Beyond the Elves: Making Intelligent Agents Intelligent

Craig Knoblock

University of Southern California
Information Sciences Institute
Contributors

- José Luis Ambite, USC
- Greg Barish, Fetch Technologies
- Hans Chalupsky, USC
- Yolanda Gil, USC
- Kristina Lerman, USC
- Steven Minton, Fetch Technologies
- Maria Muslea, USC
- Jean Oh, CMU
- David V. Pynadath, USC
- Thomas A. Russ, USC
- Paul Scerri, USC
- Milind Tambe, USC
Research Support

Electric Elves
- DARPA CoABS program
- DARPA Active Templates program

Research based on lessons learned
- DARPA PAL Program
- AFOSR
Outline

- The Electric Elves
  - Overview
- The Travel Elves
  - Constraint-based data integration
  - Efficient plan execution
- Beyond the Elves
  - Deployment to DARPA
  - Some lessons
    - Learning expectations
    - Finding alternative sources
    - Predicting prices
- Discussion
Agent-Facilitated Human Organizations
Goal

Agent-Facilitated Human Organizations

Develop software agents that automate routine tasks within organization
- E.g., obtaining info, monitoring, distributing information

Enable software agents and humans to act coherently within the organization
- Humans have agent proxies that assist in coordinating with other agents

Coordination of tasks within the organization
- Efficient use of resources (physical and human)
- Monitoring progress of both individual and overall tasks
- Suggest/execute corrective actions when goals appear endangered
Example Tasks

Supporting our own research organization:

- Coordinating routine project activities
  - Eliminate email about meeting delays/postponements
  - Eliminate email about who will present at next meeting
  - Track visitor’s flights and arrival times

- Support for external meetings
  - Monitor all aspects of a traveler’s plans
    - Notify traveler of prices changes, schedule changes, flight delays
    - Send fax to hotel if traveler is delayed
  - Organize meetings for people with similar research interests in other organizations
  - Organize meeting at external site (e.g., a restaurant)

- Similar issues arise in many organizations: corporations, military, etc.
Agent Organization is Grounded in the Real World

- **On-line information sources**
  - Calendar and schedule information
  - Flight schedules, restaurants, etc.

- **“Sensors” and “actuators”**
  - GPS, “finger”, infrared, etc.
  - Postpone/cancel meetings, faxes to hotels and restaurants, etc.

- **Interaction with people**
  - Communicate through personal portable devices (PDAs, phones, etc.)
Summary

Electric Elves: successful use of agent technology to support human organization

- Interact with their environment:
  - Data: on-line information sources, calendars, GPS, etc.
  - Interaction with people: faxes, email, PDAs, phones, etc.

- Support routine activities
  - Arrange meetings, lunch orders
  - Monitoring and suggesting/executing corrective actions

Integrates powerful techniques in diverse areas within AI:

- Teamwork, human proxies [Teamcore]
- Ontologies & problem solving [Expect]
- Information agents [Ariadne]
- Knowledge representation & information retrieval [PowerLoom]
Outline

- The Electric Elves
  - Overview
- The Travel Elves
  - Constraint-based data integration
  - Efficient plan execution
- Beyond the Elves
  - Deployment to DARPA
  - Some lessons
    - Learning expectations
    - Finding alternative sources
    - Predicting prices
- Discussion
Interactive Trip Planning

- Commercial systems provide support to select flights, hotels and cars
  - Integrates the planning at the level of dates and locations

- There are many more factors involved in planning a trip
  - Which airports to fly into and out of
  - Whether to drive or take a taxi to the airport
  - How to get from the airport to the destination
  - Proximity of hotel to meeting
  - Etc…

- Ideally a system will
  - Provide all of the data required to make these decisions
  - Provide a way to consider the tradeoffs of the various choices
Heracles Constraint-based Planning

- Framework for building integrated applications
- Extract and integrate data for a given task
  - Live access to online sources using the wrappers
- Constraint-based system decides what sources to query and how to integrate the results
  - Tight integration of user choices
The Travel Assistant
Supports Informed Choices

Fly

From
2700 University Park
Street
Los Angeles
City
CA
State

To
1120 19th ST NW
Street
Washington
City
DC
State

Getting to Airport

Parking
Terminal Parking
Daily Rate (dollars)

24.00

Duration (days)

2

Total (dollars)

