Information Integration with Uncertainty: Performance

Inder K. Dumpa   Raja S. Kota   Fereidoon Sadri
Department of Computer Science
University of North Carolina at Greensboro
Greensboro, NC, USA
ikdumpa@uncg.edu, raja_k999@yahoo.com, f_sadri@uncg.edu

***Abstract*—Information integration and modeling and management of uncertain information have been active research areas for decades, with both areas receiving significant renewed interest in recent years. Research on information integration with *uncertainty*, on the other hand, is quite recent. In this paper we concentrate on recent works on uncertain-data integration. We present experimental results on the efficiency of recent algorithms for information integration from sources that contain uncertain data. Our experiments show the algorithms to be efficient, demonstrating a near linear performance in the total size of the uncertain data to be integrated.**

Keywords: Information integration; uncertain data; possible worlds; integration performance.

I. INTRODUCTION

The importance of information integration *with uncertainty*, has been realized recently [1]-[4]. It has been observed that [2]: *"While in traditional database management managing uncertainty and lineage seems like a nice feature, in data integration it becomes a necessity."* Research on information integration with uncertainty, is quite recent [5]-[11]. Researchers have concentrated on two main aspects of information integration with uncertainty. The first category considers integration of definite data with uncertain schema mappings [6], [7]. The second category considers integration of uncertain data [5], [8], [9], [11]. Our work in [11] falls in the second category. We presented algorithms for the integration of uncertain data, and justified the correctness of the algorithms by showing they coincided with the integration formalism presented in the foundational work [5]. In this paper, we report the implementation and experimental results of these algorithms. Note that information integration has many dimensions and involves a number of important tasks such as data cleaning, data linkage, schema mapping, data standardization, query translation, and query optimization. We concentrate only on the issue of information integration from sources that contain uncertain data in this paper.

This paper is organized as follows. In Section II, we present an introduction to integration of information from sources with uncertain data, and briefly discuss the uncertain data integration algorithms of [11]. Our implementation is presented in Section III, and experimental results in Section IV. Our experiments show the algorithms to be efficient, demonstrating a near linear performance in the total size of the uncertain data to be integrated. Conclusion and future work are presented in Section V.

II. PRELIMINARIES

In this section we will review some of the recent works regarding issues and algorithms for the integration of uncertain data.

A. Information Integration with Uncertainty: Foundations

Foundations of information integration with uncertainty have been discussed in [5], [11]. We will present a brief summary here. We begin with an example from [11].

Example 1: John and Jane are talking about fellow student Bob. John says "I am taking CS100 and CS101, and Bob is in one of them, but not in both." Jane says "I am taking CS101 and CS102 and Bob is in one of them, but not in both."

Intuitively, if we integrate the information from these two sources (John and Jane), we should infer that Bob is either taking CS101, or he is taking both CS100 and CS102.

The model used in [5], [11] for the representation of uncertain information is the well-known *possible-worlds* model [12]. In Example 1, the information presented by the two sources (John and Jane) is represented by the possible worlds shown in Figures 1 and 2. The possible worlds of the result of integration is shown in Figure 3.

student	course
Bob	CS100

D1

student	course
Bob	CS101

D2

Figure 1: Possible Worlds of source S1

student	course
Bob	CS101

D3

student	course
Bob	CS102

D4

Figure 2: Possible Worlds of source S2

student	course
Bob	CS101

student	course
Bob	CS100
Bob	CS102

Figure 3: Possible Worlds of the Integration for Example 1

Here, we will summarize the integration approach from [11] which uses a simple logic-based technique. It has been

 ISBN: 978-1-61208-334-6

shown to be equivalent to the integration formalism of [5] which is based on the concept of superset-containment. Interested readers are referred to [11] for details.

First, we should point out that the pure possible world model is not adequate for uncertain-data integration applications. We need additional information, namely, the set of all tuples. The following example demonstrates the possible-worlds with tuple sets model.

Example 2: Andy and Jane are talking about fellow student Bob. Andy says "I am taking CS100, CS101, and CS102 and Bob is in either CS100 or CS101 but not in both." Jane says "I am taking CS101 and CS102 and Bob is in one of them, but not in both."

Intuitively, if we integrate the information from these two sources, we should infer that Bob is taking CS101. The second possibility from Example 1 is not valid anymore since Andy's statement rules out the possibility that Bob is taking 102.

