Planning for
Information Gathering

Craig Knoblock
University of Southern California

These slides are based in part on slides from José Luis Ambite and Rao Kambhampati, which are in turn based in part on slides from Alon Halevy.

Planning on the Web

- Part I: Planning for Information Gathering
- Part II: Plan Execution for Information Gathering
Outline

- Information Gathering
  - Planning for Information Gathering
    - View Integration
    - Query Reformulation
    - Source Capabilities
  - Optimizing Information Gathering Plans
    - Removing Redundant Sources
    - Optimizing Sources and Queries
  - Interleaving Planning and Sensing
    - Sensing to Handle Incomplete Information
    - Sensing to Optimize Plans
  - Contingent Planning for Information Gathering
  - Planning to Compose Web Sources
- Discussion
Wrappers for Accessing Online Information Sources

- Wrappers provide uniform querying and data extraction

![Yellow Pages Image]

NAME Casablanca Restaurant
STREET 220 Lincoln Boulevard
CITY Venice
PHONE (310) 392-5751

- State of the art in wrapper induction
  - Data extraction is based on Web page layout
    (Muslea et al. 1999, Kushmerick et al. 1997)
  - User labels examples of data on pages
  - Induction algorithm learns extraction rules for data

Planning for Information Gathering

- Database query access planning
  - Specialized planner optimized for task
  - Sources are fixed
  - Mappings predefined in global schema
  - Complete plan is generated and then executed
  - Assumes closed-world and complete information

- Distributed, heterogeneous environments:
  - Sources and mappings are not fixed
  - Sources are autonomous
  - Overlapping and redundant sources
  - Sources may be incomplete
  - Sources may be unavailable
  - Additional information may be required to access a source
  - Access to sources may be costly
Database Query Access Plans

Declarative SQL query

Imperative query execution plan:

```
SELECT S.buyer
FROM Purchase P, Person Q
WHERE P.buyer=Q.name AND
  Q.city='seattle' AND
  Q.phone > '5430000'
```

Inputs:
- the query
- statistics about the data (indexes, cardinalities, selectivity factors)
- available memory

Ideally: Want to find best plan. Practically: Avoid worst plans!

Outline

- Information Gathering
- Planning for Information Gathering
  - View Integration
  - Query Reformulation
  - Source Capabilities
- Optimizing Information Gathering Plans
  - Removing Redundant Sources
  - Optimizing Sources and Queries
- Interleaving Planning and Sensing
  - Sensing to Handle Incomplete Information
  - Sensing to Optimize Plans
- Contingent Planning for Information Gathering
- Planning to Compose Web Sources
- Discussion
Virtual Integration Architecture

- Leave the data in the sources
- When a query comes in:
  - Determine the relevant sources to the query
  - Break down the query into sub-queries for the sources
  - Get the answers from the sources, and combine them appropriately
- Data is fresh. Approach scalable
- Issues:
  - Relating Sources & Mediator
  - Reformulating the query
  - Efficient planning & Execution

Garlic [IBM], Hermes[UMD]; Tsimmis, InfoMaster[Stanford]; DISCO[INRIA]; Information Manifold [AT&T]; SIMS/Ariadne[USC]; Emerac/Havasu[ASU]

Desiderata for Relating Source-Mediator Schemas

- Expressive power: distinguish between sources with closely related data. Hence, be able to prune access to irrelevant sources.
- Easy addition: make it easy to add new data sources.
- Reformulation: be able to reformulate a user query into a query on the sources efficiently and effectively.
- Nonlossy: be able to handle all queries that can be answered by directly accessing the sources

- Given:
  - A query Q posed over the mediated schema
  - Descriptions of the data sources

- Find:
  - A query Q' over the data source relations, such that:
    - Q' provides only correct answers to Q, and
    - Q' provides all possible answers to Q given the sources.
Source Descriptions

Elements of source descriptions:
- Contents: source contains movies, directors, cast.
- Constraints: only movies produced after 1965.
- Completeness: contains all American movies.
- Capabilities:
  - Negative: source requires movie title or director as input
  - Positive: source can perform selections, joins, ...

