaluation:
  - Express the query as a safe plan with the extensional operators
  - Compute the query results and their probabilities via the plan
  - The probabilities are correct because the plan is safe
- Summary of safe operators:
  - Product and join of syntactically independent queries

    $\rightarrow$ Product of independent probabilities
  - Union of syntactically independent queries

    $\rightarrow$ Independent OR of the probabilities
  - Projecting away a separator attribute

    $\rightarrow$ Independent OR because the tuples in each group are independent
  - Applying selection in the straightforward way
  - Also other rules: negation, inclusion-exclusion, etc.

$\rightarrow$ Not all queries have safe plans

Table of contents

Idea of intensional query evaluation

- We cannot always compute directly the probabilities of results
- Idea:
  - Compute a lineage expression for each output tuple describing the possible worlds where it appears
  - Compute the probability of these lineage expressions

Idea of intensional query evaluation

- We cannot always compute directly the probabilities of results
- Idea:
  - Compute a lineage expression for each output tuple describing the possible worlds where it appears
  - Compute the probability of these lineage expressions
- Advantages:
  - Intensional evaluation is always possible (but not always efficient)
  - Intensional evaluation is more modular:
    - → Compute the lineage expression (no probabilities)
    - → Use any model counting method or software

Idea of intensional query evaluation

- We cannot always compute directly the probabilities of results
- Idea:
  - Compute a lineage expression for each output tuple describing the possible worlds where it appears
  - Compute the probability of these lineage expressions
- Advantages:
  - Intensional evaluation is always possible (but not always efficient)
  - Intensional evaluation is more modular:
    - $\rightarrow$ Compute the lineage expression (no probabilities)
    - $\rightarrow$ Use any model counting method or software
- Disadvantages:
  - Two steps: (1.) compute the lineage; (2.) compute the probability
  - The lineage expression loses information about the query

Reminder: pc-tables

Remember that a TID is a special case of a pc-table:

U

date	**prof**	$x_1 : 0.8, x_2 : 0.2$
04	S	x_1
04	A	x_2

Remember that pc-tables are a strong representation system (same rules as for pc-tables for relational algebra operators)

pc-table query example

U

date	prof	$x_1 : 0.8, x_2 : 0.2$
04	S	x_1
04	A	x_2

pc-table query example

U

date	**prof**	$x_1 : 0.8, x_2 : 0.2$
04	S	x_1
04	A	x_2

$\pi_{\textbf{date}}(U)$

date	$x_1 : 0.8, x_2 : 0.2$
04	$x_1 \lor x_2$

Lineage expression

	$\pi_{\textbf{date}}(U)$
date	$x_1 : 0.8, x_2 : 0.2$
04	$x_1 \vee x_2$

- The lineage expression $x_1 \vee x_2$ describes the possible worlds where the tuple 04 appears.

Lineage expression

$\pi_{\text{date}}(U)$

date	$x_1 : 0.8, x_2 : 0.2$
04	$x_1 \lor x_2$

- The lineage expression $x_1 \lor x_2$ describes the possible worlds where the tuple 04 appears.
- The probability that $x_1 \lor x_2$ is true is exactly the probability that this tuple is in the result

Intensional query evaluation

- Translate the TID to a pc-table

Intensional query evaluation

- Translate the TID to a pc-table
- Evaluate the query on the pc-table using c-table rules

Intensional query evaluation

- Translate the TID to a pc-table
- Evaluate the query on the pc-table using c-table rules
- Compute the probability $\mathrm{P}(\phi)$ of the lineage expression ϕ of the output tuple under consideration

Intensional query evaluation

- Translate the TID to a pc-table
- Evaluate the query on the pc-table using c-table rules
- Compute the probability $\mathrm{P}(\phi)$ of the lineage expression ϕ of the output tuple under consideration

$\rightarrow$ We have reduced probabilistic query evaluation to computing the probability that a Boolean formula is true

How to compute the probability of a lineage expression?

Many ways to compute the probability $\mathrm{P}(\phi)$:

- Naive method: enumerate all possibilities (exponential)

How to compute the probability of a lineage expression?

