Hardening Soft Information Sources

William W. Cohen*
AT&T Labs–Research
Shannon Laboratory
180 Park Avenue
Florham Park, NJ 07932
wcohen@whizbanglabs.com

Henry Kautz†
AT&T Labs–Research
Shannon Laboratory
180 Park Avenue
Florham Park, NJ 07932
kautz@cs.uw.edu

David McAllester
AT&T Labs–Research
Shannon Laboratory
180 Park Avenue
Florham Park, NJ 07932
dmac@research.att.com

ABSTRACT
The web contains a large quantity of unstructured information. In many cases, it is possible to heuristically extract structured information, but the resulting databases are "soft": they contain inconsistencies and duplication, and lack unique, consistently-used object identifiers. Examples include large bibliographic databases harvested from raw scientific papers or databases constructed by merging heterogeneous "hard" databases. Here we formally model a soft database as a noisy version of some unknown hard database. We then consider the hardening problem, i.e., the problem of inferring the most likely underlying hard database given a particular soft database. A key feature of our approach is that hardening is global — many sources of evidence for a given hard fact are taken into account. We formulate hardening as an optimization problem and give a nontrivial nearly linear time algorithm for finding a local optimum.

Categories and Subject Descriptors
H.4.m [**Information Systems**]: Miscellaneous

General Terms
data integration

1. INTRODUCTION
The web contains a large quantity of unstructured information. In many cases, it is possible to heuristically extract structured information, but the resulting databases are "soft": they contain duplication and lack unique consistently-used object identifiers.

As an example of a soft database, consider the large bibliographic databases harvested from raw scientific papers

*Current address: Whizbang Labs-Research, East, 4616 Henry St, Pittsburgh, PA 15213.

†Current address: Dept. of Computer Science, Univ. of Washington, Seattle WA.

KDD '00 Boston, Massachusetts USA

A "soft" database S:
author("Bart Selman", "Critical behavior in satisfiability").
affiliation("Bart Selman", "Cornell University").
author("B. Selman", "Critical behavior for satisfiability").
author("B. Selman", "BLACKBOX theorem proving").

A "hard" database H:
author(bart_selman, critical).
affiliation(bart_selman, cornell).
author(bart_selman, blackbox).

Figure 1: A "soft" bibliographic database and the underlying "hard" data

by systems such as ResearchIndex [11] or Cora [4]. These databases contain information about thousands of technical papers. However, it is often hard to determine if two citations refer to the same paper, or if two names (e.g. "William Cohen" and "W. E. Cohen") refer to the same person. This makes it difficult to perform basic operations such as counting the number of citations to a particular paper or person.

Here we formalize a soft database as one in which distinct identifiers may refer to the same entity. We formalize hardening as the problem of determining the (most likely) co-references between the soft identifiers—that is, determining which pairs of soft identifiers refer to the same real-world object. The result of this is a hard (conventional) database. An example is shown in Figure 1. Notice that more conclusions can be drawn from the hard database H than the soft database S. In particular, H implies that an author of the paper "BLACKBOX theorem proving" is affiliated with Cornell, where as S does not.

Although we will use bibliographic data as a running example, we wish to emphasize that soft data arises in many different contexts. For instance, soft data might be created by automatically extracting facts from classified ads [2], or newsgroup postings [5]. A soft database might also arise from merging the contents of several heterogeneous, autonomously-created "hard" databases.

Formally, we will assume soft facts are given along with some measure of how likely two identifiers are to be co-referent. (For example, the strings "W. E. Cohen" and "William Cohen" are a-priori plausibly co-referent, whereas the strings "William Cohen" and "Bart Selman" are not.) We then formalize the problem of finding the best "hard" model of a set of soft facts probabilistically. We define a prior probability distribution over possible hard databases and a prior probability distribution over name co-references. Given these distributions, the task of constructing the most likely hard database becomes a well-defined optimization problem. This optimization problem involves minimizing the sum of the number of tuples in the hard database, plus a cost associated with the set of co-reference assumptions required.

