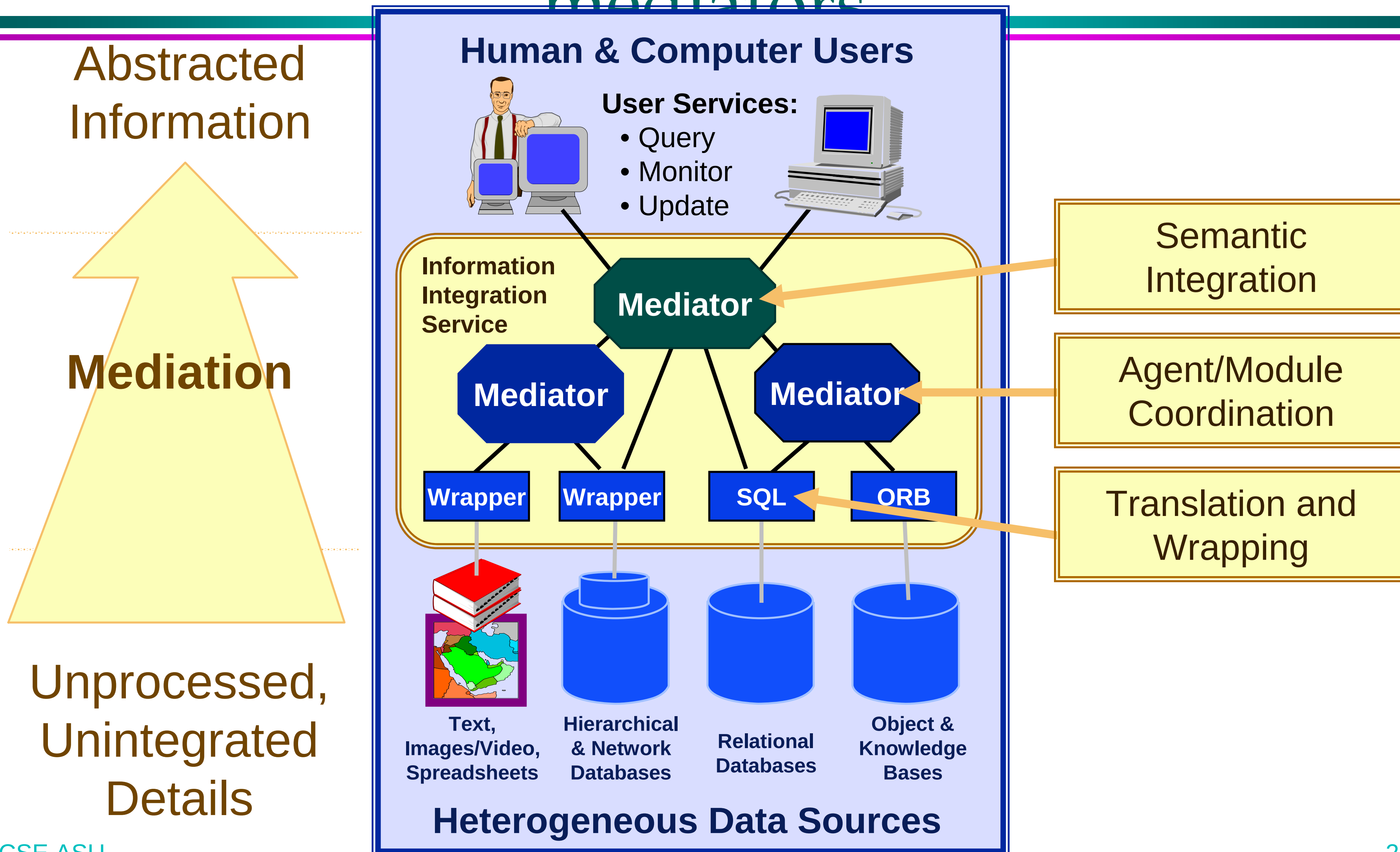
Learning for Semantic Query Optimization in Information Mediators