Approaches to Specification of Source Descriptions

- Global-as-View (GAV):
  Mediator relation defined as a view over source relations
  Ex: TSIMMIS (Stanford), HERMES (Maryland)
- Local-as-View (LAV):
  Source relation defined as view over mediator relations
  Ex: Information Manifold (AT&T), Tukwila(UW), InfoMaster (Stanford), Ariadne (USC)

View ~ named query ~ logical formula
Views and Conjunctive Queries

CREATE VIEW Big-LA-buyers AS
SELECT buyer, seller, price
FROM Person, Purchase
WHERE Person.city = “Los Angeles” AND
Person.name = Purchase.buyer AND
Purchase.price > 10000

big-LA-buyers(Buyer,Seller, Price) :-
person(Buyer, “Los Angeles”),
purchase(Buyer, Seller, Product, Price),
Price > 10000.

Datalog rule ~ view definition
Rule body ~ select-from-where construct of SQL

Outline

• Information Gathering
• Planning for Information Gathering
  • View Integration
  • Query Reformulation
  • Source Capabilities
• Optimizing Information Gathering Plans
  • Removing Redundant Sources
  • Optimizing Sources and Queries
• Interleaving Planning and Sensing
  • Sensing to Handle Incomplete Information
  • Sensing to Optimize Plans
• Contingent Planning for Information Gathering
• Planning to Compose Web Sources
• Discussion
Query Reformulation

Problem: rewrite the user query expressed in the mediated schema into a query expressed in the source schemas

Given a query $Q$ in terms of the mediated-schema relations, and descriptions of the information sources,

Find a query $Q'$ that uses only the source relations, such that

- $Q' \models Q$ (i.e., answers are correct; i.e., $Q' \subseteq Q$) and
- $Q'$ provides all possible answers to $Q$ given the sources

Global-as-View (GAV)

Each mediator relation is defined as a view over source relations.

- MovieActor(title, actor) ← 
  - DB1(id, title, actor, year)
- MovieActor(title, actor) ← 
  - DB2(title, director, actor, year)
- MovieReview(title, review) ← 
  - DB1(id, title, actor, year) ^ DB3(id, review)
Query Reformulation in GAV

Query reformulation = rule unfolding+simplification
Query: Find reviews for ‘DeNiro’ movies
q(title, review) :- MovieActor(title, ‘DeNiro’),
                  MovieReview(title, review)
1. q’(title, review) :- DB1(id, title, ‘DeNiro’, year),
                          DB1(id, title, actor, year’), DB3(id, review)
2. q’(title, review) :-
                          DB2(title, director, ‘DeNiro’, year),
                          DB1(id, title, actor, year’), DB3(id, review)

Local-as-View (LAV)

- Each source relation is defined as a view over mediator relations

V1(title, year, director) ⇐ Movie(title, year, director, genre)
  ^ American(director) ^ year ≥ 1960 ^ genre = ‘Comedy’

V2(title, review) → Movie(title, year, director, genre) ^
  year ≥ 1990 ^ MovieReview(title, review)
Query Reformulation in LAV

Query: Reviews for comedies produced after 1950
q(title,review) :- Movie(title,year,director,’Comedy’), year ≥1950, MovieReview(title,review)

Reformulated query:
q’(title,review) :- V1(title,year,director),
V2(title,review)

\( V1(title, year, director) \rightarrow Movie(title, year, director, genre) \land American(director) \land year ≥1960 \land genre = ‘Comedy’ \)
\( V2(title, review) \rightarrow Movie(title, year, director, genre) \land year ≥1990 \land MovieReview(title, review) \)

Inverse-Rules Algorithm

Idea: Construct an equivalent logic program which evaluation yields the answer to the query

- The antecedent of the query and views is in term of mediator predicates
- Would like to have source predicates in antecedent so that program can be evaluated

⇒ Invert the rules
(simply by using standard logical manipulations)
The Inverse-Rules Algorithm: Example

\[ V_1(\text{dept}, \text{course}) \rightarrow \text{Enrolled(\text{student}, \text{dept})} \land \text{Registered(\text{student}, \text{course})} \]

\[ \forall D, C \ [v_1(D, C) \rightarrow \exists S \ [e(S, D) \land r(S, C)]] \]
\[ \equiv \neg v_1(D, C) \lor [e(f(D, C), D) \land r(f(D, C), C)] \]
\[ \equiv [\neg v_1(D, C) \lor e(f(D, C), D)] \land [\neg v_1(D, C) \lor r(f(D, C), C)] \]
\[ \equiv [v_1(D, C) \rightarrow e(f(D, C), D)] \land [v_1(D, C) \rightarrow r(f(D, C), C)] \]
\[ \equiv \]
\[ e(f(D, C), D) \leftarrow v_1(D, C) \]
\[ r(f(D, C), C) \leftarrow v_1(D, C) \]

