Efficient Query Computing for Uncertain Possibilistic Databases with Provenance

Angelos Vasilakopoulos
National Technical University of Athens

Verena Kantere
Cyprus University of Technology

Abstract

We propose an extension of possibilistic databases that also includes provenance. The introduction of provenance makes our model closed under selection with equalities, projection and join. In addition the computation of query computing with possibilities is *polynomial*, in contrast with current models that combine provenance with probabilities and have $\#P$ complexity.

1 Introduction

Modeling, representing and manipulating uncertain data has gained a lot of research attention. There has been a plethora of models that capture different kinds of uncertainty: i) when an attribute can take a value from a finite set of *alternatives* (model of or-sets), ii) when the existence of a whole tuple is not certain (model of ?-tuples). The last two kinds can be combined yielding the *x-tuple* and *x-relation* model $R^a_?$ [1, 8]. On top of uncertain models we can also put "confidence" values. For example in the model of ?-tuples if we attach on each tuple the probability of the event that this tuple is indeed present in our data we yield the model of probabilistic databases [5].

One of the key aspects of an uncertain database framework is how efficient it can compute queries. Consider a query Q and an uncertain database U. An uncertain database U represents a set of *possible worlds* $PW(U)$. One naive approach in order to compute Q over U would be to compute first all the possible worlds of $PW(U)$ and pose the query over each one of them. This approach is not efficient since computing the possible worlds of an uncertain database can be intractable in the size of the data. For example if we have an uncertain relation with n x-tuples and each one of which has m different alternatives then the possible worlds are m^n. In contrast we would like to be able to efficiently compute query Q posed directly on uncertain database U and the result $Q(U)$ to be a new uncertain database which can be represented in our model with the correct semantics, i.e., we have that $PW(Q(U)) = Q(PW(U))$ [9]. If this holds for an uncertainty model and a query language L we say that this model is *closed* under L.

Possibilistic databases extend the x-tuple model by attaching on each alternative value degrees from a possibility distribution. In [2] it was shown that possibilistic databases are not closed under: i) selection with a condition that involves different attributes, ii) projection that performs duplicate elimination in the tuples of the answer and iii) under the join operator. In addition existing database models with provenance that attach "belief values" by using probabilities have high complexity $\#P$ [5, 8, 11, 12]. We solve the first problem and for the second problem we offer a suitable alternative to probabilities by proposing a new model which extends possibilistic databases by adding provenance. The proposed model has the following benefits:

- *Closed* under: i) selection involving equalities even over different attributes, ii) projection even after duplicate elimination and iii) join. This property is a result of the introduction of provenance in the possibilistic model.
- The possibility values of each tuple alternative in the answer of a query involving the above three operators are computed in polynomial time.

Our main contribution is that we define operators for equality select, projection (wth duplicate elimination) and join that can be posed directly on a database expressed in our model of provenance and possibilities without the need to compute first all the possible worlds. The result of each operator is a new database of our model that has the correct semantics: its possible worlds are the same with the ones we would have if we first computed all the possible worlds and pose queries over them. Moreover our operators compute data and possibilities for each alternative of the result in polynomial time.

We think that the employment of possibilities instead of probabilities in our model offers more suitable mod-

eling of alternative belief values, due to the qualitative nature of possibilities. For example suppose that we want to model the fact that a witness Amy is uncertain of whether she saw a $Mazda$ or a $Toyota$ car but she believes that more likely it was a $Mazda$. These kinds of real-life situations are well-represented though possibilistic theory. In addition even when only probabilities of alternatives are available, there exists a way to "translate" probability values to possibilities such that the more probable events will also be more possible, as it is intuitively expected [6].

1.1 Related Work

The possibilistic model is not closed for SPJ queries because it is not powerful enough to pose logical constraints on the alternative values that tuples can take of the answer of a query (e.g., indicate that two alternatives of two different tuples cannot coexist in a possible world [2]). Recent work [3] efficiently computes SPJ queries over a limited possibilistic model (specifically where only one alternative has possibility 1 and all others have $1 - a$) and the answers of the queries include only tuples appearing in a complete possible world (a world with possibility 1). In contrast our approach returns all tuples appearing in any possible world and does not require the initial data to have this limitation in its possibilities.