2. A FORMAL VIEW OF HARDENING

We will assume a set of "references", each of which is some string referring to some real-world object. A tuple of references is written $R(r_1, \ldots, r_n)$ where R ranges over some fixed set of relations and $r_1, \ldots, r_n$ are references. Unlike a conventional hard database, we will allow distinct references to refer to the same real-world entity. We define a *soft database* to be a set of tuples of the form $R(r_1, \ldots, r_n)$, where R ranges over some fixed set of relations and $r_1, \ldots, r_n$ are references.

In the bibliographic domain the same string might denote different entities when used in different contexts, e.g., the string "Michael Jordan" might denote a different person in a news story than in a paper on graphical models. To circumvent this problem we can construct references by concatenating a string with some indication of the context from which that string was extracted; for instance, the author name "Michael Jordan" extracted from a postscript paper p would be represented as the reference "Michael Jordan"$_p$, and would be distinct from the reference "Michael Jordan"$_b$ extracted from a news story b.

To return to the example of Figure 1, suppose one has processed two postscript papers $p1$ and $p2$. The following facts have been extracted from the title page of $p1$:

author("Bart Selman"$_{p1}$, "Critical behavior in satisfiability"$_{p1}$).
affil("Bart Selman"$_{p1}$, "Cornell University"$_{p1}$).

and suppose the following facts have been extracted from the bibliography section of $p2$:

author("B. Selman"$_{p2}$, "Critical behavior for satisfiability"$_{p2}$).
author("B. Selman"$_{p2}$, "BLACKBOX: Applying theorem proving to problem solving"$_{p2}$).

Together these facts form a soft database S.

Hardening determines the co-reference relationships between references in the soft database. The co-reference relation is perhaps most naturally represented by an equivalence relation on the soft references. However, here it is technically more convenient to work with an interpretation function — a function mapping each soft reference to a hard interpretation. In order to incrementally construct interpretation functions we formulate an interpretation function as a set of *interpretation arcs* each of which is an arc of the form $r_1 \stackrel{w}{\to} r_2$ where r_1 and r_2 are distinct references, and w is a non-negative real number weight (or cost) discussed in more detail below. An *interpretation* I is an acyclic set of interpretation arcs, such that for all references r_1 appearing in I there is at most one arc of the form $r_1 \stackrel{w}{\to} r_2 \in I$. This definition allows interpretation arcs to be chained, i.e., it is possible that $r_1 \stackrel{w}{\to} r_2 \in I$ and $r_2 \stackrel{w'}{\to} r_3 \in I$. In this case r_1 is indirectly interpreted as r_3. For any given interpretation I and any reference r, we define $I(r)$ to be the ultimate interpretation of r:

$$I(r) = \begin{cases} I(r') & \text{if } I \text{ contains } r \stackrel{w}{\to} r' \\ r & \text{otherwise} \end{cases}$$

Note that an interpretation I defines a hard database $I(S)$ derived by replacing each reference in S by its ultimate interpretation under I.

Of course not all interpretations are possible — we would not want to interpret "B. Selman" as "H. Kautz". The weight w in the arc $r_1 \stackrel{w}{\to} r_2$ is the cost (or unlikeliness) of interpreting r_1 as r_2. Formally we assume the costs are provided in the form of a *potential interpretation set*, i.e., a set I_{pot} of (weighted) potential interpretation arcs. Given a soft database S and a set I_{pot} of potential interpretations our goal will be to find an interpretation $I \subseteq I_{\text{pot}}$ minimizing the cost function $c(I)$ defined as follows where λ_1 and λ_2 are parameters of the cost function; $|I(S)|$ is the number of distinct tuples in the hard database $I(S)$; and $|I|$ is the number of interpretation arcs in I.

$$\begin{aligned} c(I) &\equiv w(I) + \lambda_1 |I(S)| + \lambda_2 |I| \qquad (1) \\ w(I) &\equiv \sum_{r_1 \stackrel{w}{\to} r_2 \in I} w \end{aligned}$$

Note that as more arcs are added to I we have that $w(I) + \lambda_2 |I|$ increases while $\lambda_1 |I(S)|$ decreases. The cost function hence represents a tradeoff between the unlikeliness of the interpretation and the compactness of the resulting hard database. This objective function is approximately derived from a probabilistic model in the following section.