---

The Inverse-Rules Algorithm: Example

\[ q(D) \leftarrow \text{Enrolled(\text{S}, \text{D})} \land \text{Registered(\text{S,}"DB")} \]
\[ v_1(D, C) \rightarrow \text{Enrolled(\text{S}, \text{D})} \land \text{Registered(\text{S,}C)} \]

\[ q(D) \leftarrow \text{Enrolled(\text{S}, \text{D})} \land \text{Registered(\text{S,}"DB")} \]
\[ \text{Enrolled}(f(D, C), D) \leftarrow v_1(D, C) \]
\[ \text{Registered}(f(D, C), C) \leftarrow v_1(D, C) \]
\[ q(D) \leftarrow v_1(D,"DB") \]

\[ \text{Ext}(v_1) = \{("CS", "DB"), ("EE", "DB"), ("CS", "AI")\} \]
\[ \text{Ext}(q) = \{("CS"), ("EE")\} \]
### GAV vs. LAV

- **Not modular**
  - Addition of new sources changes the mediated schema
- **Can be awkward to write mediated schema without loss of information**
- **Query reformulation easy**
  - reduc... of mediated schemas
- **Best when**
  - Few, stable, data sources
  - well-known to the mediator (e.g. corporate integration)
  - Garlic, TSIMMIS, HERMES

- **Modular--adding new sources is easy**
- **Very flexible--power of the entire query language available to describe sources**
- **Reformulation is hard**
  - involves answering queries only using views (can be intractable)
- **Best when**
  - Many, relatively unknown data sources
  - possibility of addition/deletion of sources
  - Information Manifold, InfoMaster, Emerac

### Outline

- Information Gathering
- **Planning for Information Gathering**
  - View Integration
  - Query Reformulation
  - Source Capabilities
- Optimizing Information Gathering Plans
  - Removing Redundant Sources
  - Optimizing Sources and Queries
- Interleaving Planning and Sensing
  - Sensing to Handle Incomplete Information
  - Sensing to Optimize Plans
- Contingent Planning for Information Gathering
- Planning to Compose Web Sources
- Discussion
Modeling Source Capabilities

Negative capabilities:
- A website may require certain inputs (in an HTML form) to answer a query
- Need to consider only valid query execution plans

Positive capabilities:
- A source may be a database (understands SQL)
- Need to decide the placement of operations according to capabilities

Problem: how to describe and exploit source capabilities

Negative Capabilities: Binding Patterns

Sources:
- AAAIdb^f(X) → AAAIPapers(X)
- CitationDB^bf(X,Y) → Cites(X,Y)
- AwardDB^b(X) → AwardPaper(X)

Query: find all the award winning papers:
- q(X) ← AwardPaper(X)
Recursive Rewritings

\[ q(X) \leftarrow \text{AwardPaper}(X) \]

- **Problem:** *Unbounded* union of conjunctive queries

\[
\begin{align*}
q_1(X) & \leftarrow \text{AAAIdb}(X), \text{AwardDB}(X) \\
q_1(X) & \leftarrow \text{AAAIdb}(X1), \text{CitationDB}(X1,X), \text{AwardDB}(X) \\
\vdots \\
q_1(X) & \leftarrow \text{AAAIdb}(X1), \text{CitationDB}(X1,X2), \ldots, \text{CitationDB}(Xn,X), \text{AwardDB}(X)
\end{align*}
\]

- **Solution:** Recursive Rewriting

\[
\begin{align*}
papers(X) & \leftarrow \text{AAAIdb}(X) \\
papers(X) & \leftarrow papers(Y), \text{CitationDB}(Y,X) \\
q'(X) & \leftarrow papers(X), \text{AwardDB}(X)
\end{align*}
\]

Sources:

\[
\begin{align*}
\text{AAAIdb}^f(X) & \rightarrow \text{AAAIPapers}(X) \\
\text{CitationDB}^{bf}(X,Y) & \rightarrow \text{Cites}(X,Y) \\
\text{AwardDB}^b(X) & \rightarrow \text{AwardPaper}(X)
\end{align*}
\]