Many models have been proposed that are able to handle uncertainty and keep track of the provenance of data which is usually modeled though semiring annotations on data [4, 7, 10]. Those models are closed under positive relational algebra but if probabilistic confidence values are added on each possible alternative tuple then the computation of the probabilities of the answer of a query that involves projection (with duplicate elimination) is intractable (specificaly #P) [5, 8, 11, 12]. The provenance used in Trio system [1] is one out of many kinds of provenance that semirings can model [7]. Our proposed model extends possibilistic databases by adding Trio's provenance used in the Uncertain and Lineage ULDB model [1] (where provenance is called "lineage"). The reason we choose this model is because it expresses tuple uncertainty and provenance tracking over the $R^a_?$ (or-set and ?-tuple) x-tuple model which is also used in possibilistic databases[1].

[1]Note that our proposed model that extends possibilistic databases with Trio's provenance can be equivalently regarded as an extension of the ULDB model with possibilistic confidence values on each alternative

2 Properties of the Proposed Model

In this section we illustrate the key aspects of possibilistic databases and of the provenance semiring of Trio. Due to space limitation we refer for more details to [1, 7]. We also investigate how we can combine uncertainty (x-tuples), provenance (Trio's lineage semiring) and possibilities. We begin with stating the basic properties of the Possibility Theory [3].

2.1 Possibility Theory

A possibility distribution is a function π from a domain X to the interval $[0, 1]$. Possibility $\pi(a)$ is a qualitative measure expressing the degree of "how possible" it is for the considered variable to take the value a. Each possibility distribution has a normalization condition posing the constraint that at least one of the values of X is completely possible, i.e., has possibility 1. We use a discrete domain of possible values and we denote with $\{a_1{:}\pi_1, \ldots, a_n{:}\pi_n\}$ the fact that for each $i = 1 \ldots n$ value a_i has possibility π_i. The axioms of possibility are the following: i) $\Pi(X) = 1$, ii) $\Pi(\emptyset) = 0$, iii) $\Pi(E_1 \cup E_2) = \max(\Pi(E_1), \Pi(E_2))$, iv) $\Pi(E_1 \cap E_2) \leq \min(\Pi(E_1), \Pi(E_2))$ and when E_1 and E_2 are not-interactive: $\Pi(E_1 \cap E_2) = \min(\Pi(E_1), \Pi(E_2))$. For the events E and $\bar{E}$ (opposite of E) the only valid relation is: $\max\{\Pi(E),\ \Pi(\bar{E})\} = 1$. Apart from possibility each event has a necessity measure N which is dual with Π and their relation is expressed through: $N(E) = 1 - \Pi(\bar{E})$.

2.2 The Proposed Model: Combining Uncertainty, Possibilities and Provenance

Possibility theory can be naturally adapted to the model of x-relations and the semiring of Trio [1]. In x-relations model we no longer have ordinary tuples. Instead we have x-tuples which include a bag of possible ordinary tuples, called *alternatives*. The semantics are the following: on each possible world at most one of the alternatives of an x-tuple can be true. If from an x-tuple we can select none of its alternatives then this is a maybe-xtuple annotated with symbol '?'. It is then straightforward that we can combine possibilistic theory and x-tuples in the following way: Suppose that we have an uncertain database which contains x-tuples. We attach on each alternative a possibility degree and on each x-tuple at least one alternative should be assigned with possibility 1 (the most possible one(s)). Furthermore for each x-tuple with a '?' symbol we attach to it a necessity degree less than 1 and to all other x-tuples necessity equal to 1 (since an alternative of each one of them is always possible). We do not have to explicitly attach a possibility degree on each

x-tuple since it is equal to the minimum possibility of each alternative. So we always begin with an uncertain database containing x-tuples with possibility degrees on each alternative and necessity degrees on each x-tuple.