3. A PROBABILISTIC MODEL

In this section we give an approximate probabilistic derivation of the cost function (1). Let $Pr(H, I, S)$ a distribution on hard databases H, interpretations I, and soft databases S. An optimal hardening of S is a pair H, I that maximizes $Pr(H, I|S)$. Notice that for a fixed S the optimal hardening can be found by simply maximizing the joint probability of H, I and S.

To define $Pr(H, I, S)$ we will assume that there is a finite set U of potential real-world objects and a finite set of relations, each of fixed arity. These assumptions imply that there is a finite set of possible tuples Φ which could appear in H. We let N be the number of possible hard tuples. In the probabilistic model we assume that both the hard data base H and the soft database S can contain duplicate tuples. We let $|H|$ and $|S|$ be the number of tuples in H and S respectively where distinct occurrences of the same tuple are treated as separate elements. We also let $|I|$ be the number of arcs in an interpretation I — note that duplication of arcs in I is ruled out by the requirement that every reference has at most one outgoing arc. We now assume that $Pr(U,\ I,\ S)$ can be decomposed as follows where ϵ_H, ϵ_I, and ϵ_S are real number parameters in the interval $(0, 1)$ and the number Z is selected so that $Pr(I)$ sums to 1.

$$Pr(H, I, S) \;=\; Pr(S|I, H) \cdot Pr(I) \cdot Pr(H) \qquad (2)$$

$$Pr(H) \;=\; \epsilon_H (1-\epsilon_H)^{|H|} \left(\frac{1}{N}\right)^{|H|} \qquad (3)$$

$$Pr(I) \;=\; \frac{1}{Z} \cdot \epsilon_I (1-\epsilon_I)^{|I|} \prod_{r_1 \overset{w_i}{\to} r_2 \in I} e^{-w_i} \qquad (4)$$

$$Pr(S|I, H) \;=\; \epsilon_S (1-\epsilon_S)^{|S|} \left(\frac{1}{|I^{-1}(H)|}\right)^{|S|} \qquad (5)$$

In (5) we have that $I^{-1}(H)$ is the set of soft tuples Φ such that $I(\Phi) \in H$ and $|I^{-1}(H)|$ is the cardinality of this set.

The above equations correspond to a certain process for generating the triple H, I, S. First H is generated by repeatedly selecting one of the N possible hard tuples at random. At each iteration the selection process stops with probability ϵ_H and continues with probability $1 - \epsilon_H$. A similar process is used to generate I by selecting arcs from $I_{\rm pot}$ where the probability of the arc $r_1 \overset{w}{\to} r_2$ is proportional to e^{-w} and where, on each selection iteration, the selection process terminates with probability ϵ_I. In the case of generating I, however, after the arcs have been selected we check that I is well formed, i.e., that it is acyclic and that each reference has at most one outgoing edge. If I is not well formed then we start over. If the weights in $I_{\rm pot}$ are normalized so that the probability of selecting $r_1 \overset{w}{\to} r_2$ is exactly e^{-w}, we have that the constant Z equals the probability that a I is well formed when the selection phase terminates. Finally, given H and I we generate S using a process analogous to that for generating H except that we select individual tuples from the set $I^{-1}(H)$.