Query: find all the award winning papers:

\[ q(X) \leftarrow \text{AwardPaper}(X) \]
Inverted Rules:

\begin{align*}
\text{AAAI Papers}(X) & \leftarrow \text{AAAldb}(X) \\
\text{Cites}(X,Y) & \leftarrow \text{dom}(X) \land \text{CitationDB}(X,Y) \\
\text{Award Paper}(X) & \leftarrow \text{dom}(X) \land \text{AwardDB}(X)
\end{align*}

Domain Rules:

\begin{align*}
\text{dom}(Y) & \leftarrow \text{dom}(X) \land \text{CitationDB}(X,Y) \\
\text{dom}(X) & \leftarrow \text{AAAldb}(X)
\end{align*}

Query:

\begin{align*}
q(X) & \leftarrow \text{Award Paper}(X)
\end{align*}

Simplyfing the program:

\begin{align*}
q(X) & \leftarrow \text{paper}(X) \land \text{AwardDB}(X) \\
paper(Y) & \leftarrow \text{paper}(X) \land \text{CitationDB}(X,Y) \\
paper(X) & \leftarrow \text{AAAldb}(X)
\end{align*}
Outline

- Information Gathering
- Planning for Information Gathering
  - View Integration
  - Query Reformulation
  - Source Capabilities
- Optimizing Information Gathering Plans
  - Removing Redundant Sources
  - Optimizing Sources and Queries
- Interleaving Planning and Sensing
  - Sensing to Handle Incomplete Information
  - Sensing to Optimize Plans
- Contingent Planning for Information Gathering
- Planning to Compose Web Sources
- Discussion

Managing Source Overlap

- Often, sources on the Internet have overlapping contents
  - The overlap is *not* centrally managed (unlike DDBMS—data replication etc.)
- Reasoning about overlap is important for plan optimality
  - We cannot possibly call all potentially relevant sources!
- Qns: How do we characterize and exploit source overlap?
Local Completeness Information

- If sources are incomplete, we may need to look at all of them
- Often, sources are *locally complete*
- Movie(title, director, year) complete for years after 1960, or for American directors
- **Question:** given a set of local completeness statements, is a query Q’ a complete answer to Q?

Using LCW rules to minimize plans

**Basic Idea:**
- If reformulation of Q leads to a union of conjunctive plans
  - \( P_1 \lor P_2 \lor \ldots \lor P_k \)
- Then, if \( P_1 \) is “complete” for Q (under the given LCW information), then we can minimize the reformulation by pruning \( P_2 \ldots P_k \)
  - \([P_1 \land \text{LCW}] \) contains \( P_1 \lor P_2 \lor \ldots \lor P_k \)
  - \([Duschka, AAAI-97]\)
- For Recursive Plans (obtained when the sources have access restrictions)
  - We are allowed to remove a rule \( r \) from a plan \( P \), if the “complete” version of \( r \) is already contained in \( P-r \)

Emerac \([\text{Lambrecht \\ Kambhampati}, 99]\)
Example

- **S1**: Movie(title, director, year) (complete after 1960)
  \[ S1(T,D,Y) \rightarrow M(T,D,Y) \]
- **S2**: Show(title, theater, city, hour) (complete for Seattle)
  \[ S2(T,Th,C,H) \rightarrow Sh(T,Th,C,H) \]
  LCW: \( S2(T,Th,C,H) \leftarrow Sh(T,Th,C,H) \land C = \text{Seattle} \)
- **S3**: Show(title, theater, city, hour)
  \[ S3(T,Th,C,H) \rightarrow Sh(T,Th,C,H) \]

- **Query**: Find movies and directors playing in Seattle
  \[ Q(T,D) \leftarrow M(T,D,Y) \land Sh(T,Th,C,H) \land C = \text{"Seattle"} \]

- **Plan**: Combine S1 with S2 or S3
  \[ Q(T,D) \leftarrow S1(T,D,Y) \land S2(T,Th,C,H) \land C = \text{"Seattle"} \]
  \[ Q(T,D) \leftarrow S1(T,D,Y) \land S3(T,Th,C,H) \land C = \text{"Seattle"} \]
- **Optimized Plan**: Use LCW to prune S3
  \[ Q(T,D) \leftarrow S1(T,D,Y) \land S2(T,Th,C,H) \land C = \text{"Seattle"} \]

