

Automatic Source Modeling

Jose Luis Ambite
University of Southern California

Based on slides by Mark Carman, Craig Knoblock
Jose Luis Ambite and Kristina Lerman



Source Modeling vs. Schema Matching

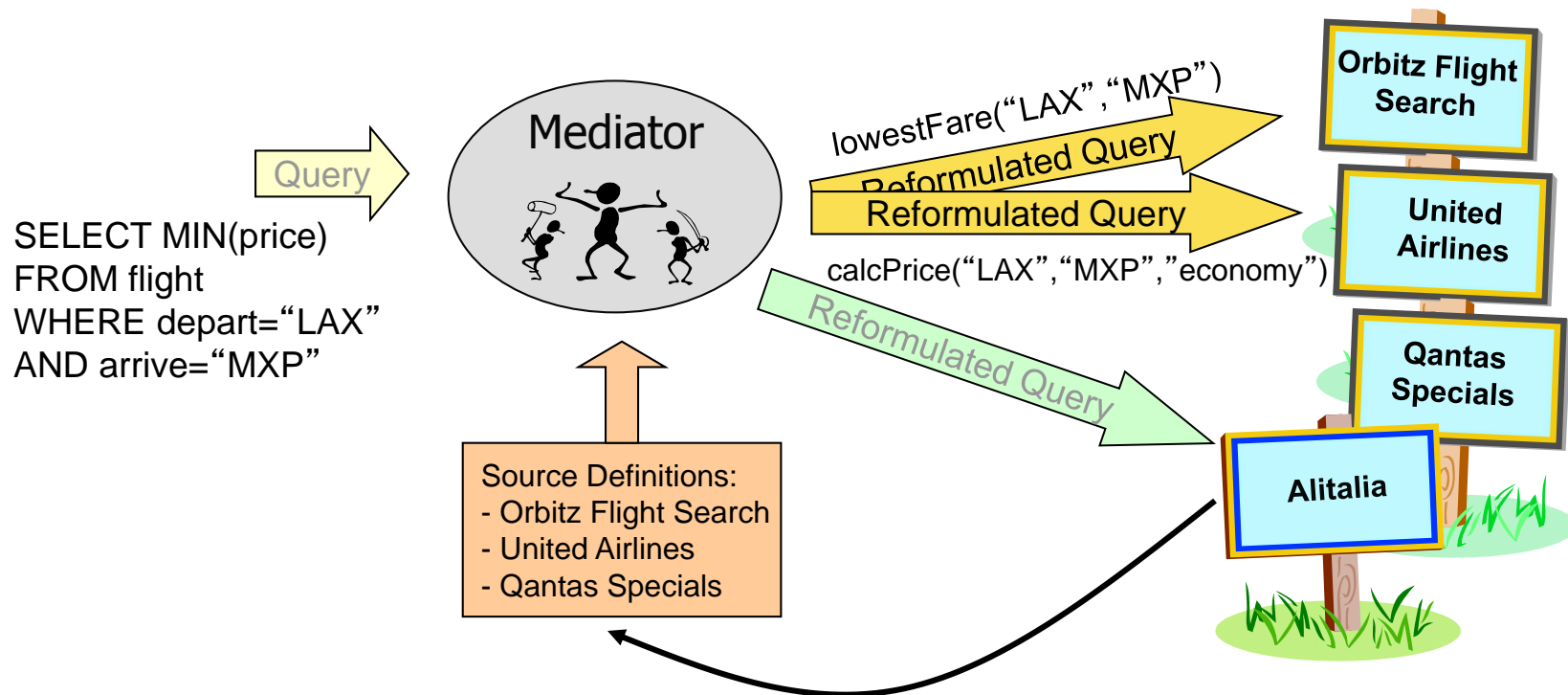
- Schema Matching/Mapping
 - Align schemas between data sources
 - Assumes static sources and complete access to data
 - Schema Mapping does provide queries to map between sources
- Source Modeling
 - Incrementally build models from partial data (e.g., web services, html forms, programs)
 - Model not just the fields but the source types and even the function of a source
 - Support richer source models (a la Semantic Web) that describe the contents of sources

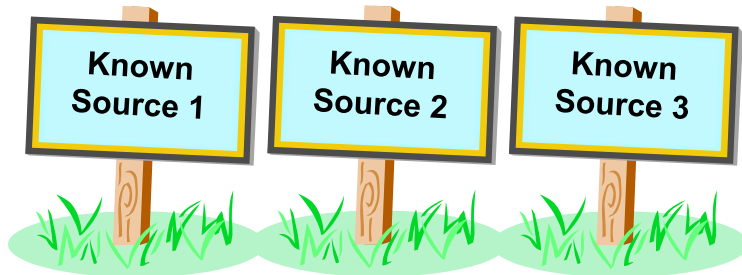
Overview

- **Learning Semantic Descriptions of Web Information Sources**
 - Integrated approach to learning semantic definitions of a source
 - Needs to invoke the source
 - Learn Datalog description of the source: LAV mapping
- **Automatically Constructing Semantic Web Services from Online Sources**
 - Deimos: End to end system for discovering and modeling sources
 - Uses system above to automatically discover semantic definitions of discovered sources

Mediators Require Source Definitions

- New service => no source definition!
- Can we discover a definition automatically?



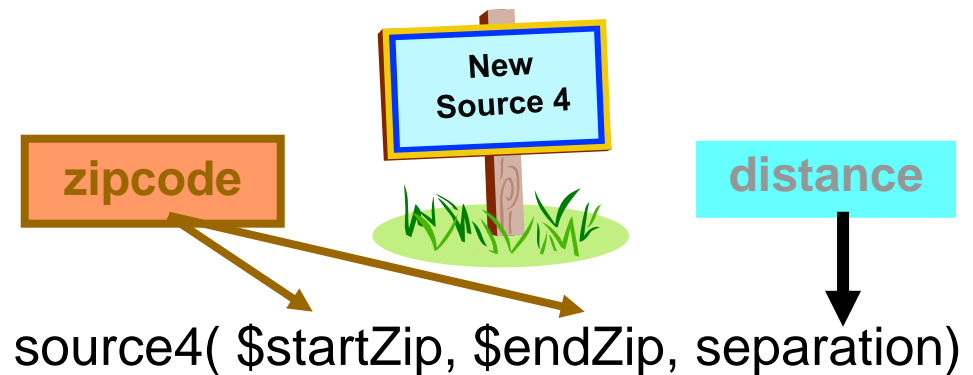


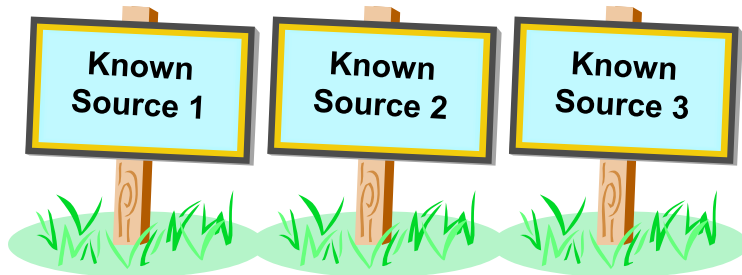
```
source1($zip, lat, long) :-  
    centroid(zip, lat, long).
```

```
source2($lat1, $long1, $lat2, $long2, dist) :-  
    greatCircleDist(lat1, long1, lat2, long2, dist).
```

```
source3($dist1, dist2) :-  
    convertKm2Mi(dist1, dist2).
```

- Step 1: classify input & output semantic types





- Step 1: classify input & output semantic types
- Step 2: generate plausible definitions

```
source1($zip, lat, long) :-  
    centroid(zip, lat, long).
```

```
source2($lat1, $long1, $lat2, $long2, dist) :-  
    greatCircleDist(lat1, long1, lat2, long2, dist).
```

```
source3($dist1, dist2) :-  
    convertKm2Mi(dist1, dist2).
```

```
source4($zip1, $zip2, dist):-  
    source1(zip1, lat1, long1),  
    source1(zip2, lat2, long2),  
    source2(lat1, long1, lat2, long2, dist2),  
    source3(dist2, dist).
```


```
source4($zip1, $zip2, dist):-  
    centroid(zip1, lat1, long1),  
    centroid(zip2, lat2, long2),  
    greatCircleDist(lat1, long1, lat2, long2, dist2),  
    convertKm2Mi(dist1, dist2).
```

- Step 1: classify input & output semantic types
- Step 2: generate plausible definitions
- Step 3: invoke service & compare output

```
source4($zip1, $zip2, dist):-  
  source1(zip1, lat1, long1),  
  source1(zip2, lat2, long2),  
  source2(lat1, long1, lat2, long2, dist2),  
  source3(dist2, dist).
```

```
source4($zip1, $zip2, dist):-  
  centroid(zip1, lat1, long1),  
  centroid(zip2, lat2, long2),  
  greatCircleDist(lat1, long1, lat2, long2, dist2),  
  convertKm2Mi(dist1, dist2).
```

match



\$zip1	\$zip2	dist (actual)	dist (predicted)
80210	90266	842.37	843.65
60601	15201	410.31	410.83
10005	35555	899.50	899.21

Searching for Definitions

- Search space of *conjunctive queries*:

$\text{target}(X) \text{ :- source1}(X_1), \text{source2}(X_2), \dots$

- For scalability don't allow negation or union
- Perform Top-Down Best-First Search

Expressive Language
Sufficient for modeling
most online sources

1. First sample the
New Source

Invoke **target** with set of random inputs;
Add empty clause to **queue**;

while (**queue** not empty)

$v :=$ best definition from **queue**;

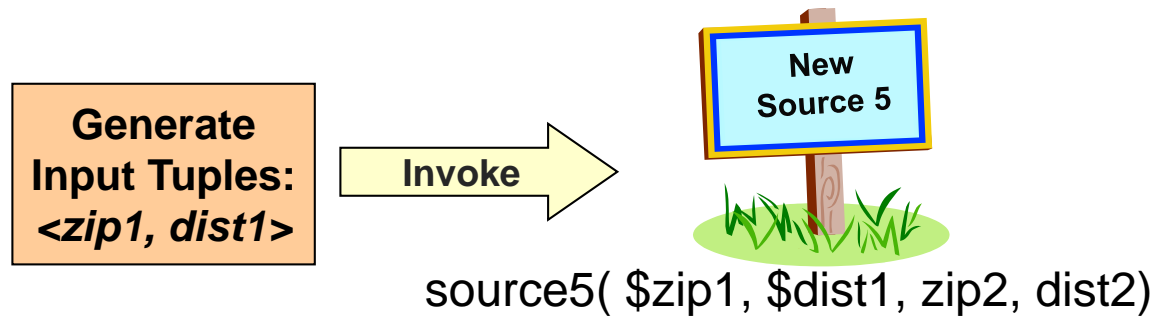
forall (v' in **Expand**(v))

if (**Eval**(v') > **Eval**(v))

insert v' into **queue**;