Given a soft data base S our objective is find the values I and H maximizing $P_r(I, H|S)$. This is equivalent to finding I and H so as to maximize $Pr(H, I, S)$. For any given value of S and I the optimal value of H is simply $I(S)$. So, given S, it suffices to find the value of I maximizing $Pr(I(S), I, S)$. This is in turn equivalent to minimizing the "complexity" of the triple H, I, and S, i.e., minimizing the quantity $-\ln Pr(I, H, S)$. For this minimization problem we can ignore terms that do not depend on the choice of I. So the problem becomes that of selecting I so as to minimize the cost $c'(I)$ defined as follows.

$$\begin{aligned} c'(I) \;&\equiv\; |I(S)| \log\left(\frac{N}{1-\epsilon_H}\right) && (6)\\ &\quad + |I| \log\left(\frac{1}{1-\epsilon_I}\right) + w(I) \\ &\quad + |S| \log |I^{-1}(I(S))| \end{aligned}$$

To derive (1) we assume that the last term in (6) can be approximated by a linear function of the form $\alpha + \beta|I|$.

4. OPTIMAL HARDENING IS NP-HARD

Here we show that the optimization problem defined by (1) is NP-hard. The problem remains NP-hard even under rather severe restrictions. In particular, we can assume that all weights in $I_{\rm pot}$ are zero and that λ_2 in (1) is also zero so that we are simply minimizing $|I(S)|$. Furthermore, we can assume that $I_{\rm pot}$ has a rather restricted form. Define a "hard reference" to mean a reference h such that $I_{\rm pot}$ contains an arc of the form $r \overset{w}{\to} h$ and define a "soft reference" to be a reference r such that $I_{\rm pot}$ contains an arc of the form $r \overset{w}{\to} h$. The graph $I_{\rm pot}$ is bipartite if and only if the hard references are disjoint from the soft references.

THEOREM 4.1. *Unless $NP = P$, there is no polynomial time algorithm for computing an interpretation $I \subseteq I_{pot}$ minimizing $|I(S)|$ even under the assumption that I_{pot} is bipartite and every soft reference occurs at most once in S.*

PROOF. The proof is by reduction of vertex cover [7]. Let $G = (V, E)$ be a graph. The vertex cover problem is that of determining if there exists $V' \subseteq V$ with $|V'| < k$ such that every edge in E contains a vertex in V'. Given the graph let S be the soft data base such that for each edge $\{x, y\} \in E$ we have that S contains the tuple $R(r_{\{x,y\}})$. Note that no reference occurs more than once in S. Let $I_{\rm pot}$ consist of all arcs of the form $r_{\{x,y\}} \overset{0}{\to} x$ and $r_{\{x,y\}} \overset{0}{\to} y$. Note that $I_{\rm pot}$ is bipartite. For any interpretation I providing a hard interpretation for every reference we have that $I(S)$ defines a subset of the vertices that covers the edges. We now have that there exists an I with $|I(S)| < k$ if and only if there exists a vertex cover of the given graph smaller than k. $\square$

5. A GREEDY HARDENING ALGORITHM

While finding an optimal hardening is intractable, it is clearly possible to heuristically search for a good hardening. A natural approach is to use a local search algorithm that begins with an empty interpretation I and iteratively extends it so as to improve the cost function (1). We will consider a simple algorithm that adds one arc at a time — at each iteration I is replaced with $I \cup \{r_1 \overset{w}{\to} r_2\}$. Each arc is selected greedily so as to induce the largest reduction in $c(I)$.

To constructing an efficient greedy algorithm it is convenient to introduce some formal notation. We abbreviate $I \cup \{r_1 \overset{w}{\to} r_2\}$ as $I + r_1 \overset{w}{\to} r_2$. We define $\texttt{candidates}(I)$ to be the set of arcs $r_1 \overset{w}{\to} r_2 \in I_{\rm pot}$ such that $I + r_1 \overset{w}{\to} r_2$ is a well formed interpretation, i.e., such that $I(r_2) \neq I(r_1)$ and $I(r_1) = r_1$. Assuming that I and S are clear from context we define $\Delta(r_1 \overset{w}{\to} r_2)$ to be the reduction in cost achieved by adding $r_1 \overset{w}{\to} r_2$, i.e., the quantity $c(I) - c(I + r_1 \overset{w}{\to} r_2)$.

The greedy algorithm can be viewed as repeatedly merging equivalence classes. At each point in time the interpretation I defines an equivalence relation on the set of references used in S, i.e., r_1 and r_2 are equivalent under I if $I(r_1) = I(r_2)$. Each new arc $r_1 \overset{w}{\to} r_2$ added to I merges two equivalences classes.

