Making a Case for Case-based Reasoning

Qiang Yang
Simon Fraser University
http://www.cs.sfu.ca/~qyang

The Setting

- The mission: NSERC industry chair program
- The problem
- Elevator test
- The methodology
  - Application domain
  - Research problem

The Problem

- Rogers cable-TV has hundreds of customer service representatives (CSR's) who solve customers' cable-TV and internet problems on the phone (call center).
- If a problem cannot be solved, Rogers must send out a truck to customer's site --> truck roll.
- Truck rolls, and training, are expensive!

Problem Resolution Example

- **Customer**: “my VCR is not working”
- **CSR** “do you have a recording problem”
- **Customer**: “yes. I cannot record channel 13”
- **CSR**: “first, turn your TV to channel 3. Now tell me what you see on your TV screen”
- **Customer**: “I see the music channel”
- **CSR**: “OK, now change to channel 13 through the remote…, finally, unplug and then plug
- the TV”
- **Customer**: “OK, problem solved”

Domain Problem

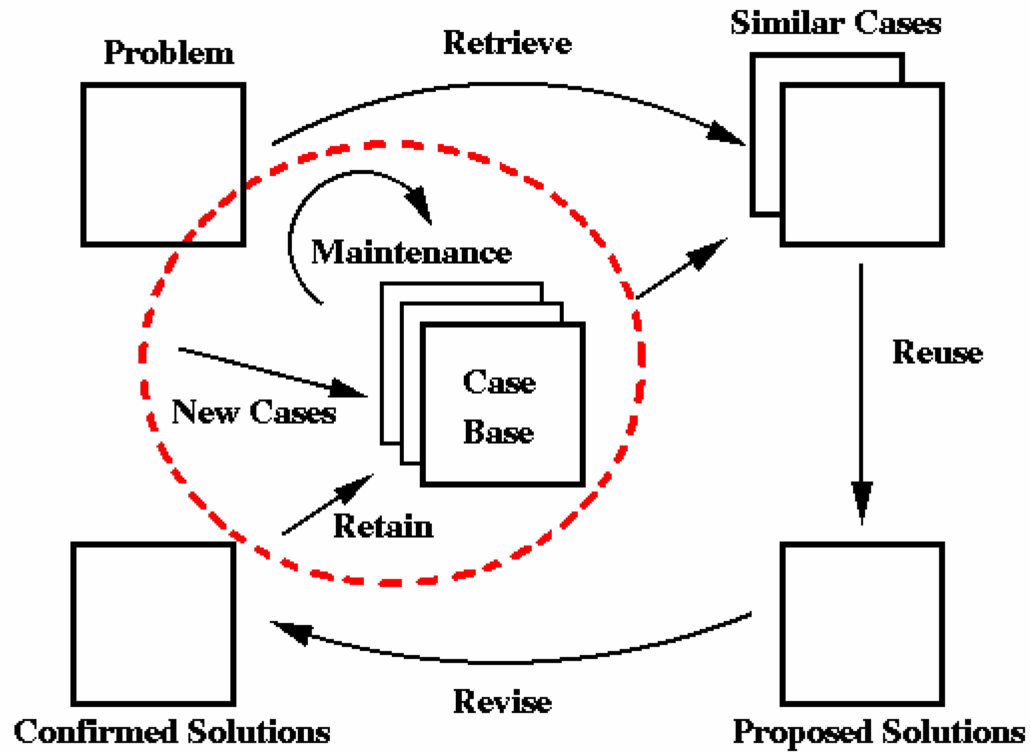
- Problem: cache and re-use the knowledge through small and focused databases and interactive retrieval
- Requirements: no formal domain model, knowledge change at fast rate, knowledge highly typical
- Solution: case based reasoning

Case Representation

- Case name: VCR not taping required channels
- Description: most likely, VCR hookup problems
  - Questions: "Does direct hookup of VCR help solve the prob?"
- Solution:
  1. Check that account is enabled for required channels
  2. Check that sub has required equipment, and is following correct recording procedures
  3. If problem continues, advise that the VCR is faulty and should be examined
- Multimedia attachment

Case Based Reasoning Cycle

- Create
- Maintain
- Retrieve
- Revise??



System Demo

CaseAdvisor is available at
http://www.cs.sfu.ca/~isa/isaresearch.html#systems

Problem 1: Unstructured Cases

Much of knowledge is stored in flat files (Text, Html, Etc)

Semi-structured Cases

- In help desk applications, knowledge is distributed among different data sources
  - User manuals
  - Database records
  - HTML files
- Cases are in semi-structured format: *<attributes, problem, solution, links…>*
- Changes are often incremental

Two Types of Cases

- Structured Cases

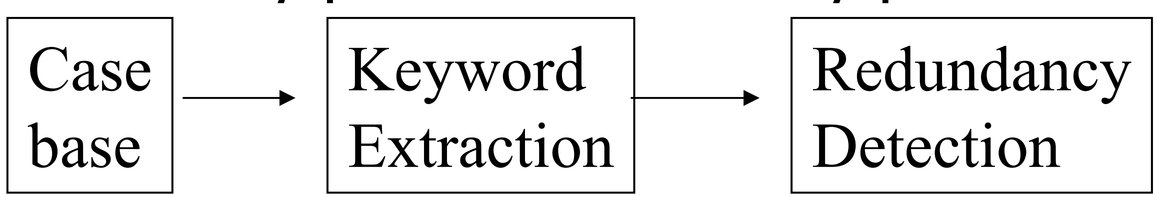
Case Id: 10056
Make: Honda
Model: Civic
Year: 1997
Price: $17 000
Number of Doors: 2
Engine Location: Rear
Engine Size: 420EL
Problem: Engine stalling
Validation: Condition of fuel injector.
Solution: Clean fuel injector.

- Unstructured Cases

Case Name: Income Funds
Case Solution: Income funds can be considered a core holding for almost all mutual fund investors. These mutual funds provide investores with a regular streeam of income, plus the potential for long-term growth. Thse are also known as "fixed income" funds. They include government bonds, corporate bonds and mortgages. The funds can also hold very short-term securites known as money market instruments. Because bonds pay interest, value tied to interest rates.

Information Retrieval

- *Task: detect cases that are similar in content*
- *Information Retrieval* (IR):
  - remove stop words
  - stem remaining terms
  - collapse terms using thesaurus
  - build inverted index
  - extract key words - build key word index
  - extract key phrases - build key phrase index



Keyword and Feature Classification

- Case Notation (P, Q, S are sets of keywords)
  - Problem Descriptions: P
  - Solution Qualifications: Q
  - Solutions: S
- Case<P, Q, S> means

  given(Q) and do(S) => solved(P)

Subsumption Rules

- Case 1 *subsumes* Case 2 if
  - Rule: P1 >= P2, Q1 <= Q2, S1 <= S2
  - Case 1 can solve all problems that Case 2 solves
  - Case 1 requires fewer preconditions and is more efficient
- Removing Case 2 does not affect the coverage of the case base!

Subsumption Example

- Case 1

Problem: fever, headache

Qualification: adult

Solution: take 2 Tylenol

- Case 2

Problem: fever

Qualification: adult

Solution: take 2 Tylenol, 2 aspirin

➡ Case 1 subsumes case 2

- Case 2 may be redundant, a candidate for removal

Empirical Testing

CaseAdvisor Redundancy Detection Module

- 210 cases generated from cable-TV domain
- 5 separate authors

	Identified	Not Identified	
Redundant	97	6	103
Not Redundant	20	87	107
	117	93	

Problem 1: Unstructured Cases

With Kersti Racine, MSc.

- ICCBR’97
- IEEE TKDE 2001

Problem 2: Case-base Coverage Problem

Lots of cases are repetitive, small variations of one anther

Maintenance Policies

- Given:

  - a large data base Z of (prob,sol) pairs

  - a constant K, the final size of a case base

  - a similarity metric defined by adaptation costs.

