Deploying Information Agents on the Web

Craig A. Knoblock

University of Southern California
and
Fetch Technologies
Acknowledgements

- **Information Agents Research Group**
  - Steve Minton, Fetch Tech.
  - Jose Luis Ambite, USC
  - Greg Barish, Fetch Tech.
  - Kristina Lerman, USC
  - Martin Michalowski, USC
  - Ion Muslea, SRI
  - Maria Muslea, USC
  - Sheila Tejada, UNO
  - Snehal Thakkar, USC
  - Rattapoom Tuchinda, USC

- **Electric Elves**
  - Hans Chalupsky, USC
  - Yolanda Gil, USC
  - Jean Oh, CMU
  - David V. Pynadath, USC
  - Thomas A. Russ, USC
  - Milind Tambe, USC

- **Funding**
  - DARPA
  - AFOSR
  - NSF
  - Microsoft
Introduction

- The Web is a tremendous resource, but designed for browsing
  - Sites provide limited capabilities for personalization
  - Few sites are designed to be integrated with others
- Goal: Develop technology to rapidly construct personal software agents
  - Build agents that can perform retrieval, integration, and monitoring tasks on any online source
Outline

- The Electric Elves: Information agents for monitoring travel
- Wrapping online sources
- Linking records across sources
- Efficiently executing agent plans
- Current and related work
- Conclusions
Outline

- The Electric Elves: Information agents for monitoring travel
  - Wrapping online sources
  - Linking records across sources
  - Efficiently executing agent plans
- Current and related work
- Conclusions
Electric Elves Project
[Chalupsky et al, 2001]

Elves project goal: Apply agent technology to support human organizations
- Develop software agents that automate routine tasks
- Enable software agents and humans to work together
- Support coordination of tasks

◆ Applications: Office Elves and Travel Elves
Agents for Monitoring Travel
[Ambite et al, 2002]

- Office Elves created as an application of the Electric Elves
- Given travel itinerary, generates set of agents for anticipating travel-related failures and opportunities:
  - Price changes
  - Schedule changes
  - Flight delays & cancellations
  - Earlier and close connections
  - Finding the closest restaurant given GPS coordinates
Monitoring Travel Plans

Monitoring Tasks

- **Monitor Flight Status**
  - Monitor Flights
  - Stop Monitoring
  - **Active**
    - Outbound flight 1
  - Notify Hotel (Fax)
  - 7038128516
  - Notify Car Rental Counter (Fax)
  - 7034948462

- **Monitor Flight Schedule**
  - Monitor Schedule
  - Stop Monitoring
  - **Active**
    - Status

- **Monitor Earlier Flights**
  - Monitor Earlier Flights
  - Stop Monitoring
  - **Active**
    - Status

- **Monitor Connecting Flights**
  - Monitor Connecting Flights
  - Stop Monitoring
  - **Active**
    - Status (Outbound)
  - **Active**
    - Status (Inbound)

- **Monitor Airfare**
  - Decrease only
  - Monitor Airfare
  - Stop Monitoring
  - **Active**
    - Airfare
    - Status
Agents Deployed to Monitor Travel Itinerary

- Travel Itinerary
- Flight Prices & Schedules
- Flight Status
- Weather
- Restaurants
Monitoring Agents

- **Flight-Status Agent:**
  - **Flight delayed message:**
    Your United Airlines flight 190 has been delayed. It was originally scheduled to depart at 11:45 AM and is now scheduled to depart at 12:30 PM. The new arrival time is 7:59 PM.
  - **Flight cancelled message:**
    Your Delta Air Lines flight 200 has been cancelled.
  - **Fax to hotel message:**
    Attention: Registration Desk
    I am sending this message on behalf of David Pynadath, who has a reservation at your hotel. David Pynadath is on United Airlines 190, which is now scheduled to arrive at IAD at 7:59 PM. Since the flight will be arriving late, I would like to request that you indicate this in the reservation so that the room is not given away.
Monitoring Agents

- **Airfare Agent: Airfare dropped message**
  The airfare for your American Airlines itinerary (IAD - LAX) dropped to $281.