Many ways to compute the probability $\mathrm{P}(\phi)$:

- Naive method: enumerate all possibilities (exponential)
- Use some simple intensional rules
  - $\rightarrow$ e.g., $\mathrm{P}(\phi(x, y, z) \wedge \psi(x', y', z')) = \mathrm{P}(\phi(x, y, z)) \times \mathrm{P}(\psi(x', y', z'))$ thanks to independence

How to compute the probability of a lineage expression?

Many ways to compute the probability $\mathrm{P}(\phi)$:

- Naive method: enumerate all possibilities (exponential)
- Use some simple intensional rules
  $\rightarrow$ e.g., $\mathrm{P}(\phi(x, y, z) \wedge \psi(x', y', z')) = \mathrm{P}(\phi(x, y, z)) \times \mathrm{P}(\psi(x', y', z'))$ thanks to independence
- Compile the lineage expression in a tractable formalism
  - read-once formulas
  - tractable circuit classes
  - binary decision diagrams

How to compute the probability of a lineage expression?

Many ways to compute the probability $\mathrm{P}(\phi)$:

- Naive method: enumerate all possibilities (exponential)
- Use some simple intensional rules
  $\rightarrow$ e.g., $\mathrm{P}(\phi(x, y, z) \wedge \psi(x', y', z')) = \mathrm{P}(\phi(x, y, z)) \times \mathrm{P}(\psi(x', y', z'))$ thanks to independence
- Compile the lineage expression in a tractable formalism
  - read-once formulas
  - tractable circuit classes
  - binary decision diagrams
- Approximate the probability of the lineage expression

How to compute the probability of a lineage expression?

Many ways to compute the probability $\mathrm{P}(\phi)$:

- Naive method: enumerate all possibilities (exponential)
- Use some simple intensional rules
  $\rightarrow$ e.g., $\mathrm{P}(\phi(x,y,z) \wedge \psi(x',y',z')) = \mathrm{P}(\phi(x,y,z)) \times \mathrm{P}(\psi(x',y',z'))$ thanks to independence
- Compile the lineage expression in a tractable formalism
  - read-once formulas
  - tractable circuit classes
  - binary decision diagrams
- Approximate the probability of the lineage expression
- Use an external weighted model counter

Naive evaluation

Example: formula $\phi = x_1 \vee x_2$ with $\mathrm{P}(x_1 = 1) = 0.8$ and $\mathrm{P}(x_2 = 1) = 0.2$

- If x_1 is true and x_2 is true, the formula is true
  - $\rightarrow$ Probability 0.8×0.2

Naive evaluation

Example: formula $\phi = x_1 \vee x_2$ with $\mathrm{P}(x_1 = 1) = 0.8$ and $\mathrm{P}(x_2 = 1) = 0.2$

- If x_1 is true and x_2 is true, the formula is true
  - $\rightarrow$ Probability 0.8×0.2
- If x_1 is true and x_2 is false, the formula is true
  - $\rightarrow$ Probability $0.8 \times (1 - 0.2)$

Naive evaluation

Example: formula $\phi = x_1 \vee x_2$ with $\mathrm{P}(x_1 = 1) = 0.8$ and $\mathrm{P}(x_2 = 1) = 0.2$

- If x_1 is true and x_2 is true, the formula is true
  - $\rightarrow$ Probability 0.8×0.2
- If x_1 is true and x_2 is false, the formula is true
  - $\rightarrow$ Probability $0.8 \times (1 - 0.2)$
- If x_1 is false and x_2 is true, the formula is true
  - $\rightarrow$ Probability $(1 - 0.8) \times 0.2$

Naive evaluation

Example: formula $\phi = x_1 \vee x_2$ with $\mathrm{P}(x_1 = 1) = 0.8$ and $\mathrm{P}(x_2 = 1) = 0.2$

- If x_1 is true and x_2 is true, the formula is true
  - $\rightarrow$ Probability 0.8×0.2
- If x_1 is true and x_2 is false, the formula is true
  - $\rightarrow$ Probability $0.8 \times (1 - 0.2)$
- If x_1 is false and x_2 is true, the formula is true
  - $\rightarrow$ Probability $(1 - 0.8) \times 0.2$
- If x_1 is false and x_2 is false, the formula is false
  - $\rightarrow$ Probability $(1 - 0.8) \times (1 - 0.2)$