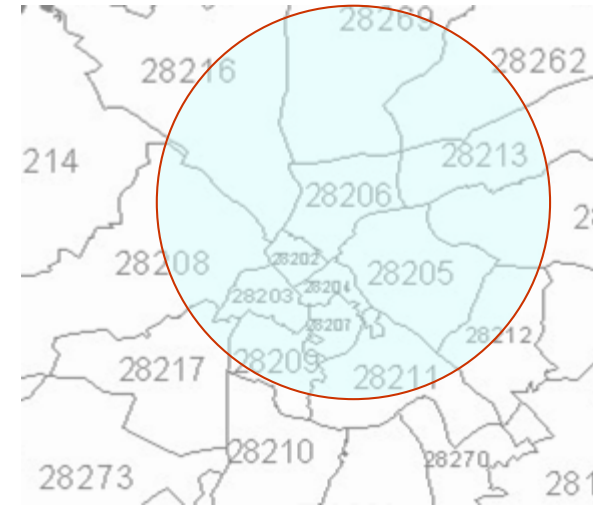
2. Then perform best-first
search through space of
candidate definitions

Invoking the Target



Invoke source with *representative* values

- Try randomly generating input tuples:
 - Combine examples of each type
 - Use distribution if available



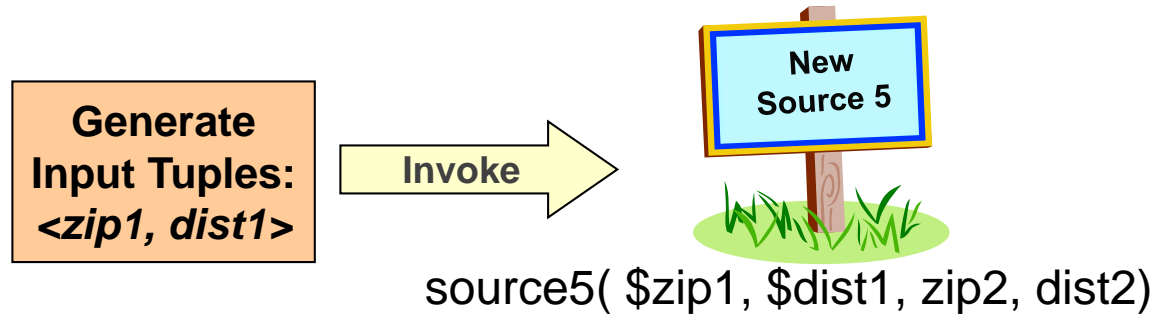
Randomly Combined Example Values

<u>Input</u> $\langle \text{zip1}, \text{dist1} \rangle$	<u>Output</u> $\langle \text{zip2}, \text{dist2} \rangle$
$\langle 07307, 50.94 \rangle$	$\{ \langle 07097, 0.26 \rangle, \langle 07030, 0.83 \rangle, \langle 07310, 1.09 \rangle, \dots \}$
$\langle 60632, 10874.2 \rangle$	$\{ \}$

Non-empty Result

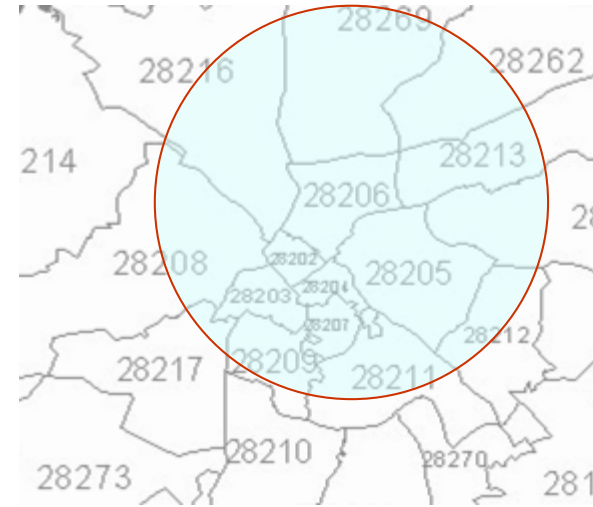
Empty Result

Invoking the Target



Invoke source with *representative* values

- Try randomly generating input tuples:
 - Combine examples of each type
 - Use distribution if available
- If *only empty invocations* result
 - Try *invoking other sources* to generate input
- Continue until sufficient non-empty invocations result



Top-down Generation of Candidates

Start with empty clause & generate specialisations by

- Adding one predicate at a time from set of sources
- Checking that each definition is:
 - Not logically redundant
 - Executable (binding constraints satisfied)



source5(_,_,_,_).

Expand

source5(\$zip1,\$dist1,zip2,dist2)

```
source5(zip1,_,_,_)      :- source4(zip1,zip1,_).  
source5(zip1,_,zip2,dist2) :- source4(zip2,zip1,dist2).  
source5(_,dist1,_,dist2) :- <(dist2,dist1).  
...
```

Best-first Enumeration of Candidates

- Evaluate each clause produced
- Then expand best one found so far
- Expand high-arity predicates incrementally

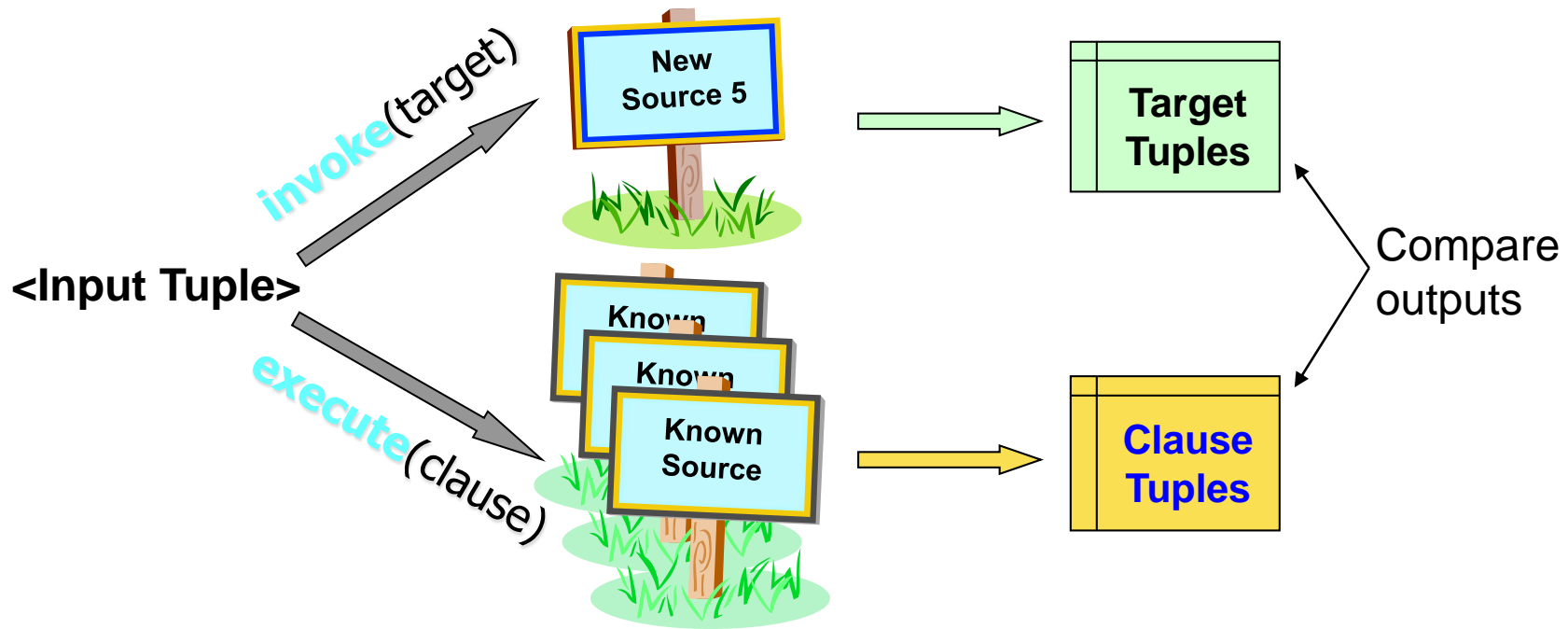


```
source5(zip1,_,zip2,dist2) :- source4(zip2,zip1,dist2).
```



```
source5(zip1,dist1,zip2,dist2) :- source4(zip2,zip1,dist2), source4(zip1,zip2,dist1).  
source5(zip1,dist1,zip2,dist2) :- source4(zip2,zip1,dist2), <(dist2,dist1).  
...
```

Evaluating Candidates



- Compare output of clause with that of target.
- Average the results across different input tuples.

Evaluating Candidates II

Candidates may return multiple tuples per input

- Need measure that compares sets of tuples!

<u>Input</u> <\$zip1, \$dist1>	<u>Target Output</u> <zip2, dist2>	<u>Clause Output</u> <zip2, dist2>	
<60632, 874.2>	{}	{<60629, 2.15>, <60682, 2.27>, <60623, 2.64>, ..}	No Overlap
<07307, 50.94>	{<07097, 0.26>, <07030, 0.83>, <07310, 1.09>, ...}	{}	No Overlap
<28041, 240.46>	{<28072, 1.74>, <28146, 3.41>, <28138, 3.97>, ...}	{<28072, 1.74>, <28146, 3.41>}	Overlap!

Approximating Equality

Allow flexibility in values from different sources

- Numeric Types like *distance*

10.6 km \approx 10.54 km

Error Bounds (eg. +/- 1%)

- Nominal Types like *company*

Google Inc. \approx Google Incorporated

String Distance Metrics (e.g. JaroWinkler Score > 0.9)

- Complex Types like *date*

Mon, 31. July 2006 \approx 7/31/06

Hand-written equality checking procedures.

Next ...