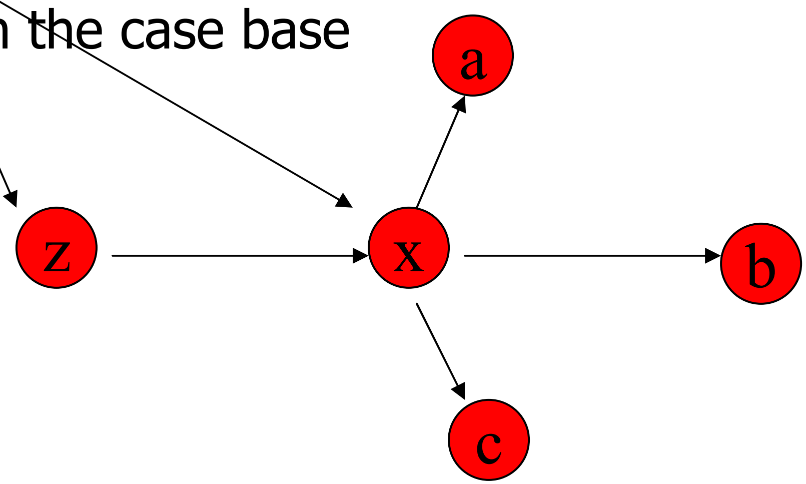
  - a frequency of problem occurrences
- Find a case base of size K with good **competence**
- Optimal solution is NP-complete
- Want: good approximate algorithm

Coverage of Cases

- Coverage(case) ={case’| Adaptable(case,case’)}
- Cases are classified into several classes:

  - **<u>Pivotal</u>**: not contained in the coverage of any other cases in the case base

  - **<u>Auxiliary</u>**: its coverage is contained in the coverage of some other case in the case base

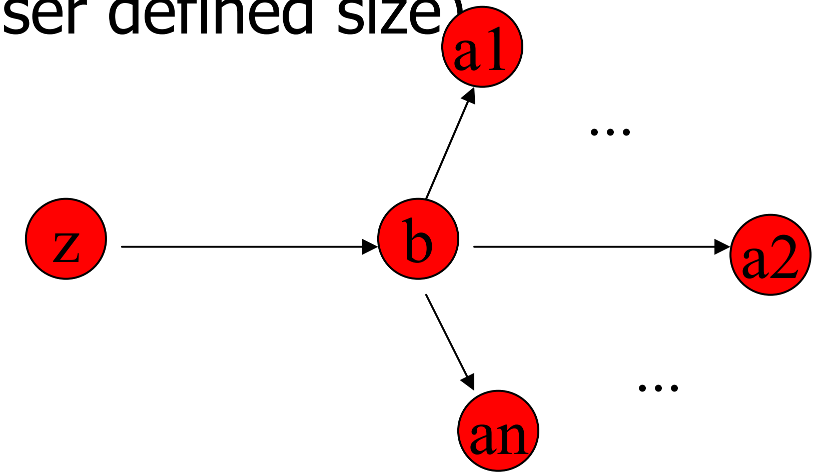


Smyth and Keane's Case Deletion Policy (IJCAI-95)

- Deletion Policy:
  - Delete auxiliary cases first
  - Delete support and spanning cases
  - Delete pivotal cases

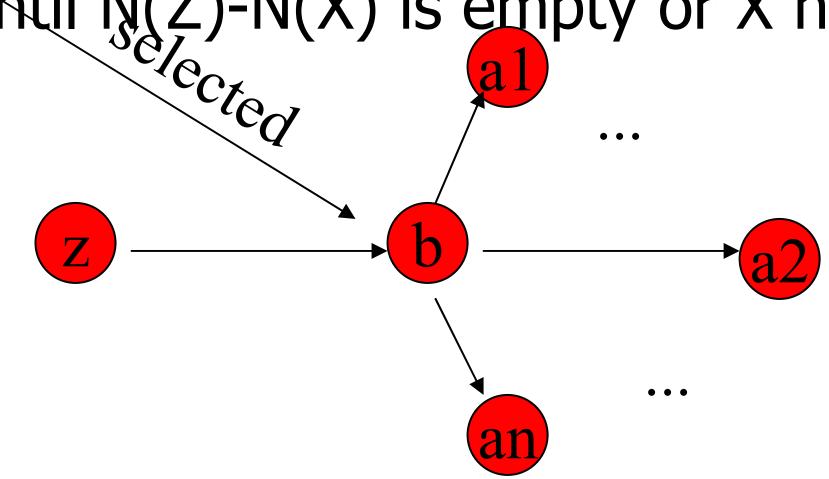
Until case base size is K (user defined size)

However, deletion-based policy can lose almost all coverage
(set K=1, case-base={Z}
coverage=1/(n+1)

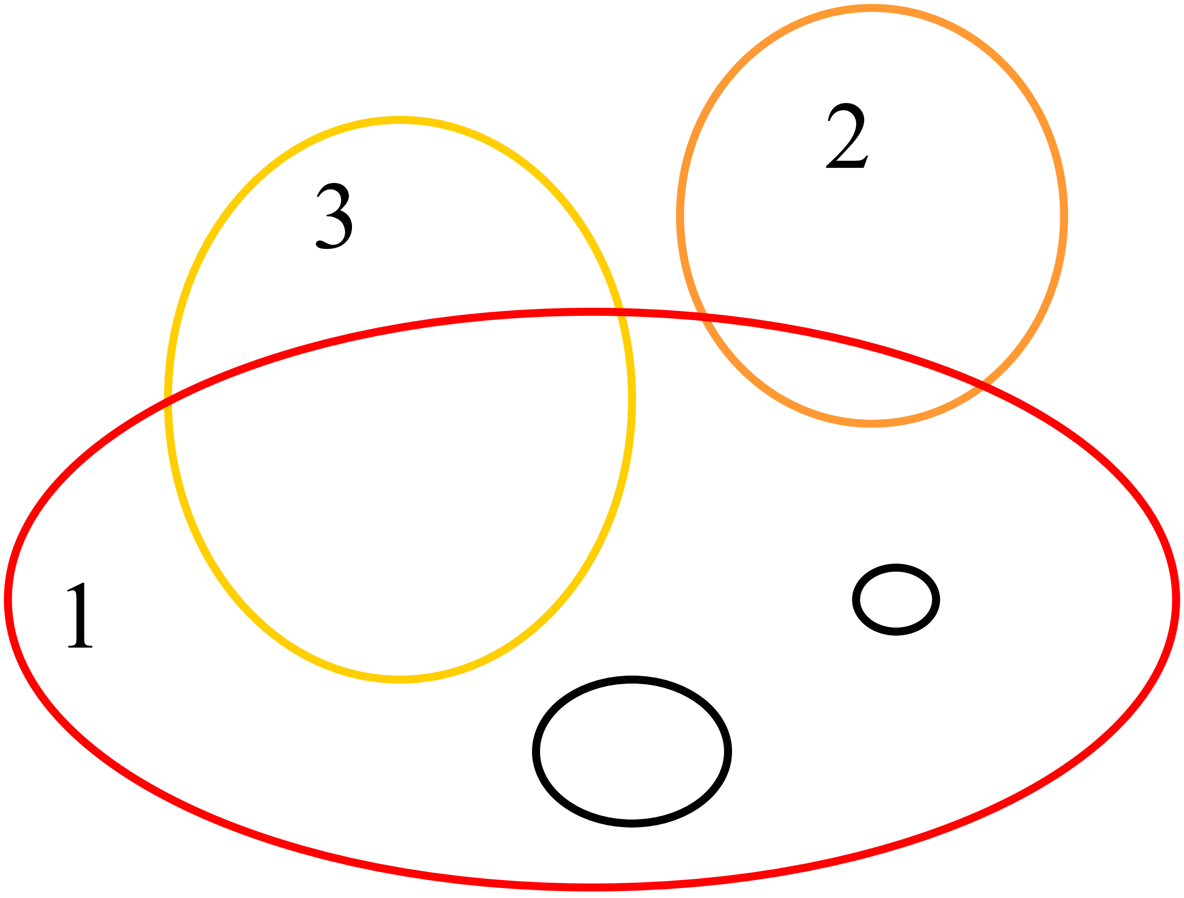


Our Case-Addition Policy

1. Find the coverage N(x) of every problem x in database Z; case base X={};
3. Select a case from Z-X with the **maximal benefit** with respect to N(X) and add it to X
4. Repeat step 3 until N(Z)-N(X) is empty or X has K elements



Case-Addition Policy



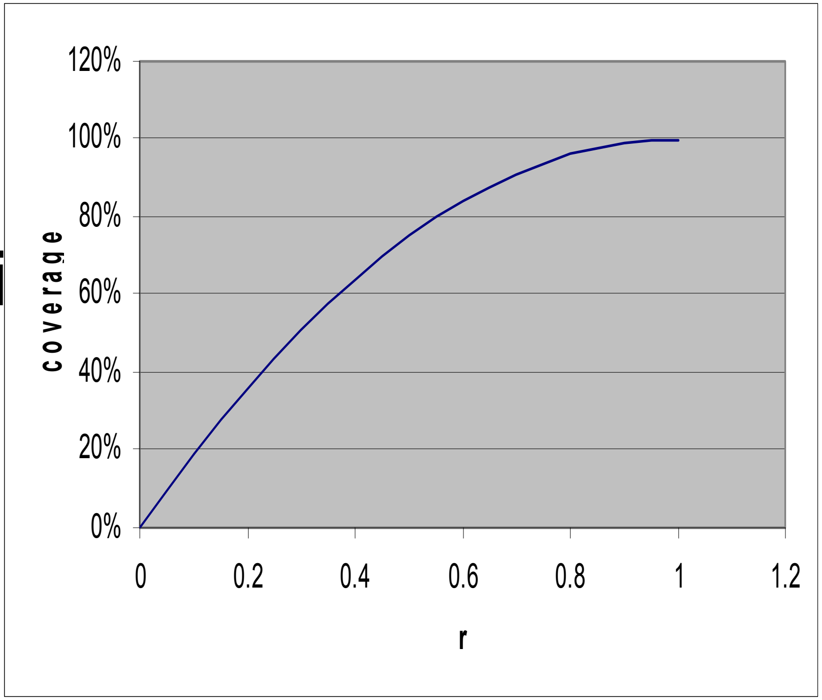
Competence Preserving Claim

- **Theorem**: The case-addition policy produces a case base X such that the coverage of X is no less than 63% of the coverage of an optimal case base

- Proof based on set-covering, also similar to one given by [Harinarayan, Rajaraman and Ullman 96] for data cube construction

How many cases are enough?