48

Taxi
Distance

12.7

19.50

Mode to
Airport
Take a Taxi
Click to Expand

Itinerary
LAX
IAD
Apr
19

From
To
Month
Day
**Supports Informed Choices**

### Fly

#### From
- Street: 2700 University Park
- City: Los Angeles
- State: CA

#### To
- Street: 1120 19th ST NW
- City: Washington
- State: DC

#### Getting to Airport
- **Parking**
  - Lot: 48
  - Daily Rate (dollars): 24.00

- **Taxi**
  - Distance: 12.7
  - Taxi fare (dollars): 19.50

- **Mode to Airport**
  - Click to Expand

#### Itinerary
- **Flights**
  - LAX
  - IAD
  - Month: Apr
  - Day: 19

### Other Flights

**Economy Lot B**
- Lot: 10
- Daily Rate (dollars): 5.00
- Duration (days): 2
- Total (dollars): 10

**Drive**
- Click to Expand

**Flights**
- LAX
- IAD
- Month: Apr
- Day: 19
Changes Propagate Throughout

Taxi

Leaving From

2700 University Park

Los Angeles

CA

Driving To

LAX

Los Angeles

CA

Suggested Departure

Apr

18

2002

10:31 PM

Predicted Arrival

Apr

18

2002

10:55 PM

Taxi fare

19.50

Total Drive

12.7

0

24

Maps

Yahoo! Map 1

Yahoo! Map 2
Changes Propagate Throughout

**Round Trip Flights**

- **Preference**: Lowest Price
- **Departs**:
  - Code: LAX
  - City, State: Los Angeles, CA
  - Month: Apr
  - Day: 19

- **Returns**:
  - Code: LAX
  - City, State: Wash, DC/Dulles, DC
  - Month: Apr
  - Day: 20

- **Price**:
  - Default: $192

- **Outbound Flight 1**
  - Time: 11:35 PM
  - Depart: LAX
  - Arrive: LAX

- **Total Drive**:
  - Dist: 12.7
  - Time: 24 mins

Maps are also shown, displaying a route from one location to another.
Changes Propagate Throughout
Constraint Network: Drive or Taxi?

- DepartureDate (Mar 15, 2001)
- ReturnDate (Mar 18, 2001)
- DestinationAddress
- OriginAddress
- DepartureAirport (LAX)
- getDistance (15.1 miles)
- findClosestAirport
- getTaxiFare ($23.00)
- getParkingRate ($16.00/day)
- parkingTotal ($64.00)
- ModeToAirport (Taxi)
- computeDuration (4 days)
- Duration
- multiply

selectModeToAirport
Summary

- Integration of wide range of data from many different sources
- Tight integration of data using constraints to capture the dependencies
- Supports better decision making
  - Easy to consider costs of specific choices
  - Easy to compare tradeoffs
Outline

- The Electric Elves
  - Overview

- The Travel Elves
  - Constraint-based data integration
  - Efficient plan execution

- Beyond the Elves
  - Deployment to DARPA
  - Some lessons
    - Learning expectations
    - Finding alternative sources
    - Predicting prices

- Discussion
Agents for Monitoring Travel

Many opportunities and possible problems can arise during travel.

Current environment:
- Wide access to data
- Abundance of computer resources
- Availability of cell phones and portable computers

Makes it possible to monitor all aspects of a trip.

Create personal assistants that monitor your travel plan to:
- exploit opportunities
- avoid problems
# Monitoring Travel Plans

## Monitoring Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Status</th>
<th>Monitor Flight Status</th>
<th>Monitor Schedule</th>
<th>Monitor Earlier Flights</th>
<th>Monitor Connecting Flights</th>
<th>Monitor Airfare</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitor Flights</td>
<td>Active</td>
<td>On</td>
<td></td>
<td></td>
<td></td>
<td>Increase only</td>
</tr>
<tr>
<td>Stop Monitoring</td>
<td></td>
<td>On</td>
<td></td>
<td></td>
<td></td>
<td>Stop Monitoring</td>
</tr>
<tr>
<td>Notify Hotel (Fax)</td>
<td></td>
<td>On</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notify Car Rental Counter (Fax)</td>
<td></td>
<td>On</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outbound flight 1</td>
<td>Active</td>
<td>On</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outbound flight 2</td>
<td>Active</td>
<td>On</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inbound flight 1</td>
<td>Active</td>
<td>On</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inbound flight 2</td>
<td>Active</td>
<td>On</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Monitor Airfare</td>
<td></td>
<td>Off</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Decrease only</td>
<td></td>
<td>Off</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**Note:**
- Monitor Flight Status options are 'On' or 'Off'.
- Monitor Schedule status is 'Active'.
- Monitor Earlier Flights status is 'Active'.
- Monitor Connecting Flights status is 'Active'.
- Monitor Airfare status is 'Off' and can be changed to 'Active'.
- Mode options are 'Increase only' and 'Decrease only'.
Agents Deployed to Monitor Travel Itinerary