To justify this answer, we observe that pairwise combinaiton of possible worlds from the two sources result in the four possible worlds of Figure 4. But only the second possible world is a valid combination, and the other three are not valid. The first world is not valid since Andy states that he is taking CS100, CS101, and CS102 and Bob is taking 100 of 101 but not both. So Bob can not be in both 100 and 101. The third and fourth worlds are not valid due to Andy's statement too. He is taking 102 (among other courses) and states that Bob is taking 100 or 101. Hence Bob can not be in 102. Note that the last world is also not valid due to Jane's statements. She says that she is in 101 and 102, and Bob is in one of them, but not both. The only valid combination is the second world: Bob must be taking CS101.

student	course
Bob	CS100
Bob	CS101

student	course
Bob	CS101

student	course
Bob	CS100
Bob	CS102

student	course
Bob	CS101
Bob	CS102

Figure 4: Pairwise combination of possible worlds from the two sources

However, the possible-worlds representations of these sources (Andy and Jane) are exactly the same as those of Example 1 (Figures 1 and 2). Only when we add the tuple-set to possible worlds of Andy, namely {(Bob, CS100), (Bob, CS101), (Bob, CS102)}, It becomes explicit that Andy's statement eliminates the possibility that Bob is taking CS102.

Hence, we will use the following definition from [5] for uncertain databases that adds tuple sets to the possible-worlds model. Note that to simplify presentation, it is assumed that possible worlds are sets of tuples in a single relation. We adopt the same convention throughout this paper.

Definition 1: (UNCERTAIN DATABASE). An *uncertain database* U consists of a finite set of tuples $T(U)$ and a nonempty set of possible worlds $PW(U) = \{D_1, \ldots, D_m\}$, where each $D_i \subseteq T(U)$ is a certain database.

B. Integration Using Logical Representation

The following definitions and results are from [11].

Given an uncertain database U, we assign a propositional variable x_i to each tuple $t_i \in T(U)$. We define the formula f_j corresponding to a possible world D_j, and the formula f corresponding to the uncertain database U as follows:

Definition 2: (LOGICAL REPRESENTATION OF AN UNCERTAIN DATABASE). Let D_j be a database in the possible worlds of uncertain Database U. Construct a formula as the conjunction of all variables x_i where the corresponding tuple t_i is in D_j, and the conjunction of $\neg x_i$ where the corresponding tuple t_i is not in D_j. That is,

$$f_j = \bigwedge_{t_i \in D_j} x_i \bigwedge_{t_i \notin D_j} \neg x_i \tag{1}$$

The *formula corresponding to the uncertain database* U is the disjunction of the formulas corresponding to the possible worlds of U. That is,

$$f = \bigvee_{D_j \in PW(U)} f_j \tag{2}$$

Now we can integrate uncertain databases using their logical representations as follows:

Let $S_1, \ldots, S_n$ be sources containing (uncertain) databases $U_1, \ldots, U_n$. Let the propositional formulas corresponding to $U_1, \ldots, U_n$ be $f_1, \ldots, f_n$. We obtain the formula f corresponding to the uncertain database resulting from integrating $U_1, \ldots, U_n$ by conjuncting the formulas of the databases: $f = f_1 \wedge \cdots \wedge f_n$.

Example 3: (INTEGRATION USING LOGICAL REPRESENTATION) Consider Example 1. The uncertain database corresponding to John's statement is represented by $(x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2)$, where x_1, and x_2 correspond to the tuples (Bob, CS100) and (Bob, CS101), respectively. The uncertain database corresponding to Jane's statement is represented by $(x_2 \wedge \neg x_3) \vee (\neg x_2 \wedge x_3)$, where x_2 is as above and x_3 corresponds to the tuple (Bob, CS102). The integration in this case is obtained as

$$\begin{aligned} &((x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2)) \wedge ((x_2 \wedge \neg x_3) \vee (\neg x_2 \wedge x_3)) \\ &= (x_1 \wedge \neg x_2 \wedge x_3) \vee (\neg x_1 \wedge x_2 \wedge \neg x_3) \end{aligned}$$

which corresponds to the possible worlds of Figure 3. The result is consistent with our intuition: Based on statements by John and Jan, Bob is taking either CS101 or both CS100 and CS102.