Trio's provenance (called "lineage" in Trio) semiring works as follows: If we pose queries over initial data we want to keep track of the provenance of the answers, i.e., from which data the answers are derived from. In order to do this efficiently we attach a *unique* identifier i over each x-tuple. We also identify the alternatives of each x-tuple: In general the pair (i, j) identifies the j-th alternative of x-tuple i. If an alternative with data t is a result from two other alternatives t_1 and t_2 but can also be the result of our query combining two other alternatives t_3 and t_4 then we have for its lineage: $\lambda(t) = (id(t_1) \wedge id(t_2)) \vee (id(t_3) \wedge id(t_4))$. We note that lineage plays a double role: it relates answers of queries to the data they are coming from and also poses logical restrictions: an alternative can be true only in a possible world in which its lineage is true. We note that initial data have empty lineage. Initial data with empty lineage is defined as *base* data. Due to limited space we refer to [1] for more details about lineage and possible worlds. We borrow from the same work the general setting of our following running example.

2.3 Running Example

Consider x-relation $Saw(witness, car)$ having two x-tuples with two alternatives each. Suppose that witness Amy saw a car near a crime-scene but she was not sure if it was a $Mazda$ or a $Toyota$ car. Moreover she believed it was more possible that the car was a $Mazda$ and a little less possible that it was a $Toyota$. The first tuple has identifier 11 and the second 12. We separate different alternatives of a same x-tuple with $||$ symbol. After each alternative we attach its possibility and after each x-tuple its necessity measure, i.e., $<t'_1{:}a||t'_2{:}\text{b}>{:}c$ is an x-tuple with necessity c that has two alternatives: alternative with data t'_1 has possibility a and alternative t'_2 has possibility b. Suppose also that in x-relation $Drives(person, car)$ we encode uncertainty about who is driving a car of a specific brand. The uncertain database U with Trio's provenance of our running example is:

Saw(witness,car)=
{11<Amy, Mazda:1||Amy, Toyota:0.8>:1,
12<Billy, Mazda:0.4 || Billy, Lexus:1>:1}
Drives(person,car)=
{21<Hank, Mazda:0.6||Hank,Toyota,:1>:1}

There is a total of $2^3 = 8$ possible worlds. Suppose that we pose query Q_2 which is a projection of attribute $person$ on the result of query Q_1 which is the join of Saw and $Drives$ over common attribute car, i.e.: $Q_1 = Saw \bowtie_{car=car} Drives$ and $Q_2 = \pi_{person}(Q_1(U))$. Only five of the possible worlds include answers over Q_1 (and Q_2). Those worlds are:

W1: Saw={11,1<Amy, Mazda:1>:0.2,
12,1<Billy, Mazda:0.4>:0}
Drives={21,1<Hank, Mazda:0.6>:0}
$\Pi(W1) = 0.4$, $N(W1) = 0$
W2: Saw={11,1<Amy, Mazda:1>:0.2,
12,2<Billy, Lexus:1>:0.6}
Drives={21,1<Hank, Mazda:0.6>:0}
$\Pi(W2) = 0.6$, $N(W2) = 0$
W3: Saw={11,2<Amy, Toyota:0.8>:0,
12,1<Billy, Mazda:0.4>:0}
Drives={21,1<Hank, Mazda:0.6>:0}
$\Pi(W3) = 0.4$, $N(W3) = 0$
W4: Saw={11,2<Amy, Toyota:0.8>:0,
12,1<Billy, Mazda:0.4>:0}
Drives={21,2<Hank, Toyota:1>:0.4}
$\Pi(W4) = 0.4$, $N(W4) = 0$
W5: Saw={11,2<Amy, Toyota:0.8>:0,
12,2<Billy, Lexus:1>:0.6}
Drives={21,2<Hank, Toyota:1>:0.4}
$\Pi(W5) = 0.8$, $N(W5) = 0$

For example the possibility of PW_1 is equal to the minimum of the possibilities of its alternatives, so with $\min\{1,\ 0.4,\ 0,6\} = 0.4$. Its necessity is equal to 1 minus the maximum possibility from the possibilities of alternatives which do *not* belong to this world have: $1-\max\{0.8,\ 1,\ 1\} = 1-1 = 0$. The necessity of x-tuple $(11, 1)$ is equal to 1 minus the maximum possibility of the other alternatives (in our case only alternative $11, 2$) of initial x-tuple 11, i.e., equal to $1 - \max\{0.8\} = 0.2$.