Naive evaluation

Example: formula $\phi = x_1 \vee x_2$ with $\mathrm{P}(x_1 = 1) = 0.8$ and $\mathrm{P}(x_2 = 1) = 0.2$

- If x_1 is true and x_2 is true, the formula is true
  - $\rightarrow$ Probability 0.8×0.2
- If x_1 is true and x_2 is false, the formula is true
  - $\rightarrow$ Probability $0.8 \times (1 - 0.2)$
- If x_1 is false and x_2 is true, the formula is true
  - $\rightarrow$ Probability $(1 - 0.8) \times 0.2$
- If x_1 is false and x_2 is false, the formula is false
  - $\rightarrow$ Probability $(1 - 0.8) \times (1 - 0.2)$

$\rightarrow$ $\mathrm{P}(\phi) = 0.8 \times 0.2 + 0.8 \times (1 - 0.2) + (1 - 0.8) \times 0.2$

Naive evaluation

Example: formula $\phi = x_1 \vee x_2$ with $\mathrm{P}(x_1 = 1) = 0.8$ and $\mathrm{P}(x_2 = 1) = 0.2$

- If x_1 is true and x_2 is true, the formula is true
  - $\rightarrow$ Probability 0.8×0.2
- If x_1 is true and x_2 is false, the formula is true
  - $\rightarrow$ Probability $0.8 \times (1 - 0.2)$
- If x_1 is false and x_2 is true, the formula is true
  - $\rightarrow$ Probability $(1 - 0.8) \times 0.2$
- If x_1 is false and x_2 is false, the formula is false
  - $\rightarrow$ Probability $(1 - 0.8) \times (1 - 0.2)$

$\rightarrow$ $\mathrm{P}(\phi) = 0.8 \times 0.2 + 0.8 \times (1 - 0.2) + (1 - 0.8) \times 0.2 = 0.84$

Intensional rule definitions

- ϕ and ψ are syntactically independent if they have no variables in common
  - $\rightarrow$ E.g., $\phi(x, y, z)$ and $\psi(x', y', z')$

Intensional rule definitions

- ϕ and ψ are syntactically independent
  if they have no variables in common
  - $\rightarrow$ E.g., $\phi(x, y, z)$ and $\psi(x', y', z')$
- ϕ and ψ are mutually exclusive if $\phi \wedge \psi$ is unsatisfiable
  - $\rightarrow$ E.g., $\phi = x \wedge y$ and $\psi = \neg x \wedge (y \vee z)$

Intensional rule definitions

- ϕ and ψ are syntactically independent if they have no variables in common
  - $\rightarrow$ E.g., $\phi(x, y, z)$ and $\psi(x', y', z')$
- ϕ and ψ are mutually exclusive if $\phi \land \psi$ is unsatisfiable
  - $\rightarrow$ E.g., $\phi = x \land y$ and $\psi = \neg x \land (y \lor z)$
- $\phi_{|x=0}$ is the result of replacing x by 0 in ϕ (and likewise for $\phi_{|x=1}$)
  - $\rightarrow$ E.g., for $\phi = \neg x \land (y \lor z)$, we have $\phi_{|x=0} = y \lor z$ and $\phi_{|x=1} = \bot$

Intensional rules

- Independent AND: if ϕ and ψ are syntactically independent then: $\mathrm{P}(\phi \wedge \psi) = \mathrm{P}(\phi) \times \mathrm{P}(\psi)$

Intensional rules

- Independent AND: if ϕ and ψ are syntactically independent then: $\mathrm{P}(\phi \wedge \psi) = \mathrm{P}(\phi) \times \mathrm{P}(\psi)$
- Independent OR: if ϕ and ψ are syntactically independent then: $\mathrm{P}(\phi \vee \psi) = 1 - (1 - \mathrm{P}(\phi)) \times (1 - \mathrm{P}(\psi))$