Chun-Nan Hsu
Dept of Computer Science & Engineering
Arizona State University
USA

Architecture of information mediators

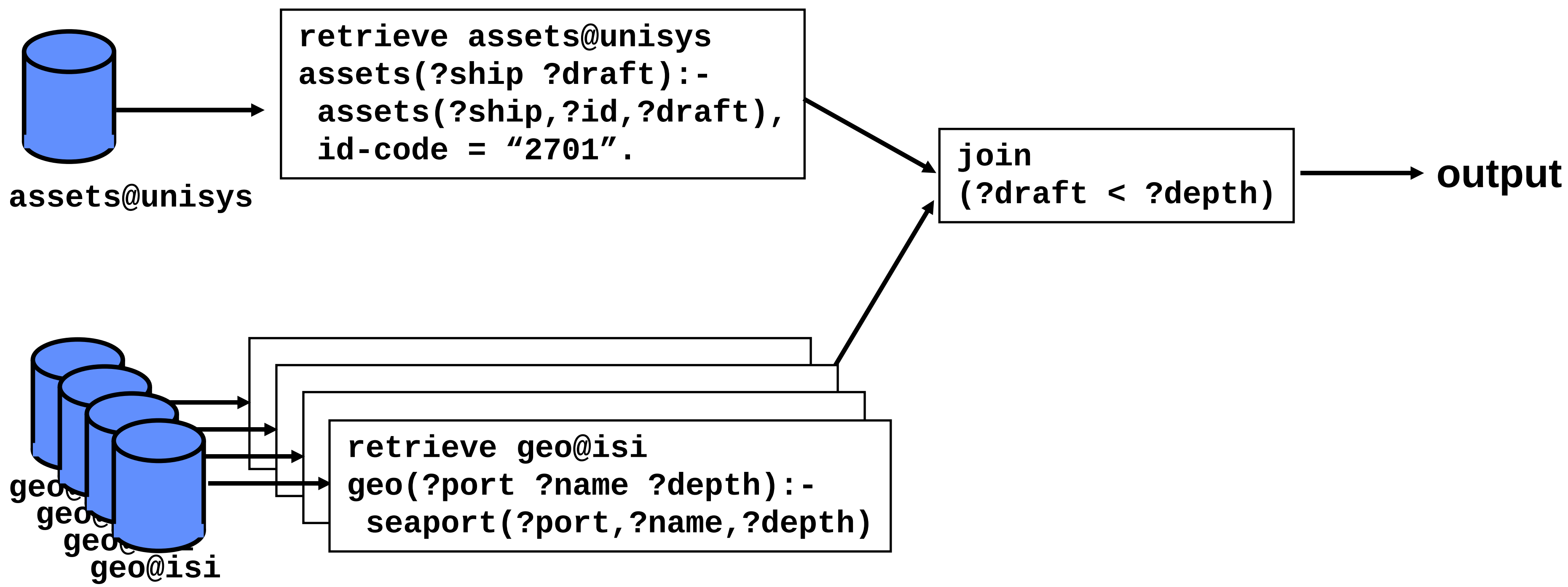


Information mediators

- Flexible integration of heterogeneous information sources (databases, texts, web pages etc.)
- Key ideas:
  - users access data through a ***domain model***
  - information sources represented by a ***source model***
  - the mediator ***reformulates*** domain model query into source model sub-queries
  - the mediator constructs a ***query plan*** that determines the orders of data flow and execution to retrieve data
- Enable new applications of information systems
  - E-commerce, global health-care IS, etc.

Query planning in information mediators

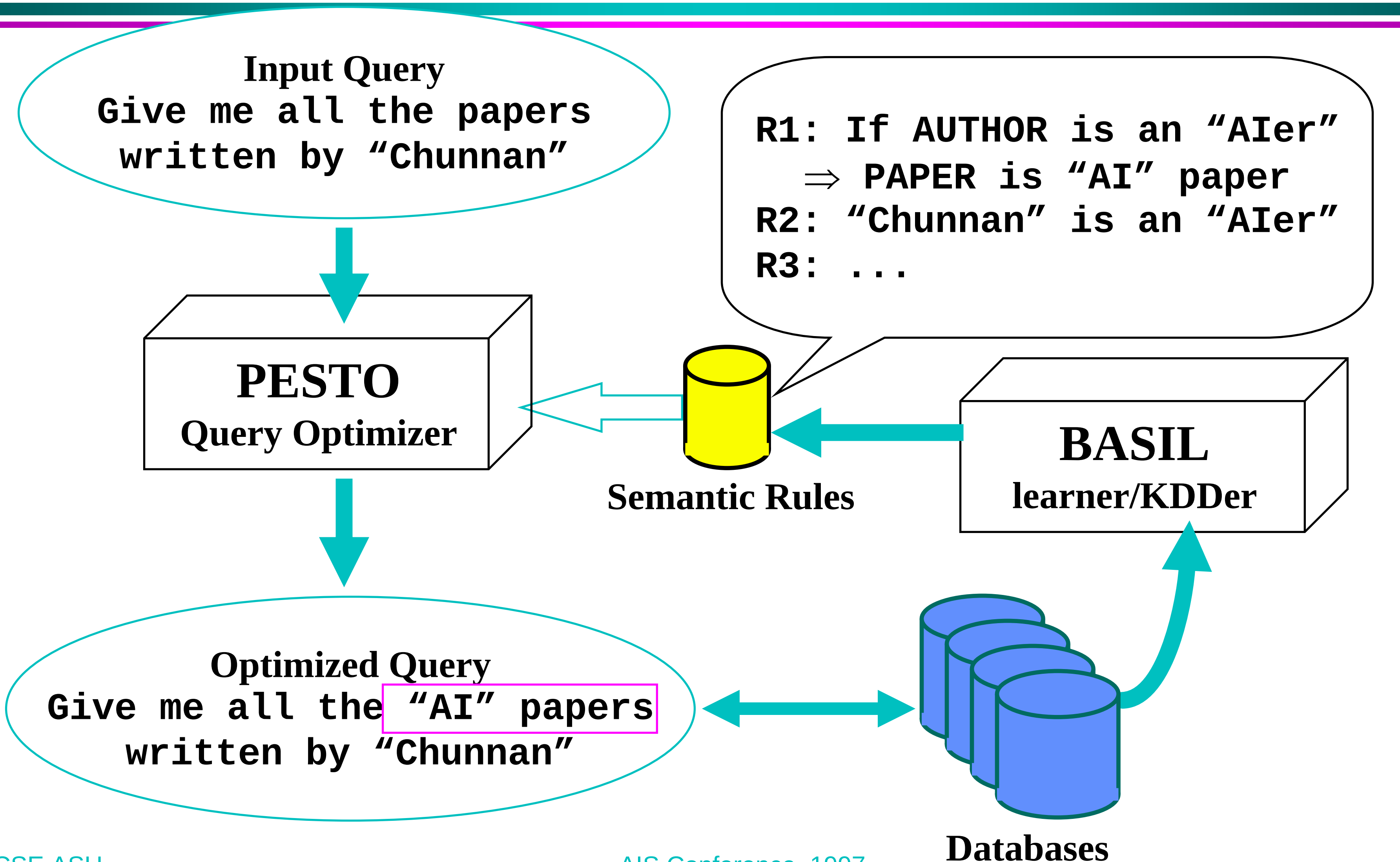
- Query: Retrieve seaports deep enough for ship “2701”.



