Knowing What to Believe and Who to Trust in the Presence of Conflicting Information

Jeff Pasternack and Dan Roth
University of Illinois at Urbana-Champaign
Introduction

- The advent of the Information Age and the Web
  - Overwhelming quantity of information
  - But uncertain quality:
    - Collaborative media
      - Blogs
      - Wikis
      - Message boards
    - Established media are losing market share
      - Reduced fact-checking
  - Information extraction tells us what a document says
  - But what do we believe?
Introduction

- Too many documents from too many sources
- Not feasible for a human to read them all
- A computational trust system can be our proxy
  - Ideally assign the same trust judgments the user would
- The user may be another system
  - A question answering system
  - A navigation system
  - Etc.
Overview

- Applying prior knowledge to fact-finders
  - Motivation
  - Fact-finders
  - A brief introduction to (integer) linear programming
  - Encoding prior knowledge as linear constraints
    - And enforcing it with a polynomial-time linear program
  - Results and discussion

- Current work
  - Constrained learning via linear programming
  - Enhanced fact-finding

- Future concerns and directions
Applying Prior Knowledge to Fact-Finders
Motivation

- Problem: Information sources assert conflicting claims
- Fact-finders evaluate a bipartite network of information sources and the claims they make
  - Determine which sources are trustworthy
  - And which claims are believable
  - Sources with believable claims are more trustworthy
  - Claims with trustworthy sources are more believable
  - E.g. TruthFinder (Yin, Han & Yu, 2008)
- Usually work much better than voting
  - But seek a global, objective “ground truth”
  - And can’t take advantage of prior knowledge
Applying Prior Knowledge to Fact-Finders

Motivation

- Truth is subjective
  - Opinions (“this restaurant is good”)
  - Facts (“man landed on the moon”)
- User’s prior knowledge biases what we should believe
  - User A believes that man landed on the moon
  - User B believes the moon landing was faked
  - Different belief in the claim “there is a mirror on the moon”
- Common-sense
  \[ \neg ManOnMoon \Rightarrow \neg MirrorOnMoon \]
- How do we apply this prior knowledge efficiently?
  - Linear programming
Applying Prior Knowledge to Fact-Finders

Motivation: Deception

"...HOLD STILL, LARRY. IT'S TAKING ANOTHER PICTURE..."
Applying Prior Knowledge to Fact-Finders

Fact-Finders: Introduction

- Fact-finders evaluate a bipartite heterogeneous information of network sources and the claims they make
  - Sources with believable claims are more trustworthy
  - Claims with trustworthy sources are more believable
  - E.g. TruthFinder (Yin, Han & Yu, 2008)

- Usually work much better than voting
  - But seek a global, objective “ground truth”
  - And can’t take advantage of prior knowledge
Applying Prior Knowledge to Fact-Finders
Fact-Finders: Graphical Representation

Information sources $S$

- $s_1$
- $s_2$
- $s_3$
- $s_4$

Claims $C$

- $c_1$
- $c_2$
- $c_3$
- $c_4$
- $c_5$

Mutual exclusion sets

- $M_1$
- $M_2$

Bipartite information network

Each source $s \in S$ asserts a set of claims $C_s \subseteq C$

Each claim $c \in C$ belongs to a mutual exclusion set $M_c \subseteq C$

(Graph adapted from a slide by Jiawei Han)
Applying Prior Knowledge to Fact-Finders

Fact-Finders: General Algorithm

Iterate:

- Calculate trust in each source $T^i(s)$ at iteration $i$ in terms of the belief in its claims in the previous iteration, $B^{i-1}(C_s)$
- Calculate belief in each claim $B^i(c)$ in terms of $T^i(S_c)$
  
  $S_c = \{ s : s \in S, c \in C_s \}$
- Finish on specific number of iterations, or stop criteria

Various possible belief priors

- $B^0_{voted}(c) = \frac{|S_c|}{\sum_{d \in M_c} |S_d|}$
- $B^0_{uniform}(c) = |M_c|^{-1}$
- $B^0_{fixed}(c) = 0.5$
Applying Prior Knowledge to Fact-Finders

Fact-Finders: Analogy to Hubs and Authorities

- Claims ↔ authorities, information sources ↔ hubs

Also know as Hyperlink-Induced Topic Search (HITS)