Now consider Example 2. The uncertain database corresponding to Andy's statement is represented by $(x_1 \wedge \neg x_2 \wedge \neg x_3) \vee (\neg x_1 \wedge x_2 \wedge \neg x_3)$, where x_1, x_2, and x_3 represent (Bob, CS100), (Bob, CS101), and (Bob, CS102), respectively. The uncertain database corresponding to Jane's statement is represented by $(x_2 \wedge \neg x_3) \vee (\neg x_2 \wedge x_3)$. The integration in this case is obtained as

$$\begin{aligned} &((x_1 \wedge \neg x_2 \wedge \neg x_3) \vee (\neg x_1 \wedge x_2 \wedge \neg x_3)) \wedge \\ &\qquad ((x_2 \wedge \neg x_3) \vee (\neg x_2 \wedge x_3)) \\ &= (\neg x_1 \wedge x_2 \wedge \neg x_3) \end{aligned}$$

corresponding to the (in this case, definite) relation consisting only of the tuple (Bob, CS101) (Figure 5). Again, this result is consistent with our intuition: Based on statements by Andy and Jane, Bob is taking CS101.

student	course
Bob	CS101

Figure 5: Possible Worlds of the Integration for Example 2

C. An Alternative View of Integration

Let $S_1, \ldots, S_n$ be sources containing (uncertain) databases $U_1, \ldots, U_n$. Let $PW(U_i)$ represent the set of possible worlds of uncertain database U_i, and T_i represent the tuple set of U_i. We can regard the integration of information from these sources as follows:

Definition 3: (COMPATIBLE SET OF POSSIBLE-WORLDS RELATIONS). Consider a set of n relations $\{w_1, \ldots, w_n\}$ where each w_i is a relation in the set of possible worlds of U_i, that is, $w_i \in PW(U_i)$, $i = 1, \ldots, n$. If there is a tuple t in a relation w_i, that it is also in $T_j - w_j$ for some other possible-world relation w_j, we say the set of possible-world relations $\{w_1, \ldots, w_n\}$ is *not compatible*. Otherwise, $\{w_1, \ldots, w_n\}$ is *compatible*.

Note that $t \in T_j - w_j$ means that according to source S_j, the tuple t can not exist (is ruled out) in w_j. Hence, if a set of possible world-relations is not compatible, they can not be integrated. A compatible set of possible-world relations $\{w_1, \ldots, w_n\}$ can be integrated, and the resulting relation contains all the tuples in the relations, that is, the result of integrating $w_1, \ldots, w_n$ is $w = \cup_{i=1}^{n} w_i$.

Hence, to integrate sources $S_1, \ldots, S_n$, we can compute the possible-worlds relations of the integration by

1) forming all possible combinations $\{w_1, \ldots, w_n\}$, $w_i \in PW(U_i)$,
2) determining compatible sets, and
3) obtaining the union of the relations in the compatible set.

This alternative characterization of integration results in a simpler integration algorithm. We use the logical formulation only to determine compatible sets of possible worlds, and then we obtain the result by calculating the union of the possible worlds in each compatible set. We have used this characterization to design our integration algorithm (Section III).

Example 4: (ALTERNATIVE VIEW OF INTEGRATION) Consider Example 1. The possible-worlds relations of the uncertain database corresponding to John's statement were shown in Figure 1, and the possible-worlds relations of the uncertain database corresponding to Jane's statement were shown in Figure 2. In this case, the compatible sets of possible worlds are $\{D_1, D_4\}$ and $\{D_2, D_3\}$. We can conveniently represent the compatibility of possible-worlds relations for two sources by a bi-partite graph, such as Figure 6. The possible-worlds of the result of integration is shown in Figure 3.

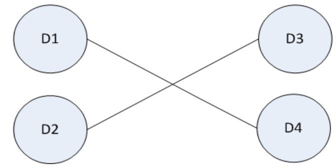


Figure 6: Compatibility graph of Example 1.

III. IMPLEMENTATION

We implemented the information integration approach of Section II-C, and ran a large number of experiments to assess the performance of the implementation. In this section we present the implementation details. Experimental results are presented and discussed in the next section.

Our implementation consists of several modules implemented in Java.

- The `GeneratePossibleWorlds` module is a utility module used to randomly generate possible world relations for the information sources. The user can specify the following parameters:
  - Number of information sources.
  - Number of possible worlds for each source.
  - Number of tuples for each possible world.
  - Number of attributes for the possible-worlds relations.