- **Earlier-Flight Agent: Earlier flights message**
  The status of your currently scheduled flight is:
  # 190 LAX (11:45 AM) - IAD (7:29 PM) 45 minutes Late
  If you would like to return earlier, the following United Airlines flights will arrive earlier than your scheduled flights:
  # 946 LAX (8:31 AM) - IAD (3:35 PM) 11 minutes Late
  # 388 LAX (9:25 AM) - DEN (12:25 PM) 10 minutes Late
  # 1534 DEN (1:20 PM) - IAD (6:06 PM) On Time
Outline

- The Electric Elves: Information agents for monitoring travel
  - Wrapping online sources
- Linking records across sources
- Efficiently executing agent plans
- Current and related work
- Conclusions
Wrappers for Live Access to Online Sources

- HTML sources turned into agent-friendly sources

Wrapper

```html
<YAHOO_WEATHER>
  - <ROW>
  <TEMP>25</TEMP>
  <OUTLOOK>Sunny</OUTLOOK>
  <HI>32</HI>
  <LO>19</LO>
  <APPARTEMP>25</APPARTEMP>
  <HUMIDITY>35%</HUMIDITY>
  <WIND>E/10 km/h</WIND>
  <VISIBILITY>20 km</VISIBILITY>
  <DEWPOINT>9</DEWPOINT>
  <BAROMETER>959 mb</BAROMETER>
</ROW>
</YAHOO_WEATHER>
```
Extraction Rules

- Wrapper defined by a set of extraction rules
- Extraction rule: sequence of landmarks
  - Define both beginning and end of required information on the page

Name: Joel’s  <p> Phone: <i> (310) 777-1111 </i><p> Review: ...
Learning the Extraction Rules

[Muslea, Minton, & Knoblock, 01]

- Hierarchical wrapper induction
  - Decomposes a hard problem into several easier ones
  - Extracts items independently of each other
Example of Rule Induction

Training Examples:

Name: Del Taco <p> Phone (toll free) : </b> ( 800 ) 123-4567 </b> <p> <b> Cuisine ...</b>

Name: Burger King <p> Phone : ( 310 ) 987-9876 <p> Cuisine: ...
Active Learning

- Problem: May require large number of examples to achieve high accuracy
- Exploit active learning
  - System selects most informative examples to label
  - Want to achieve 100% accuracy with as few examples as possible
Which Example to Label Next

Training Examples

Name: Joel’s  Phone: (310) 777-1111  Review: The chef...
Name: Kim’s  Phone: (213) 757-1111  Review: Korean ...

Unlabeled Examples

Name: Chez Jean  Phone: (310) 666-1111  Review: ...
Name: Burger King  Phone: (818) 789-1211  Review: ...
Name: Café del Rey  Phone: (310) 111-1111  Review: ...
Name: KFC  Phone: (800) 111-7171  Review: ...
Multi-view Learning
[Muslea, Minton, Knoblock ’00]

- Two ways to find start of the phone number:
  
  SkipTo( **Phone:** )  
  BackTo( **Nmb** )

  Name: KFC  
  Phone: (310) 111-1111  
  Review: Fried chicken ...
Multi-view Learning: Co-Testing

RULE 1

Unlabeled data

Labeled data

RULE 2
<table>
<thead>
<tr>
<th>Name</th>
<th>Phone</th>
<th>Review</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joel’s</td>
<td>(310) 777-1111</td>
<td>...</td>
</tr>
<tr>
<td>Kim’s</td>
<td>(213) 757-1111</td>
<td>...</td>
</tr>
<tr>
<td>Chez Jean</td>
<td>(310) 666-1111</td>
<td>...</td>
</tr>
<tr>
<td>Burger King</td>
<td>(818) 789-1211</td>
<td>...</td>
</tr>
<tr>
<td>Café del Rey</td>
<td>(310) 111-1111</td>
<td>...</td>
</tr>
<tr>
<td>KFC</td>
<td>(800) 111-7171</td>
<td>...</td>
</tr>
</tbody>
</table>
Not All Queries are Equally Informative
Learn “content description” for item to be extracted