Outline

- Information Gathering
- Planning for Information Gathering
  - View Integration
  - Query Reformulation
  - Source Capabilities
- Optimizing Information Gathering Plans
  - Removing Redundant Sources
  - Optimizing Sources and Queries
- Interleaving Planning and Sensing
  - Sensing to Handle Incomplete Information
  - Sensing to Optimize Plans
- Contingent Planning for Information Gathering
- Planning to Compose Web Sources
- Discussion
Planning by Rewriting

[Ambite & Knoblock, 1998]

- Efficiently generate an initial solution plan (possibly of low quality)
- Iteratively rewrite the current plan
  - using a set of declarative plan rewriting rules
  - improving plan quality
  - until an acceptable solution or resource limit reached

Efficient High-Quality Planning

Planning by Rewriting as Local Search

- PbR: efficient high-quality planning using local search
- Main issues:
  - Selection of initial feasible point: Initial plan generation
  - Generation of a local neighborhood: Set of plans obtained from application of the plan rewriting rules
  - Cost function to minimize: Measure of plan quality
  - Selection of next point: Next plan to consider -- determines how the global space is explored

Start
Neighborhood
Planning by Rewriting for Query Planning in Mediators

- **Initial plan generation**: random parse of the query
- **Plan rewriting rules**: based on properties of:
  - relational algebra,
  - distributed environment,
  - integration axioms
- **Plan quality**: query execution time (size estimation)
- **Search Strategies**: gradient descent+restart, simulated annealing, variable-depth rewriting, ...

---

Query Planning in PbR

\[ a(name \text{ sal proj}) \vdash Emp(name \text{ ssn}) \land Payroll(ssn \text{ sal}) \land Projects(name \text{ proj}) \]
Rewriting Rules: Distributed Environment
remote-join-eval

\[(\text{define-rule } \text{remote-join-eval})\]
\[(\text{if } (\text{operators } ((?n1 \ (\text{retrieve } ?\text{source } ?\text{query1}))))
(\text{?n2 } (\text{retrieve } ?\text{source } ?\text{query2}))
(\text{?n3 } (\text{join } ?\text{join-conds } ?\text{query0 } ?\text{query1 } ?\text{query2})))
]:\text{constraints} (\text{capability } ?\text{source join})
):\text{replace} (\text{operators } ((?n1 \ ?n2 \ ?n3))
):\text{with} (\text{operators } (((?n4 \ (\text{retrieve } ?\text{source } ?\text{query0})))))

Rewriting Rules: Relational Algebra
join-associativity

\[(\text{define-rule } \text{name } \text{join-associativity})\]
\[(\text{if } (\text{operators } ((?n1 \ (\text{join } ?jc34 \ ?q1 \ ?q3 \ ?q4))))
(\text{?n2 } (\text{join } ?jc12 \ ?q0 \ ?q1 \ ?q2)))
]:\text{constraints} (\text{join-swappable } ?jc34 \ ?q1 \ ?q3 \ ?q4 \ ?jc12 \ ?q0 \ ?q2
\text{; in}
?jc24 \ ?jc35 \ ?q5)\text{; out}
):\text{replace} (\text{operators } (?n1 \ ?n2))
):\text{with} (\text{operators } ((?n3 \ (\text{join } ?jc24 \ ?q5 \ ?q4 \ ?q2))))
(\text{?n4 } (\text{join } ?jc35 \ ?q0 \ ?q3 \ ?q5)))
Rewriting Rules: Integration Axioms

- Rules computed from integration axioms relevant to query:
  \[
  \text{Restaurant}(\text{name cuisine rating lat long}) = \\
  \quad \text{a) } \text{Zagat}(\text{name address cuisine rating}) \land \text{Geocoder(address lat long)} \\
  \quad \text{b) } \text{Fodors}(\text{name street zip cuisine rating}) \land \text{Mapblast(street zip lat long)}
  \]

PbR in Query Planning: Summary

- Operators: output, retrieve, assign, select, join, union
- Plan rewriting rules:
  - Distributed environment: source-swap, remote-join-eval, remote-selection-eval, and remote-assignment-eval.
  - Integration axioms: computed automatically from the relevant integration axioms for classes in the query
- Search: gradient descent + random restart
  - first-improvement
  - steepest descent
### Outline