Can be adapted as a fact-finder (we call it Sums):

- \[ T^i(s) = \sum_{c \in Cs} B^{i-1}(c) \]  
  Trustworthiness of source = sum of belief in its claims

- \[ B^i(c) = \sum_{s \in Sc} T^i(s) \]  
  Belief in claim = sum of belief in its sources

(Adapted from a slide by Jiawei Han)
Applying Prior Knowledge to Fact-Finders

Fact-Finders: Existing Algorithms

- **TruthFinder (Yin, Han & Yu, 2008)**
  - Simplified version—see paper for complete version
  - \[ T^i(s) = \sum_{c \in C_s} B^{i-1}(c) / |C_s| \]  
    Mean probability of claims
  - \[ B^i(c) = 1 - \prod_{s \in S_c} (1 - T^i(s)) \]  
    1 – P(all sources are wrong)

- **3-Estimates (Galland et al., 2010)**
  - Based on summation
  - Extends the typical model by adding a third set of parameters
    - These capture the “difficulty” of a claim
    - Some truths are harder to discern than others
      - \( 1 + 1 = 2 \)
      - \( P =? NP \)
Applying Prior Knowledge to Fact-Finders

Our Contribution

- Next, we present three novel fact-finders
- Then, we’ll introduce our framework for incorporating prior knowledge into any fact-finder
Applying Prior Knowledge to Fact-Finders

Fact-Finders: Novel Algorithm #1

- **Average·Log**
  - \( T^i(s) = \log |C_s| \cdot \sum_{c \in C_s} B^{i-1}(c)/|C_s| \)
  - \( B^i(c) = \sum_{s \in Sc} T^i(s) \)

\( \log(#\text{claims}) \cdot \text{mean belief} \)

\( \text{sum trust in sources} \)
Investment

- Sources "invest" trustworthiness uniformly among their claims
- Belief in each claim grows according to a non-linear function $G$
- Source trustworthiness is sum of belief in its claims, weighted by its relative investment in each
- We use $G(x) = x^g$, with $g = 1.2$

\[ T^i(s) = \sum_{c \in C_s} B^{i-1}(c) \cdot \frac{T^{i-1}(s)}{|C_s|} \cdot \sum_{r \in S_c} \frac{T^{i-1}(r)}{|C_r|} \]

\[ B^i(c) = G \left( \sum_{s \in S_c} \frac{T^i(s)}{|C_s|} \right) \]

Proportion of total "investment" in $c$ provided by $s$ at time $i-1$

Each source "invests" uniformly among all its claims
Applying Prior Knowledge to Fact-Finders
Fact-Finders: Novel Algorithm #3

- **Pooled Investment**
  - Similar to Investment
  - Sources invest trustworthiness uniformly among their claims
  - $T^i(s)$ remains the same
  - Difference: claims in each $M$ are linearly scaled so their sum remains the same after being grown by $G$
  - We use $G = x^{1.4}$

$$H^i(c) = \sum_{s \in S_c} \frac{T^i(s)}{|C_s|}$$  
Investment in claim $c$

$$B^i(c) = H^i(c) \cdot \frac{G(H^i(c))}{\sum_{d \in M_c} G(H^i(d))}$$
Normalize belief in $c$
(Integer) Linear Programming
A One-Slide Introduction

**Given variables** $x_1, x_2, x_3, \ldots x_n$

**With constraints** $i = 1 \ldots m$

of the form $a_{i,1}x_1 + a_{i,2}x_2 + a_{i,3}x_3 + \ldots a_{i,n}x_n \geq b_i$

**Minimize cost** $w_1x_1 + w_2x_2 + w_3x_3 + \ldots w_nx_n$

**Integer LP:**
Each $x \in \mathbb{Z}$

(Graphs provided by James Clarke)
Applying Prior Knowledge to Fact-Finders

Prior Knowledge

- Prior knowledge comes in two flavors
  - Common-sense reasoning
    - Cities generally grow over time
    - A person has two biological parents
  - Specific knowledge
    - John was born in 1970 or 1971
    - The population of Los Angeles is greater than Phoenix