- Too general for extraction
  - ( Nmb ) Nmb – Nmb can’t tell a phone number from a fax number

- Useful at discriminating among query candidates

- Learned content descriptions
  - Starts with: ( Nmb )
  - Ends with: Nmb – Nmb
  - Contains: Nmb Punct
  - Length: [6,6]
Naïve & Aggressive Co-Testing

- **Naïve Co-Testing:**
  - Query: randomly chosen contention point
  - Output: rule with fewest mistakes on queries

- **Aggressive Co-Testing:**
  - Query: contention point that most violates weak view
  - Output: committee vote (2 rules + weak view)
Results for Random Sampling

- 33 most difficult of the 140 tasks from [Kushmerick ’97]
Results for Active Learning

Examples to 100% accuracy

Extraction Tasks

Naïve Co-Testing
Random sampling

Craig Knoblock
University of Southern California
Results for Active Learning with Weak Views

Extraction Tasks

Examples to 100% accuracy

- Aggressive Co-Testing
- Naïve Co-Testing
- Random sampling
Outline

- The Electric Elves: Information agents for monitoring travel
- Wrapping online sources
  - Linking records across sources
- Efficiently executing agent plans
- Current and related work
- Conclusions
## Record Linkage
(Object Consolidation)

### Zagat’s Restaurants

<table>
<thead>
<tr>
<th>Name</th>
<th>Street</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art’s Deli</td>
<td>12224 Ventura Boulevard</td>
<td>818-756-4124</td>
</tr>
<tr>
<td>Teresa’s</td>
<td>80 Montague St.</td>
<td>718-520-2910</td>
</tr>
<tr>
<td>Steakhouse The</td>
<td>128 Fremont St.</td>
<td>702-382-1600</td>
</tr>
<tr>
<td>Les Celebrites</td>
<td>155 W. 58th St.</td>
<td>212-484-5113</td>
</tr>
</tbody>
</table>

### Dept. of Health

<table>
<thead>
<tr>
<th>Name</th>
<th>Street</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art’s Delicatessen</td>
<td>12224 Ventura Blvd.</td>
<td>818/755-4100</td>
</tr>
<tr>
<td>Teresa’s</td>
<td>103 1st Ave. between 6th and 7th Sts.</td>
<td>212/228-0604</td>
</tr>
<tr>
<td>Binion’s Coffee Shop</td>
<td>128 Fremont St.</td>
<td>702/382-1600</td>
</tr>
<tr>
<td>Les Celebrites</td>
<td>160 Central Park S</td>
<td>212/484-5113</td>
</tr>
</tbody>
</table>
Active Learning to Determine Matched Records

[Tejada, Knoblock, Minton ’01,’02]

- Learn importance of attributes for matching records

<table>
<thead>
<tr>
<th>Name</th>
<th>Street</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zagat’s</td>
<td>Art’s Deli 12224 Ventura Boulevard</td>
<td>818-756-4124</td>
</tr>
<tr>
<td>Dept of Health</td>
<td>Art’s Delicatessen 12224 Ventura Blvd.</td>
<td>818/755-4100</td>
</tr>
</tbody>
</table>

Mapping rules:

- Name > .9 & Street > .87 => mapped
- Name > .95 & Phone > .96 => mapped
Mapping Rule Learner