Latest work in information mediators

- IM
  - » Levy, Srivastava, Kirk, et al. At AT&T Lab
  - » query reformulation, relevant source selections
- TSIMMS
  - » Hammer, Garcia-Molina, Papakonstantinou, Ullman at Stanford
  - » object-based data modeling
- SIMS
  - » Arens, Knoblock, Chunnan Hsu, et al. at ISI of USC
  - » flexible query planner, ***adaptive semantic query optimizer***

Basic idea of adaptive semantic query optimization



Novel features and contributions of PESTO

- Use more expressive relational rules NEW
- Optimize a larger class of queries NEW
  - » queries with arbitrary join topology
  - » joins with multiple comparand attributes
  - » unions, intersections, other set operators
- Therefore…
  - » detect more optimization opportunities
  - » execute queries faster
- See
  - » Hsu & Knoblock 93 (CIKM93)
  - » Hsu & Knoblock 97 (Submitted to IEEE TKDE)

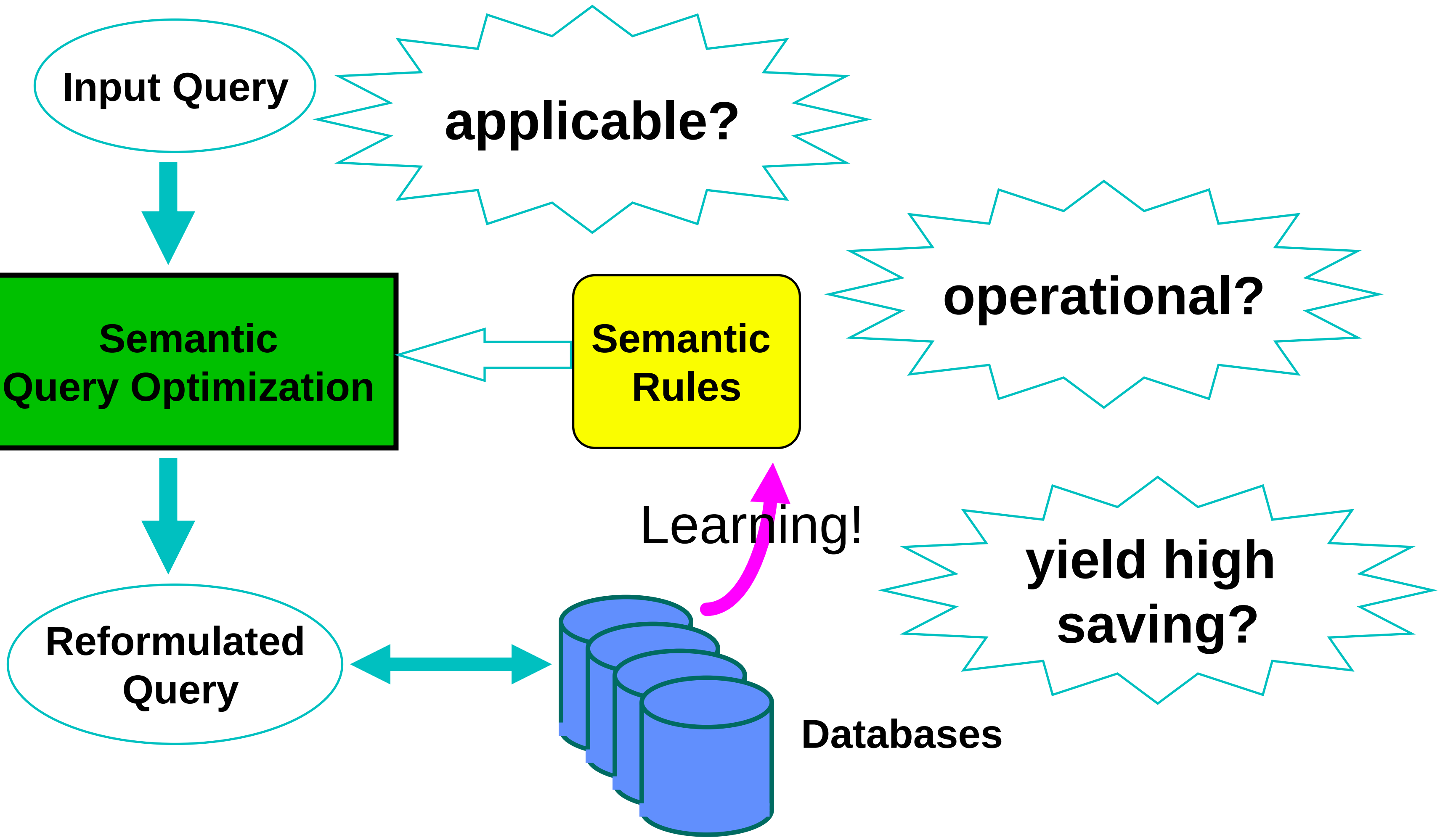
Using relational rules in semantic query optimization