- Can be represented with first-order logic
  - Population grows over time \([\text{pop(city,population, year)}]\):
    - \(\forall v,w,x,y,z \ \text{pop}(v,w,y) \land \text{pop}(v,x,z) \land z > y \Rightarrow x > w\)
  - Tom is older than John
    - \(\forall x,y \ \text{Age}(\text{Tom}, x) \land \text{Age}(\text{John}, y) \Rightarrow x>y\)
Applying Prior Knowledge to Fact-Finders
Prior Knowledge as Linear Constraints

- We can convert first-order logic to linear constraints
  - Similar to what is done for ILPs (Yih, 2004)
- Convert FOL to propositional logic
- Convert propositional logic to conjunctive normal form
- Each claim $c$ will be represented by a proposition
  - And ultimately a $[0,1]$ variable in the linear program
  - This variable can be informally thought of as $P(c)$
Applying Prior Knowledge to Fact-Finders
Prior Knowledge as Linear Constraints

- Repeat:
  - Run one iteration of the fact-finder
    - Calculate $T_i(S)$ given $B_{i-1}(C)$
    - Obtain new belief values $B_i(C)'$
  - Apply the linear program to “correct” $B_i(C)' \rightarrow B_i(C)$
Applying Prior Knowledge to Fact-Finders
Prior Knowledge as Linear Constraints

- LP versus ILP
  - Why use a linear program instead of an integer linear program?
    - We want a belief assignment that “minimally” corrects the output of the fact-finder
      - ILP, however, would assign a \{0,1\} value to every variable
        - Best possible assignment at that moment, but “truncates” the information available to later iterations
      - The LP assigns \([0,1]\) value, giving us a “corrected” distribution
    - LPs can be solved in polynomial time (Karmarkar, 1984) whereas ILP is NP-hard
Applying Prior Knowledge to Fact-Finders
Prior Knowledge as Linear Constraints

- For each disjunctive clause with set $P$ of positive literals (claims) and a set $N$ of negations of literals:
  - Add the constraint $\sum_{c \in P} c_v + \sum_{c \in N} (1 - c_v) \geq 1$, where $c_v$ denotes a $[0,1]$ LP variable corresponding to each $c$

- LHS is a union bound of at least one claim being true
  - Optimistic in some cases
  - $x \lor y$ translates to $x + y \geq 1$
    - Exact if $x \oplus y$
    - But what if $x \Leftrightarrow y$?
Applying Prior Knowledge to Fact-Finders
Prior Knowledge as Linear Constraints

- Many common constraints can be represented exactly
  - For example, $q \Rightarrow r^1 \lor r^2 \lor \ldots$
    - Where the $r$ literals are mutually exclusive
  - Translates exactly to $r^1_v + r^2_v + \ldots \geq q_v$
  - Mutual exclusion among $n$ claims $c^1, c^2, \ldots, c^n$ can be compactly written as $c^1_v + c^2_v + \ldots + c^n_v = 1$
Applying Prior Knowledge to Fact-Finders
The Cost Function

- So linear constraints encode our prior knowledge
- But we do not want just *any* consistent belief distribution
- We want one that is “close” to the belief distribution produced by our fact-finder
- This distance will be our cost function
  - Must be linear
- First, we need to convert the belief scores produced by the fact-finder into a number of “votes”
Applying Prior Knowledge to Fact-Finders
The Cost Function

- Each claim $c$ receives $\omega_c = \omega(B(c))$ votes
- The vote function depends on the fact-finder
  - TruthFinder is probabilistic, so we use $\omega_{inv}(x) = \min ( (1-x)^{-1}, m_{inv} )$ with $m_{inv} = 10^{10}$
    - # votes scales with error; e.g. 0.1 error = 10 votes, 0.01 error = 100
  - Our other fact-finders have “linear” beliefs
    - Use identify function: $\omega_{idn}(x) = x$
Applying Prior Knowledge to Fact-Finders
The Cost Function

- Now we can write our distance (cost) to be minimized
- We use \textbf{VoteDistance}, a weighted Manhattan distance
  - The cost for increasing belief in a claim is proportional to the number of votes against it
  - The cost for decreasing belief in a claim is proportional to the number of votes for it
- \[ \sum_{c \in C} \max \left( \begin{array}{c}
\left( \omega_{M_c} - \omega_c \right) \cdot \left( c_v - \omega_c/\omega_{M_c} \right), \right.
\left. \omega_c \cdot \left( \omega_c/\omega_{M_c} - c_v \right) \right) \]
  - Cost of increasing belief in \( c \)
  - Cost of decreasing belief in \( c \)
- Where \( \omega_{M_c} = \sum_{d \in M_c} \omega_d \)
- Gives the “least objectionable” correction to the fact-finder’s output, frustrating the fewest votes
Applying Prior Knowledge to Fact-Finders From Values to Votes to Belief