For the answers of queries we have similar semantics with the ones defined for probabilities in [5]: The answer of a query Q is a set of alternatives and their possibilities. Intuitively for the answer of Q_1 we should have:
$\mathbf{Q_1}(\mathbf{U})$={31<Amy, Mazda, Hank:0.6>:0,
32<Billy, Mazda,Hank:0.4>:0
33<Amy, Toyota,Hank:0.8>:0}
For example alternative $(Amy, Mazda, Hank)$ appears in W_1, a world with possibility 0.4 and in W_2 with possibility 0.6 (while both necessities are 0 - note that only a world whose all tuples have possibility 1 has necessity greater than 0). As a result in the answer of query Q_1 we want to have a tuple with data $(Amy, Mazda, Hank)$ with possibility the union of the events that this tuple appears in W_1 or in W_2. So with the maximum of the possibilities of 0.4 and 0.6. According to Trio's semiring provenance we attach the following lineage on each alternative:
$\lambda(31) = (11, 1) \wedge (21, 1)$
$\lambda(32) = (12, 1) \wedge (21, 1)$

$\lambda(33) = (11,2) \wedge (21,2)$.
Similarly in the answer of query Q_2 we expect:
$\mathbf{Q_2}(\mathbf{Q_1}(\mathbf{U}))$={41<Hank:0.8>:0}
$\lambda(41) = \{((11,1) \wedge (21,1)) \vee ((12,1) \wedge (21,1)) \vee ((11,2) \wedge (21,2))\}$

We would like to be able to directly compute those answers of Q_1 and Q_2 without having to compute all (exponentially many) possible worlds. As we already mentioned, existing work about possibilistic theory, join or projection with duplicate elimination was not possible due to the fact that possibistic sets were not powerful enough to express the disjunction of two different tuples occuring in the answer [2]. For example possibility theory could not model the fact that, e.g., tuples 31 and 33 in the answer of Q_1 could not coexist. Provenance poses additional logical restrictions to where an alternative can exist, thus overcoming this obstacle.

On the other hand until now provenance has only been combined with probabilistic theory and not with possibilistic. But probabilities have high complexity: for example if we want to compute the probability of alternative 41 $Hank$ we must compute the probability of $\{((11,1) \wedge (21,1)) \vee (12,1) \wedge (21,1) \vee ((11,2) \wedge (21,2))\}$. In general computing the probability of a DNF boolean formula is #-P complete [1, 5, 11]. In contrast in our model which uses possibilities we can compute answers of selection with equality, projection and join in polynomial time. Note in particular that the possibility of the union of two events is always equal to the maximum of their possibilities. We use provenance only to restrict data. The computation of possibilities and necessities is not based on provenance; instead, they are computed directly from initial data. Provenance (which includes only possibilities of alternatives and not necessities of x-tuples) is inadequate of computing x-tuple necessities.

3 The Operators

In this section we give the definitions of selection, projection and join operators. These definitions enable us to directly compute the answers of SPJ queries posed over an uncertain database of our model with x-tuples and possibilities without having to compute first its possible worlds. In addition the computation of the possibilities of the answers is polynomial.

Let r be an uncertain relation of our model, A an attribute and $(A = q)$ a logical selection condition where q can be another attribute or a constant. With $alt(t)$ we denote the alternatives of x-tuple t:

Selection
select$(r, A = q) = \{< restict(alt(t), A = q) >: N'$
such that $t{:}N \in r$ and where:
N'=min$\{1 - \max_{t'_i \in alt(t) \wedge t'_i \not\models (A=q)} \{\Pi(t'_i)\}, N(t)\}$ and:
$restrict(alt(t), A = q) =$
$\{t'{:}\Pi,\ \lambda(t')$ such that: $t' \in alt(t)$, where $t \in r$, and $t' \models (A = q)$ and $\Pi = \Pi(t')$ and $\lambda(t') = Id(t')\}$.