Automatically build semantic models for data and services available on the larger Web

- Construct models of these sources that are sufficiently rich to support querying and integration
- Current focus:
 - Build models for the vast amount of structured and semi-structured data available
 - *Not just web services, but also form-based interfaces*
 - *E.g., Weather forecasts, flight status, stock quotes, currency converters, online stores, etc.*
 - Learn models for information-producing web sources and web services

Integrated Approach

- Start with an some initial knowledge of a domain
 - Sources and semantic descriptions of those sources
- Automatically
 - Discover related sources
 - Determine how to invoke the sources
 - Learn the syntactic structure of the sources
 - Identify the semantic types of the data
 - Build semantic models of the source
 - Construct semantic web services

Seed Source

Washington, District of Columbia (20502) Conditions & Forecast : Weather Underground

file:///Users/tar/Projects/Calo/SourceDiscovery/icdm-wunderground-1.html RSS Google

Twiki APIs Apple (125) TinyURL! Zip PL-GUI Heracles GoogleGroups Mantis Shop Popular News (1368) CAL-FIRE

Welcome to Weather Underground! [Sign In](#) or [Create an Account](#). Edit my [Page Preferences](#). Other Wunders: [Mobile](#) - [iPhone](#) - [Lite](#) - [Download](#)

Search: City, State, Zip, Airport Code, or Country Weather Conditions Go


Features: [Tropical / Hurricane](#) [NEXRAD Radar](#) [Zoom Satellite](#) [Ski / Snow](#) [Marine](#) [Climate Change](#) [Tornadoes](#) [WX Radio](#) [Sports](#)
[Weather Stations](#) [Regional Radar](#) [Severe](#) [WunderBlogs](#) [WunderPhotos](#) [Trip Planner](#) [History Data](#) [Webcams](#) [Maps](#)

Washington, District of Columbia [Add to My Favorites](#) - [ICAL](#) [RSS](#)


Local Time: 1:07 PM EST — [Set My Timezone](#) Lat/Lon: 38.9° N 77.0° W (Google Map)

Tropical Weather: [Invest 96](#) (North Atlantic)






Current Conditions
Eckington Pl, NE, Washington, District of Columbia (PWS)
Updated: 1:06 PM EST on November 25, 2008

 **46.8 °F / 8.2 °C**
Mostly Cloudy

Windchill: 43 °F / 6 °C
Humidity: 41%
Dew Point: 24 °F / -4 °C
Wind: 8.0 mph / 12.9 km/h / 3.6 m/s from the WSW
Wind Gust: 15.0 mph / 24.1 km/h / 9.3 m/s
Pressure: 29.78 in / 1008.4 hPa (Steady)
Visibility: 10.0 miles / 16.1 kilometers
UV: 2 out of 16
Clouds: Mostly Cloudy 6000 ft / 1828 m
Mostly Cloudy 14000 ft / 4267 m (Above Ground Level)
Elevation: 90 ft / 27 m


Radar [Webcam](#)

[Click Radar to Enlarge](#)
[Local Radar](#)
[WunderMap new!](#)
[Regional Radar](#)
[Local Satellite](#)
[Marine Forecast](#)
[Ski Conditions](#)
[Trip Planner](#)
[Weather Stations](#)


5-Day Forecast for ZIP Code 20502 [Customize Your Icons!](#)


Tuesday	Wednesday	Thursday	Friday	Saturday
 45° F 32° F 7° C 0° C Mostly Cloudy Hourly	 47° F 31° F 8° C -1° C Partly Cloudy Hourly	 50° F 31° F 10° C -1° C Clear Hourly	 50° F 34° F 10° C 1° C Partly Cloudy Hourly	 47° F 34° F 8° C 1° C Chance of Rain 30% chance of precipitation Hourly


Today is forecast to be **Cooler** than yesterday.

Forecast for District of Columbia [↑↓](#)
Updated: 10:48 am EST on November 25, 2008


 Active Notice: [Public Information Statement](#) ([US Severe Weather](#))


 **Rest of Today**
Becoming partly sunny. Highs in the upper 40s. West winds 10 to 15 mph with gusts up to 25 mph.
» [ZIP Code Detail](#)

 **Tonight**
Mostly cloudy. Lows in the lower 30s. Southwest winds 10 to 15 mph.

 **Wednesday**
Partly sunny. Highs in the upper 40s. West winds 10 to 15 mph.
» [ZIP Code Detail](#)

Website Spotlight
[Weather Maps](#)
[Solar Calculator new!](#)
[Forecast Flyer](#)
[Community Chat](#)
[Education](#)
[Astronomy](#)
[Print This Page](#)
[Developer's Blog](#)

WunderPhotos

Washington
District of Columbia
[Browse All Photos](#)

WunderMap

[View WunderMap](#)

Weather Underground
You don't appear to have any favorites yet, or your cookies may be disabled.
[Edit My Favorites](#)

Automatically Discover and Build Semantic Web Services for Related Sources

Unisys Weather

http://weather.unisys.com/

Twiki APIs Apple (125) TinyURL Zip PL-GUI Heracles GoogleGroups Mantis Shop

UNISYS
imagine it. done.

Unisys Home Page
Unisys Transportation
Weather Solutions
Unisys Weather
Home
Information
Contents
Analyses
Satellite Images
Surface Data
Upper Air Data
Radar Data
Forecasts
Model Statistics
NGM Model
NAMWrf Model
GFS/Avn Model
GFSx/MRF Model
RUC Model
ECMWF Model
Miscellaneous
Hurricane Data
Archive of Images
USGS Maps

Enter a zip code or city name to get forecast:

GO SETUP

UNISYS
WXP Weather Analysis
UNISYS
WeatherMax Resources

ICRA

Unisys Weather

ES7000 Servers
True Flexibility

UNISYS
Internet Weather Data
UNISYS
NOAAPORT Solutions

00Z 11 DEC 08

Current satellite image and surface map (Click on map for forecast) [loop]

Visible Satellite Image Enh IR Satellite Image Satellite Surface Map
US Radar Summary NAM Model Forecast GFSx 10 day Forecast

NEWS

FAQ
First Time User
Guest Book

The intent of this weather site is to provide a complete source of graphical weather information. This is intended to satisfy the needs of the weather professional but can be a tool for the casual user as well. The graphics and data are displayed as a meteorologist would expect to see. For the novice user, there are detailed explanation pages to guide them through the various plots, charts and images. The data on this site are provided from the [National Weather Service](#) via the [NOAAPORT](#) satellite data service. All the images are generated using the [Weather Processor \(WXP\)](#) analysis package which is available from Unisys.

© Unisys Corp. 2005
- For questions and information on this server, NOAAPORT and WXP, contact Dan Vietor at devo@ks.unisys.com
- For sales information on Unisys weather solutions, contact Robert Benedict at robert.benedict@unisys.com
- Last modified February 7, 2007

Unisys Weather: Forecast for Washington, DC (20502) [0] 2

file:///Users/tar/Projects/Calo/SourceDiscovery/icdm-unisys/

Twiki APIs Apple (125) TinyURL Zip PL-GUI Heracles GoogleGroups Mantis Shop

Unisys Weather

Unisys Home Page
Unisys Transportation
Weather Solutions
Unisys Weather
Home
Information
Contents
Analyses
Satellite Images
Surface Data
Upper Air Data
Radar Data
Forecasts
Model Statistics
NGM Model
NAMWrf Model
GFS/Avn Model
GFSx/MRF Model
RUC Model
ECMWF Model
Miscellaneous
Hurricane Data
Archive of Images
USGS Maps

Enter a zip code or city name to get forecast:

Latest Observation for Washington, DC (20502)

Partly Cloudy Site: KDCa (Washington/Nati, VA) Almanac
Time: 4 PM EST 25 NOV 08 Sunrise: 7:02 AM
Temp: 45 F (7 C) Dewpt: 22 F (-5 C) Sunset: 4:48 PM
Rel Hum: 40%
Winds: W at 7 knt
Wind chill: 41 F
Pressure: 1010.1 mb (29.84 in)
Visibility: 10 mi
Skies: partly cloudy
Weather:

Alerts
No alerts

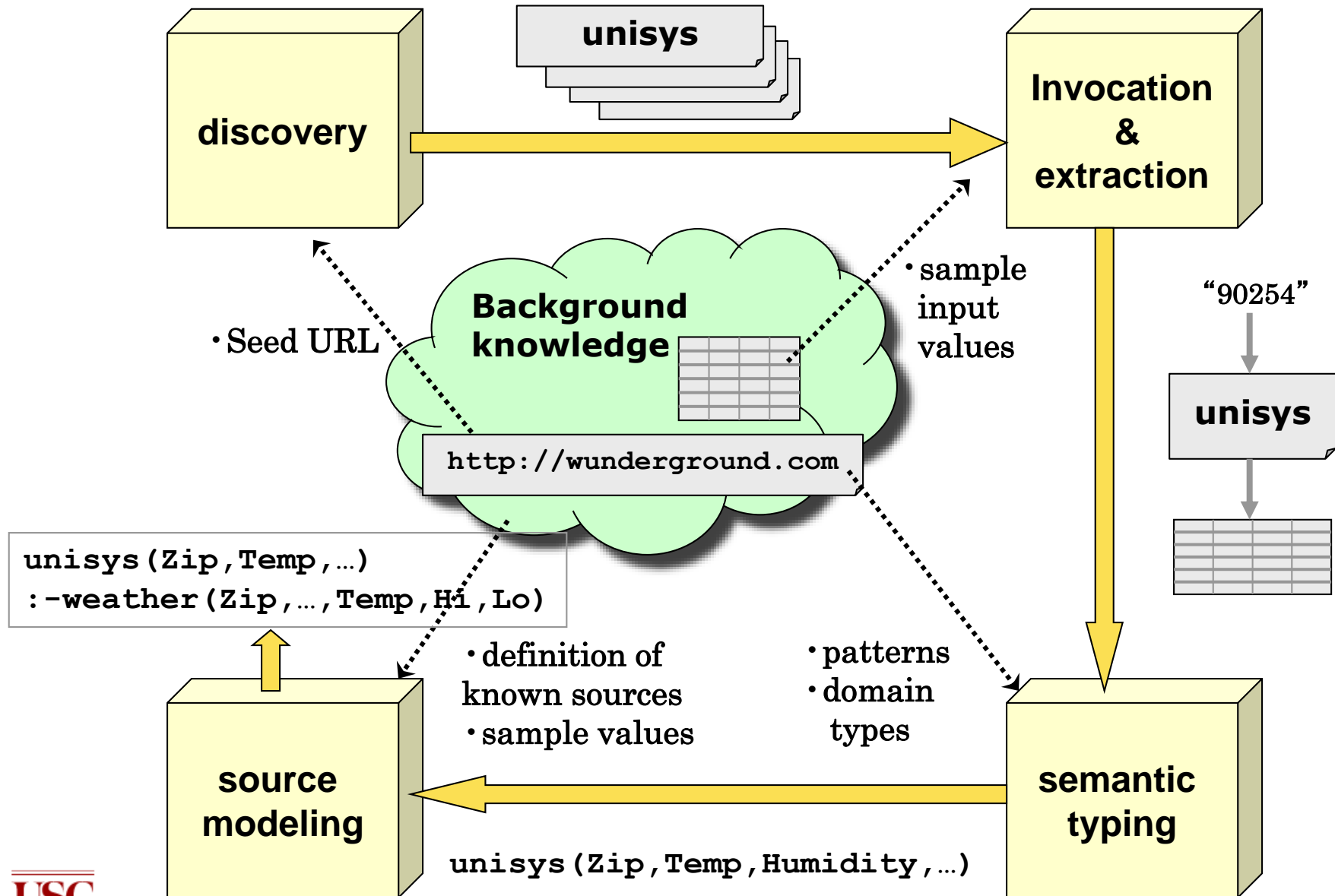
Forecast Summary

WEDNESDAY	THURSDAY	FRIDAY	SATURDAY	SUNDAY	MONDAY	TUESDAY
Sunny	Sunny	Rainy		Sunny	Sunny	Sunny
Hi: 45 Lo: 32	Hi: 52 Lo: 35	Hi: 52 Lo: 35	Hi: 48 Lo: 35	Hi: 48 Lo: 35	Hi: 45 Lo: 32	Hi: 45 Lo: 32