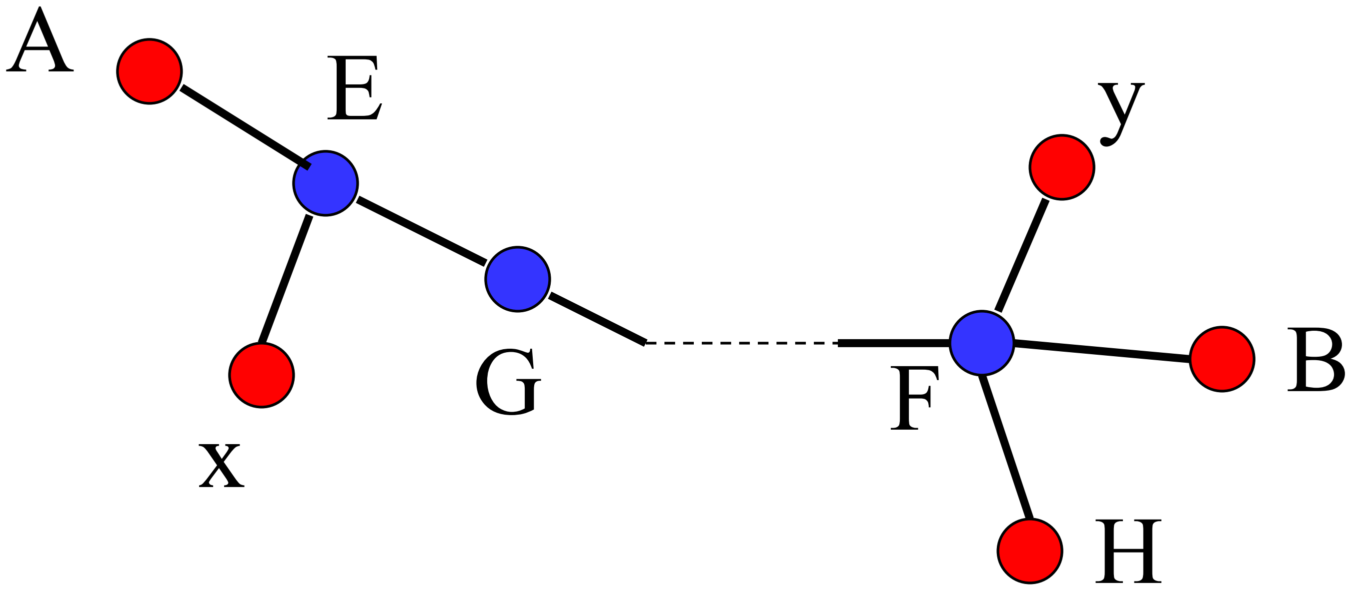
- Let the size of database be *n*; size of case base be *k*;
- Let $r=k/n$ be the rati
- Suppose when adding cases into a case, the benefits decreases *linearly*
- Then: **coverage=*r(2-r)***



How to compute case-coverage?

- Count the number of adaptation steps needed,
- State-based similarity metric for path planning:

Dist(x, y) = min # of steps added/deleted from x to y



Problem 2: Case-base Coverage Problem

Jun Zhu, MSc.

- IJCAI ’99
- Computational Intelligence Journal

Problem 3: Feature Weight Learning

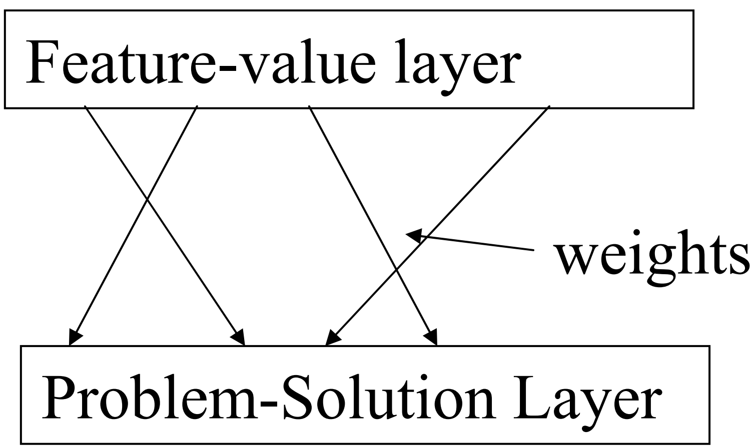
Experts pay attention to some problem features more than others

Maintaining Indexes

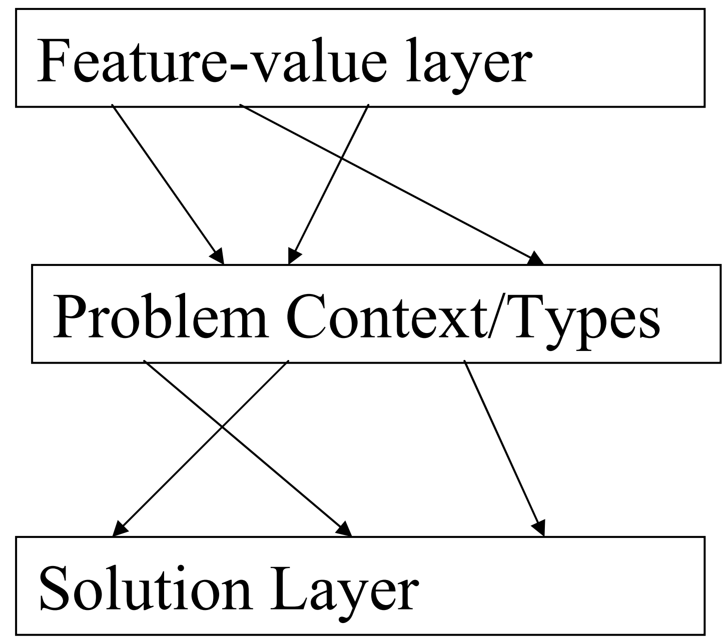
- Weights to question-answers set by domain expert may be inaccurate, change over time
- Adjust weights to refine case associations based on usage patterns
  - close the feedback loop
- Different type of users have different preferences, usage behavior
  - agents vs. customers visiting web site