We keep in the select result only the alternatives that satisfy our select condition, with the same possibility that they had in our initial database. We set as their lineage, the lineage pointing to the identifiers of the initial alternatives. As for necessity of each resulting x-tuple, it is the minimum of: i) the necessity of the original x-tuple they belonged to, ii) the initial necessity of the original alternatives and iii) of 1 minus the maximum possibility that an alternative of the same original x-tuple that does *not* satisfy our selection condition has. The proof that our system is closed under selection with equality conditions uses a combination of the closure of Trio system [1] and the closure of selection on Possibilistic databases [2].

Projection
project$(r, X) = \{< t'.X : \Pi' >: N'$
such that: $t \in r$ and $t' \in alt(t)$ and $N' = \max_{A_i}\{N_i\}$
where: $A_i = \{N_i \mid t_i{:}N_i \in r$ and $\exists t'_i \in alt(t_i)$ with $t'_i.X = t'.X\}$
and $\Pi' = \max_{B_i}\{\Pi_i\}$ where:
$B_i = \{\Pi_i \mid t'_i{:}\Pi_i$ where $t'_i \in alt(t_i)$ and $t_i \in r$ and $t'_i.X = t.X\}$.
We also set: $\lambda(t'.X) = \vee_{C_i} id(t'_i)$ where $C_i = \{id(t'_i) \mid t'_i \in alt(t_i)$ where $t'_i \in alt(t_i)$ and $t_i \in r$ and $t'_i.X = t.X\}$.

We project the set of attributes X from every alternative and we perform alternative duplicate elimination. Thus the necessity of each resulting x-tuple is equal to the maximum necessity of each original x-tuple that includes an alternative that has the same projected value as the one alternative of our resulting x-tuple has. The same holds for the new possibility as well. Finally we set as lineage the disjunction of alternatives that give the same projected value. The proof that our system is closed under projection is easy. Let us just mention that the use of lineage allows duplicate elimination without losing the correct possible worlds. In addition for the possibilities we can easily use the maximum for the union of two alternatives with same data when we perform duplicate elimination.

Join
join$(r_1, r_2, A = B) = \{restrict(alt(t_1) \oplus alt(t_2), A = B)$
such that: $t_1 : N_1 \in r_1$ and $t_2 : N_2 \in r_2$ where:
$restrict(alt(t_1) \oplus alt(t_2), A = B) =$
$< t'_1 \oplus t'_2 : \Pi', \lambda'(t'_1 \oplus t'_2) >: N'$ such that: $t_1 \in r_1$ and $t'_1 \in alt(t_1)$ and $t_2 \in r_2$ and $t'_2 \in alt(t_2)$ and

$t'_1 \oplus t'_2 \models (A = B)$ and $\Pi' = \min\{\Pi(t'_1), \Pi(t'_2)\}$ and $N' = min\{1 - \max_{t1''_i \in alt(t_1)/t'_1}\{\Pi(t''_{i1})\}$, $1 - \max_{t''_{i2} \in alt(t_2)/t'_2}\{\Pi(t''_{i2})\}, N_1, N_2\}$ and $\lambda'(t'_1 \oplus t'_2) = id(t'_1) \wedge id(t'_2)$.

We note that $\oplus$ denotes the concatenation of tuples. Also the above definition can be easily adopted to the case where the join condition involves a conjunction of attribute equalities. We restrict in the join results only the tuples that satisfy the join condition and we perform duplicate elimination on alternatives. The new possibility of each alternative of the result is equal to the minimum of the possibilities of the original alternatives that contributed to its value. The necessity of each resulting x-tuple is the minimum of: i) 1 minus the maximum possibility of each other alternative that exists in the original contributing x-tuples and ii) the necessities of the original contributing x-tuples. We also set as lineage the conjunction of the initial contributing alternatives.