- Information Gathering
  - Planning for Information Gathering
    - View Integration
    - Query Reformulation
    - Source Capabilities
  - Optimizing Information Gathering Plans
    - Removing Redundant Sources
    - Optimizing Sources and Queries
  - Interleaving Planning and Sensing
    - Sensing to Handle Incomplete Information
    - Sensing to Optimize Plans
  - Contingent Planning for Information Gathering
  - Planning to Compose Web Sources
  - Discussion

---

### Planning for the Internet Softbot


- XII and Puccini planners for the Internet Softbot
- Plans both gathering and manipulation actions
  - e.g., `ls -a`, `chmod +r *`
- Used to model Internet resources such as netfind
- Each resource modeled as a operator

Name: (netfind ?person)

Preconds:

- (current.shell csh)
- (isa netfind.server ?server)
- (firstname ?person ?firstname)
- (lastname ?person ?lastname)
- (or
  - (person.city ?person ?keyword)
  - (person.institution ?person ?keyword))

Postconds:

- (userid ?person !userid)
- (person.machine ?person !machine)

Netfind Operator from XII
Observational Effects and Knowledge Preconditions

- **Observational Effects**
  - Effect that changes the state of the world
    \[ \text{chmod} + r \ foo.tex \text{ -- cause(readable(foo.tex))} \]
  - Effect that changes the agent’s model of the world
    \[ \text{wc} \text{ -- observe(word.count(file, Iword))} \]

- **Knowledge Preconditions**
  - Information goal -- find-out(length (paper.tex, l))
  - Goals of achievement -- satisfy(readable(f) False)

- **Verification Links**
  - Alternative to knowledge preconditions
  - Assume secondary condition is true and then use an observational effect to determine whether it is true after execution

Sensing for Locally Complete Information

- **Reasons about incomplete information**
  - Uses LCW to reason about what it knows and what it doesn’t know
  - e.g., \text{ls -a *} gives it locally complete information about the current directory

- **Interleaves sensing actions to gather LCW information**
  - LCW statements are a way of satisfying universally quantified goals

- **Provides fine-grained reasoning**
  - e.g., can request all recent techreports by X not already stored locally
Outline

- Information Gathering
- Planning for Information Gathering
  - View Integration
  - Query Reformulation
  - Source Capabilities
- Optimizing Information Gathering Plans
  - Removing Redundant Sources
  - Optimizing Sources and Queries
- Interleaving Planning and Sensing
  - Sensing to Handle Incomplete Information
  - Sensing to Optimize Plans
- Contingent Planning for Information Gathering
- Planning to Compose Web Sources
- Discussion

Sensing to Determine Relevant Sources [Ashish, Knoblock, & Levy, 1997]

Technical Report Repositories

Carnegie Mellon

CMU-1
  - year<1980
CMU-2
  - year>=1980
  - year<1990
CMU-3
  - year>=1990

Stanford

dept=“CS”

AT&T Labs

AT&T-1
  - year=1996
AT&T-2
  - year=1995
AT&T-3
  - year=1994
Building a Discrimination Matrix

- Discrimination matrix specifies the relevant sources for each region of each attribute
- Approach:
  - Analyze source descriptions to build a discrimination matrix
  - Matrix partitions sources along some attribute
  - Discrimination matrix used to estimate the cost of querying with and without sensing
- Useful discriminations provided when:
  - Sources can be partitioned by some attribute
  - Exists another source that provides that attribute
- Example: Information about the year of a tech report reduces the relevant sources from 7 to 3

## Discrimination Matrix

<table>
<thead>
<tr>
<th>Region</th>
<th>Relevant Sources</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 1980</td>
<td>CMU-1, Stanford</td>
</tr>
<tr>
<td>[1980,1990)</td>
<td>CMU-2, Stanford</td>
</tr>
<tr>
<td>[1990,1994]</td>
<td>CMU-3, Stanford</td>
</tr>
<tr>
<td>[1994,1994]</td>
<td>CMU-3, Stanford, AT&amp;T-1</td>
</tr>
<tr>
<td>&gt; 1996</td>
<td>CMU-3, Stanford</td>
</tr>
</tbody>
</table>
Planning with Discriminating Queries