Intensional rules

- Independent AND: if ϕ and ψ are syntactically independent then: $\mathrm{P}(\phi \wedge \psi) = \mathrm{P}(\phi) \times \mathrm{P}(\psi)$
- Independent OR: if ϕ and ψ are syntactically independent then: $\mathrm{P}(\phi \vee \psi) = 1 - (1 - \mathrm{P}(\phi)) \times (1 - \mathrm{P}(\psi))$
- Mutually exclusive OR: if ϕ and ψ are mutually exclusive then: $\mathrm{P}(\phi \vee \psi) = \mathrm{P}(\phi) + \mathrm{P}(\psi)$

Intensional rules

- Independent AND: if ϕ and ψ are syntactically independent then: $\mathrm{P}(\phi \wedge \psi) = \mathrm{P}(\phi) \times \mathrm{P}(\psi)$
- Independent OR: if ϕ and ψ are syntactically independent then: $\mathrm{P}(\phi \vee \psi) = 1 - (1 - \mathrm{P}(\phi)) \times (1 - \mathrm{P}(\psi))$
- Mutually exclusive OR: if ϕ and ψ are mutually exclusive then: $\mathrm{P}(\phi \vee \psi) = \mathrm{P}(\phi) + \mathrm{P}(\psi)$
- Negation: for any ϕ, we have $\mathrm{P}(\neg\phi) = 1 - \mathrm{P}(\phi)$

Intensional rules

- Independent AND: if ϕ and ψ are syntactically independent then: $\mathrm{P}(\phi \wedge \psi) = \mathrm{P}(\phi) \times \mathrm{P}(\psi)$
- Independent OR: if ϕ and ψ are syntactically independent then: $\mathrm{P}(\phi \vee \psi) = 1 - (1 - \mathrm{P}(\phi)) \times (1 - \mathrm{P}(\psi))$
- Mutually exclusive OR: if ϕ and ψ are mutually exclusive then: $\mathrm{P}(\phi \vee \psi) = \mathrm{P}(\phi) + \mathrm{P}(\psi)$
- Negation: for any ϕ, we have $\mathrm{P}(\neg\phi) = 1 - \mathrm{P}(\phi)$
- Shannon expansion: for any ϕ and variable x, we have: $\mathrm{P}(\phi) = \mathrm{P}(x = 0) \times \mathrm{P}(\phi_{|x=0}) + \mathrm{P}(x = 1) \times \mathrm{P}(\phi_{|x=1})$

Application of intensional rules

- We can always compute probabilities with intensional rules

→ But Shannon expansions are costly and may be exponential

- The efficiency of these rules depends:
    - on how the lineage is written
    - on the order in which they are applied
- Note that these rules are a bit similar to the extensional rules

Tractable lineage formalisms

- Read-once formula: each variable occurs at most once
  - → If the lineage is written in this way, we can compute the probability with independent AND, independent OR, negation

Tractable lineage formalisms

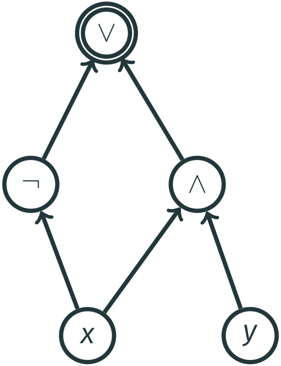
- Read-once formula: each variable occurs at most once
  - $\rightarrow$ If the lineage is written in this way, we can compute the probability with independent AND, independent OR, negation
- Tractable Boolean circuit representations of lineages

Tractable lineage formalisms

- Read-once formula: each variable occurs at most once
  - $\rightarrow$ If the lineage is written in this way, we can compute the probability with independent AND, independent OR, negation
- Tractable Boolean circuit representations of lineages
- Tractable representations as Binary decision diagrams

Boolean circuit representations

Circuits are just a way to represent Boolean formulas
while factoring common subexpressions (more concise)