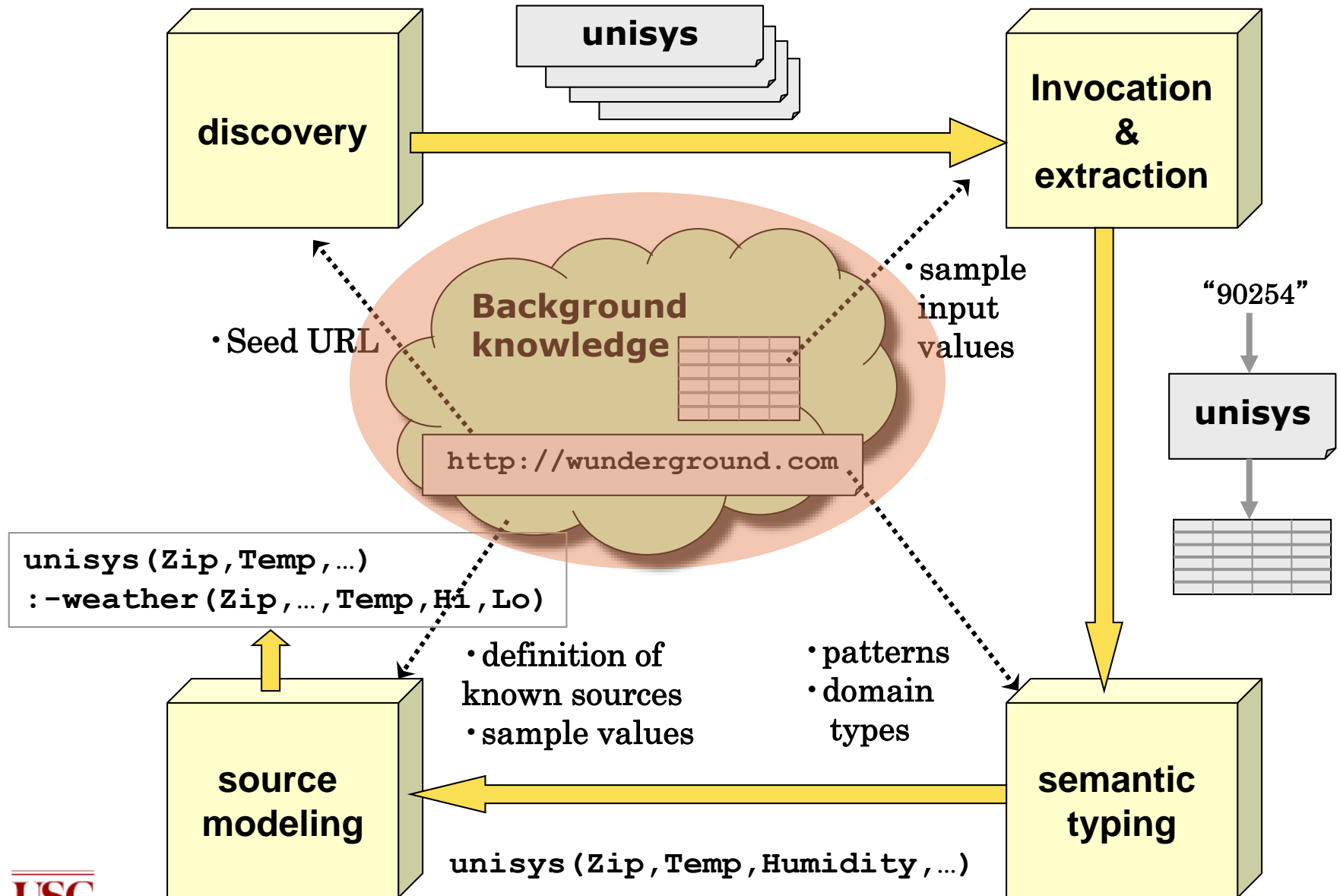
Detailed forecast from National Weather Service
DISTRICT OF COLUMBIA-ARLINGTON/FALLS CHURCH/ALEXANDRIA-
INCLUDING THE CITIES OF...WASHINGTON...ALEXANDRIA...FALLS CHURCH
306 PM EST TUE NOV 25 2008

	TONIGHT LO: 32 MOSTLY CLOUDY. LOWS IN THE LOWER 30S. SOUTHWEST WINDS AROUND 10 MPH.
Sunny	WEDNESDAY Hi: 45 MOSTLY SUNNY. HIGHS IN THE MID 40S. WEST WINDS 10 TO 15 MPH.
	WEDNESDAY NIGHT LO: 35 PARTLY CLOUDY. LOWS IN THE MID 30S. WEST WINDS 5 TO 10 MPH.
Sunny	THANKSGIVING DAY Hi: 52 SUNNY. HIGHS IN THE LOWER 50S. SOUTHWEST WINDS 5 TO 10 MPH.
	THURSDAY NIGHT LO: 35 PARTLY CLOUDY. LOWS IN THE MID 30S. SOUTH WINDS AROUND 5 MPH.
Rainy	FRIDAY Hi: 52

Integrated Approach



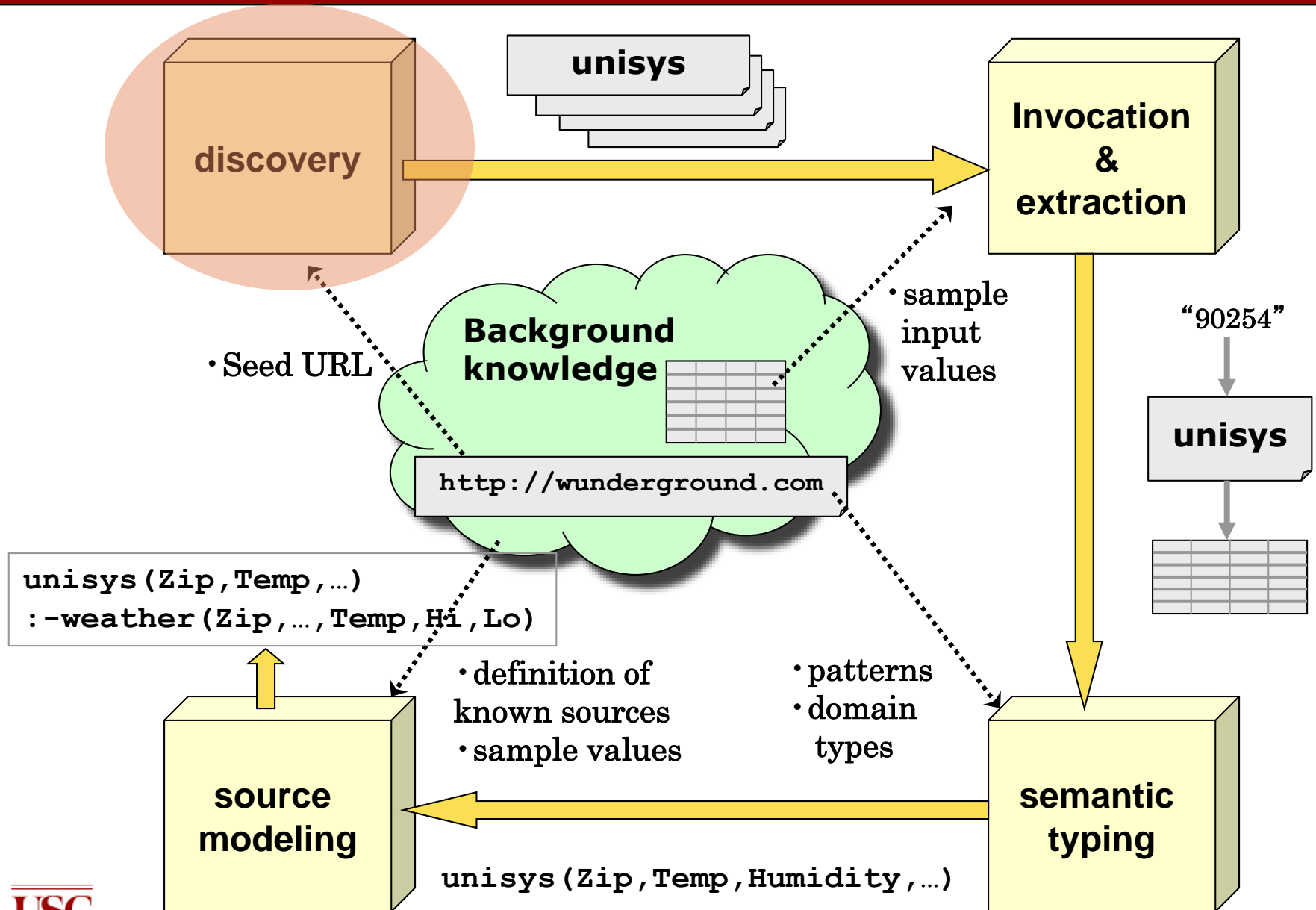
Background Knowledge



Background Knowledge

- Ontology of the inputs and outputs
 - e.g., TempF, Humidity, Zipcode;
- Sample values for each semantic type
 - e.g., “88 F” for TempF, and “90292” for Zipcode
- Domain input model
 - a weather source may accept Zipcode or City and State as input
 - Sample input values
- Known sources (seeds)
 - e.g., <http://wunderground.com>
- Source descriptions in Datalog or RDF
 - wunderground(\$Z,CS,T,F0,S0,Hu0,WS0,WD0,P0,V0,FL1,FH1,S1,FL2,FH2,S2,FL3,FH3,S3,FL4,FH4,S4,FL5,FH5,S5) :-
weather(0,Z,CS,D,T,F0,_,_,S0,Hu0,P0,WS0,WD0,V0)
weather(1,Z,CS,D,T,_,FH1,FL1,S1,_,_,_,_,_),
weather(2,Z,CS,D,T,_,FH2,FL2,S2,_,_,_,_,_),
weather(3,Z,CS,D,T,_,FH3,FL3,S3,_,_,_,_,_),
weather(4,Z,CS,D,T,_,FH4,FL4,S4,_,_,_,_,_),
weather(5,Z,CS,D,T,_,FH5,FL5,S5,_,_,_,_,_).