Architectural Changes

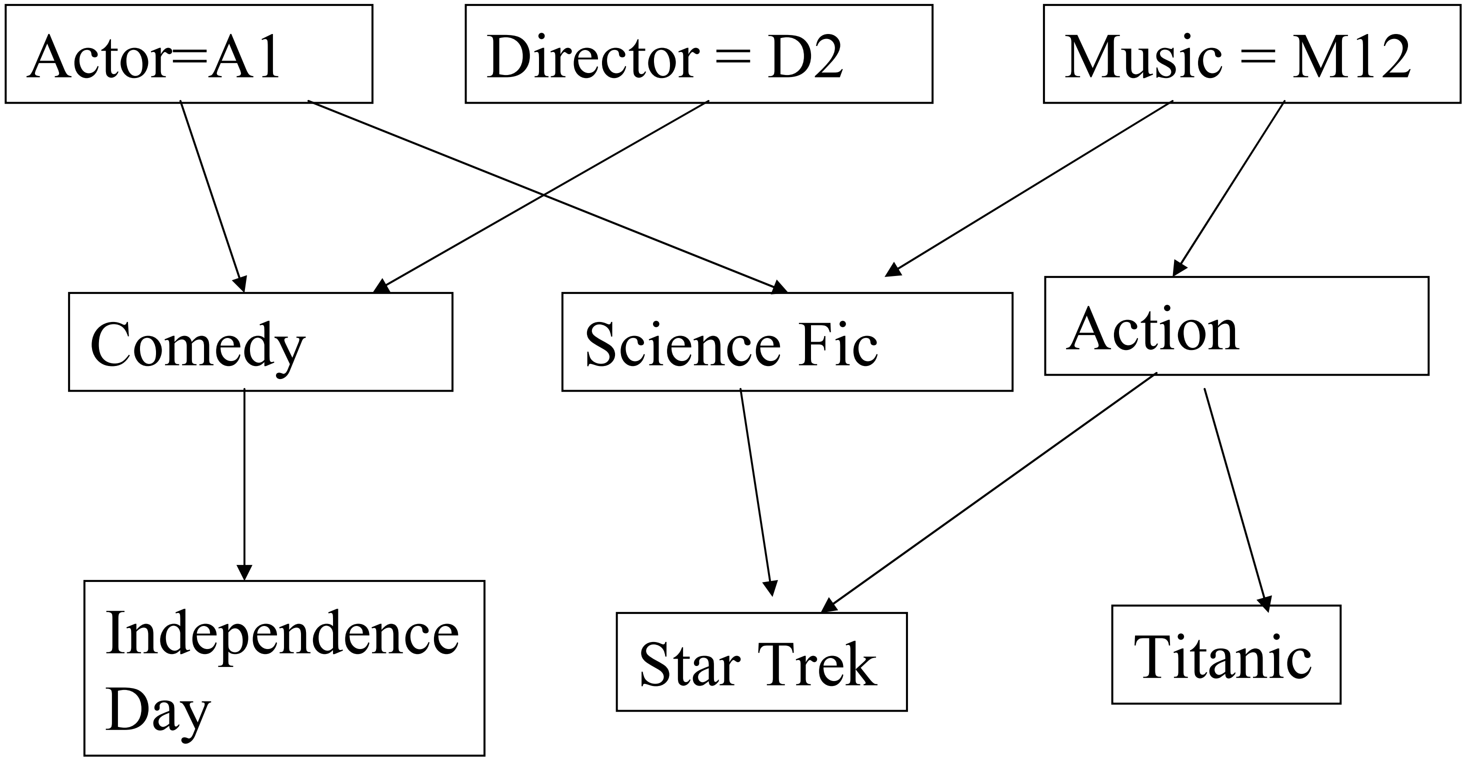
- Two layer case base



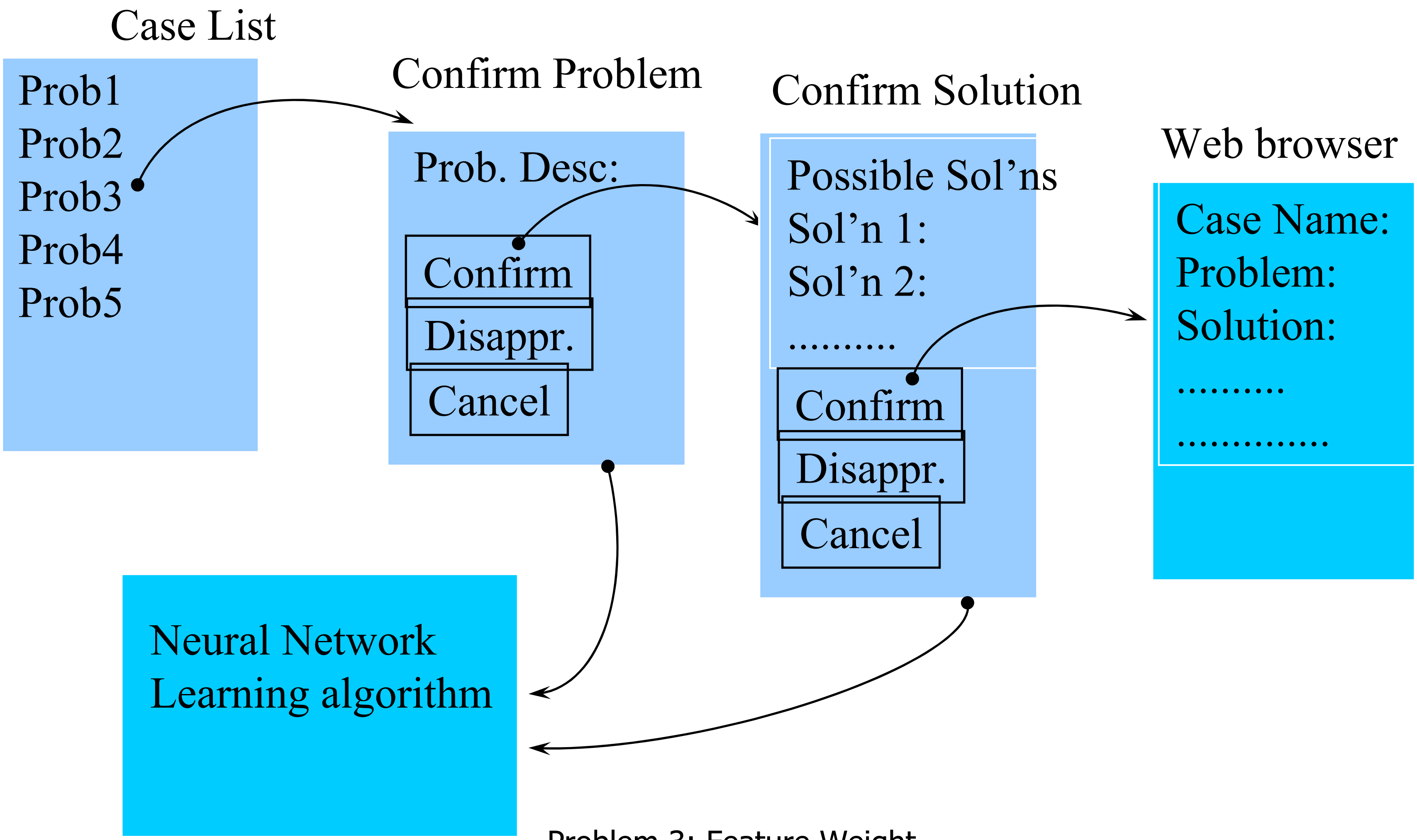
- Three Layer case base



A Video Rental Domain Ex



Problem Resolution and Learning

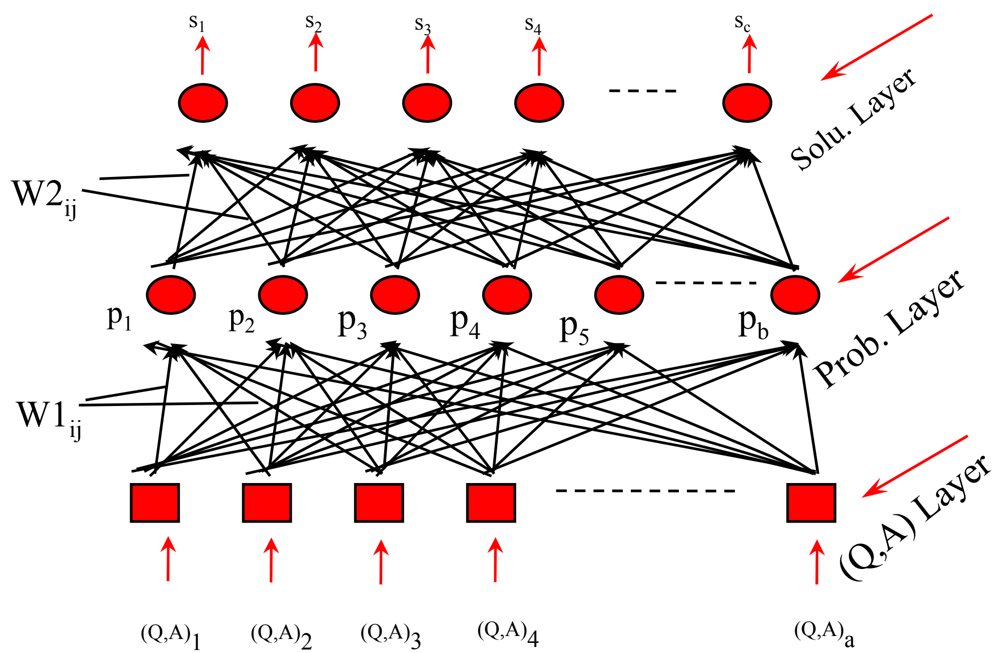


Back-propagation Network

Step 1:

$$S_j = \frac{1}{1+e^{-\sum_i w2_{ij} P_i}}$$

$$P_j = \frac{1}{1+e^{-\sum_i w1_{ij} (Q,A)_i}}$$



Step 2:

$$\delta 2_j = S_j(1-S_j)(y_j - S_j)$$

$$\delta 1_j = P_j(1-P_j)\sum_i \delta 2_i \cdot w2_{ji}$$

Step 3:

$$\Delta w2_{ij} = \eta \cdot \delta 2_j \cdot P_i$$

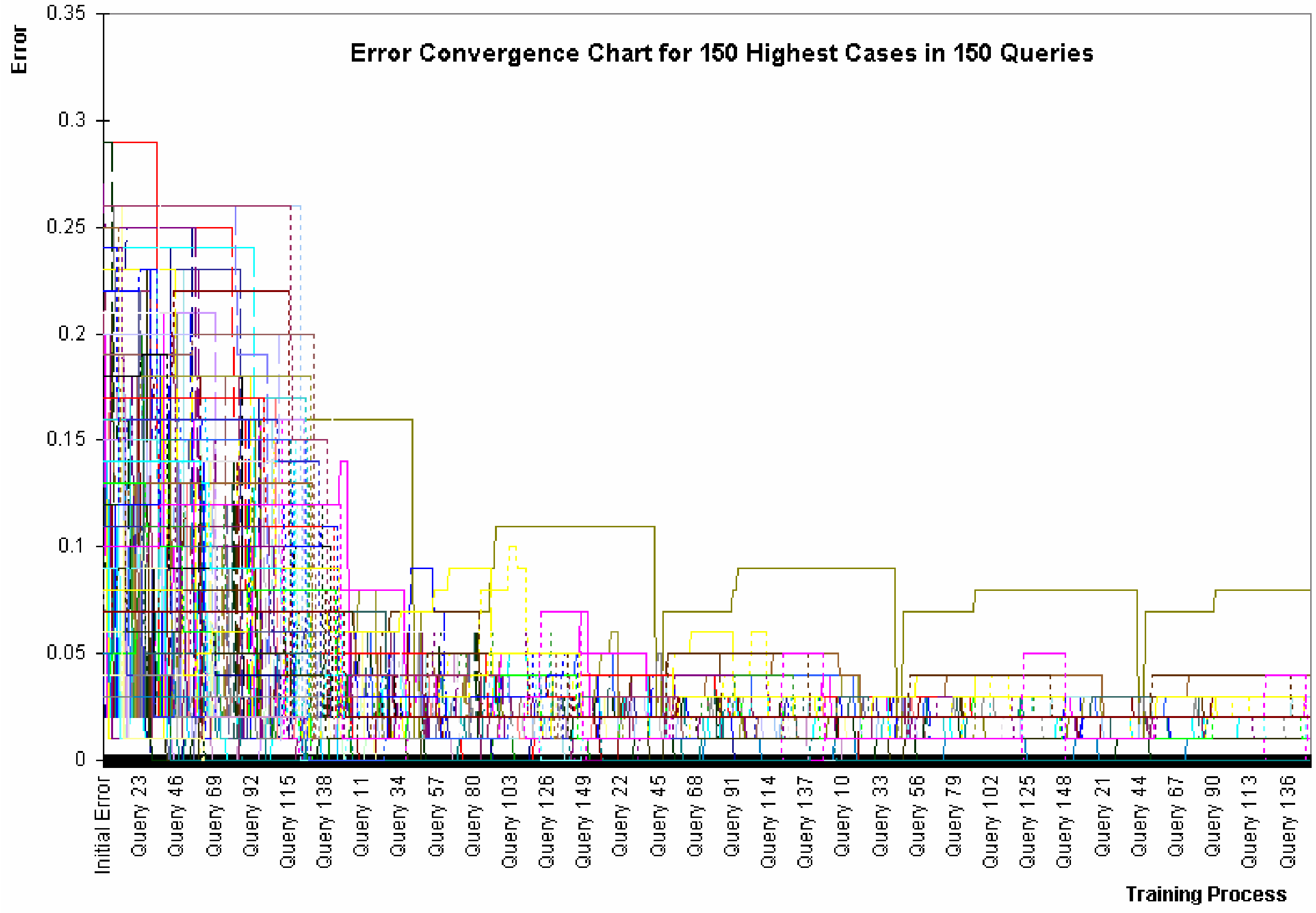
$$\Delta w1_{ij} = \eta \cdot \delta 1_j \cdot (Q, A)_i$$

y_j:the target output

Test the Index Learning Module:

- Rogers Cable-TV Case Base (30 Q/A)
- Video Rental Case Base (25 Q/A)
- UCI Data

Test Results



Training time: quadratic with CB-size

Problem 3: Feature Weight Learning

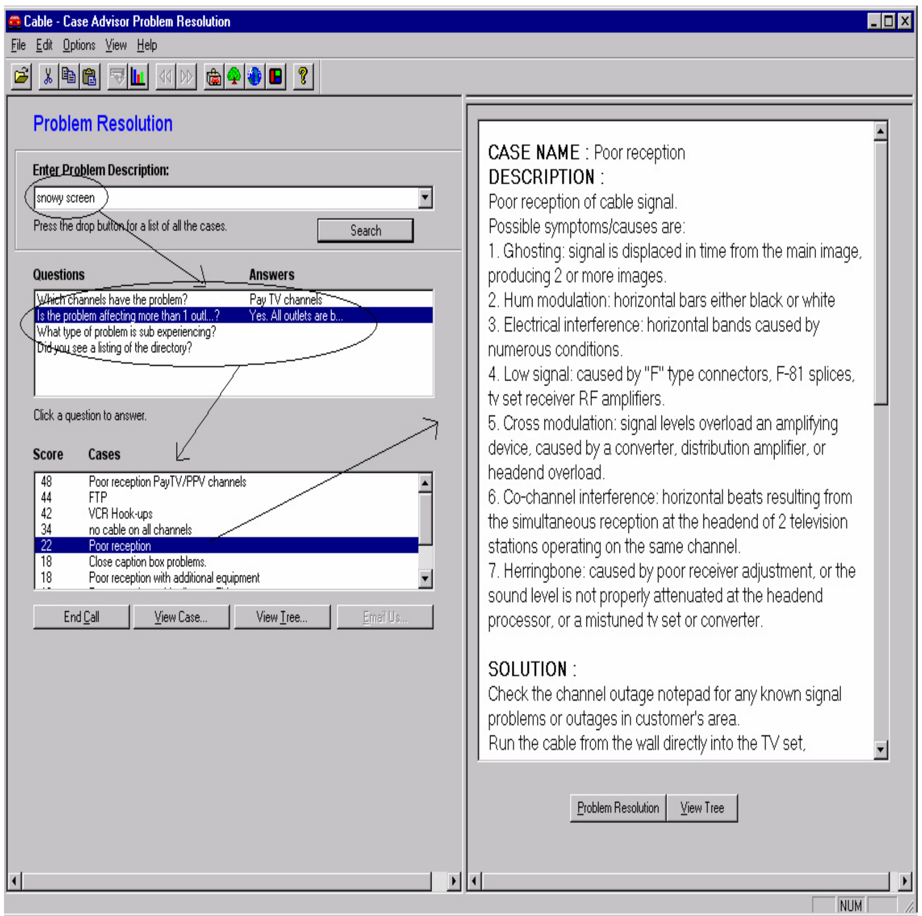
Zhong Zhang, Msc.

- IJCAI '99
- International Journal of Information Systems, Kluwer

Problem 4: Interactive Retrieval

In case-retrieval, experts usually ask a small number of key questions to find problems

Retrieval



Retrieval Issues:

-Given a set of candidate clusters that may share attributes

-Find: A small set of attributes that can distinguish the clusters

-Problem: similar to decision-tree construction

Information Theory

- Information (Entropy): given a probability distribution $P = \{P_1, P_2, \ldots, P_n\}$, information conveyed by this distribution is

$$Info(P) = -(p_1 \log(p_1) + p_2 \log(p_2) + \ldots + p_n \log(p_n))$$

- Gain: $Gain(X,T) = Info(T) - Info(X,T)$

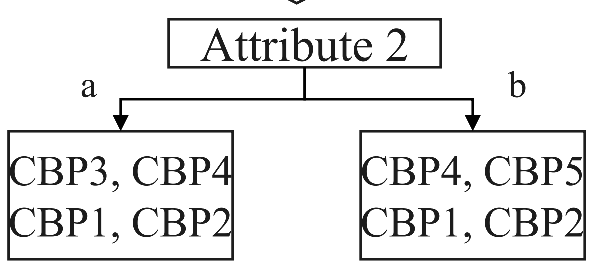
  where

$$Info(X,T) = -\sum_{i=1}^{m} \frac{T_i}{T} Info(T_i)$$

Cluster Retrieval Example

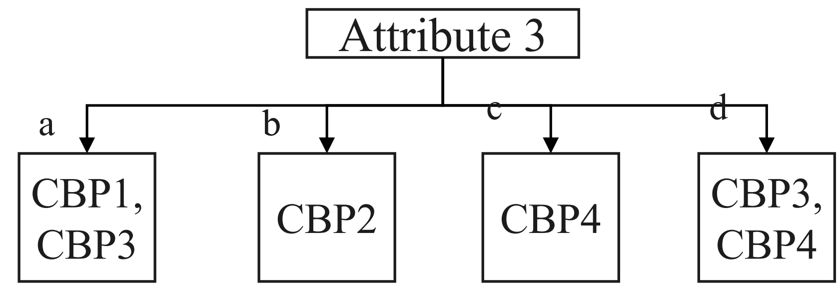
CBP No.	Attr 1	Attr 2	Attr 3	Attr 4
CBP 1	(a,100)		(a,100)	(a,50) (b,50)
CBP 2	(a,100)		(b,100)	(c,50),(d,50)
CBP 3		(a,50)	(a,50) (d,50)	(a,50),(b,50)
CBP 4		(a,50),(b,50)	(d,50) (c,50)	(a,100)
CBP 5		(b,100)		(b,50),(c,50)