- Range rules are propositional
  - » IF seaport(?port-name,?city,?storage,_,_) $\wedge$ city(?city,“Malta”,_,_)

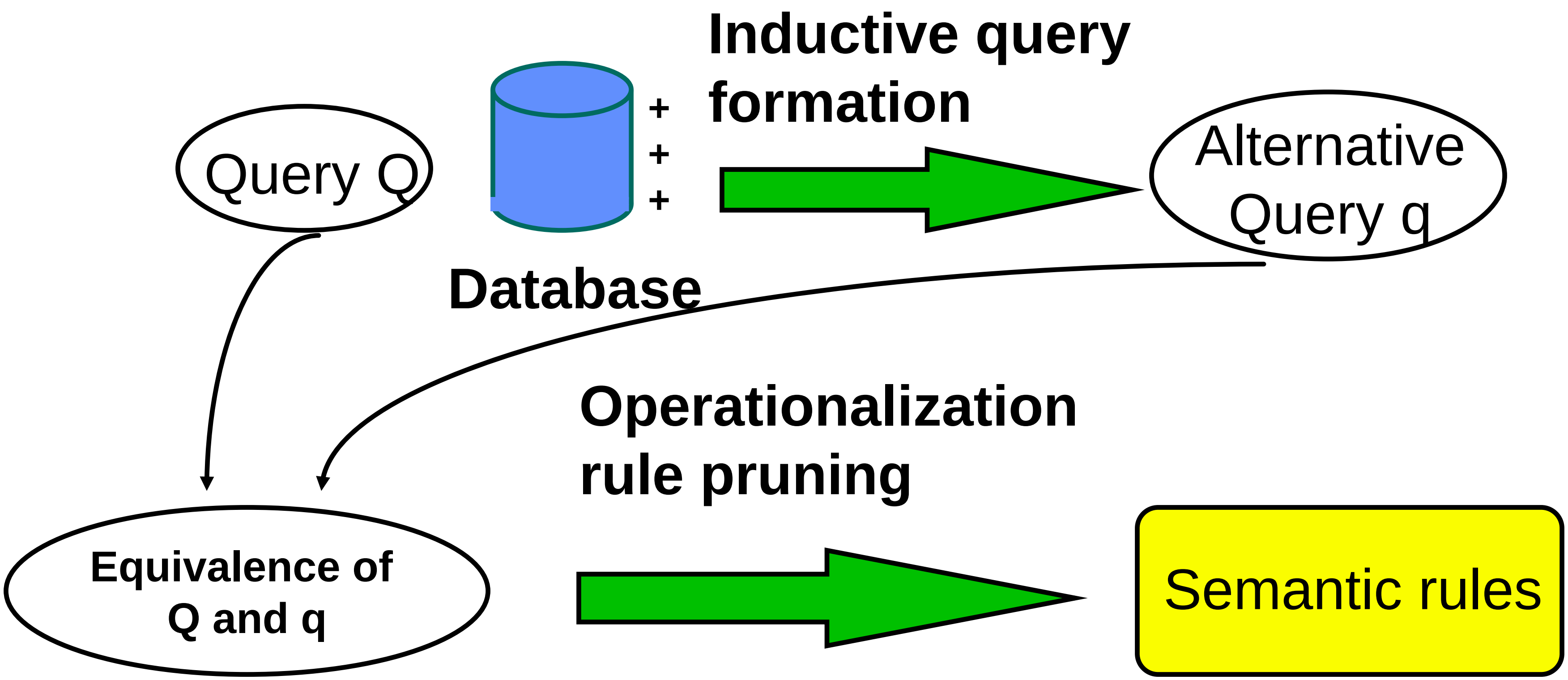
    $\Rightarrow$ **?storage > 2,000,000**
- Relational rules are first-ordered, predicate logic
  - » IF city(?city,?population,_,_) $\wedge$?population > 3,000,000

    $\Rightarrow$ **airport(?airport-name,?city,_,_)**
- Relational rules are useful in detecting unnecessary relational joins
  - » the dominant cost factor of query execution

Desiderata of learning



Induce alternative query and operational rules



Inductive formation of efficient equivalent query

Database DB:

A1 *	A2	A3	
A	1.5	2	-
B	1.8	2	-
C	0.7	2	+
B	1.4	2	-
B	0.8	1	-
C	0.6	2	+
A	1.6	2	-
A	2.8	2	-

Candidate sub-goals:

Candidates	gain	cost	h	
?A2=0.7 or 0.6	6	16	0.38	
0.5 < ?A2 < 1	5	16	0.31	
?A2 < 1	5	8	0.62	
?A3 = 2	1	8	0.12	
?A1 = "C"	6	1	6.00	*

Induced new query: Q'(?A1,?A2,?A3):-
DB(?A1,?A2,?A3), <u>?A1 = "C"</u>. (cost=1)

Input query: **Q(?A1,?A2,?A3):-**
DB(?A1,?A2,?A3), ?A2 < 1, ?A3 = 2. (cost=9)

Induce operational rules

- Induce an equivalent query Q’ for Q from data

  Q(?A1,?A2,?A3) :- DB(?A1,?A2,?A3), ?A2 < 1, ?A3 = 2.

  Q’(?A1,?A2,?A3) :- DB(?A1,?A2,?A3), ?A1 = “C”.

- Equivalence of Q’ and Q:

  $DB(?A1,?A2,?A3) \wedge (?A1 = \text{“C”})$

  $\Leftrightarrow DB(?A1,?A2,?A3) \wedge (?A2 < 1) \wedge (?A3 = 2)$

- Derive Rules:

  $DB(?A1,?A2,?A3) \wedge (?A1 = \text{“C”}) \Rightarrow (?A2 < 1)$

  $DB(?A1,?A2,?A3) \wedge (?A1 = \text{“C”}) \Rightarrow (?A3 = 2)$

  $DB(?A1,?A2,?A3) \wedge (?A2 < 1) \wedge (?A3 = 2) \Rightarrow (?A1 = \text{“C”})$