Travel Itinerary

GRID

Flight Prices & Schedules
Flight Status
Restaurants
Weather
Example 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
Theseus

A plan language and execution system for building Web-based information-gathering and monitoring agents [Barish & Knobock, JAIR’05]

- *Efficient* enough for near-real-time monitoring: streaming dataflow
- *Expressive* enough for integrating a variety of sources (web sites, XML, databases, …)
Streaming Dataflow

Plans consist of a network of operators

- **Examples**: Wrapper, Select, etc.
- Operators produce and consume data
- Operators “fire” upon any input data

**Input relation**

<table>
<thead>
<tr>
<th>City</th>
<th>State</th>
<th>Max Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Santa Monica</td>
<td>CA</td>
<td>200000</td>
</tr>
</tbody>
</table>

**Plan**

- Wrapper
- Select
- Join

**Output relation**

<table>
<thead>
<tr>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 Main St., Santa Monica, 90292</td>
</tr>
<tr>
<td>520 4th St. Santa Monica, 90292</td>
</tr>
<tr>
<td>2 Ocean Blvd, Venice, 90292</td>
</tr>
</tbody>
</table>
Theseus: Efficiency Streaming Dataflow

Dataflow-style execution
- Operators execute when inputs become available
- Optimizes *horizontal parallelism*
  - Plan is as parallel as its data dependencies allow

Data Streaming
- Data in the system represented as *relations*
  - Producer operators pipeline *tuples* to consumers
- Optimizes *vertical parallelism*
  - Multiple operators can work on same relation concurrently
Theseus Monitoring Agent: Flight Status Plan

flight #
connecting flight #

WRAPPER
ITN

WRAPPER
Timezone

DBQUERY
SELECT old_depart

COMPARE
|old_depart - curr_depart| > threshold

EMAIL
user@isi.edu

UN_SCHEDULE

EMAIL
user@isi.edu

SCHEDULE
Flight-connection agent

SELECT
status=departed

SELECT
status=pending

SELECT
status=arrived

SELECT
status=cancelled

SELECT
Arrive > 5 PM

DBQUERY
UPDATE old_depart=curr_depart

FAX
Hotel

FAX
Car rental agency
Outline

- The Electric Elves
  - Overview
- The Travel Elves
  - Constraint-based data integration
  - Efficient plan execution
- Beyond the Elves
  - Deployment to DARPA
  - Some lessons
    - Learning expectations
    - Finding alternative sources
    - Predicting prices
- Discussion
Deploying the Elves at DARPA

The office elves were successful and received a lot of press.

DARPA decided they wanted a version running in their environment.

But there were issues:
- Integration into the DARPA environment
- Privacy concerns
- Software integration
- …
The Travel Elves

So we created the Travel Elves
- No integration into the DARPA environment
- Could run on ISI servers
- Communicate via PDAs

Trained the DARPA travel person to enter travel data using Heracles

System deployed…
- Used by program managers and office directors
- …for a while
What Went Wrong and Why?

- **Agent failures due to problems with the data**
  - “Why didn’t I receive any updates from the Elves?”
  - Source changed, unexpected data, source unavailable

- **Agents failures due to problems in the logic**
  - “Why did the message arrive after my flight left?”
  - Time zone exception

- **Unexpected behaviors**
  - “Why are the Elves pestering me with price changes?”
  - Frequent price changes

- **Getting the definition “right”**
  - “Those !*?&%* Elves sent a message to my cell phone at 3am to tell me my 8am flight was delayed!”
  - Agent worked as designed…

- **Personalized agents**
  - “Could you build an agent to do <X>?”
  - Every user wants something different
Outline

The Electric Elves
  - Overview

The Travel Elves
  - Constraint-based data integration
  - Efficient plan execution

Beyond the Elves
  - Deployment to DARPA
  - Some lessons
    - Learning expectations
    - Finding alternative sources
    - Predicting prices

Discussion
Learning Data Prototypes
[Lerman et al., JAIR 2003]

- Approach to learning the structure of data
- Token-level syntactic description
  - descriptive but compact
  - computationally efficient
- Structure is described by a sequence (pattern) of general and specific tokens.
- Data prototype = starting & ending patterns

**STREET_ADDRESS**
- start with: _NUM _CAPS
- end with: _CAPS Blvd

220 Lincoln Blvd
420 S Fairview Ave
2040 Sawtelle Blvd

_CODES _CAPS
Token Syntactic Hierarchy

- Tokens = words
- Syntactic types
  - e.g., NUMBER, ALPHA
- Hierarchy of types allows generalization
- Extensible
  - new types
  - domain-specific information
Prototype Learning Algorithm