Choose initial examples

Generate committee of learners

Learn Rules
Classify Examples
Votes

Choose Example

Set of Mapped Objects

Learn Rules
Classify Examples
Votes

Learn Rules
Classify Examples
Votes

Label

USER

Label
Committee Disagreement

- Chooses an example based on the disagreement of the query committee

<table>
<thead>
<tr>
<th>Examples</th>
<th>Committee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art’s Deli, Art’s Delicatessen</td>
<td>M1: Yes</td>
</tr>
<tr>
<td>CPK, California Pizza Kitchen</td>
<td>M2: Yes</td>
</tr>
<tr>
<td>Ca’Brea, La Brea Bakery</td>
<td>M3: Yes</td>
</tr>
<tr>
<td></td>
<td>M1: Yes</td>
</tr>
<tr>
<td></td>
<td>M2: No</td>
</tr>
<tr>
<td></td>
<td>M3: Yes</td>
</tr>
<tr>
<td></td>
<td>M1: No</td>
</tr>
<tr>
<td></td>
<td>M2: No</td>
</tr>
<tr>
<td></td>
<td>M3: No</td>
</tr>
</tbody>
</table>

- CPK, California Pizza Kitchen is the most informative example
Exploiting Secondary Sources for Record Linkage

[Michalowski, Thakkar, Knoblock ’03]

- Primary data source may be insufficient to determine mappings
- Secondary sources can help reduce the uncertainty
- Examples of secondary sources
  - Geocoder
    - Maps street addresses into lat/long coordinates
  - Business directories
    - Provide company officers and locations
  - Area code updates
    - Provide changes in area codes over time
Missing Matches

Record Linkage

Matched Records

<table>
<thead>
<tr>
<th>Data Source</th>
<th>Name</th>
<th>Address</th>
<th>City</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zagats</td>
<td>Chart House</td>
<td>13950 Panay Way</td>
<td>Marina del Rey</td>
<td>CA</td>
</tr>
<tr>
<td>Dinesite</td>
<td>Chart House</td>
<td>13950 Panay Way</td>
<td>Marina del Rey</td>
<td>CA</td>
</tr>
<tr>
<td>Zagats</td>
<td>Killer Shrimp</td>
<td>523 Washington Blvd.</td>
<td>Marina del Rey</td>
<td>California</td>
</tr>
<tr>
<td>Dinesite</td>
<td>Killer Shrimp</td>
<td>523 Washington Boulevard</td>
<td>Marina del Rey</td>
<td>California</td>
</tr>
<tr>
<td>Zagats</td>
<td>CAFE DEL REY</td>
<td>4451 Admiralty Way</td>
<td>Marina del Rey</td>
<td>CA</td>
</tr>
<tr>
<td>Dinesite</td>
<td>Cafe Del Rey</td>
<td>4451 Admiralty Way</td>
<td>Marina del Rey</td>
<td>California</td>
</tr>
<tr>
<td>Zagats</td>
<td>Koo Koo Roo</td>
<td>4325 Glencoe Ave.</td>
<td>Marina del Rey</td>
<td>CA</td>
</tr>
<tr>
<td>Dinesite</td>
<td>Koo Koo Roo</td>
<td>4325 Glencoe Avenue</td>
<td>Marina del Rey</td>
<td>California</td>
</tr>
</tbody>
</table>
Exploiting a Geocoder

Record Linkage

Matched Records

Secondary Source

26 Beach Cafe
26 Washington St.
Venice, CA

26 Beach Cafe
26 Washington Boulevard
Marina Del Rey, Calif
Preliminary Results: Secondary Sources

<table>
<thead>
<tr>
<th># Labeled Examples</th>
<th>Total Correct Matches</th>
<th>Without Secondary Source</th>
<th>With Secondary Source</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Precision</td>
<td>Recall</td>
</tr>
<tr>
<td>25</td>
<td>109</td>
<td>51%</td>
<td>33%</td>
</tr>
<tr>
<td>35</td>
<td>109</td>
<td>73%</td>
<td>57%</td>
</tr>
<tr>
<td>50</td>
<td>109</td>
<td>83%</td>
<td>81%</td>
</tr>
</tbody>
</table>

- Secondary source reduces the depth of the decision tree that needs to be learned
Outline

- The Electric Elves: Information agents for monitoring travel
- Wrapping online sources
- Linking records across sources
  - Efficiently executing agent plans
- Current and related work
- Conclusions
Efficiently Executing Agent Plans

Problem

- Information gathering may involve accessing and integrating data from many sources
- Total time to execute these plans may be large

Why?