Source Discovery



Source Discovery

[Plangprasopchok and Lerman]

- Leverage user-generated tags on the social bookmarking site del.icio.us to discover sources similar to the seed

The screenshot shows the del.icio.us interface for a specific bookmark. The URL is <http://delicious.com/url/296e5ade5b8f4d5ac0423343475c783f>. The page title is "Welcome to The Weather Underground : Weather Underground" with the URL www.wunderground.com/. It states that the page has been saved 3242 times and 378 notes were written. The "History" tab is active, showing a list of users who bookmarked the page. Each user's entry includes their name and a set of tags. For example, "jonwalker" has the tag "weather", while "crazy-pills" has tags "reference", "tools", "maps", "travel", "news", "weather", "forecast", and "radar".

Annotations on the image highlight specific tag data:

- A "Tags" sidebar on the right lists the "Top 10 Tags" with their counts:

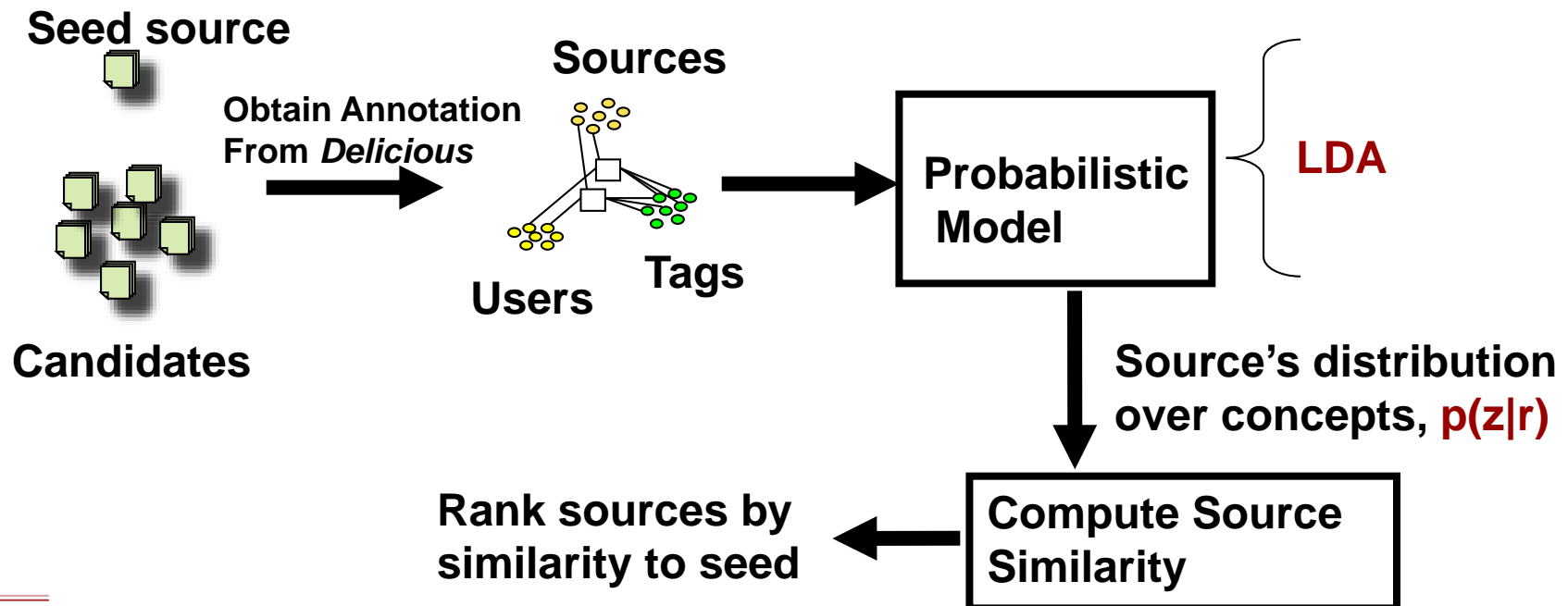
Tag	Count
weather	2314
forecast	536
travel	417
reference	386
news	285
tools	213
science	200
maps	124
world	62
meteo	53

An arrow points from the text "Most common tags" to this list.

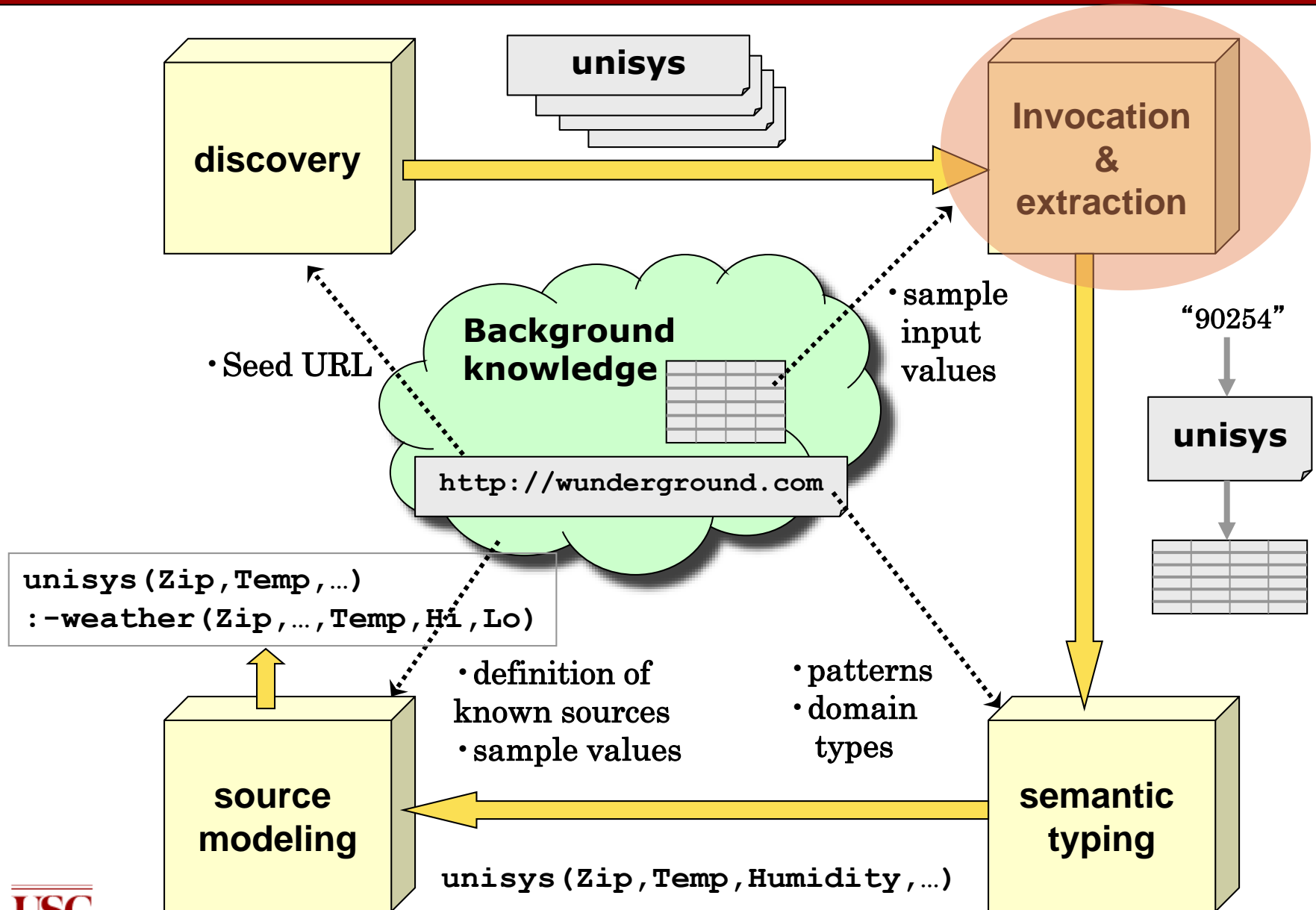
- An arrow points from the text "User-specified tags" to the tags of the user "crazy-pills" in the history list.

Exploiting Social Annotations for Resource Discovery

- **Resource discovery task** : “*given a seed source, find other most similar sources*”
 - Gather a corpus of <user, source, tag> bookmarks from del.icio.us
 - Use probabilistic modeling to find hidden topics in the corpus
 - Rank sources by similarity to the seed within topic space



Source Invocation & Extraction



Target Source Invocation

- To invoke the target source, we need to locate the form and determine the appropriate input values
 1. Locate the form
 2. Try different data type combinations as input
 - *For weather, only one input - location, which can be zipcode or city/state*
 3. Submit Form
 4. Keep successful invocations

Form
Input



The screenshot shows the Unisys Weather website in a web browser. The browser's address bar displays "http://weather.unisys.com/". The website features a large weather map of the United States with various weather symbols and pressure systems. To the left of the map is a sidebar with navigation links including "Unisys Home Page", "Weather Solutions", "Analyses", "Forecasts", and "Miscellaneous". A red circle highlights a form labeled "Enter a zip code or city name to get forecast:" with a "GO" button. Below the map, there are links for "Visible Satellite Image", "Enh IR Satellite Image", "Satellite Surface Map", "US Radar Summary", "NAM Model Forecast", and "GFSx 10 day Forecast". At the bottom, there is a "NEWS" section with links for "FAQ", "First Time User", and "Guest Book".