Each step of the form $I\ := I + r_1 \overset{w}{\to} r_2$ can also be viewed as a kind of heuristic inference. For example, if S contains a tuple stating that "B. Selman"$_{p1}$ is an author of paper x and another tuple stating that the (different) reference "Bart Selman"$_{p2}$ is also an author of x then merging these two references reduces the number of tuples in $I(S)$. If two references have several overlapping facts, e.g., they are both authors of the same *two* papers, then we have even stronger evidence that they are the same. The above algorithm iteratively finds the "safest" heuristic inference, i.e., the one for which the positive evidence in the form of shared tuples most strongly outweighs the cost of the identification.

Note that as more inferences are made additional evidence can be generated for further inferences. For example, originally we may have that r_1 authors papers x and y; r_2 authors papers y and z; and r_3 authors papers x and z. The evidence for any single merger is originally one shared paper, e.g., r_1 and r_2 share paper x. Once we have merged r_1 and r_2, however, the evidence for merging r_3 with the class $\{r_1,\ r_2\}$ becomes stronger — it is now based on two shared papers.

The above comments imply that even for a fixed arc $r_1 \stackrel{w}{\rightarrow} r_2$, the value of $\Delta(r_1 \stackrel{w}{\rightarrow} r_2)$, which can be viewed as the current weight of evidence in favor of $r_1 \stackrel{w}{\rightarrow} r_2$, can change during the execution of the algorithm. So it is not immediately clear that finding the next greedy arc (the safest next inference) can be done without searching though all the potential arcs. Assuming that I_{pot} is roughly proportional to $|S|$, the naive implementation would run in time quadratic in the size of S. For soft data bases of sizes typical for web applications quadratic behavior is prohibitive. Our main result is that the greedy algorithm can be implemented in a way that runs to completion in nearly linear time.

We now define the ambiguity of a reference r_1 to be the number of arcs of the form $r_1 \stackrel{w}{\rightarrow} r_2$ in I_{pot}. The ambiguity of a potential interpretation set I_{pot} is the maximal ambiguity of any reference r_1 appearing in I_{pot}. Let d be the ambiguity of I. Let k be the maximum number of arguments of any relation symbol. We now have the following.

THEOREM 5.1. *It is possible to run the greedy search algorithm to completion in time $O(|S|k^3 d \log(|S|kd))$.*

The efficient implementation keeps the elements of $\mathtt{candidates}(I)$ in a priority queue where the priority of $r_1 \stackrel{w}{\rightarrow} r_2$ is $\Delta(r_1 \stackrel{w}{\rightarrow} r_2)$. The algorithm incrementally updates priorities on each iteration. The efficient version can be written as follows where Q is a priority queue containing arcs in I_{pot}. Each time an arc is removed from Q it is either determined to no longer be a candidate or is added to I. The algorithm maintains priorities so the priority of an arc $r_1 \stackrel{w}{\rightarrow} r_2$ in $\mathtt{candidates}(I)$ is always $\Delta(r_1 \stackrel{w}{\rightarrow} r_2)$. The algorithm also uses a union-find data structure to maintain the current equivalence relation on references. The operation $\mathtt{union}(r_1,\ r_2)$ declares two references to be equivalent, and two references r_1 and r_2 have been made equivalent if and only if $\mathtt{find}(r_1)$ is the same as $\mathtt{find}(r_2)$. The efficient implementation is summarized below.