We now show that our system with Trio's lineage, x-tuples, possibilities and necessities is closed for the join operation. Moreover it follows from the definitions of our operators that their complexity is polynomial to the size of the data (alternatives) of our initial uncertain database.

Theorem 1: The possibilistic database model with provenance is *closed* under the join operation.

Proof: We want to prove that $PW(\text{join}(r_1, r_2, A = B)) = \text{join}(PW(r_1, r_2), A = B)$. Suppose that r_1 and r_2 both contain a single x-tuple. So suppose that r_1 has x-tuple t_1:N_1 and r_2 has x-tuple t_2:N_2. Note that we have no loss of generality: The possibilities that alternatives have in every x-tuple in base relations form a possibilistic distribution. As a result in every base x-tuple always exists (at least one) alternative with possibility equal to 1. Suppose now that t' is an alternative in the result of a join query, resulting from two alternatives t'_1 of t_1 and t'_2 of t_2. The possibility of t' in the join result according to our definition is equal to the minimum of possibilities of t'_1 and t'_2. If r_1 and r_2 had more x-tuples then t'_1 and t'_2 would exist in more possible worlds resulting from the choices of alternatives from the other x-tuples. According to our semantics in the join result t' should have the possibility of the union of all its occurrences in every possible world. From the definition of possibility union this would be equal to the maximum of the possibilities of all possible worlds in which t'_1 and t'_2 both exist. But the maximum possibility exists in the possible world where t'_1 and t'_2 are selected from t_1 and t_2 and for all the other x-tuples the alternative with possibility equal to 1 has been selected. Hence the possibility of this possible world is equal to $\min\{\Pi(t'_1), \Pi(t'_2), 1, \ldots, 1\} = \min\{\Pi(t'_1), \Pi(t'_2)\}$, so equal to the case where r_1 and r_2 had only one x-tuple.

So suppose that r_1 has x-tuple t_1:N_1 and r_2 has x-tuple t_2:N_2. We remind that with $alt(t_1)$ we denote the set of alternatives that exist in x-tuple t_1 (respectively for t_2). We first show that $PW(\text{join}(r_1, r_2, A = B)) \subset \text{join}(PW(r_1, r_2), A = B)$. Let W_k be a possible world of $PW(\text{join}(r_1, r_2, A = B))$ and π_k its possibility. We want to show that W_k is also a world of $\text{join}(PW(r_1, r_2), A = B)$ with the same possibility. We consider two cases:

• $W_k \neq \emptyset$: We denote with t' an arbitrary alternative in W_k. From the definition of join the data of t' comes from the concatenation of two alternatives t'_1 of x-tuple t_1 and t'_2 of t_2 that satisfy the join condition (i.e., $t' = t_1 \oplus t_2$). These two alternatives also exist in a $PW(r_1, r_2)$, let us denote it with W'_k [2]. On the other hand if there exists a combination of two alternatives of x-tuples of r_1 and r_2 in a possible world W'_k of $PW(r_1, r_2)$ that satisfy the join condition then the join answer resulting from them appears in $\text{join}(PW(r_1, r_2), A = B)$. So there exists a possible world exactly equal to W_k as concerns data (and with lineage pointing to the same base data) in $PW(\text{join}(r_1, r_2, A = B))$. Note that as concerns data (and lineage) a similar result was also proven in [1].

Now for the possibilities of W_k and W'_k: Again let t' be an arbitrary alternative in $W_k \in PW(\text{join}(r_1, r_2, A = B))$. As we just showed, a tuple with same data also appears in $W'_k \in \text{join}(PW(r_1, r_2), A = B)$. Its possibility in W'_k is associated with the possibilities of $t'_1 \in PW(r_1)$ and $t'_2 \in PW(r_2)$. Specifically it is equal to the minimum of possibilities of t'_1 and t'_2 since a possible world that produces t' must include them both (conjunction of possibilities). The same choices have been made in W_k to derive t' and according to our join definition the possibility degrees of the join query is equal to the minimum of possibilities of alternatives that produce the result. So the possibilities are the same in W'_k and W_k.