  To generate random tuples for possible worlds relations, we stored several files of domain values. Each file contains a large number of values from a specific domain, such as names, course numbers, course titles, semesters, and years. The user can specify the number of attributes for the possible world relations. The system forms a random tuple by randomly picking values from the domain of each attribute. The `GeneratePossibleWorlds` module stores the possible worlds relations in Oracle databases, one for each source. The sizes of the relations, and the total size of the integration instance are also recorded.

  These features were used to generate desired cases for the experiments by altering the number of sources, number of possible worlds, size of each possible world, and total size of the sources data. Hence we can evaluate the impact of each parameter on the performance of the algorithm.
- The `TableIntegration` module performs the following tasks for each dataset generated by the `GeneratePossibleWorlds` module discussed above.
  - The module accesses the possible world relations in the databases of the sources. Each source is represented by an Oracle database that contains the possible world relations for thet source.
  - The tuple set for each source is computed as the union of the possible world relations for that source.
  - The module generates the logical formula for each possible world relation for the sources

 ISBN: 978-1-61208-334-6

according to the algorithm of Section II-B The formulas are conveniently represented by vectors, which can be used to easily implement logical operations over the expressions.

- The module determines which sets of possible world relations, one from each source, are *compatible* and hence can produce a possible world relation in the integration. This is done by computing the conjunction (logical and) of the corresponding formulas of the possible world relations. If the result is *false*, the set of possible worlds are not compatible. Otherwise they are compatible.
- For all compatible sets of possible worlds, the modul generates the resulting relation by unioning the possible worlds relations in the set. It stores the integrated relation in the Oracle database for the integration result.
- Once all compatible possible wrolds sets are processed, the module displays the total time for the integration.

IV. Experimental Evaluation

We carried out a large number of experiments to evaluate the performance of the integration algorithm. The experiments were executed on a 2.10 GHz Intel i3 CPU with 4.00 GB RAM, 64-bit Windows 7 Operating System using Java 1.7 and Oracle XE 10g. The first few experiments evaluated the performance of the integration algorithm for integrating information from two sources.

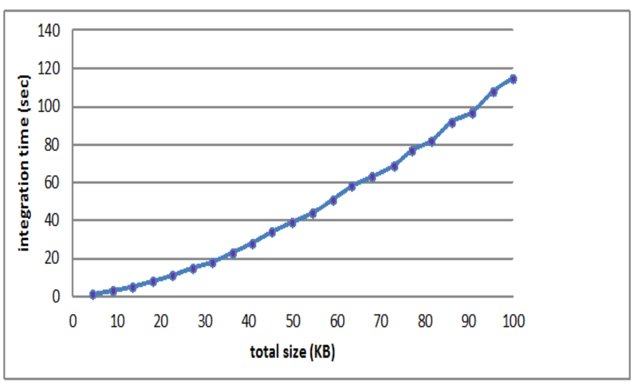


Figure 7: integration times; fixed number of possible worlds for each source.

In the first set of experiments the number of possible world relations of the two sources were kept constant, and test cases were generated by varying the number of tuples in the possible world relations (and hence, varying the size of uncertain databases to be integrated). Figure 7 shows the result of these experiments. The horizontal axis shows the total size (KB) of databases to be integrated. The vertical axis shows the time needed for the integration (sec). The experiments show that the integration algorithm is almost linear in the total size of databases to be integrated.

In the second set of experiments, we varied the number of possible world relations of the two sources while keeping the number of tuples constant. Figure 8 shows the result of these experiments. The horizontal axis shows the total size (KB) of databases to be integrated. The vertical axis shows the time needed for the integration (sec). Again, the experiments show that the integration algorithm is almost linear in the total size of databases to be integrated (no matter whether the size increase is due to larger number of possible worlds per sources, or larger possible world relation sizes.)

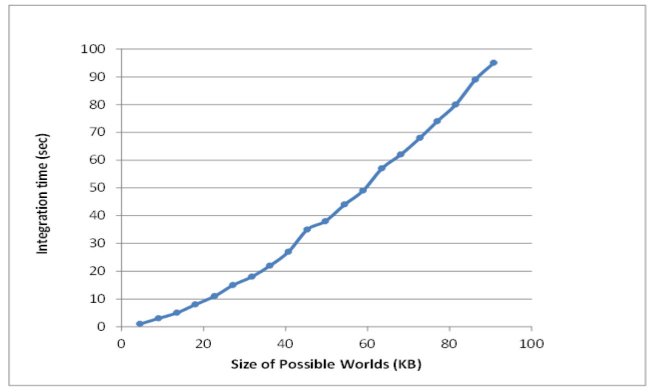


Figure 8: integration times; variable number of possible worlds for each source.