- Each $c_v$ has now been assigned a $[0,1]$ value by the LP
- Let $\omega^{-1}$ be the inverse vote function (votes $\rightarrow$ belief)
  \[ \omega_{\text{inv}}^{-1}(x) = 1 - (1 + y)^{-1} \quad \omega_{\text{idn}}^{-1}(x) = x \]
- How do we use these values to redistribute belief?
  - Vote Conservation: $B(c) = \omega^{-1}(c_v \cdot \omega_{\text{Mc}})$
    - Redistribute votes among the members of each mutual exclusion set
    - Total # votes remains constant
  - Vote Loss: $B(c) = \omega^{-1}( \min(\omega_c, c_v \cdot \omega_{\text{Mc}}))$
    - Claims can only lose votes
      - A claim is not more believable relative to the claims in other mutual exclusion sets simply because the claims in its own mutual exclusion set have lost belief
    - Works slightly better in practice
Applying Prior Knowledge to Fact-Finders

Experiments

- Three domains with four datasets
  - City population (Wikipedia infobox & synthetic data)
  - Basic biographies (Wikipedia infobox data)
  - American vs. British Spelling (articles)
    - British National Corpus, Reuters, Washington Post
  - Sometimes choosing the right fact-finder has the most significant impact on performance
    - Our three novel fact-finders usually do best
  - At other times, prior knowledge is more important
    - And prior knowledge almost always helps
Applying Prior Knowledge to Fact-Finders Experiments: Wikipedia Infoboxes

- Semi-structured data source
  - Relatively easy information extraction
  - Lots of claims
- We know who wrote what
- But still have question of attribution
  - Does an edit outside the infobox mean that the editor read and verified the content of the infobox too?
Applying Prior Knowledge to Fact-Finders
Learning + Inference vs. Inference Based Training

- **Learning + Inference (L+I)**
  - Run the fact-finder to convergence or stop condition
  - Apply global inference (prior knowledge)
  - Works better when our prior knowledge is “noisy”
    - “Cities grow over time” isn’t always true
    - L+I avoids spreading such mistakes

- **Inference Based Training (IBT)**
  - Interleave iterations of the fact-finder with global inference
  - Works better when our prior knowledge is reliable
    - Allows fact-finder to spread corrections over subsequent iterations
Applying Prior Knowledge to Fact-Finders
Population Infobox Dataset

- (City, Population, Year) tuples
- Goal: determine true population of each city in each year
- 44,761 claims, 4,107 authors
- Common-sense: cities grow over time
- Specific knowledge: city X is larger than city Y in year Z
  - Can do even better with more prior knowledge
  - Investment $L+I$ reaches 90.91% with 10,000 such inequalities

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Prior Knowledge</th>
<th>Vote</th>
<th>Sum</th>
<th>3Est</th>
<th>TF$^s$</th>
<th>TF$^c$</th>
<th>A·L</th>
<th>Inv$^{1,2}$</th>
<th>Pool$^{1,4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pop</td>
<td>∅</td>
<td>81.49</td>
<td>81.82</td>
<td>81.49</td>
<td>82.79</td>
<td>84.42</td>
<td>80.84</td>
<td>87.99</td>
<td>80.19</td>
</tr>
<tr>
<td>Pop</td>
<td>Growth$_{IBT}$</td>
<td>82.79</td>
<td>79.87</td>
<td>77.92</td>
<td>82.79</td>
<td><strong>86.36</strong></td>
<td>80.52</td>
<td>85.39</td>
<td>79.87</td>
</tr>
<tr>
<td>Pop</td>
<td>Growth$_{L+I}$</td>
<td>82.79</td>
<td>79.55</td>
<td>77.92</td>
<td>83.44</td>
<td>85.39</td>
<td>80.52</td>
<td><strong>89.29</strong></td>
<td>80.84</td>
</tr>
<tr>
<td>Pop</td>
<td>Larger$<em>{2500}$$^1</em>{IBT}$</td>
<td>85.39</td>
<td>85.06</td>
<td>80.52</td>
<td>86.04</td>
<td>87.34</td>
<td>84.74</td>
<td><strong>89.29</strong></td>
<td>84.09</td>
</tr>
<tr>
<td>Pop</td>
<td>Larger$<em>{2500}$$^1</em>{L+I}$</td>
<td>85.39</td>
<td>85.06</td>
<td>80.52</td>
<td>86.69</td>
<td>86.69</td>
<td>84.42</td>
<td><strong>89.94</strong></td>
<td>84.09</td>
</tr>
</tbody>
</table>
Applying Prior Knowledge to Fact-Finders
Population Synthetic Datset