• W_k=$\emptyset$: We can have two subcases: either $\text{join}(r_1, r_2, A = B)$ is empty (t' does not exist) or the necessity N' of t' is less than 1. If it is empty then $(PW(r_1, r_2), A = B)$ is also empty and the possibility of the empty world is in both cases equal to the maximum possibility of any possible world, i.e., : $\min\{\max_{t''_{i1} \in alt(t_1)}\{\Pi(t''_{i1})\}, \max_{t''_{i2} \in alt(t_2)}\{\Pi(t''_{i2})\}\}$.

If $\text{join}(r_1, r_2, A = B)$ is not empty then the necessity degree N' of t' is less than 1 and the empty world W_k has possibility $1 - N'$. The possibility of W_k must correspond to a world of $PW(r_1, r_2)$ with the most

[2] Unless they have extraneous lineage, but in that case they also not exist in $PW(\text{join}(r_1, r_2, A = B)$, we refer to [1] for more details).

possible choices of alternatives t'_1 of r_1 and t'_2 of r_2 that do not satisfy the join condition, i.e., with possibility: $\max\{\max_{t1''_i \in alt(t_1)/t'_1}\{\Pi(t''_{i1})\}, \max_{t''_{i2} \in alt(t_2)/t'_2}\{\Pi(t''_{i2})\}, 1-N_1, 1-N_2\}$. From our definition of join we see than indeed this would be the possibility of the empty world W_k. Using a similar logic it is now easy to also prove that join$(PW(r_1, r_2), A = B) \subset PW($join$(r_1, r_2, A = B))$.

3.1 Examples of Operators

We present in this subsection that if our join and project operators are posed over the initial data of our running example, their result directly computes the results $Q_1(U)$ and $Q_2(Q_1(U))$ with the correct data and provenance that we expected and presented in subsection 2.3.

Join

We illustrate the use of our operators join and project though our running example. Query Q_2 is a projection of attribute $person$ on the result of query Q_1 which is the join of Saw and $Drives$ over common attribute car. We begin with join query Q_1. According to our definition we have the query $join(Saw, Drives, car = car)$. In our result we naturally restrict the combinations of all possible alternatives of the two relations Saw and $Drives$ to the ones that satisfy the condition $car = car$. In our example there exist three such combinations. For the first one we have: Alternatives $11, 1$ and $21, 1$ from x-tuples 11, with necessity $N_{11} = 1$ and 21 with necessity $N_{21} = 1$, yield alternative $(Amy, Mazda, Hank)$. According to our join definition its possibility is $\Pi' = \min\{\Pi(11,1), \Pi(21,1)\} = \min\{1, 0.6\} = 0.6$. Respectively its necessity is $N' = \min\{1 - \max\{\Pi(11,2)\}, 1 - \max\{\Pi(21,2)\}, N_{11}, N_{21}\} = \min\{1 - \max\{0.8\}, 1 - \max\{1\}, 1, 1\} = \min\{1 - 0.8, 1 - 1, 1, 1\} = \min\{0.2, 0, 1, 1\} = 0$. The lineage of $(Amy, Mazda, Hank)$ is equal to the conjunction of the identifiers of the alternatives that produce it, i.e., with $(11,1) \wedge (21,1)$. For the other two combinations of alternatives that satisfy our join condition, the procedure is similar. In order to succinctly denote lineage we attach a new fresh identifier to each tuple-alternative of the answer. The final result is:

$\mathbf{Q_1}(\mathbf{U})$={31<Amy, Mazda, Hank:0.6>:0,
32<Billy, Mazda,Hank:0.4>:0
33<Amy, Toyota,Hank:0.8>:0}
$\lambda(31) = (11,1) \wedge (21,1)$
$\lambda(32) = (12,1) \wedge (21,1)$
$\lambda(33) = (11,2) \wedge (21,2)$