In the next set of experiments we evaluated the impact of the number of possible world relations and their sizes on the integration algorithm. The total size was kept constant (approximately) by changing both the number of possible world relations and the number of tuples in these relations accordingly. The values of these parameters and the integration time are shown in Table I. The columns are, respectively, number of possible worlds for each source, number of tuples in each possible world, total size, and integration time. Total size is almost constant – it ranges between 99.6 and 100.4 KB.

TABLE I: Integration experiments; with total size (almost) constant

PWs	tuples	size	time
20	110	99.9	114
19	116	99.8	114
18	122	99.6	109
17	130	100.4	113
16	138	100.2	116
15	147	100.0	114
14	158	100.3	111
13	170	100.3	116
12	184	100.2	113
11	200	99.8	113
10	220	100.0	114
9	245	100.0	114
8	275	100.0	114
7	315	99.9	111
6	368	100.2	115
5	442	100.3	116
4	553	100.1	119
3	737	100.4	120

Figures 9 and 10 plot the integration time in the experiments of Table I against the number of possible worlds

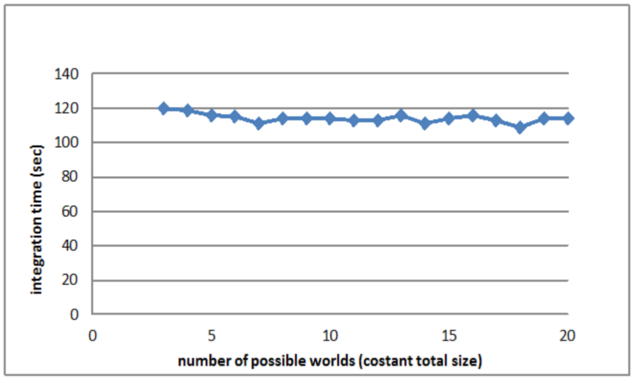


Figure 9: integration times vs number of possible worlds; total size is constant

and the number of tuples in each possible world. The x-axis of Figure 10 (number of tuples in each possible world) is logarithmic to better demonstrate the effect of number of tuples, ranging from 110 to 737 in the experiments. These experiments show that the number of possible worlds and their sizes are not factors in the performance of the integration algorithm when the total size is constant. In other words, integration time is almost constant when number of possible world relations and their sizes change while the total size is fixed. This observation is counter-intuitive since the integration algorithm needs to determine, for every pair of possible worlds (w_1, w_2), whether they are compatible, where w_1 and w_2 belong to source 1 and source 2, respectively. But the impact of number of tuples (smaller number of tuples for larger number of possible world relations) counterbalances the impact of number of possible worlds.

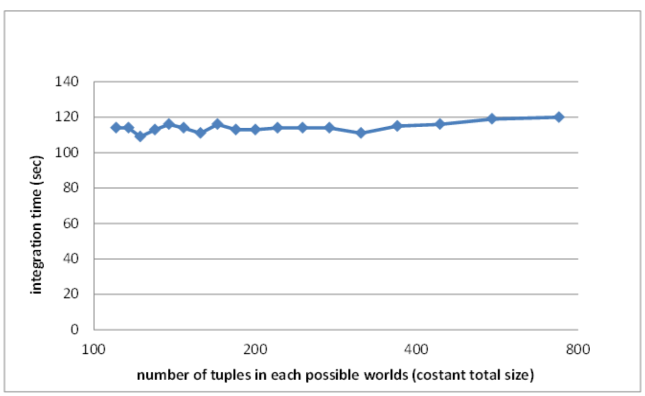
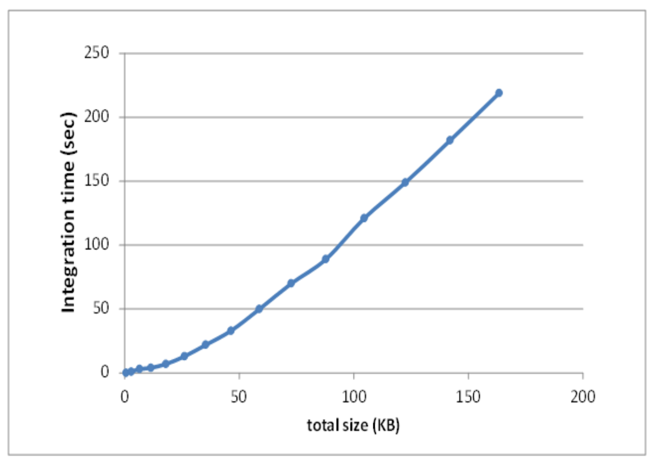