- Slow remote sources
- Unpredictable network latencies
- Binding patterns
  - Source cannot be queried until a previous query has been answered
- Result: execution is often I/O-bound
Theseus Agent Execution System
[Barish & Knoblock, ’02]

- **Plan language** and **execution system** for Web-based information integration
- Expressive enough for monitoring a variety of sources
- Efficient enough for real-time monitoring

```
PLAN myplan {
    INPUT: x
    OUTPUT: y
    BODY {
        Op (x : y)
    }
}
```

![Diagram showing the flow of input data through the Theseus Executor to plan execution](image)
Streaming Dataflow

- Plans consist of a network of operators
  - Examples: Wrapper, Select, etc.
  - Operators produce and consume data
  - Operators “fire” upon any input data
- Data passed as tuples of a relation
Parallelism in Streaming Dataflow

**Dataflow**
- Operations scheduled by data availability
  - Independent operations execute in parallel
  - **Maximizes horizontal parallelism**
- **Example**: computing \((a*b) + (c*d)\)

**Streaming**
- Operations emit data as soon as possible
  - Independent data processed in parallel
  - **Maximizes vertical parallelism**
CarInfo Agent

Agent for recommending used cars:

- Combine information from
  - Prices of used cars
  - Safety ratings
  - Reviews

- Example:
  - 2002 Midsize coupe/hatchback
  - $4K-$12K,
  - No Oldsmobiles
The CarInfo agent

1. Locate cars that meet criteria - Edmunds.com
1. Locate cars that meet criteria
   - Edmunds.com

2. Filter out Oldsmobiles
The CarInfo agent

1. Locate cars that meet criteria
   - Edmunds.com

2. Filter out Oldsmobiles

3. Gather safety reviews for each
   - NHSTA.gov
The CarInfo agent

1. Locate cars that meet criteria
   - Edmunds.com

2. Filter out Oldsmobiles

3. Gather safety reviews for each
   - NHSTA.gov

4. Gather detailed reviews of each
   - ConsumerGuide.com
Requires navigating through multiple pages
Dataflow-style CarInfo agent plan

(Midsize coupe/hatchback, $4000 to $12000, 2002)

((Midsize coupe/hatchback, $4000 to $12000, 2002),
(Dodge Stratus),
(Pontiac Grand Am),
(Mercury Cougar))

search criteria

WRAPPER
Edmunds Search

SELECT
maker != "Oldsmobile"

WRAPPER
ConsumerGuide Search

WRAPPER
ConsumerGuide Summary

WRAPPER
ConsumerGuide Full Review

JOIN

(safety reports)

(car reviews)

((http://cg.com/summ/20812.htm),
other summary review URLs)

((http://cg.com/full/20812.htm),
other full review URLs)
Speculative Execution
[Barish & Knoblock ’02, ’03]

Basic idea
• Exploit idle resources to execute future instructions in advance of when they are normally issued

Challenges
• How to augment plans for speculation
• How to ensure correctness and fairness
• How to decide what to speculate on
How to speculate?

- General problem
  - Means for issuing and confirming predictions

- Two new operators
  - **Speculate**: Makes predictions based on "hints"
  - **Confirm**: Prevents errant results from exiting plan

```
  hints  ───> Speculate  ───> predictions/additions
        /         /                     /
answers ───> Confirm  ───> actual results
```
How to speculate?

- **Example: CarInfo**
  - Predict cars based on search criteria
  - Makes practical sense:
    - Same criteria yields same cars

BEFORE

![Diagram showing relationships between W, S, W, W, W, J, and W nodes].
How to speculate?