- Consider inserting a discriminating query for any subquery that:
  - Requires accessing multiple sources
  - There exists a discriminating attribute in the matrix
- Compare the cost of no discrimination to the combined cost of discriminating and querying
- Since we cannot know the results of the discrimination, use the average estimated cost
- Potentially relevant sources: S = S1,...,S6
- Discriminating queries: R1, R2
- Possible plans: S, R1 S', R2 S'', R1 R2 S'''
  - R1: {{S1, S2}, {S3, S4, S5}, {S6}}
  - R2: {{S1}, {S2, S3}, {S4, S5, S6}}
  - R1 R2: {{S1}, {S2}, {S3}, {S4, S5}, {S6}}

Plan without Sensing

Retrieve GNP where Org=NATO from Afghanistan Page of the World Factbook

... 267 countries ...

Retrieve GNP where Org=NATO from Zimbabwe Page of the World Factbook

Union GNP of Countries where Org=NATO

Average GNP of Countries where Org=NATO
Plan with Sensing

Outline

- Information Gathering
  - Planning for Information Gathering
    - View Integration
    - Query Reformulation
    - Source Capabilities
  - Optimizing Information Gathering Plans
    - Removing Redundant Sources
    - Optimizing Sources and Queries
  - Interleaving Planning and Sensing
    - Sensing to Handle Incomplete Information
    - Sensing to Optimize Plans
- Contingent Planning for Information Gathering
- Planning to Compose Web Sources
- Discussion
Contingent Planning for Information Gathering [Friedman & Weld ’97]

- Use subsumption relationships to make a plan more resource conscious
  - Determined based on LCW statements
- Execution policies:
  - Brute force – ignore subsumption and execute everything greedily
  - Aggressive – execute multiple alternatives and cancel others once a subsumed source is successful
  - Frugal – execute the most general source first and only execute others if it fails

Augmenting the Plans

- Contingent plans
  - Operator can fire when its guard is true
  - Status variable for each operator
    - Sleeping, running, failed, and done
  - Approach:
    - Nodes initialized to running
    - Running nodes fired when input is available
    - Update status based on guards
  - Guards
    - Aggressive policy:
    - Frugal policy:
      - Failed(Y)
Outline

- Information Gathering
  - Planning for Information Gathering
    - View Integration
    - Query Reformulation
    - Source Capabilities
  - Optimizing Information Gathering Plans
    - Removing Redundant Sources
    - Optimizing Sources and Queries
  - Interleaving Planning and Sensing
    - Sensing to Handle Incomplete Information
    - Sensing to Optimize Plans
  - Contingent Planning for Information Gathering
  - Planning to Compose Web Sources
  - Discussion

Composing Web Services

- Information sources only have inputs and outputs
  - Possibly with some additional constraints on those
- Services have:
  - Inputs and outputs
  - Preconditions and effects
- Could be cast as a traditional planning problem with preconditions and effects
- Example:
  - To purchase a book on Amazon has a precondition of having the money and has the effects of having the book and less money
- Services can be composed into compound services [McIlraith & Fadel, 2002]
  - Stored and reused similar to Macrops [Elkas, 1972]
Outline

- Information Gathering
  - Planning for Information Gathering
    - View Integration
    - Query Reformulation
    - Source Capabilities
  - Optimizing Information Gathering Plans
    - Removing Redundant Sources
    - Optimizing Sources and Queries
  - Interleaving Planning and Sensing
    - Sensing to Handle Incomplete Information
    - Sensing to Optimize Plans
  - Contingent Planning for Information Gathering
  - Planning to Compose Web Sources
- Discussion

Discussion

- Is this planning?
  - Not in the sense of composing sequences of actions with interacting effects
  - Certainly in the broader sense of formulating a scheme or program for the accomplishment or attainment of some goal
- Good ideas can be shared across fields
  - Planning by rewriting based on traditional approaches to query planning
- Lots of interesting problems with real world applications
  - Optimizing the plans (e.g., interleaving sensing actions)
  - Interleaving source selection and plan optimization
  - Efficient execution of the plans (next class)
Bibliography

- Planning for Information Gathering
  - View Integration

- Traditional Planning Approaches
Bibliography

• Optimizing Information Gathering Plans
  • Removing Redundant Sources
  • Optimizing Sources and Queries

• Interleaving Planning and Sensing
  • Sensing to Handle Incomplete Information
  • Sensing to Optimize Plans
Bibliography

• Contingent Planning for Information Gathering

• Planning to Compose Web Sources