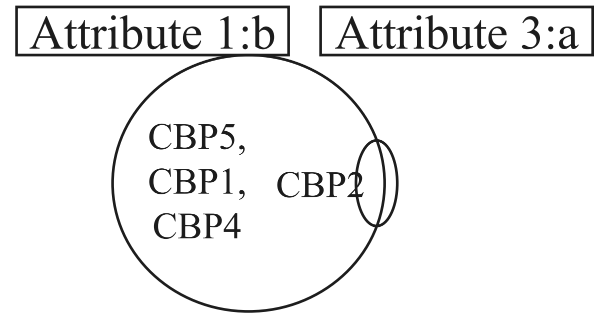
	CBC ID	Information Gain Ratio
1	2	8.72
2	4	6.99
3	3	4.15
4	1	0



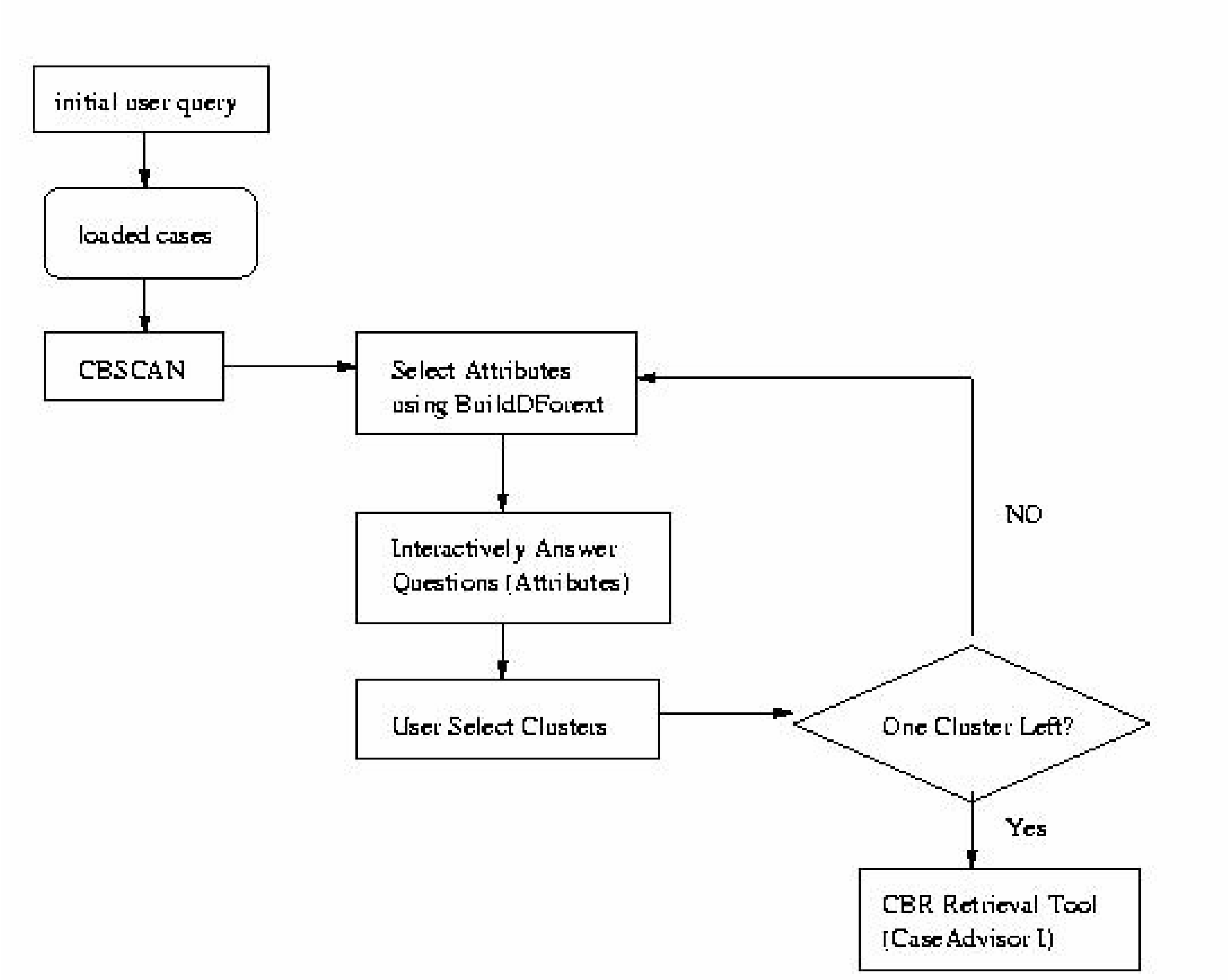
	CBC ID
1	3
2	4
3	1

For CBP1 and CBP2

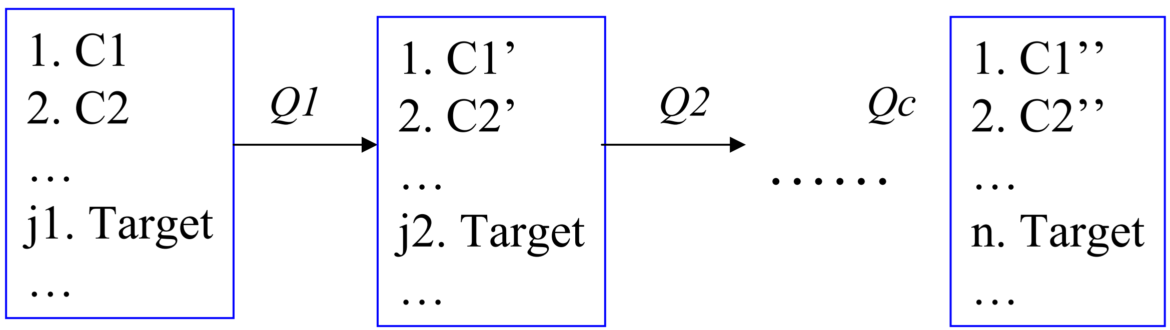




System Process



Ablation Study Evaluation



- Precision = *(1-n/10)*
  - if we set 10 to be the number of cases shown
- Interactive Efficiency= $1 - \frac{Qc}{Q_{all}}$

Experimental Results

UCI Thyroid CB	CA	Cluster	Info Gain	Cluster+Info Gain
Precision	0%	0%	45%	44%
Interactive Efficiency	56%	58%	97%	96%
Time (CPU sec)	448	4.3	62.3	17

UCI Mushroom	CA	Cluster	Info Gain	Cluster+Info Gain
Precision	6%	83%	92%	92%
Interactive Efficiency	59%	56%	92%	89%
Time (CPU sec)	5374	29	201	10

Problem 4: Interactive Retrieval

Jing Wu, MSc.

- Canadian AI 2000
- Applied Intelligence Journal, 2001

Problem 5: Information Gathering and ActiveCBR

- Lots of answers are available in various databases already
- Thus, no need to ask customers again!

A Typical Interactive-CBR Scenario

1. Agent: "What is your name and address?"
   Customer: "John, 9004 Lyra Place…"
2. Agent: "What is the nature of your problem?"
   Customer: "Fuzzy picture on Ch. 3"
3. Agent: "Let me check your payment status…OK, you are a paid customer."
4. Agent: "Let me check if there is an outage in your area…"
5. Agent: "Has the problem occurred before?"
   Customer: "Yes, but I can't remember how it was fixed."
6. Agent: "No outage. How many outlets do you have…"

A Typical Interactive-CBR Scenario

Answered from telephone number and customer database

1. Agent: "What is your name

   Customer: "John, 9004 Lyra Place…"

Answered from Sensor Database

2. Agent: "What is the nature

   Customer: "Fuzzy picture on Ch. 3"

Answered from customer database

3. Agent: "Let me check y
are a paid customer."

Answered from outage database

4. Agent: "Let me check if th

5. Agent: "Has the problem occurred before?"

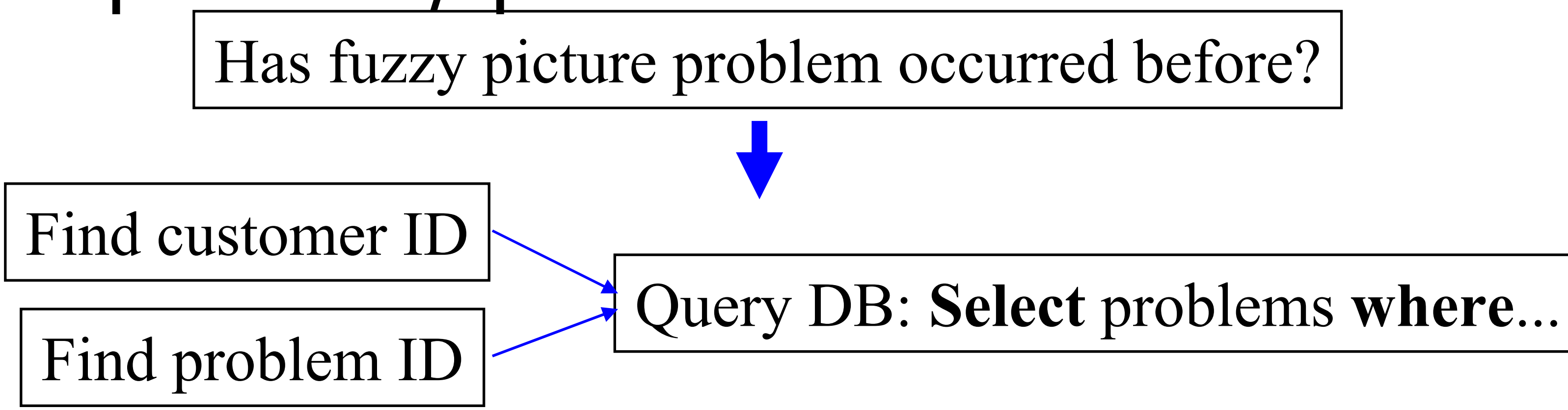
   Customer: "Yes, but I … s
   fixed."