© Unisys Corp. 2005

- For questions and information on this server, NOAAPORT and WXP, contact [Dan Vietor at devo@ks.unisys.com](mailto:Dan.Vietor@devo.ks.unisys.com)

- For sales information on Unisys weather solutions, contact [Robert Benedict at robert.benedict@unisys.com](mailto:Robert.Benedict@robert.benedict@unisys.com)

- Last modified February 7, 2007

Inducing Extraction Templates

- Template: a sequence of alternating slots and stripes
 - stripes are the common substrings among all pages
 - slots are the placeholders for data
- Induction: Stripes are discovered using the Longest Common Subsequence algorithm

Sample Page 1

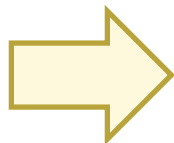
```
<br>
<font face="Arial, Helvetica, sans-serif">
  <small><b>Temp: 72F (22C)</b></small></font>
<font face="Arial, Helvetica, sans-serif">
  <small>Site: <b>KSMO (Santa_Monica_Mu, CA)</b><br>
    Time: <b>11 AM PST 10 DEC 08</b>
```



Sample Page 2

```
<br>
<font face="Arial, Helvetica, sans-serif">
  <small><b>Temp: 37F (2C)</b></small></font>
<font face="Arial, Helvetica, sans-serif">
  <small>Site: <b>KAGC (Pittsburgh/Alle, PA)</b><br>
    Time: <b>2 PM EST 10 DEC 08</b>
```

Induction



Template

```
<br>
<font face="Arial, Helvetica, sans-serif">
  <small><b>Temp: * (*)</b></small></font>
<font face="Arial, Helvetica, sans-serif">
  <small>Site: <b>* (*, *)</b><br>
    Time: <b>* 10 DEC 08</b>
```

Slot

Stripe

Data Extraction with Templates

[Gazen& Minton]

- To extract data: Find data in slots by locating the stripes of the template on unseen page:

Unseen Page

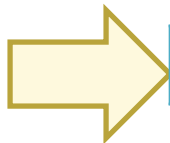
```
<br>
<font face="Arial, Helvetica, sans-serif">
  <small><b>Temp: 71F (21C)</b></small></font>
<font face="Arial, Helvetica, sans-serif">
  <small>Site: <b>KCQT (Los_Angeles_Dow, CA)</b><br>
    Time: <b>11 AM PST 10 DEC 08</b>
```



Induced Template

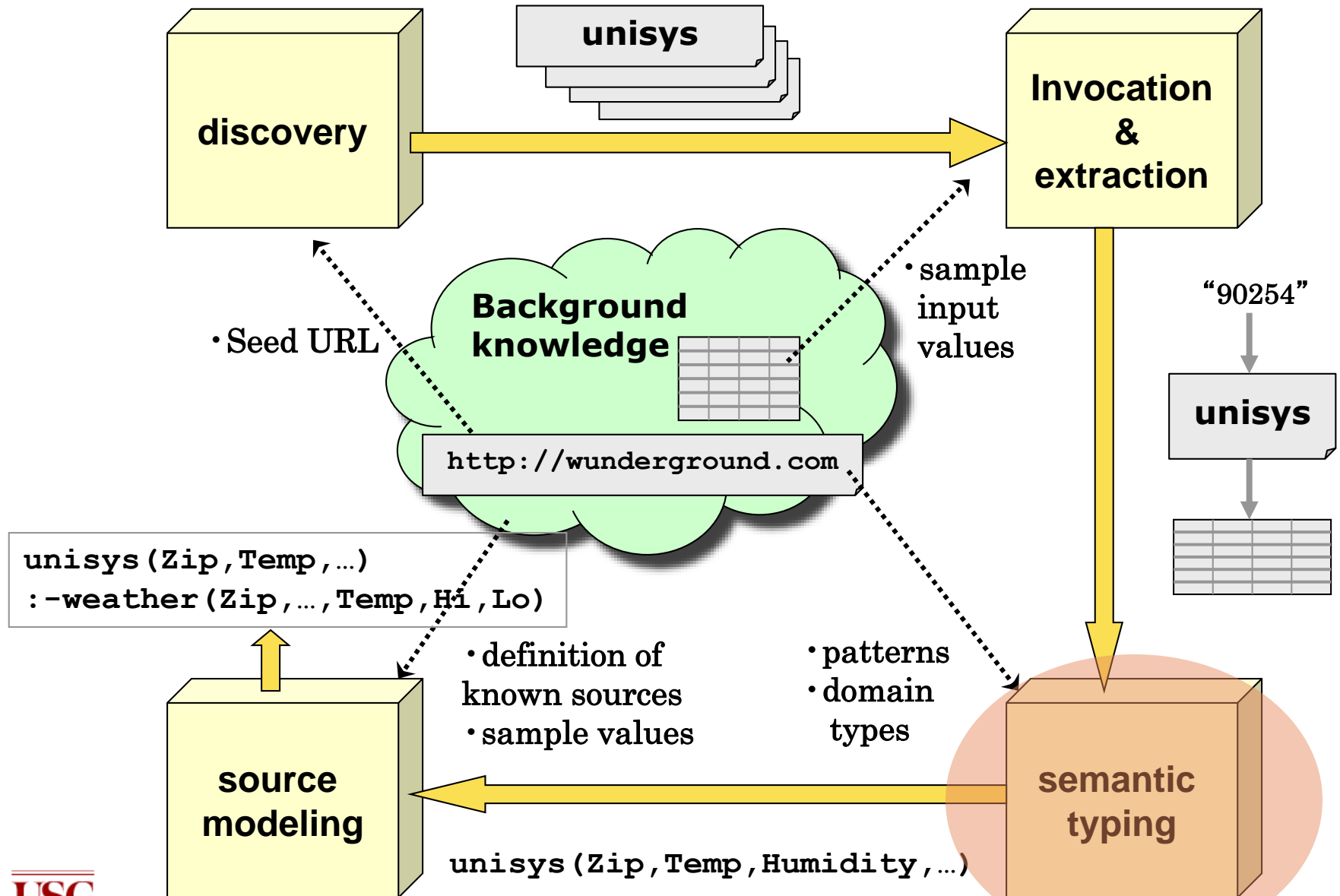
```
<br>
<font face="Arial, Helvetica, sans-serif">
  <small><b>Temp: * (*)</b></small></font>
<font face="Arial, Helvetica, sans-serif">
  <small>Site: <b>* (*, *)</b><br>
    Time: <b>* 10 DEC 08</b>
```

Extracted Data



Sun	Sunny	71F	21C	KCQT	Los_Angeles_Dow	CA	11 AM PST
-----	-------	-----	-----	------	-----------------	----	-----------

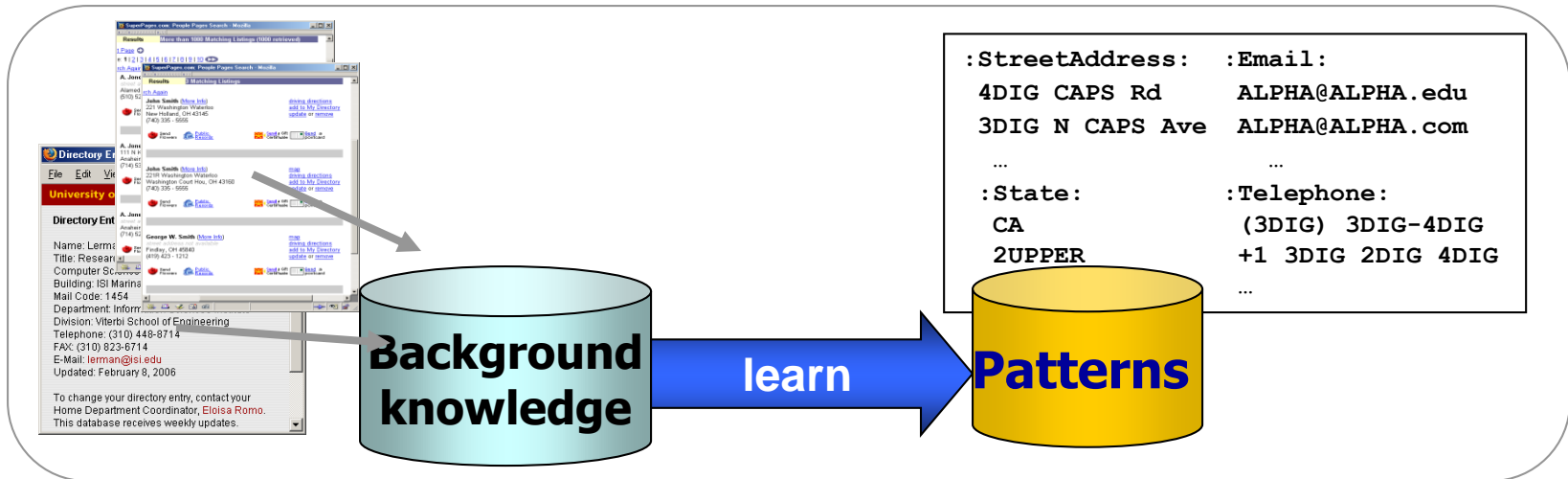
Semantic Typing



Semantic Typing

[Lerman, Plangprasopchok, & Knoblock]

- ✓ Idea: Learn a model of the content of data and use it to recognize new examples



Person	Address	Work		:FullName:	:StreetAddress:	:Telephone:
E Lewis	3518 Hilltop Rd	(419) 531 - 0504	label	E Lewis	3518 Hilltop Rd	(419) 531 - 0504
Andrew Lewis	3543 Larchmont Pkwy	(518) 474 - 4799		Andrew Lewis	3543 Larchmont Pkwy	(518) 474 - 4799
C. S. Lewis	555 Willow Run Dr	(612) 578 - 5555		C. S. Lewis	555 Willow Run Dr	(612) 578 - 5555
Carmen Jones	355 Morgan Ave N	(612) 522 - 5555		Carmen Jones	355 Morgan Ave N	(612) 522 - 5555
John Jones	3574 Brookside Rd	(555) 531 - 9566		John Jones	3574 Brookside Rd	(555) 531 - 9566
Location	State_prov	Postal_code		:City:	:State:	:Zipcode:
Toledo	OH	64325-3000		Toledo	OH	64325-3000
Toledo	OH	64356		Toledo	OH	64356
Seattle	WA	8422		Seattle	WA	8422
Seattle	WA	8435		Seattle	WA	8435
Omaha	NE	52456-6444		Omaha	NE	52456-6444