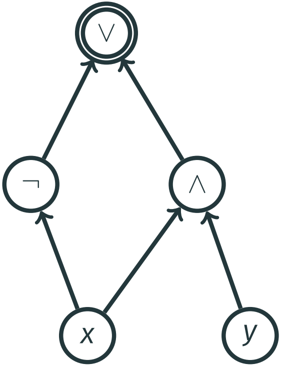


- Directed acyclic graph of gates
- Output gate: 
- Variable gates: 

- Internal gates: 

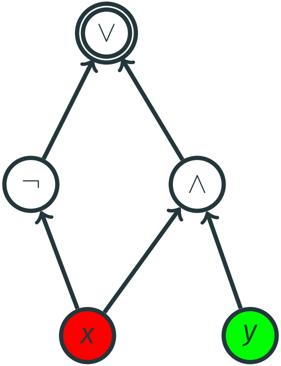
Boolean circuit representations

Circuits are just a way to represent Boolean formulas
while factoring common subexpressions (more concise)



- Directed acyclic graph of gates
- Output gate:
- Variable gates: x
- Internal gates: $\lor$ $\land$ $\lnot$
- Valuation: function from variables to $\{0, 1\}$
  Example: $\nu = \{x \mapsto 0,\ y \mapsto 1\}$...

Circuits are just a way to represent Boolean formulas
while factoring common subexpressions (more concise)



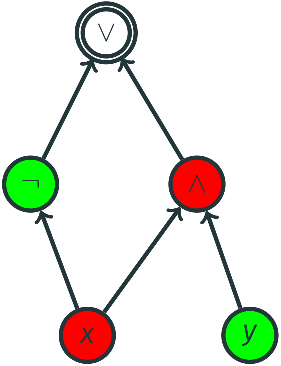
- Directed acyclic graph of gates
- Output gate: 
- Variable gates: 

- Internal gates: 

- Valuation: function from variables to $\{0, 1\}$
  Example: $\nu = \{x \mapsto 0,\ y \mapsto 1\}$...

Boolean circuit representations

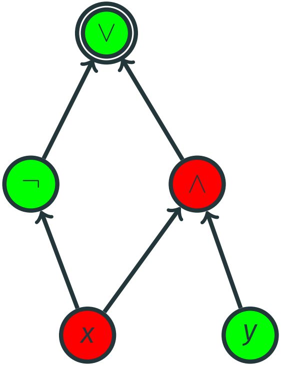
Circuits are just a way to represent Boolean formulas
while factoring common subexpressions (more concise)



- Directed acyclic graph of gates
- Output gate: 
- Variable gates: x
- Internal gates: $\vee$ $\wedge$ $\neg$
- Valuation: function from variables to $\{0, 1\}$
  Example: $\nu = \{x \mapsto 0,\ y \mapsto 1\}$...

Boolean circuit representations

Circuits are just a way to represent Boolean formulas while factoring common subexpressions (more concise)



- Directed acyclic graph of gates
- Output gate: 
- Variable gates: 

- Internal gates: 

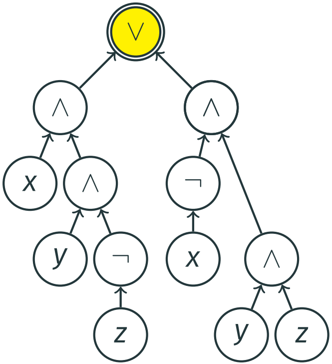
- Valuation: function from variables to $\{0, 1\}$
  Example: $\nu = \{x \mapsto 0,\ y \mapsto 1\}$... mapped to 1

Circuit restrictions

Tractable circuit class: **d-DNNF:**

- $\lor$ are all deterministic:

The inputs are mutually exclusive (= no valuation ν makes two inputs simultaneously evaluate to 1)



Circuit restrictions

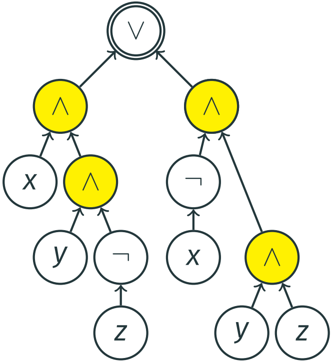
Tractable circuit class: **d-DNNF:**

- $\lor$ are all deterministic:

The inputs are mutually exclusive
(= no valuation ν makes two inputs simultaneously evaluate to 1)

- $\land$ are all decomposable:

The inputs are independent
(= no variable x has a path to two different inputs)



Circuit restrictions

Tractable circuit class: **d-DNNF:**

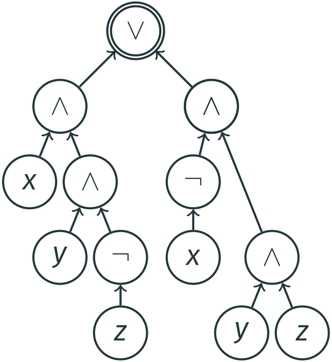
- $\vee$ are all deterministic:

The inputs are mutually exclusive
(= no valuation ν makes two inputs simultaneously evaluate to 1)

- $\wedge$ are all decomposable:

The inputs are independent
(= no variable x has a path to two different inputs)

$\rightarrow$ We can **compute** the probability of a d-DNNF with the **intensional rules**



Ordered Binary Decision Diagram (OBDD)

OBDD for a Boolean query Q on database I:
ordered decision diagram on the facts of I to decide whether Q holds

Ordered Binary Decision Diagram (OBDD)

OBDD for a Boolean query Q on database I:
ordered decision diagram on the facts of I to decide whether Q holds

$$Q : \pi_{\emptyset}(R \bowtie S \bowtie T)$$

R	
a	r_1
b	r_2
c	r_3

S		
a	a	s_1
b	v	s_2
b	w	s_3

T	
v	t_1
w	t_2
b	t_3

Ordered Binary Decision Diagram (OBDD)

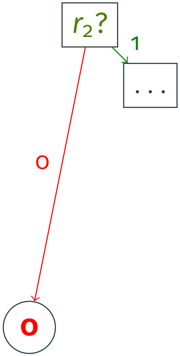
OBDD for a Boolean query Q on database I:
ordered decision diagram on the facts of I to decide whether Q holds

$$Q : \pi_{\emptyset}(R \bowtie S \bowtie T)$$

R	
a	r_1
b	r_2
c	r_3

S		
a	a	s_1
b	v	s_2
b	w	s_3

T	
v	t_1
w	t_2
b	t_3



Ordered Binary Decision Diagram (OBDD)

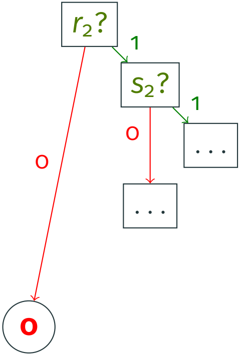
OBDD for a Boolean query Q on database I:
ordered decision diagram on the facts of I to decide whether Q holds

$$Q : \pi_\emptyset(R \bowtie S \bowtie T)$$

R	
a	r_1
b	r_2
c	r_3

S		
a	a	s_1
b	v	s_2
b	w	s_3

T	
v	t_1
w	t_2
b	t_3



Ordered Binary Decision Diagram (OBDD)

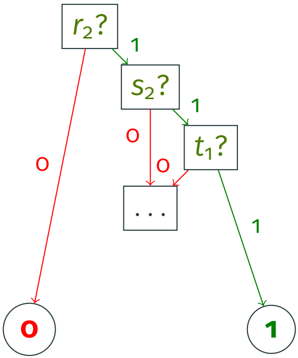
OBDD for a Boolean query Q on database I:
ordered decision diagram on the facts of I to decide whether Q holds

$$Q : \pi_{\emptyset}(R \bowtie S \bowtie T)$$

R	
a	r_1
b	r_2
c	r_3

S		
a	a	s_1
b	v	s_2
b	w	s_3

T	
v	t_1
w	t_2
b	t_3



Ordered Binary Decision Diagram (OBDD)