Answered from problem history database

6. Agent: "No outage. How many outlets do you have…"

Related Issues

- Decomposing composite questions/queries



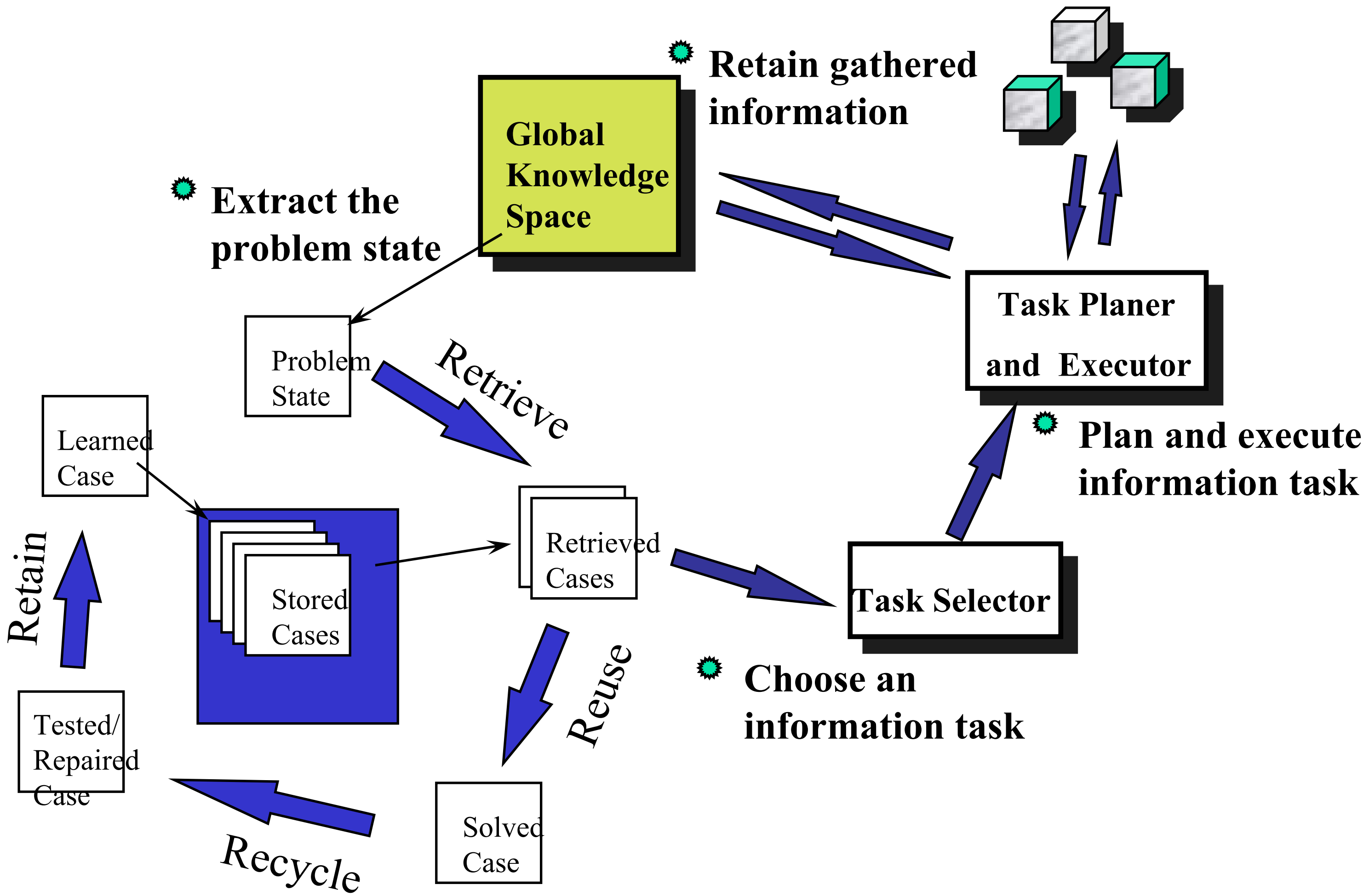
- Deciding on an order in which to ask questions

Our Aim: Summary

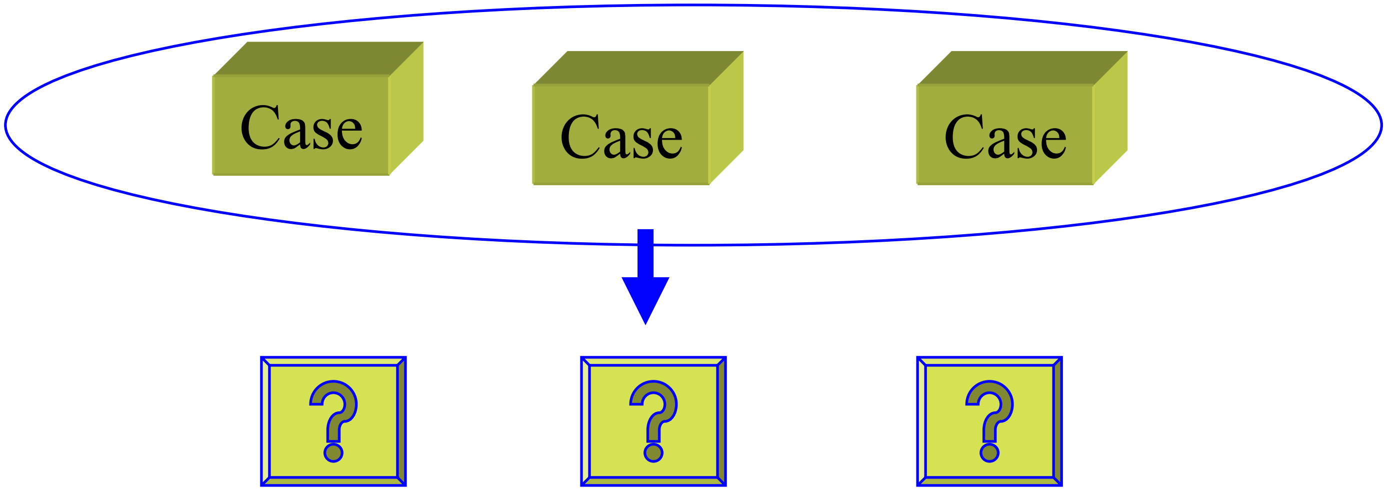
To increase *interactive efficiency* (Aha and Breslow '97) through automated information gathering:

- reduce the number of questions posed to customer
- answer as many questions as possible by gathering information from on-line sources
- answer first the questions which will most speed up diagnosis