Efficient Implementation

$I := \emptyset$;
$Q :=$ a priority queue containing all arcs in I_{pot}
  where the priority of $r_1 \stackrel{w}{\rightarrow} r_2$ is $\Delta(r_1 \stackrel{w}{\rightarrow} r_2)$.
while Q is not empty and the largest priority is positive
  Remove the highest priority arc $r_1 \stackrel{w}{\rightarrow} r_2$ from Q;
  If $\mathtt{find}(r_1) \neq \mathtt{find}(r_2)$ and $I(r_1) = r_1$ then
    execute $I := I + r_1 \stackrel{w}{\rightarrow} r_2$; $\mathtt{union}(r_1,\ r_2)$; and
    update the priorities of arcs on Q

The remainder of this section describes how the priority updates on each iteration can be done in sublinear time (amortized over all iterations). We first reduce the problem of maintaining priorities to the conceptually simpler problem of maintaining the effect of each arc on the quantity $|I(S)|$. Note that the equivalence relation maintained in the union-find data structure is the same as that defined by I, i.e., we have that $I(r_1) = I(r_2)$ if and only if $\mathtt{find}(r_1) = \mathtt{find}(r_2)$. For any arc $r_1 \stackrel{w}{\rightarrow} r_2 \in I_{\mathrm{pot}}$ define $\Delta_H(r_1 \stackrel{w}{\rightarrow} r_2)$ to be the change in $|I(S)|$. We then have the following where $\mathtt{find}(S)$ is the result of replacing each soft reference r in S by $\mathtt{find}(r)$ and $\mathtt{find} + \mathtt{union}(r_1,\ r_2)$ denotes the find map that results from performing the union operation on r_1 and r_2.

$$\Delta_H(r_1 \stackrel{w}{\rightarrow} r_2) = |\mathtt{find}(S)| - |[\mathtt{find} + \mathtt{union}(r_1,\ r_2)](S)|$$

Since $\Delta(r_1 \stackrel{w}{\rightarrow} r_2)$ can be written as $\lambda_1 \Delta_H(r_1 \stackrel{w}{\rightarrow} r_2) - w - \lambda_2$ it now suffices to incrementally maintain $\Delta_H(r_1 \stackrel{w}{\rightarrow} r_2)$.

To efficiently update $\Delta_H(r_1 \stackrel{w}{\rightarrow} r_2)$ we maintain a set E of effect assertions each of which is a triple $\langle \Phi,\ r_1 \stackrel{w}{\rightarrow} r_2,\ \Psi \rangle$ where $\Phi \in \mathtt{find}(S)$, $r_1 \stackrel{w}{\rightarrow} r_2 \in I_{\mathrm{pot}}$ with $\mathtt{find}(r_1)$ occurring in Φ, and where Ψ is the result of replacing $\mathtt{find}(r_1)$ by $\mathtt{find}(r_2)$ in Φ. Note that we do not require the arc to be in $\mathtt{candidates}(I)$. The assertion $\langle \Phi,\ r_1 \stackrel{w}{\rightarrow} r_2,\ \Psi \rangle$ can be viewed as a kind of meta-assertion about the effect of adding the arc $r_1 \stackrel{w}{\rightarrow} r_2$ to I. The assertion $\langle \Phi,\ r_1 \stackrel{w}{\rightarrow} r_2,\ \Psi \rangle$ states that if we union r_1 and r_2 and change the find map so that the find of elements in the equivalence class of r_1 are set to the find of r_2, then Φ will be converted to Ψ. The algorithm maintains the invariant that E is the set of all effect assertions (for the current values of I and $\mathtt{find}$). The initial set E_0 is the set of triples $\langle \Phi,\ r_1 \stackrel{w}{\rightarrow} r_2,\ \Psi \rangle$ such that $\Phi \in S$, r_1 occurs in Φ and Ψ is the result of replacing r_1 by r_2 in Φ.