Projection

We continue with query Q_2 which is a projection of attribute $person$ from the result $Q_1(U)$, i.e., $project(Q_1(U), person)$. Our result has a singe x-tuple with one alternative $Hank$. Its necessity, according to our definition is equal to the maximum of the necessities of the x-tuples that produce the same result $Hank$ (we perform duplicate elimination). In our case all x-tules 31, 32 and 33 produce $Hank$, so the set A_i includes the necessities of all of them. So the necessity of $Hank$ in the result is: $N' = \max\{N(31), N(32), N(33)\} = \max\{0\} = 0$. The possibility of $Hank$ is equal to the maximum of the possibilities of all the alternatives that give result $Hank$. In our case the set B_i of such alternatives includes $(31,1)$, $(32,1)$ and $(33,1)$. So the possibility of $Hank$ in the result is: $\Pi' = \max\{\Pi(31,1), \Pi(32,1), \Pi(33,1)\} = \max\{0.4, 0.8, 0.6\} = 0.8$. The lineage of $Hank$ is equal to the disjunction of the identifiers of the alternatives that have $Hank$ in the data of the projection result. In order to succinctly denote lineage we attach new identifier 41 to x-tuple $Hank$ of the answer. So we have: $\lambda(41) = \{(31,1) \vee (32,1) \vee (33,1)\}$. As noted in [1] we can use the lineage information of the initial $Q_1(U)$ and expand with polynomial complexity lineage back to base data. Hence we can replace, e.g., $(31,1)$ with its lineage in $Q_1(U)$, which is: $\{(11,1) \wedge (21,1)\}$. The final result is:

$\mathbf{Q_2}(\mathbf{Q_1}(\mathbf{U}))$={41<Hank:0.8>:0}
$\lambda(41) = \{((11,1) \wedge (21,1)) \vee ((12,1) \wedge (21,1)) \vee ((11,2) \wedge (21,2))\}$

References

[1] BENJELLOUN, O., SARMA, A. D., HALEVY, A. Y., THEOBALD, M., AND WIDOM, J. Databases with uncertainty and lineage. *VLDB J. 17*, 2 (2008), 243–264.

[2] BOSC, P., AND PIVERT, O. About projection-selection-join queries addressed to possibilistic relational databases. *IEEE T. Fuzzy Systems 13*, 1 (2005), 124–139.

[3] BOSC, P., PIVERT, O., AND PRADE, H. A model based on possibilistic certainty levels for incomplete databases. In *SUM* (2009), pp. 80–94.

[4] BUNEMAN, P., AND TAN, W. C. Provenance in databases. In *SIGMOD Conference* (2007), pp. 1171–1173.

[5] DALVI, N. N., AND SUCIU, D. Efficient query evaluation on probabilistic databases. In *VLDB* (2004), pp. 864–875.

[6] DUBOIS, D., FOULLOY, L., MAURIS, G., AND PRADE, H. Probability-possibility transformations, triangular fuzzy sets, and probabilistic inequalities. *Reliable Computing 10*, 4 (2004), 273–297.

[7] GREEN, T. J., KARVOUNARAKIS, G., AND TANNEN, V. Provenance semirings. In *PODS* (2007), pp. 31–40.

[8] GREEN, T. J., AND TANNEN, V. Models for incomplete and probabilistic information. In *EDBT Workshops* (2006), pp. 278–296.

[9] IMIELINSKI, T., AND LIPSKI, W. Incomplete information in relational databases. *J. ACM 31*, 4 (1984), 761–791.

[10] KARVOUNARAKIS, G., IVES, Z. G., AND TANNEN, V. Querying data provenance. In *SIGMOD Conference* (2010), pp. 951–962.

[11] ROY, S., PERDUCA, V., AND TANNEN, V. Faster query answering in probabilistic databases using read-once functions. In *ICDT* (2011), pp. 232–243.

[12] SARMA, A. D., THEOBALD, M., AND WIDOM, J. Exploiting lineage for confidence computation in uncertain and probabilistic databases. In *ICDE* (2008), pp. 1023–1032.