- Example: CarInfo
  - Predict cars based on search criteria
  - Makes practical sense:
    - Same criteria yields same cars
Detailed example

2002
Midsize coupe
$4000-$12000

W Speculate S W W J W W Confirm

Time = 0.0 sec
Issuing predictions

- Oldsmobile Alero T1
- Dodge Stratus T2
- Pontiac Grand Am T3
- Mercury Cougar T4

Time = 0.1 sec
Speculative parallelism

Time = 0.2 sec
Answers to hints

Oldsmobile Alero
Dodge Stratus
Pontiac Grand Am
Mercury Cougar

Time = 1.0 sec
Continued processing

Additions (corrections), if any

T1 → Speculate ← S → W
W → W → W → J
J → Confirm

Time = 1.1 sec
Generation of final results

Time = 3.2 sec
Confirmation of results

Time = 3.3 sec

Dodge Stratus (safety) (review)
Pontiac Grand Am (safety) (review)
Mercury Cougar (safety) (review)
Safety and fairness

- **Safety**
  - **Confirm** blocks predictions (and results of) from exiting plan before verification

- **Fairness**
  - **CPU**
    - Speculative operations use "speculative threads"
      - Lower priority threads
  - **Memory and bandwidth**
    - Speculative operations allocate "speculative resources"
      - Drawn from "speculative pool" of memory / objects
Cascading Speculation

- Use predicted cars to speculate about the ConsumerGuide summary and full URLs

- Optimistic performance
  - Execution time: $\max \{1.2, 1.4, 1.5, 1.6\} = 1.6$ sec
  - Speedup over streaming dataflow: $(4.2/1.6) = 2.63$
Automatic plan transformation

- Agent plans are automatically modified for speculative execution
  - Successive runs of the plan benefit
    - Even with different input data

- Leverage Amdahl's Law:
  - Consider optimizing only the most expensive path (MEP)

- Algorithm continually refines MEP
  - Until overhead of further optimization outweighs benefits
Learning for Speculative Execution

- **Caching**
  - Associate a hint with a predicted value
    - 2002 Midsize coupe 4K-12K
      - Olds Alero, Dodge Stratus, Pontiac Grand Am, Mercury Cougar

- **Classification**
  - Use features of a hint to predict value
    - **EXAMPLE**: Predicting car list from Edmunds

<table>
<thead>
<tr>
<th>Year</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Car list</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002</td>
<td>Midsize</td>
<td>8000</td>
<td>15000</td>
<td>(Oldmobile Alero, Dodge Stratus)</td>
</tr>
<tr>
<td>2002</td>
<td>Midsize</td>
<td>7500</td>
<td>14500</td>
<td>(Oldmobile Alero, Dodge Stratus)</td>
</tr>
<tr>
<td>2002</td>
<td>SUV</td>
<td>14000</td>
<td>20000</td>
<td>(Nissan Pathfinder, Ford Explorer)</td>
</tr>
<tr>
<td>2001</td>
<td>Midsize</td>
<td>11000</td>
<td>18000</td>
<td>(Honda Accord, Toyota Camry)</td>
</tr>
<tr>
<td>2002</td>
<td>SUV</td>
<td>18000</td>
<td>22000</td>
<td>(Nissan Pathfinder, Ford Explorer)</td>
</tr>
</tbody>
</table>

- **Decision list**
  - type = SUV : (Nissan Pathfinder, Ford Explorer)
  - type = Midsize :
    - ...min <= 10000 : (Olds Alero, Dodge Stratus)
    - min > 10000 : (Honda Accord, Toyota Camry)
Learning for Speculative Execution