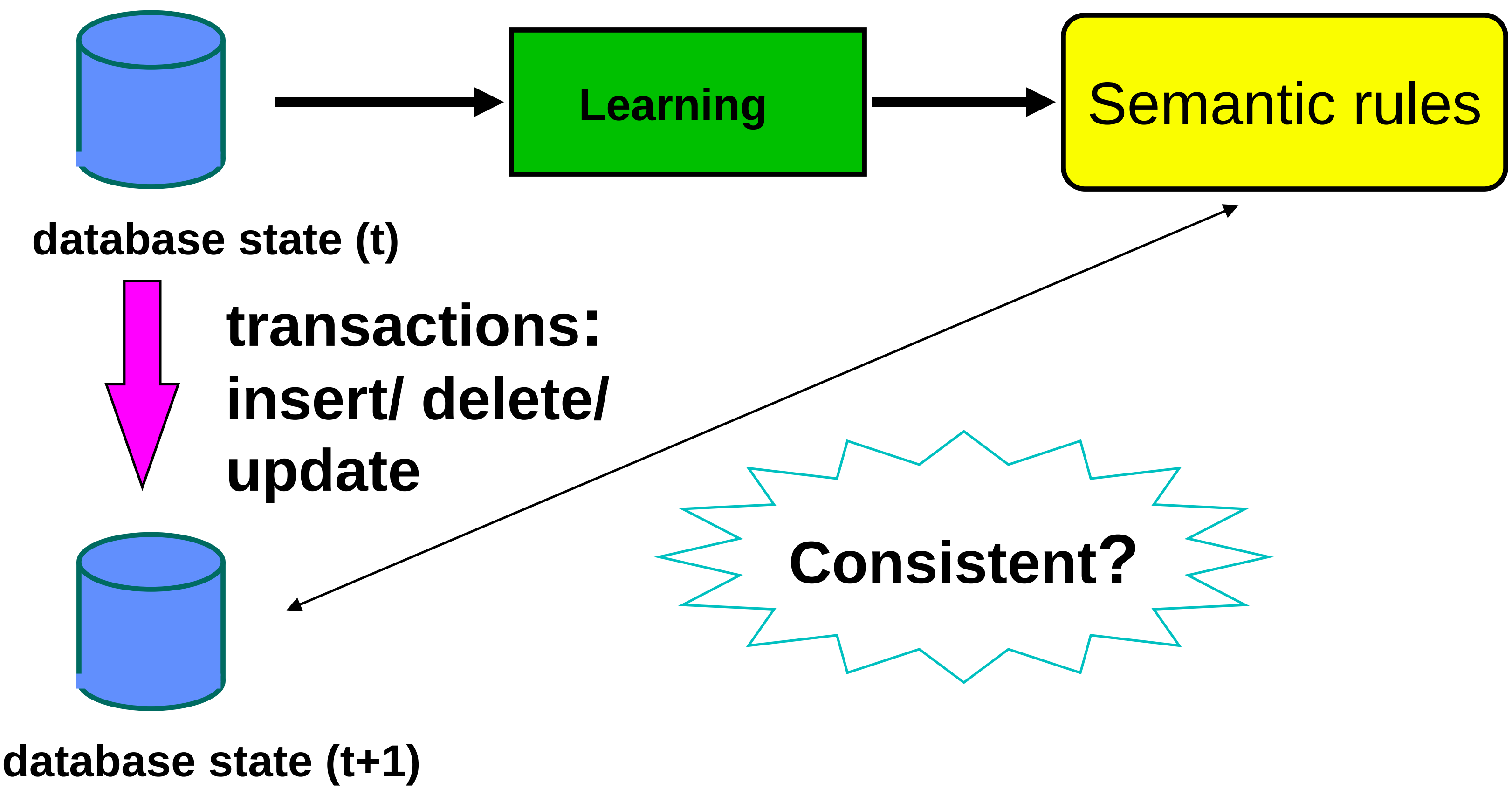
Learning relational rules

- Apply ***Inductive logic programming*** techniques (e.g., FOIL by Quinlan, 1990) in alternative query formation and operationalization
- Key ideas:
  - » construct ***database sub-goals*** (e.g., db(?x,?y)) as well as ***built-in sub-goals*** (e.g., ?x > 100) as candidates
  - » use uniform evaluation heuristics for both types of sub-goals
  - » use a join-path graph to assure that resulting rules are valid in operationalization
- See
  - » Hsu & Knoblock, 1994, Machine Learning Conference
  - » Hsu & Knoblock, 1996, New KDD book, MIT Press

Novel features and contributions of BASIL

- Learn relational rules
- Adapt to changes of query patterns
- Yield effective rules for optimization
- Yield *ROBUST* rules, so that they will remain valid after database changes NEW
- About robustness of knowledge, See
  - » Hsu & Knoblock 1995, KDD Conference
  - » Hsu & Knoblock 1996, AAAI Conference
  - » Hsu & Knoblock 1997, (invited to submit to new Data Mining / KDD journal)

Dealing with database changes



Robustness of knowledge

- Intuitively, robustness can be estimated as

$$\frac{\text{\# of database states consistent with the rule}}{\text{\# of possible database states}}$$

- Alternatively, a rule is *robust* given a current database state if transactions that invalidate the rule are unlikely to be performed.
- New definition of robustness is $1 - Pr(t|d)$
  - » t: transactions that invalidate the rule are performed
  - » d: database is in the current database state

Robustness estimation

- Step 1: Identify the class of invalidating transactions
- Step 2: Decompose each transaction into local variables based on a ***Bayesian network model*** of database transactions
- Step 3: Estimate local probabilities using
  - » *Laplace Law of Succession* (Laplace 1820) or
  - » *m-Probability* (Cestnik & Bratko 1991)
- Use information available in a database:
  - » transaction log
  - » expected size of tables, attribute range, distribution

Step 1: Find Transactions that Invalidate the Input Rule

- R1: The latitude of a Maltese Geographic location is greater than or equal to 35.89.

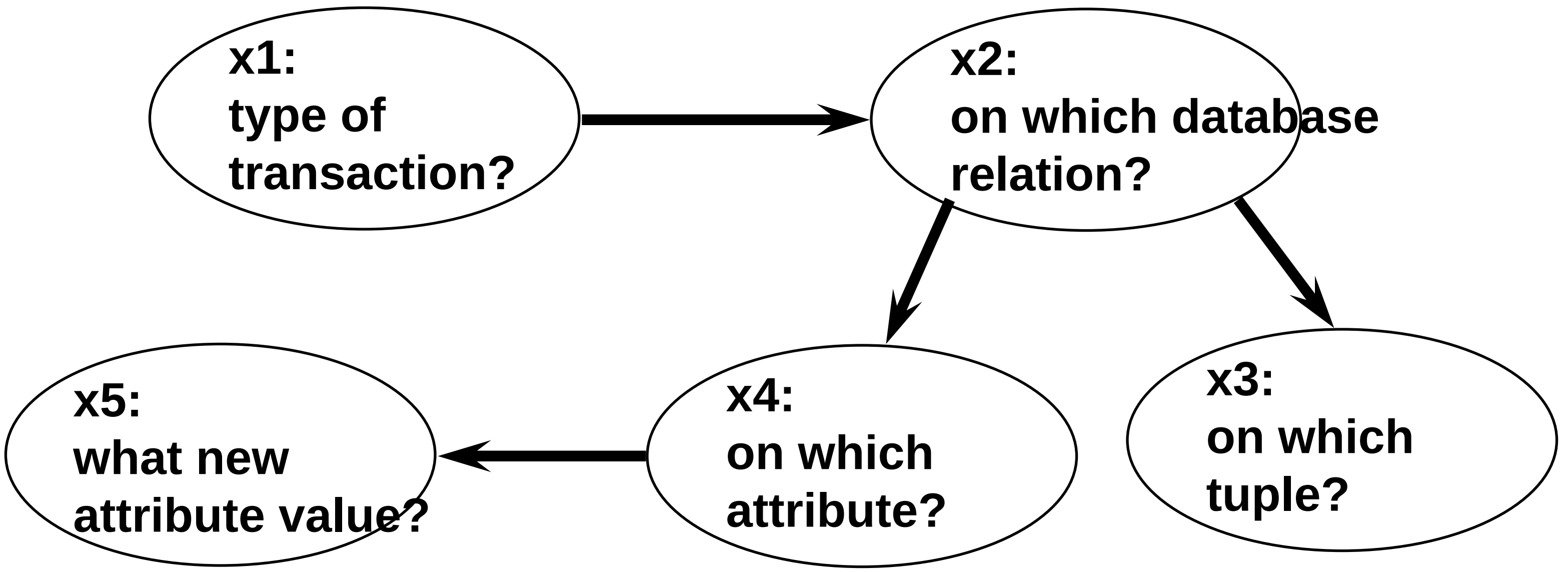
geoloc(_,_,?country,?latitude,_) & (?country = “Malta”)