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Prior Knowledge</th>
<th>Vote</th>
<th>Sum</th>
<th>3Est</th>
<th>TForest</th>
<th>TFc</th>
<th>A·L</th>
<th>Inv&lt;sup&gt;1.2&lt;/sup&gt;</th>
<th>Pool&lt;sup&gt;1.4&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>SynPop</td>
<td>∅</td>
<td>73.45</td>
<td>87.76</td>
<td>84.87</td>
<td>56.12</td>
<td>87.07</td>
<td></td>
<td>90.23</td>
<td>89.41</td>
</tr>
<tr>
<td>SynPop</td>
<td>Pop±8%&lt;sub&gt;IBT&lt;/sub&gt;</td>
<td>88.31</td>
<td>95.46</td>
<td>92.16</td>
<td>96.42</td>
<td>95.46</td>
<td></td>
<td>96.15</td>
<td>95.46</td>
</tr>
<tr>
<td>SynPop</td>
<td>Pop±8%&lt;sub&gt;L+I&lt;/sub&gt;</td>
<td>88.31</td>
<td>94.77</td>
<td>92.43</td>
<td>82.39</td>
<td>95.32</td>
<td></td>
<td>95.59</td>
<td>96.29</td>
</tr>
</tbody>
</table>

- What if we had denser data?
  - More claims per source
  - Population claims for each year
- 100 (real) cities, 100 authors
  - 1 to 10 claims per city per year between 2000 and 2008
- Common-sense: no city’s population changes by more than 8% per year
  - Not always true, but more reliable than “cities always grow”
Applying Prior Knowledge to Fact-Finders

Biographies Infobox Dataset (1)

- 129,847 claimed birth dates, 34,201 death dates, 10,418 parent-child pairs, and 9,792 spouses
- Goal: determine people’s true birth/death dates
- Common-sense:
  - Nobody dies before they are born
  - Nobody born/died after 2008
  - People are infertile before the age of 7
  - Nobody lives past 125
  - All spouses have overlapping lifetimes
  - No child is born more than a year after a parent's (father's) death
  - Nobody has more than two parents

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Prior Knowledge</th>
<th>Vote</th>
<th>Sum</th>
<th>3Est</th>
<th>TF_s</th>
<th>TF_c</th>
<th>A.L</th>
<th>Inv_{1,2}</th>
<th>Pool_{1,4}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bio</td>
<td>$\emptyset$</td>
<td>89.80</td>
<td>89.53</td>
<td>89.80</td>
<td>73.04</td>
<td>90.09</td>
<td>89.24</td>
<td>88.34</td>
<td>90.01</td>
</tr>
<tr>
<td>Bio</td>
<td>CS_{IBT}</td>
<td>89.20</td>
<td>89.61</td>
<td>89.20</td>
<td>72.44</td>
<td>89.91</td>
<td>89.35</td>
<td>88.60</td>
<td>90.20</td>
</tr>
<tr>
<td>Bio</td>
<td>CS_{L+I}</td>
<td>89.20</td>
<td>89.61</td>
<td>89.20</td>
<td>57.10</td>
<td>90.09</td>
<td>89.35</td>
<td>88.49</td>
<td>90.24</td>
</tr>
</tbody>
</table>
Applying Prior Knowledge to Fact-Finders Biographies Infobox Datset (2)

- Data sparsity
  - Plenty of birthdays, fewer death dates, fewer still familial linkages
  - Limits performance benefit of common-sense on its own, but it does roughly halve convergence times