- Transduction
  - Transducers are FSM that translate hints into predictions

To create full review URL:
1. Insert "http://cg.com/full/
2. Extract & insert the dynamic part of the summary URL (e.g., 20812)
3. Insert ".htm"

http://cg.com/summary/20812.htm

http://cg.com/full/20812.htm
Speculation Results: Last Tuple

The bar chart shows the time to last tuple (ms) for different information categories: CarInfo, RepInfo, TheaterLoc, FlightStatus, and StockInfo. The chart compares three scenarios:
- No speculation
- 50% correct
- 100% correct

The data indicates varying times across different information categories for each speculation scenario.
Outline of talk

- The Electric Elves: Information agents for monitoring travel
- Wrapping online sources
- Linking records across sources
- Efficiently executing agent plans
- Current and related work
- Conclusions
Planning to Compose Web Services
[Thakkar, Knoblock, & Ambite, ’03]

- **Goal:** Automatically compose new services from existing web services
- **We developed services** that can dynamically compose information producing services
  - Builds on data integration techniques to construct plans
  - Turns the plans into Theseus plans for efficient execution
- **We are extending this work** to more complex services that can change the world (side effects)
Learning to Make Predictions: To Buy or Not To Buy

- Agents can go beyond gathering and monitoring online sources
  - They can help make decisions by exploiting the wealth of online information

American Airlines flights 192 & 223, LAX-BOS, departing on Jan. 2 & 9
Learning to Make Predictions: To Buy or Not To Buy

- Agents can go beyond gathering and monitoring online sources
  - They can help make decisions by exploiting the wealth of online information
- Developed a learning system, Hamlet, to predict whether it is better to wait or buy [Etzioni, Knoblock, Tuchinda, Yates, KDD’03]
- Collected data on airline prices over several months
- Learned a model of the pricing
- In our simulation on collected data, Hamlet saved $198,074 out of a possible $320,572 (61.8% of optimal)
Related Agent Systems

- Some notable deployed systems
  - Internet Softbot [Etzioni & Weld, ‘94]
  - BargainFinder [Krulwich, ‘96]
  - ShopBot [Perkowitz et al. ‘96]
  - Warren [Decker et al., ‘97]
  - Electric Elves [Chalupsky et al., ‘01]
  - and many others…
Related Work

- **Wrapper learning**
  - Supervised [Kushmerick ‘97, Hsu & Dung ‘98]
  - Unsupervised [Lerman et al. ’01, Crescenzi ’01]

- **Record linkage**
  - Learning [Cohen ’00, Sarawagi & Bhamidipaty ’02]
  - Statistics [Winkler ’98]
  - Name matching [Bilenko et al. ’03, Cohen et al. ’03]

- **Efficient plan execution**
  - Network query engines [Ives et al. 1999, Naughton et al. 2000, Hellerstein et al. 2001]
  - Agent execution systems [Firby ’94, Myers et al. 1996]
Outline of talk

- The Electric Elves: Information agents for monitoring travel
- Wrapping online sources
- Linking records across sources
- Efficiently executing agent plans
- Current and related work

• Conclusions
Conclusions

- Web provides the ideal environment for developing and testing software agents
  - Noted by Etzioni, AAAI’96 in his talk on Softbots
- Yet few have seized this opportunity...why?
- Like robotics, wide variety of hard technical problems
- With Web Services, the Semantic Web, etc. the infrastructure is improving
- Great opportunity for AI
  - Ability to demonstrate and test technologies in a real-world setting
  - Opportunity to apply technologies to make a difference in people’s lives
Conclusions (cont.)

Many interesting technical challenges for building software agents:

- Wrapping online sources
- Linking records across sites
- Efficiently executing agent plans
- Extraction from text documents
- Aligning ontologies across sources
- Planning to integrate data sources
- Learning to improve performance and capabilities
- Integrating these capabilities in a robust architecture that can:
  - Respond to failures
  - Explain its behavior
  - Communicate appropriately
More Information

- My home page: http://www.isi.edu/~knoblock

- IJCAI’03 Workshop on Information Integration on the Web
  - Proceedings available online (pointer from my homepage)
The End