Figure 10: integration times vs number of tuples in each possible world; total size is constant

In the next set of experiments we generated test cases by varying both the number of possible worlds and the number of tuples in each possible worlds (and hence, varying the total size). The results are summarized in Table II. The columns are, respectively, number of possible worlds for each source, number of tuples in each possible world, total size, and integration time. Figure 11 plots the integration time in these experiments against total size. It confirms a near linear performance of the algorithms as a function of the total size of the integration. Figure 12 plots the integration time against the number of tuples in each possible-world relation. Note that the same figure is also the plot of integration time against number of possible worlds, since we are using the same numbers for the two parameters for each data point (See Table II). This figure suggests integration time is a quadratic function of number of possible worlds (or number of tuples in each possible world). This is no surprise, since by varying both these parameters (and using the same numbers) we obtain a total size that is quadratic in each of these parameters. So, again, we confirm that total size is the important parameter in the performance of the algorithm.

TABLE II: Integration experiments; varying number of possible worlds and number of tuples in each possible world

PWs	tuples	size	time
4	4	0.7	0
8	8	2.9	1
12	12	6.5	3
16	16	11.5	4
20	20	18	7
24	24	26.1	13
28	28	35.4	22
32	32	46.4	33
36	36	58.8	50
40	40	72.7	70
44	44	87.8	89
48	48	104.5	121
52	52	122.5	149
56	56	142	182
60	60	163.4	219



negvs

Figure 11: integration times vs total time

In the next set of experiments we evaluate the performance of the integration algorithm when integrating data from more than two information sources. We generated test cases by varying the number of information sources, while keeping the total size constant. Figure 13 plots the integration time against the number of possible worlds. This performance was very unexpected. As seen from this graph, the algorithm has an almost constant time up to about 10 information sources, then the integration time increases sharply. We postulated that the reason for the sharp increase is memory saturation, which

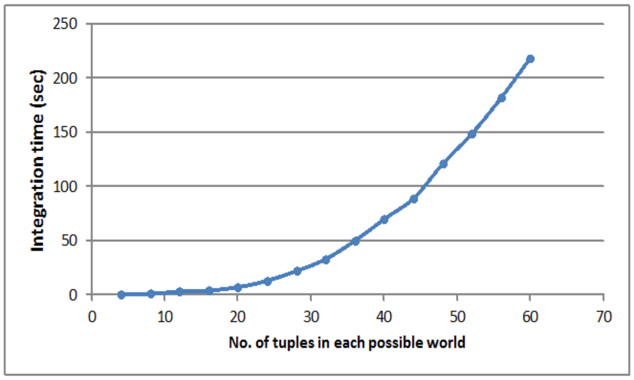


Figure 12: integration times vs number of tuples in possible-world relations

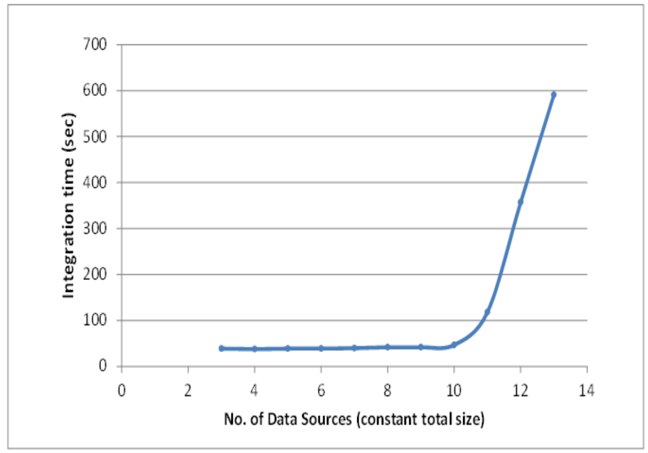


Figure 13: integration times vs number of sources

forces the execution to use virtual memory. In our experiments, the number of possible world relations for each source was kept constant at 3, and constant total size was achieved by varying the number of tuples in each possible world. So, with 10 sources, the number of integration combinations to generate was $3^{10} = 59,049$. This number increases to $3^{11} = 177,147$ for 11 sources, and to $3^{12} = 531,441$ for 12 sources. The memory of our testbed system saturates at about 10 sources.