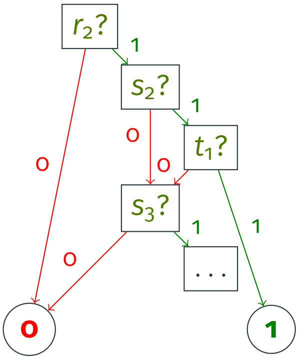
OBDD for a Boolean query Q on database I:
ordered decision diagram on the facts of I to decide whether Q holds

$$Q : \pi_\emptyset(R \bowtie S \bowtie T)$$

R	
a	r_1
b	r_2
c	r_3

S		
a	a	s_1
b	v	s_2
b	w	s_3

T	
v	t_1
w	t_2
b	t_3



Ordered Binary Decision Diagram (OBDD)

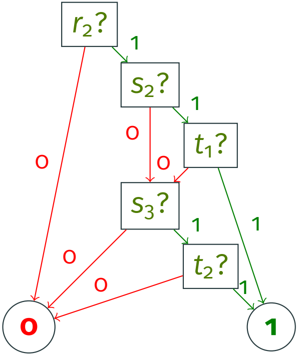
OBDD for a Boolean query Q on database I:
ordered decision diagram on the facts of I to decide whether Q holds

$$Q : \pi_\emptyset(R \bowtie S \bowtie T)$$

R	
a	r_1
b	r_2
c	r_3

S		
a	a	s_1
b	v	s_2
b	w	s_3

T	
v	t_1
w	t_2
b	t_3



Ordered Binary Decision Diagram (OBDD)

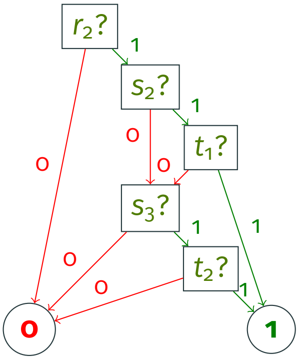
OBDD for a Boolean query Q on database I:
ordered decision diagram on the facts of I to decide whether Q holds

$$Q : \pi_\emptyset(R \bowtie S \bowtie T)$$

R		
a	r_1	
b	r_2	
c	r_3	

S		
a	a	s_1
b	v	s_2
b	w	s_3

T	
v	t_1
w	t_2
b	t_3



$\rightarrow$ We can compute the probability of an OBDD bottom-up

Approximation

- When it’s too hard to compute the exact probability, we can approximate it

Approximation

- When it's too hard to compute the exact probability, we can approximate it
- One possibility is to compute a lower bound and upper bound:
  - $\max(\mathrm{P}(\phi), \mathrm{P}(\psi)) \leq \mathrm{P}(\phi \vee \psi) \leq \min(\mathrm{P}(\phi) + \mathrm{P}(\psi), 1)$
  - $\max(0, \mathrm{P}(\phi) + \mathrm{P}(\psi) - 1) \leq \mathrm{P}(\phi \wedge \psi) \leq \min(\mathrm{P}(\phi), \mathrm{P}(\psi))$ (by duality)
  - $\mathrm{P}(\neg\phi) = 1 - \mathrm{P}(\phi)$ (reminder)

Approximation by sampling

Another possibility is to approximate via Monte-Carlo sampling:

- Pick a random valuation according to the variable probabilities:
  - $\rightarrow$ Set $x_1 = 0$ with probability on $\mathrm{P}(x_1 = 0)$ and $x_1 = 1$ otherwise
  - $\rightarrow$ Repeat for the other variables

Approximation by sampling

Another possibility is to approximate via Monte-Carlo sampling:

- Pick a random valuation according to the variable probabilities:
  - $\rightarrow$ Set $x_1 = 0$ with probability on $\mathrm{P}(x_1 = 0)$ and $x_1 = 1$ otherwise
  - $\rightarrow$ Repeat for the other variables
- Evaluate the lineage formula ϕ under this valuation

Approximation by sampling

Another possibility is to approximate via Monte-Carlo sampling:

- Pick a random valuation according to the variable probabilities:
  - $\rightarrow$ Set $x_1 = 0$ with probability on $\mathrm{P}(x_1 = 0)$ and $x_1 = 1$ otherwise
  - $\rightarrow$ Repeat for the other variables
- Evaluate the lineage formula ϕ under this valuation
- Approximate the probability of the formula ϕ as the proportion of times when it was true