$\Rightarrow$?latitude > or = 35.89

- Transactions that invalidate R1:
  - » T1: One of the existing tuples of geoloc with its country = “Malta” is updated such that its latitude < 35.89
  - » T2: Insert an inconsistent tuple...
  - » T3:Update a tuple whose latitude < 35.89 into “Malta”
- Robust(R1) = 1 - Pr(t|d)

  = 1 - (Pr(T1|d) + Pr(T2|d) + Pr(T3|d))

Step 2: Decompose the Probabilities of Invalidating Transactions



Bayesian network model of rule invalidating transactions

$$
\begin{aligned}
Pr(t|d) &= Pr(x1,x2,x3,x4,x5|d) \\
&= Pr(x1|d)\ Pr(x2|\ x3,d)\ Pr(x3|x2,d)\ Pr(x4|\ x2,d)\ Pr(x5|\ x4,d)
\end{aligned}
$$

Step 3: Estimate Local Probabilities

- Estimate local probabilities using *Laplace Law of Succession* (Laplace 1820)

$$\frac{r + 1}{n + k}$$

- Useful information for robustness estimation:
  - » transaction log
  - » expected size of tables
  - » information about attribute ranges, value distributions
- When no information is available, use database schema information

Example of Robustness Estimation

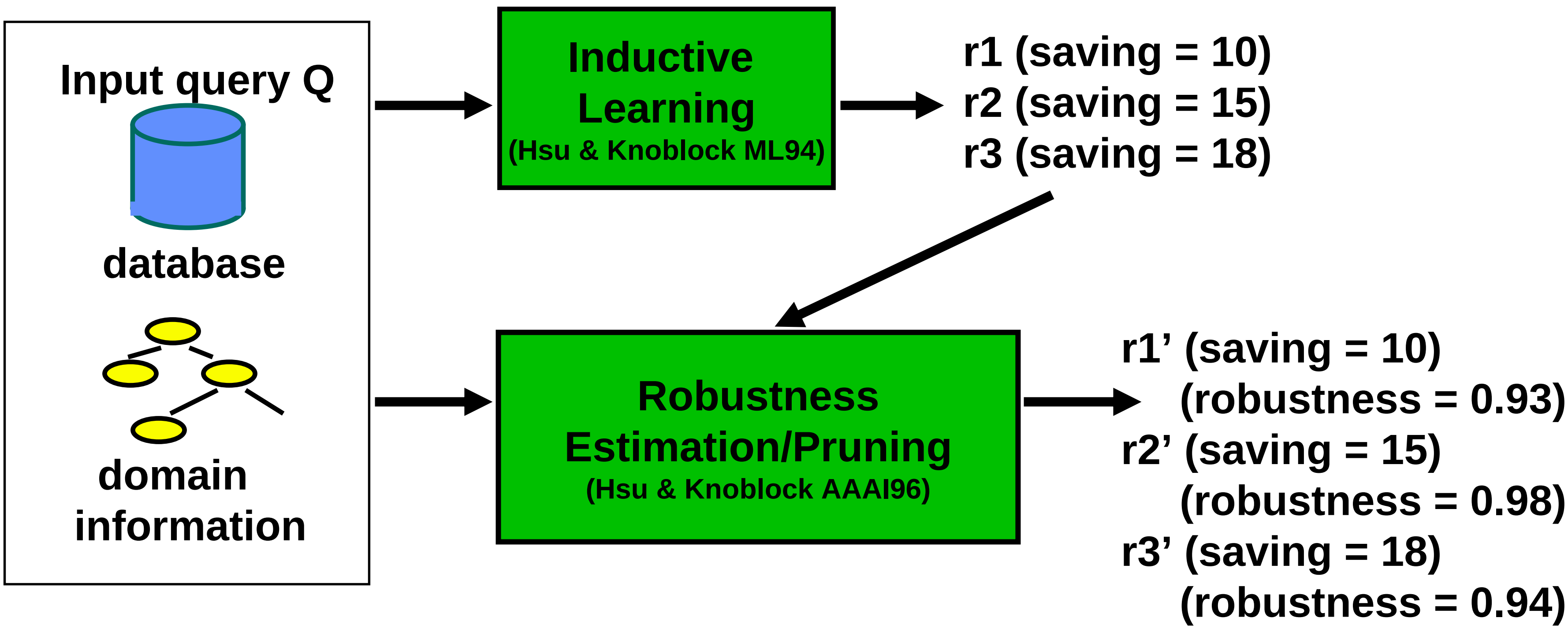
- R1: geoloc(_,_,?country,?latitude,_) & (?country = “Malta”) $\Rightarrow$?latitude > or = 35.89
- T1: One of the existing tuples of geoloc with its country = “Malta” is updated such that its latitude < 35.89
  - » p1: update? 1/3 = 0.33
  - » p2: geoloc? 1/2 = 0.50
  - » p3: geoloc, country = “Malta”? 4/80 = 0.05
  - » p4: geoloc, latitude to be updated? 1/5 = 0.20
  - » p5: latitude updated to < 35.89? 1/2 = 0.5
- Pr(T1|d) = p1 * p2 * p3 * p4 * p5 = 0.008
- Pr(T2|d) and Pr(T3|d) can be estimated similarly