We now describe the relationship between the set E of effect assertions and the quantities $\Delta_H(r_1 \stackrel{w}{\rightarrow} r_2)$. Define, for any arc x, the set $\mathtt{range}(x)$ to be the set of Ψ such that there exists a $\Phi \in \mathtt{find}(S)$ such that E contains a triple of the form $\langle \Phi,\ x,\ \Psi \rangle$. For any Ψ in $\mathtt{range}(x)$ we define $\mathtt{count}(x,\ \Psi)$ to be the number of triples in E of the form $\langle \Phi,\ x,\ \Psi \rangle$ plus one if $\mathtt{find}(S)$ contains Ψ. It is possible to show the following.

$$\Delta_H(x) = \sum_{\Psi \in \mathtt{range}(x)} [\mathtt{count}(x,\ \Psi) - 1] \qquad (7)$$

By maintaining appropriate indices, it is possible to efficiently incrementally maintain the set E, the set $\mathtt{find}(S)$, and the values of $\mathtt{count}(x,\ \Psi)$ and $\Delta_H(x)$. One key detail is that in merging equivalence classes for references r_1 and r_2 we change the value of the find map on the smaller of the two classes. This ensures that each time the find of a reference changes the size of its equivalence class at least doubles. So the number of find changes for a given reference can be at most the log of the number of references, i.e., at most $\log(|S|k)$. Another key observation is that in any triple $\langle \Phi, x, \Psi \rangle \in E$ we have that Φ and x determine Ψ. This implies that are at most $|S|kd$ triples in E. Details are given in the full paper.

6. CONCLUSIONS

Previous work has considered reasoning directly with soft databases (e.g. [3, 6, 1]). One disadvantage of this approach is that queries to a soft, probabilistic database are generally more expensive to answer than queries to a conventional "hard" database. There is also close connection between the work described here and well-studied problem of *record linkage* (e.g. [10, 8, 9]). In record linkage the goal to determine which entity descriptions are co-referent (that is, refer to the same real-world object). It is generally assumed that each entity is represented by a "record"—a vector of atomic values—and the similarity or dissimilarity of two records is measured by comparing these vectors. However, there are many situations where the similarity of two entity names depends on properties that cannot be easily represented in a single record. For instance, "B. Selman" and "Bart Selman" are clearly more likely to be variants of the same name if "B. Selman" and "Bart Selman" have authored similarly-titled technical papers; however, a publication history is not easily stored as a record attribute. This heuristic could be easily incorporated into our approach by extending a soft bibliographic database with facts about co-authorship.

In this paper we have considered the following more general problem: given a soft database S, and a structure I_{pot} indicating which name co-reference relationships are possible, find the hard database H that is most likely given S. We show that a natural formalization of this optimization problem is NP-complete but that optimally greedy hardening can be done in time nearly linear in $|S|$.

7. REFERENCES

[1] D. Barbara, H. Garcia-Molina, and D. Porter. The management of probabilistic data. *IEEE Transations on knowledge and data engineering*, 4(5):487–501, October 1992.

[2] M. E. Califf and R. Mooney. Relational learning of pattern-match rules for information extraction. In *Working Papers of ACL-97 Workshop on Natural Language Learning*, 1997.

[3] W. W. Cohen. Integration of heterogeneous databases without common domains using queries based on textual similarity. In *Proceedings of ACM SIGMOD-98*, Seattle, WA, 1998.

[4] Cora: Computer science research paper search engine. http://www.cora.justresearch.com, 2000.

[5] D. Freitag. Multistrategy learning for information extraction. In *Proceedings of the Fifteenth International Conference on Machine Learning.* Morgan Kaufmann, 1998.

[6] N. Fuhr. Probabilistic Datalog—a logic for powerful retrieval methods. In *Proceedings of the 1995 ACM SIGIR conference on research in information retrieval*, pages 282–290, New York, 1995.

[7] M. R. Gary and D. S. Johnson. *Computers and Intractibility: A Guide to the Theory of NP-completeness.* W. H. Freeman and Company, New York, 1979.

[8] M. Hernandez and S. Stolfo. The merge/purge problem for large databases. In *Proceedings of the 1995 ACM SIGMOD*, May 1995.

[9] A. Monge and C. Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *The proceedings of the SIGMOD 1997 workshop on data mining and knowledge discovery*, May 1997.

[10] H. B. Newcombe, J. M. Kennedy, S. J. Axford, and A. P. James. Automatic linkage of vital records. *Science*, 130:954–959, 1959.

[11] Researchindex: The NECI scientific literature digital library. http://www.researchindex.com, 2000.