Labeling New Data

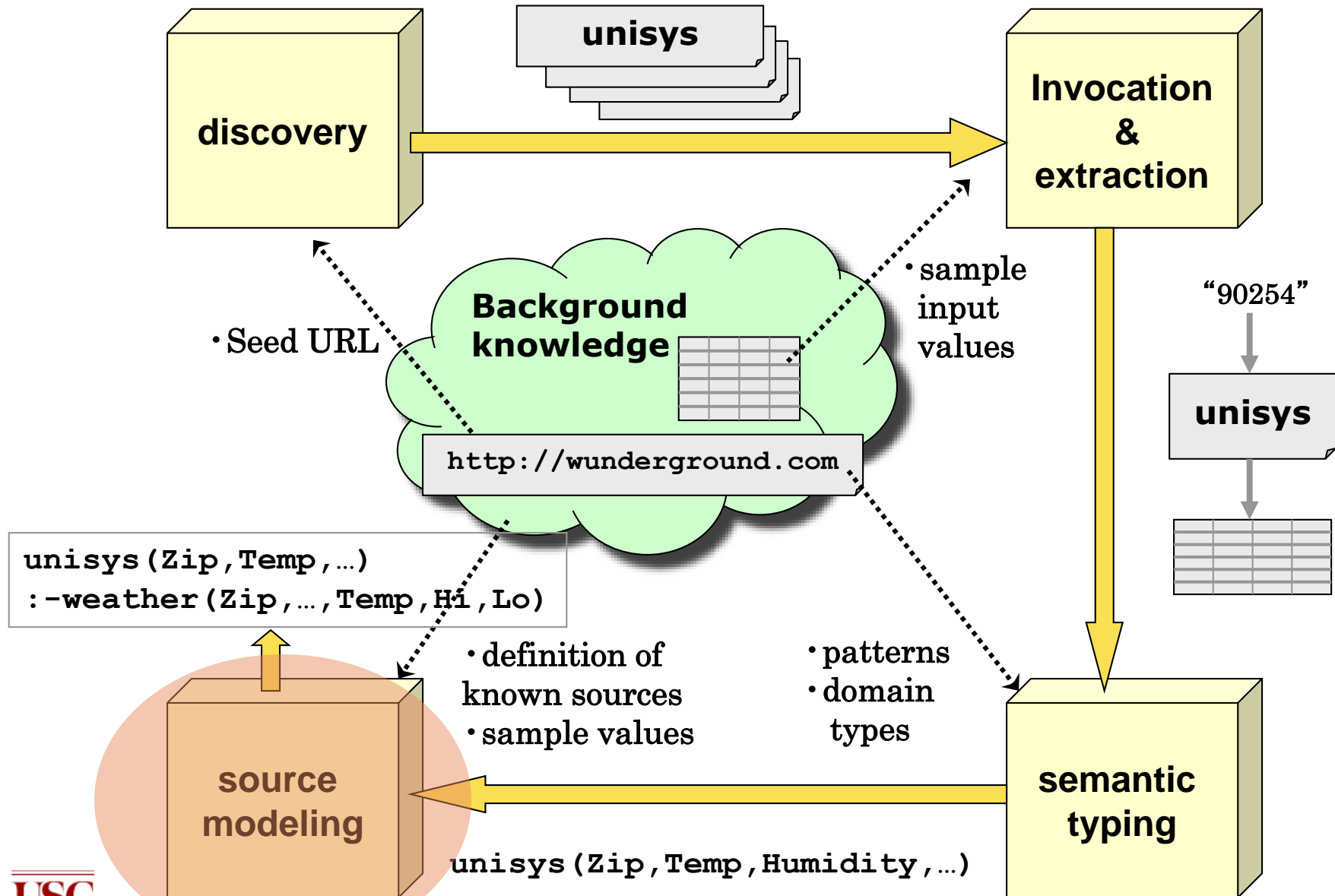
- Use learned patterns to link new data to types in the ontology
 - Score how well patterns describe a set of examples
 - *Number of matching patterns*
 - *How many tokens of the example match pattern*
 - *Specificity of the matched patterns*
 - Output top-scoring types

Person	Address	Work
E Lewis	3518 Hilltop Rd	(419) 531 - 0504
Andrew Lewis	3543 Larchmont Pkwy	(518) 474 - 4799
C. S. Lewis	555 Willow Run Dr	(612) 578 - 5555
Carmen Jones	355 Morgan Ave N	(612) 522 - 5555
John Jones	3574 Brookside Rd	(555) 531 - 9566
Location	State_prov	Postal_code
Toledo	OH	64325-3000
Toledo	OH	64356
Seattle	WA	8422
Seattle	WA	8435
Omaha	NE	52456-6444

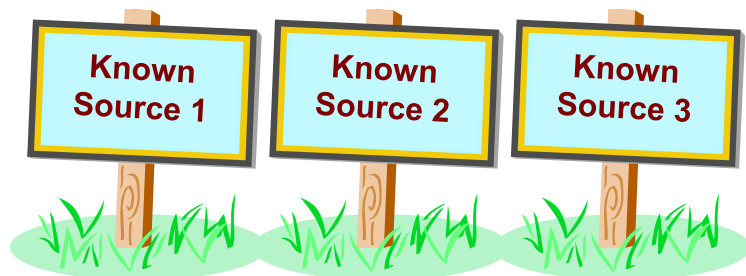
patterns

:StreetAddress:	:Email:
4DIG CAPS Rd	ALPHA@ALPHA.edu
3DIG N CAPS Ave	ALPHA@ALPHA.com
...	...
:State:	:Telephone:
CA	(3DIG) 3DIG-4DIG
2UPPER	+1 3DIG 2DIG 4DIG
...	...

Source Modeling [Carman & Knoblock]



Inducing Source Definitions

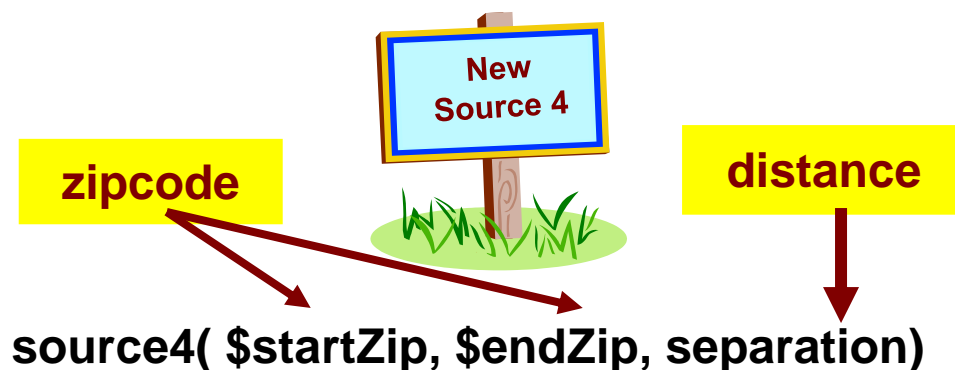


**source1(\$zip, lat, long) :-
centroid(zip, lat, long).**

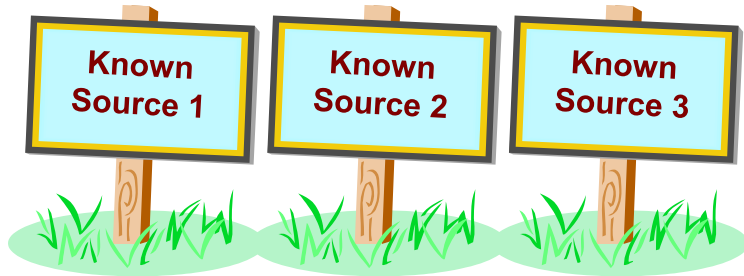
**source2(\$lat1, \$long1, \$lat2, \$long2, dist) :-
greatCircleDist(lat1, long1, lat2, long2, dist).**

**source3(\$dist1, dist2) :-
convertKm2Mi(dist1, dist2).**

- Step 1: classify input & output semantic types



Generating Plausible Definition



- Step 1: classify input & output semantic types
- Step 2: generate plausible definitions

```
source1($zip, lat, long) :-  
    centroid(zip, lat, long).
```

```
source2($lat1, $long1, $lat2, $long2, dist) :-  
    greatCircleDist(lat1, long1, lat2, long2, dist).
```

```
source3($dist1, dist2) :-  
    convertKm2Mi(dist1, dist2).
```

```
source4($zip1, $zip2, dist):-  
    source1(zip1, lat1, long1),  
    source1(zip2, lat2, long2),  
    source2(lat1, long1, lat2, long2, dist2),  
    source3(dist2, dist).
```

```
source4($zip1, $zip2, dist):-  
    centroid(zip1, lat1, long1),  
    centroid(zip2, lat2, long2),  
    greatCircleDist(lat1, long1, lat2, long2, dist2),  
    convertKm2Mi(dist1, dist2).
```

Invoke and Compare the Definition

- Step 1: classify input & output semantic types
- Step 2: generate plausible definitions
- Step 3: invoke service & compare output

```
source4($zip1, $zip2, dist):-  
    source1(zip1, lat1, long1),  
    source1(zip2, lat2, long2),  
    source2(lat1, long1, lat2, long2, dist2),  
    source3(dist2, dist).
```

```
source4($zip1, $zip2, dist):-  
    centroid(zip1, lat1, long1),  
    centroid(zip2, lat2, long2),  
    greatCircleDist(lat1, long1, lat2, long2, dist2),  
    convertKm2Mi(dist2, dist).
```



\$zip1	\$zip2	dist (<i>actual</i>)	dist (<i>predicted</i>)
80210	90266	842.37	843.65
60601	15201	410.31	410.83
10005	35555	899.50	899.21

Example of a Learned Source Model for Weather Domain

- Given a set of known sources and their descriptions
 - wunderground(\$Z,CS,T,F0,S0,Hu0,WS0,WD0,P0,V0) :-
weather(0,Z,CS,D,T,F0,_,_,S0,Hu0,P0,WS0,WD0,V0)
 - convertC2F(C,F) :- centigrade2fahrenheit(C,F)
- Learn a description of a new source in terms of the known sources
 - unisys(\$Z,CS,T,F0,C0,S0,Hu0,WS0,WD0,P0,V0) :-
wunderground(Z,CS,T,F0,S0,Hu0,WS0,WD0,P0,V0),
convertC2F(C0,F0)

Background Source Descriptions

wunderground(\$Z,CS,T,F0,C0,S0,Hu0,WS0,WD0,P0,V0,FL1,FH1,S1,
FL2,FH2, S2,FL3,FH3,S3,FL4,FH4,S4,FL5,FH5,S5):-

Weather(_w0),hasForecastDay(_w0,0),hasZIP(_w0,Z),
hasCityState(_w0,CS),hasTimeWZone(_w0,T),
hasCurrentTemperatureFarenheit(_w0,F0),
hasCurrentTemperatureCentigrade(_w0,C0),
hasSkyConditions(_w0,S0),hasHumidity(_w0,Hu0),
hasPressure(_w0,P0), hasWindSpeed(_w0,_ws1),
WindSpeed(_ws1), hasWindSpeedInMPH(_ws1,WS0),
hasWindDir(_ws1,WD0), hasVisibilityInMi(_w0,V0),
Weather(_w1), hasForecastDay(_w1,1), hasZIP(_w1,Z),
hasCityState(_w1,CS), hasLowTemperatureFarenheit(_w1,FL1),
hasHighTemperatureFarenheit(_w1,FH1), hasSkyConditions(_w1,S1),

...