- No explicit negative examples
- Learn from positive examples of data
- Find patterns that
  - describe many of the positive examples of data
  - highly unlikely to describe a random token sequence (implicit negative examples)
- are statistically significant patterns at $\alpha=0.05$ significance level
- **DataPro** – efficient (greedy) algorithm
Outline

The Electric Elves
  ➢ Overview

The Travel Elves
  ➢ Constraint-based data integration
  ➢ Efficient plan execution

Beyond the Elves
  ➢ Deployment to DARPA
  ➢ Some lessons
    – Learning expectations
    – Finding alternative sources
    – Predicting prices

Discussion
Learning New Sources of Data
[Carman & Knoblock, 2005]

Source Definitions:
- United
- Lufthansa
- Qantas

Find cheapest Flights between MXP and PIT

Query

Mediator

United
Lufthansa
Qantas
Alitalia
new service

Web Services
Learning New Sources of Data
[Carman & Knoblock, 2005]

Source Definitions:
- United
- Lufthansa
- Qantas

Mediator

Query

Find cheapest Flights between MXP and PIT

Reformulated Query

lowestFare(“MXP”, “PIT”)
calcPrice(“MXP”, “PIT”, ”economy”)

United
Web Services
Lufthansa
Qantas
Alitalia
new service
Learning New Sources of Data
[Carman & Knoblock, 2005]

- Need source definitions to incorporate new data
- Time consuming and difficult to write these descriptions
- Can we discover definitions automatically?

Source Definitions:
- United
- Lufthansa
- Qantas

Find cheapest Flights between MXP and PIT

Query

Mediator

Reformulated Query

lowestFare("MXP","PIT")

calcPrice("MXP","PIT","economy")

Web Services

United

Lufthansa

Qantas

Alitalia

ew service
The Framework

**Intuition**: Services often have similar semantics, so we should be able to use what we know to induce that which we don’t

**Two phase algorithm**

For each operation provided by the new service:

1. Classify its input/output data types
   - Classify inputs based on metadata similarity
   - Invoke operation & classify outputs based on data

2. Induce a source definition
   - Generate candidates via Inductive Logic Programming
   - Test individual candidates by reformulating them
Inducing Source Definitions: A Simple Example

Semantic Types:
currency ⊆ \{USD, EUR, AUD\}
rate ⊆ \{1936.2, 1.3058, 0.53177\}

Predicates:
exchange(currency, currency, rate)

RateFinder($fromCountry, $toCountry, val):- ?
LatestRates($country1, $country2, rate):-
exchange($country1, $country2, rate)
Inducing Source Definitions: A Simple Example

Step 1: use metadata to classify input types

Mediator

Semantic Types:
currency ⊆ {USD, EUR, AUD}
rate ⊆ {1936.2, 1.3058, 0.53177}

Predicates:
exchange(currency, currency, rate)

LatestRates($country1, $country2, rate):-
exchange($country1, $country2, rate)

RateFinder($fromCountry, $toCountry, val):- ?
Inducing Source Definitions: A Simple Example

- Step 1: use metadata to classify input types
- Step 2: invoke service and classify output types

Mediator

Semantic Types:
currency ⊆ {USD, EUR, AUD}
rate ⊆ {1936.2, 1.3058, 0.53177}

Predicates:
exchange(currency,currency,rate)

RateFinder($fromCountry,$toCountry,val):- ?
{<EUR,USD,1.30799>, <USD,EUR,0.764526>, …}

LatestRates($country1,$country2,rate):-
exchange($country1,$country2,rate)
Inducing Source Definitions: A Simple Example

- Step 3: generate plausible source definitions

**Predicates:**
- exchange(currency,currency,rate)

**Mediator**

**Diagram:**
- New source:
  - `currency`
  - `rate`

- Rule:
  - `RateFinder($fromCountry,$toCountry,val):- ?`
Inducing Source Definitions: A Simple Example

- Step 3: generate plausible source definitions

Predicates:
exchange(currency,currency,rate)

Mediator

RateFinder($fromCountry,$toCountry,val):- ?

def_1($from, $to, val) :- exchange(from,to,val)
def_2($from, $to, val) :- exchange(to,from,val)
Inducing Source Definitions: A Simple Example

- Step 3: generate plausible source definitions
- Step 4: reformulate in terms of other sources

Predicates:

- exchange(currency, currency, rate)