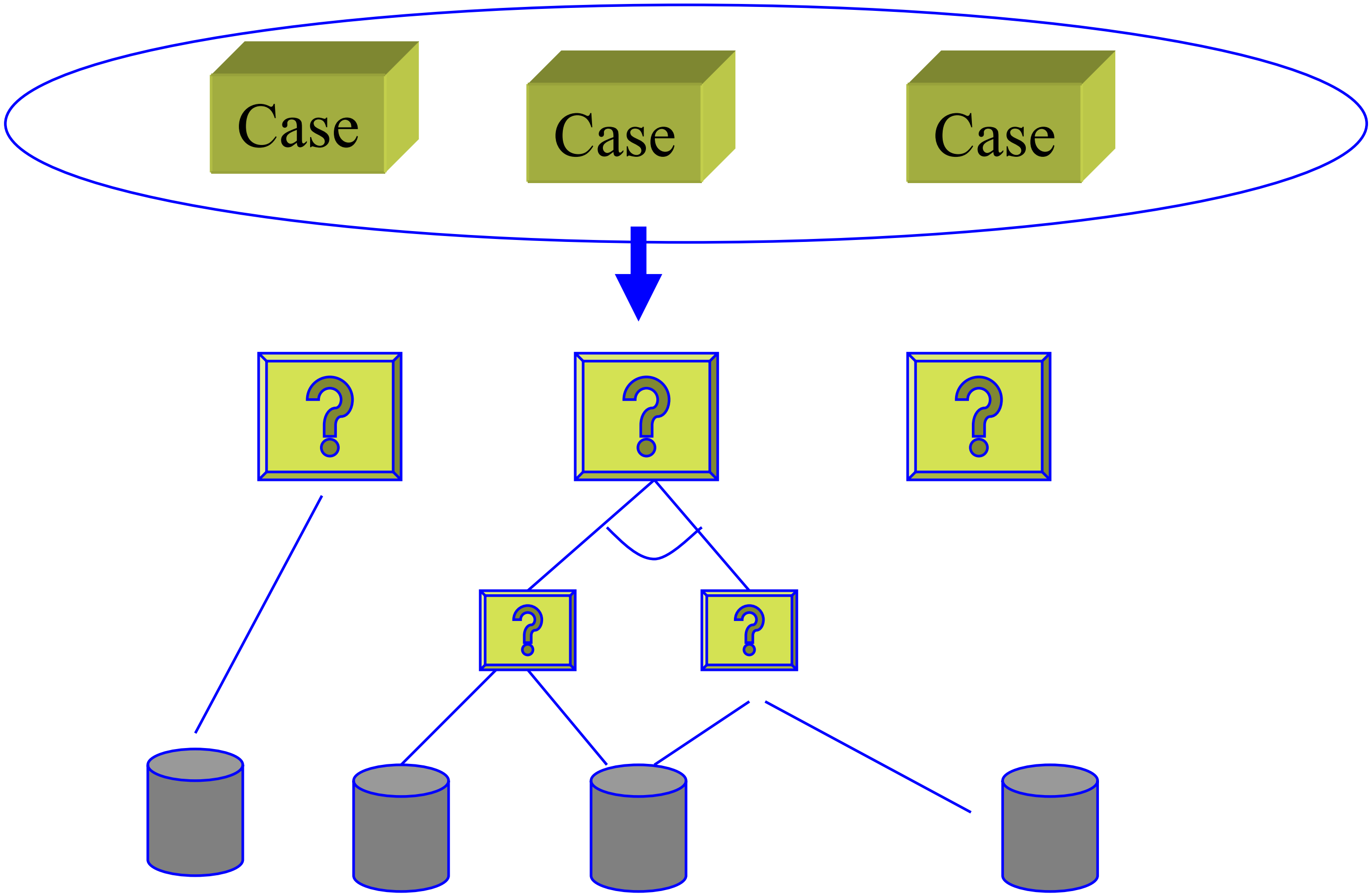
System Processes



Key Idea:



Key Idea:



Step 1. Initial Retrieval

- Initial retrieval by keywords in problem description
- Additional attributes focus retrieval through K-nearest neighbor search
- Retrieved cases indicate hypotheses
- Example:

Hypothesis: Parental control switch on

Attributes:

problem description:	*poor reception of the cable signal*	1.0
channels affected:	*channel 50*	0.7
uses parental control:	*yes*	0.8
has cable box:	*yes*	0.4
outlets concerned:	*1*	0.3

Step 2. Generating Queries from Retrieved Cases

- Select an attribute with high estimated utility as a query, based on the following two values:
  - Information value $v(a)$
    - the number of times the question appears in the candidate cases,
    - the weights of the question in the candidate cases, and
    - the ranks of the cases containing the question
  - Cost of evaluating the attribute $c(a)$
- Score of the attribute is $v(a)$ $c(a)$
- System selects the attribute with the maximal value as the *information task* for subsequent planning

Step 2: Query Ordering

Signal Case

Score = 80%

Attribute	Value	Weight
Problem?	Poor recep	1.00
Channels?	3-10	0.80
Local signal?	clear	0.95

Parental Control Case

Score = 90%

Attribute	Value	Weight
Problem?	Poor recep	0.5
Channels?	50-52	0.1
parental control?	yes	1.0

Information Value of attributes:

*V(Channels)=(0.8*0.8+0.9*0.1)=0.73*

*V(local signal)=0.8*0.95=0.76*

*V(parental control)=1.0*0.9=0.9*

Step 2: Decomposing composite queries

- Given: a library of information-task schemata

Use-parental-control :- Ask(customer)

Use-parental-control :- Check-online

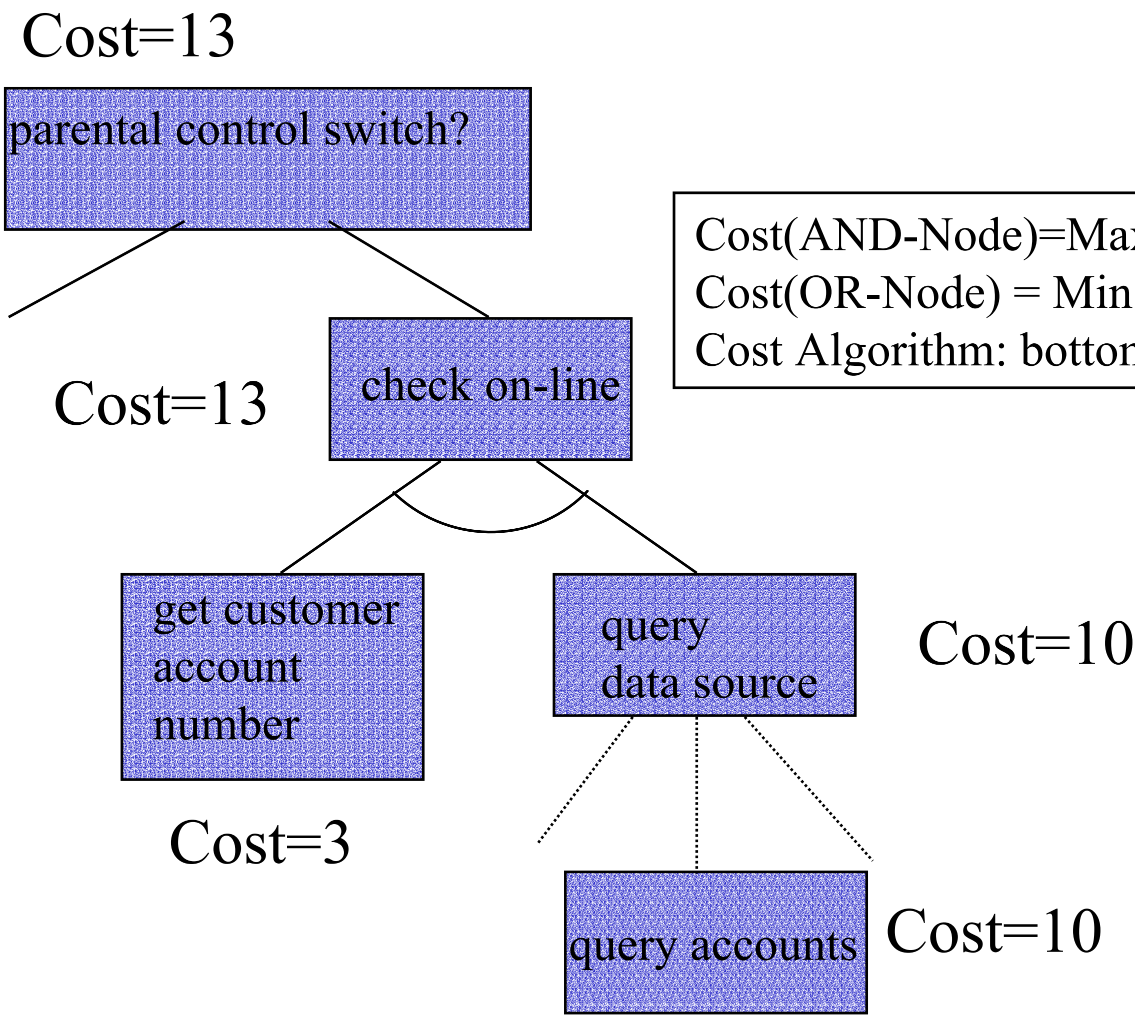
Check-online :- Query(account) **and** Query-data-source

Query-data-source :- Query(customer-profile)

Query-data-source :- Query(work_log)

- The schema is used to expand the information task into an AND-OR Tree

Example of AND-OR Tree



Cost(AND-Node)=Max{Cost(Children-Node)}
Cost(OR-Node) = Min{Cost(Children-Node)}
Cost Algorithm: bottom-up

Cost Models at Leaf Nodes

- Defined or learned from database characteristics
  - propagated up the task hierarchy
- Costs include
  - time to access data source
  - reliability of source
  - intrusion (querying customer)

Problem 5: Information Gathering and ActiveCBR

C. Carrick, Sheng Li, I. Abi-Zeid and L. Lamontagne

- ICCBR '99
- EWCBR '00
- International Journal of Knowledge and Information Systems, Kluwer

Field test

Objective?

- Real-time problem solving
- Junior CSR training
- New technology education
- Consistent answers

Status

- Rogers Cable Systems Ltd.
- Help Desks
- Educational Systems
  - Experimental testbed
  - Tool to learn about CBR
  - CBR for software requirement engineering
  - Other uses

Conclusions

- Problem-driven research methodologies
- Case-base maintenance main objective
  - Hard problem
  - CBR without maintenance???
- Case-adaptation practical?
- Future: Case mining