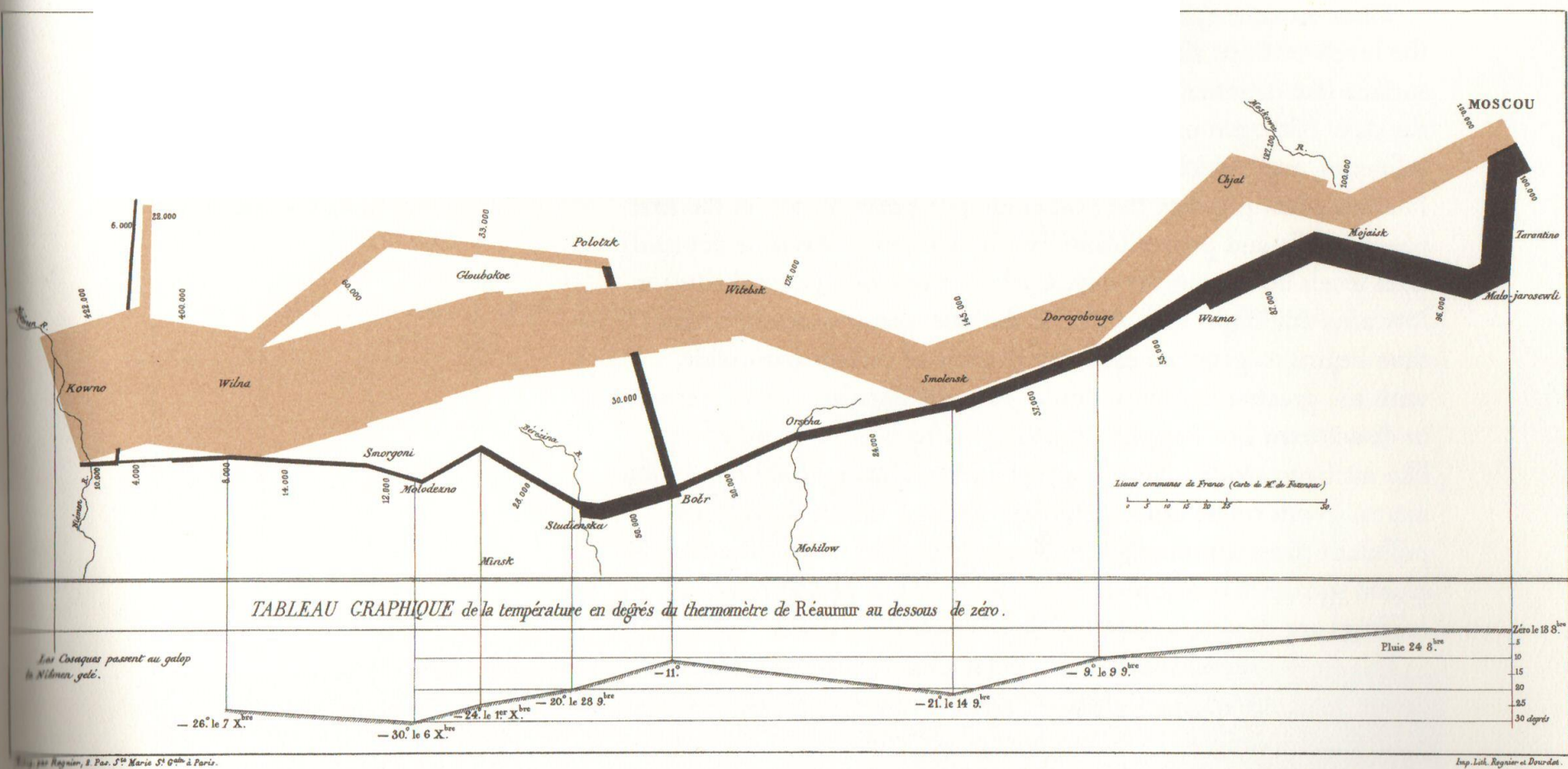
convertC2F(\$C,F) :- centigrade2fahrenheit(C,F)

Target explained using background sources

```
unisys($Z,_,_,_,_,_,_,_,F9,_,C,_,F13,F14,Hu,_,F17,_,_,_,_,S22,_,S24,  
_,_,_,_,_,_,_,_,S35,S36,_,_,_,_,_,_,_) :-  
wunderground(Z,_,_,F9,_,Hu,_,_,_,_,F14,F17,S24,_,_,S22,_,_,  
S35,_,_,S36,F13,_,_),  
convertC2F(C,F9)
```

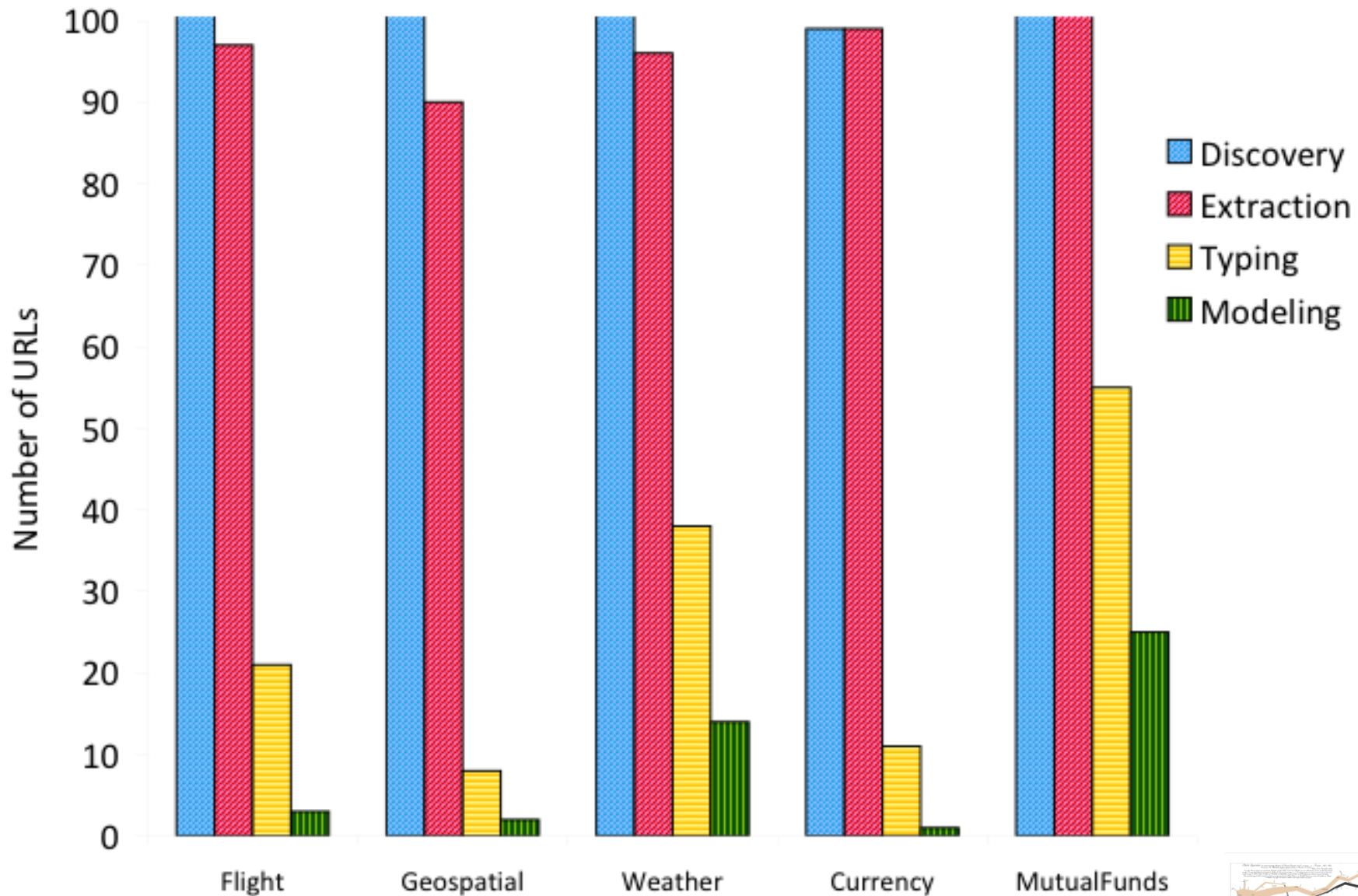
Experimental Evaluation

- Experiments in 5 domains
 - Flight – lookup the current status of a flight
 - Geospatial – map street addresses into lat/long coordinates
 - Weather – find the current and forecasted weather
 - Currency – convert between various currencies
 - Mutual Funds – look up current data on a mutual fund
- Evaluation:
 - 1) Can the system correctly learn a model for those sources that perform the same task
 - 2) What is the precision and recall of the attributes in the model



[Edward Tufte, *The Visual Display of Quantitative Information*]

Candidate Sources after Each Step



Evaluation of the Models

domain	Precision	Recall	F ₁ -measure
<i>weather</i>	0.64	0.29	0.39
<i>geospatial</i>	1.00	0.86	0.92
<i>flights</i>	0.69	0.35	0.46
<i>currency</i>	1.00	1.00	1.00
<i>mutualfund</i>	0.72	0.30	0.42

Conclusions

- Integrated approach to learning:
 - *How to invoke a web service*
 - *The semantic types of the output*
 - *A definition of what the service does*
- Provides an approach to generate source descriptions for the Semantic Web
 - Little motivation for providers to annotate services
 - Instead we generate metadata automatically
- Also provides an approach to automatically discover new sources of data