Mediator

<table>
<thead>
<tr>
<th>New Source</th>
<th>RateFinder($fromCountry,$toCountry,val):-</th>
</tr>
</thead>
<tbody>
<tr>
<td>currency</td>
<td></td>
</tr>
<tr>
<td>rate</td>
<td></td>
</tr>
</tbody>
</table>

| def_1($from, $to, val) :- exchange(from,to,val) |
| def_2($from, $to, val) :- exchange(to,from,val) |

| def_1($from, $to, val) :- LatestRates(from,to,val) |
| def_2($from, $to, val) :- LatestRates(to,from,val) |
Inducing Source Definitions: A Simple Example

- Step 3: generate plausible source definitions
- Step 4: reformulate in terms of other sources
- Step 5: invoke service and compare output

**Predicates:**
- exchange(currency, currency, rate)

<table>
<thead>
<tr>
<th>Input</th>
<th>RateFinder</th>
<th>Def_1</th>
<th>Def_2</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;EUR,USD&gt;</td>
<td>1.30799</td>
<td>1.30772</td>
<td>0.764692</td>
</tr>
<tr>
<td>&lt;USD,EUR&gt;</td>
<td>0.764526</td>
<td>0.764692</td>
<td>1.30772</td>
</tr>
<tr>
<td>&lt;EUR,AUD&gt;</td>
<td>1.68665</td>
<td>1.68979</td>
<td>0.591789</td>
</tr>
</tbody>
</table>

**New Source:**
def_1($from, $to, val) :- exchange(from, to, val)
def_2($from, $to, val) :- exchange(to, from, val)
def_1($from, $to, val) :- LatestRates(from, to, val)
def_2($from, $to, val) :- LatestRates(to, from, val)

**Inducing Source Definitions: A Simple Example**

- Step 3: generate plausible source definitions
- Step 4: reformulate in terms of other sources
- Step 5: invoke service and compare output

**Predicates:**
- `exchange(currency,currency,rate)`

**Input | RateFinder | Def_1 | Def_2**
--- | --- | --- | ---
EUR,USD | 1.30799 | 1.30772 | 0.764692
USD,EUR | 0.764526 | 0.764692 | 1.30772
EUR,AUD | 1.68665 | 1.68979 | 0.591789

**New Source**
- `newSource(fromCountry,$toCountry,val):-` ?
- `def_1($from, $to, val) :- exchange(from,to,val)`
- `def_2($from, $to, val) :- exchange(to,from,val)`
- `def_1($from, $to, val) :- LatestRates(from,to,val)`
- `def_2($from, $to, val) :- LatestRates(to,from,val)`
Outline

- The Electric Elves
  - Overview

- The Travel Elves
  - Constraint-based data integration
  - Efficient plan execution

- Beyond the Elves
  - Deployment to DARPA
  - Some lessons
    - Learning expectations
    - Finding alternative sources
    - Predicting prices

- Discussion
Predicting Airline Prices
[Etzioni et al., KDD 2003]

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

- Collected airline flight data over several months
- Developed a learning algorithm to predict whether to buy immediately or wait to buy a ticket
- Exploits the fact that airline pricing is done with a relatively static, but unknown algorithm
- Pricing can be learned by considering the pricing on the same flight on previous days
Data Set

- Extracted data from online sources using wrappers
- Collected over 12,000 price observations:
  - Lowest available fare for a one-week roundtrip
  - LAX-BOS and SEA-IAD
  - 6 airlines including American, United, etc.
  - 21 days before each flight, every 3 hours
Learning Algorithm

Stacking with three base learners:
1. Rule learning (Ripper) (e.g., R=wait)
2. Time series
3. Q-learning (e.g., Q=buy)

Ripper used as the meta-level learner.
Output: classifies each decision point as ‘buy’ or ‘wait’.
Savings by Method

- **Net** savings = cost now – cost at purchase point.
- Penalty for sell out = upgrade cost. 0.42% of the time.
- Total ticket cost is $4,579,600.

![Net Savings by Method](chart)

Legend:
- **Time Series**
- **Q-Learning**
- **By Hand**
- **Ripper**
- **Hamlet**
- **Optimal**
Status

- Patent granted on predictive pricing of air fares based on historical data
- Technology licensed to Hamlet, Inc. started by Oren Etzioni
- Company raised $7M in VC funding and is now called Farecast.
- Stay tuned…
Discussion

We need to build agents that:

- Robustly accomplish their tasks, responding appropriately to failures
- Learn from their past experience
- Rapidly build personalized agents without manual programming
- Communicate flexibly with humans and software agents
- Explain their behavior both on success or failure
- Dynamically compose new agents and behaviors from existing agents