- What if we add specific knowledge?
  - X was born before Y
  - Common sense plus specific knowledge works quite well:
    - PooledInvestment (L+I)
      - 90.72% with 20,000 such pairs
      - 93.22% with 200,000

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Prior Knowledge</th>
<th>Vote</th>
<th>Sum</th>
<th>3Est</th>
<th>TFS</th>
<th>TFc</th>
<th>A.L</th>
<th>Inv^1:2</th>
<th>Pool^1:4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bio</td>
<td>∅</td>
<td>89.80</td>
<td>89.53</td>
<td>89.80</td>
<td>73.04</td>
<td>90.90</td>
<td>89.24</td>
<td>88.34</td>
<td>90.01</td>
</tr>
<tr>
<td>Bio</td>
<td>CS_{IBT}</td>
<td>89.20</td>
<td>89.61</td>
<td>89.20</td>
<td>72.44</td>
<td>89.91</td>
<td>89.35</td>
<td>88.60</td>
<td>90.20</td>
</tr>
<tr>
<td>Bio</td>
<td>CS_{L+I}</td>
<td>89.20</td>
<td>89.61</td>
<td>89.20</td>
<td>57.10</td>
<td>90.09</td>
<td>89.35</td>
<td>88.49</td>
<td>90.24</td>
</tr>
</tbody>
</table>
Applying Prior Knowledge to Fact-Finders
British vs. American Spelling (1)

- “Color” vs. “colour”: 694 such pairs
- An author claims a particular spelling by using it in an article
- Goal: find the “true” British spellings
  - British viewpoint
  - American spellings predominate by far
  - No single objective “ground truth”
- Without prior knowledge the fact-finders do very poorly
  - Predict American spellings instead
Applying Prior Knowledge to Fact-Finders

British vs. American Spelling (2)

Prior knowledge: true spelling of 100 random words

Not very effective

But what if we add common-sense?

Given spelling A, if |A| ≥ 4 and A is a substring of B, A ⇔ B

- e.g. colour ⇔ colourful

Alone, common-sense hurts performance

- Makes the system better at finding American spellings!

Need both common-sense and specific knowledge

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Prior Knowledge</th>
<th>Vote</th>
<th>Sum</th>
<th>3Est</th>
<th>TFe</th>
<th>TFe</th>
<th>A·L</th>
<th>Inv1·2</th>
<th>Pool1·4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spell</td>
<td>(\emptyset)</td>
<td>13.54</td>
<td>9.37</td>
<td>11.96</td>
<td>41.93</td>
<td>7.93</td>
<td>10.23</td>
<td>9.36</td>
<td>9.65</td>
</tr>
<tr>
<td>Spell</td>
<td>Words(^{100})(_{IBT})</td>
<td>13.69</td>
<td>9.02</td>
<td>12.72</td>
<td>44.28</td>
<td>8.05</td>
<td>9.98</td>
<td>11.11</td>
<td>8.86</td>
</tr>
<tr>
<td>Spell</td>
<td>Words(^{100})(_{L+I})</td>
<td>13.69</td>
<td>8.86</td>
<td>12.08</td>
<td>46.54</td>
<td>8.05</td>
<td>9.98</td>
<td>9.34</td>
<td>7.89</td>
</tr>
<tr>
<td>Spell</td>
<td>CS+Words(^{100})(_{IBT})</td>
<td>35.10</td>
<td>31.88</td>
<td>35.10</td>
<td>56.52</td>
<td>29.79</td>
<td>32.85</td>
<td>73.59</td>
<td>80.68</td>
</tr>
<tr>
<td>Spell</td>
<td>CS+Words(^{100})(_{L+I})</td>
<td>35.10</td>
<td>31.72</td>
<td>34.62</td>
<td>55.39</td>
<td>22.06</td>
<td>32.21</td>
<td>30.92</td>
<td>29.95</td>
</tr>
</tbody>
</table>
Conclusion

- Three new fact-finding algorithms:
  - Average-Log, Investment, PooledInvestment
  - High performance vs. existing algorithms
- New framework for incorporating prior knowledge into any fact-finder
  - Highly expressive declarative constraints
  - Tractable (polynomial time)
- Prior knowledge can almost always improve results
  - But becomes absolutely essential when the user’s judgment varies from the norm