Approximation by sampling

Another possibility is to approximate via Monte-Carlo sampling:

- Pick a random valuation according to the variable probabilities:
  - $\rightarrow$ Set $x_1 = 0$ with probability on $\mathrm{P}(x_1 = 0)$ and $x_1 = 1$ otherwise
  - $\rightarrow$ Repeat for the other variables
- Evaluate the lineage formula ϕ under this valuation
- Approximate the probability of the formula ϕ as the proportion of times when it was true
- Theoretical guarantees: on how many samples suffice so that, with high probability, the estimated probability is almost correct

Using external tools

- Specialized software to compute the probability of a formula: weighted model counters
- Examples (ongoing research):
  - c2d: `http://reasoning.cs.ucla.edu/c2d/download.php`
  - d4: `https://www.cril.univ-artois.fr/KC/d4.html`
  - dsharp: `https://bitbucket.org/haz/dsharp`

Table of contents

Summary

- We have seen probabilistic query evaluation on TID instances: compute the marginal probability of each query output tuple

Summary

- We have seen probabilistic query evaluation on TID instances: compute the marginal probability of each query output tuple
- We can enumerate naively all possible worlds, but it is inefficient

Summary

- We have seen probabilistic query evaluation on TID instances: compute the marginal probability of each query output tuple
- We can enumerate naively all possible worlds, but it is inefficient
- For some queries, we can do better:

Summary

- We have seen probabilistic query evaluation on TID instances: compute the marginal probability of each query output tuple
- We can enumerate naively all possible worlds, but it is inefficient
- For some queries, we can do better:
  - Extensional evaluation: find a safe plan so we can correctly compute all probabilities as the query is being evaluated

Summary

- We have seen probabilistic query evaluation on TID instances: compute the marginal probability of each query output tuple
- We can enumerate naively all possible worlds, but it is inefficient
- For some queries, we can do better:
  - Extensional evaluation: find a safe plan so we can correctly compute all probabilities as the query is being evaluated
  - Intensional evaluation:
    - compute the lineage of each result via pc-tables
    - compute the probability of each lineage expression

Acknowledgements

Partly inspired by slides by Silviu Maniu

http://silviu.maniu.info/teaching/m2_dk_udm_query_processing.pdf

Abiteboul, S., Hull, R., and Vianu, V. (1995).
Foundations of Databases.
Addison-Wesley.
http://webdam.inria.fr/Alice/pdfs/all.pdf.

Barbará, D., Garcia-Molina, H., and Porter, D. (1992).
The management of probabilistic data.
IEEE Transactions on Knowledge and Data Engineering, 4(5).
http://www.iai.uni-bonn.de/III/lehre/AG/IntelligenteDatenbanken/Seminar/SS05/Literatur/%5BBGP92%5DProbData_IEEE_TKDE.pdf.

Dalvi, N. N. and Suciu, D. (2007).
Efficient query evaluation on probabilistic databases.
VLDB Journal.
http://www.vldb.org/conf/2004/RS22P1.PDF.

Green, T. J. and Tannen, V. (2006).
Models for incomplete and probabilistic information.
IEEE Data Eng. Bull.
http://sites.computer.org/debull/A06mar/green.ps.

Huang, J., Antova, L., Koch, C., and Olteanu, D. (2009).
MayBMS: a probabilistic database management system.
In *SIGMOD.*
https://www.cs.ox.ac.uk/dan.olteanu/papers/hako-sigmod09.pdf.

Lakshmanan, L. V. S., Leone, N., Ross, R. B., and Subrahmanian, V. S. (1997).
ProbView: A flexible probabilistic database system.
ACM Transactions on Database Systems.
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.53.293&rep=rep1&type=pdf.

Ré, C. and Suciu, D. (2007).
Materialized views in probabilistic databases: for information exchange and query optimization.
In *VLDB.*
http://www.cs.stanford.edu/people/chrismre/papers/prob_materialized_views_TR.pdf.

Suciu, D., Olteanu, D., Ré, C., and Koch, C. (2011).
Probabilistic Databases.
Morgan & Claypool.
Unavailable online.