Example (cont.): When additional information is available

- Naive
  - » p1: update? $1/3 = 0.33$
- Laplace
  - » p1: update? $$\frac{\text{\# of previous updates} + 1}{\text{\# of previous transactions} + 3}$$
- m-Probability (Cestnik & Bratko 1991)
  - » p1: update? $$\frac{\text{\# of previous updates} + m * Pr(U)}{\text{\# of previous transactions} + m}$$
  - » m is an expected number of future transactions
  - » Pr(U) is a prior probability of updates

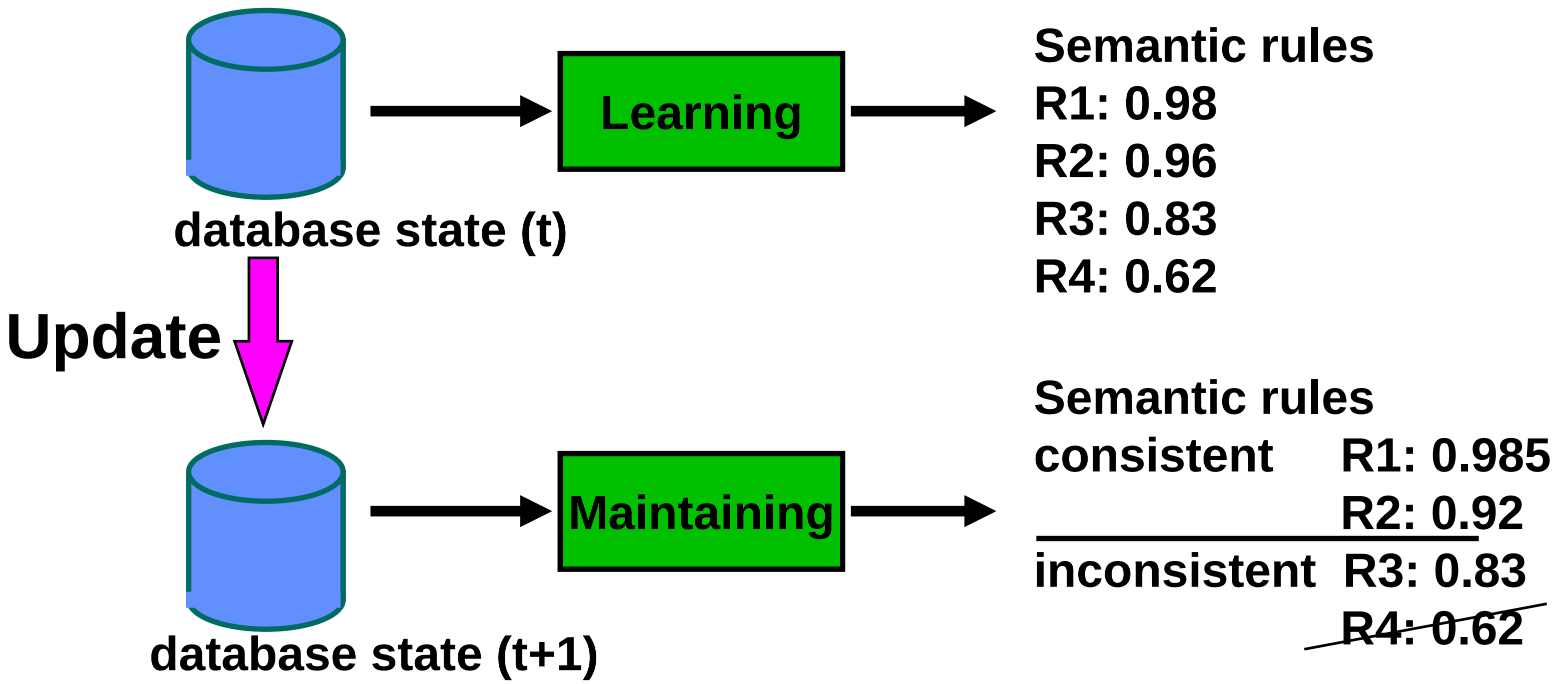
Applying robustness estimation in rule induction

- Learning effective and robust rules



Rule maintenance

- Rule Maintenance: Identify and repair inconsistent rules



Finale

- PESTO saves up to 97%, and 41+% on average for simple multi-database query plans
- Higher saving expected for complex, expensive query plans to web sources
- All rules learned automatically by BASIL
- Totally invisible from users
- Will be essential of information mediators like SIMS
- For more information:
  - » Chunnan Hsu, PhD Thesis, 1996, U of Southern California
  - » mailto: chunnan@asu.edu
  - » http://www.isi.edu/sims/chunnan/