To test our hypothesis, we executed the exact same experiments on systems with lower (2GB) and higher (8GB) memory sizes. The graphs for these experiments have the same shape, except at lower memory size the graph is shifted to the left, and at higher memory size the graph is shifted to the right. In other words, the sharp increase happens at a lower number of sources for the lower memory size, and at a higher number of sources for higher memory size. These additional experiments confirm our hypothesis that the change in the performance of the integration algorithm, from constant time to almost linear, is a result of memory saturation. So, our final conclusion is that, given adequate memory, the performance of the integration algorithm is a linear function (approximately) of the total size of the integration instance. It is not sensitive to the other factors, namely, number of information sources, number of possible worlds relations in the sources, and number of tuples in the possible world relation, when the total size is kept constant.

V. Conclusion

We presented our implementation and experimental evaluation of the uncertain-data integration algorithms of [11]. Our experiments show the algorithms to be efficient, demonstrating a near linear performance in the total size of the uncertain data to be integrated.

There are a number of important issues that require further investigation. First, uncertain schema mappings is another source of uncertainty in information integration. We would like to develop integration algorithms for this case, with definite or uncertain data. The integration algorithm is a good candidate for parallel computation, in particular, using the map-reduce framework [13]. A future direction would be to implement the integration using Hadoop running on a large number of computers. More importantly, we would like to devise integration algorithms to work with compact representations of uncertain data, such as the probabilistic relational model of [14], [15].

References

[1] L. M. Haas, "Beauty and the beast: The theory and practice of information integration," in Proceedings of International Conference on Database Theory, 2007, pp. 28–43.

[2] A. Y. Halevy, A. Rajaraman, and J. Ordille, "Data integration: The teenage years," Proceedings of International Conference on Very Large Databases, 2006, pp. 9–16.

[3] M. Magnani and D. Montesi, "Uncertainty in data integration: current approaches and open problems," in Proceedings of VLDB Workshop on Management of Uncertain Data, 2007, pp. 18–32.

[4] ——, "A survey on uncertainty management in data integration," ACM Journal of Data and Information Quality, vol. 2, no. 1, 2010.

[5] P. Agrawal, A. D. Sarma, J. D. Ullman, and J. Widom, "Foundations of uncertain-data integration," Proceedings of the VLDB Endowment, vol. 3, no. 1, 2010, pp. 1080–1090.

[6] X. L. Dong, A. Halevy, and C. Yu, "Data integration with uncertainty," in Proceedings of International Conference on Very Large Databases, 2007, pp. 687–698.

[7] X. L. Dong, A. Y. Halevy, and C. Yu, "Data integration with uncertainty," The VLDB Journal, vol. 18, no. 2, 2009, pp. 469–500.

[8] A. A. Eshmawi and F. Sadri, "Information integration with uncertainty," in Proceedings of International Database Engineering and Applications, IDEAS, 2009, pp. 284–291.

[9] R. Fagin, B. Kimelfeld, and P. G. Kolaitis, "Probabilistic data exchange," Journal of the ACM, vol. 58, no. 4, 2011, p. 15.

[10] D. Florescu, D. Koller, and A. Y. Levy, "Using probabilistic information in data integration," in Proceedings of International Conference on Very Large Databases, 1997, pp. 216–225.

[11] F. Sadri, "On the foundations of probabilistic information integration," in Proceedings of International Conference on Information and Knowledge Management, 2012, pp. 882–891.

[12] S. Abiteboul, P. C. Kanellakis, and G. Grahne, "On the representation and querying of sets of possible worlds," in Proceedings of ACM SIGMOD International Conference on Management of Data, 1987, pp. 34–48.

[13] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," in Proceedings of Operating System Design and Implementation, 2004, pp. 137–150.

[14] N. N. Dalvi and D. Suciu, "Efficient query evaluation on probabilistic databases," in Proceedings of International Conference on Very Large Databases, 2004, pp. 864–875.

[15] ——, "Efficient query evaluation on probabilistic databases," The VLDB Journal, vol. 16, no. 4, 2007, pp. 523–544.

 ISBN: 978-1-61208-334-6