DATA ANALYTICS

Unit 4: Knowledge-Based Recommender System

Jyothi R.
Department of Computer Science and Engineering

"Core" Recommendation Techniques

Technique	Background	Input	Process
Collaborative	Ratings from U of items in I	Ratings from U of items in I	Identify users in U similar to u, and extrapolate from their ratings of i
Content-based	Features of items in I	U's ratings of items in I	Generate a classifier that fits U's rating behavior and use it on I
Demographic	Demographic information about U and their ratings of items in I	Demographic information about U	Identify users that are demographically similar to U, ad extrapolate from their ratings of i
Utility-based	Features of items in I	A Utility function over items in I that describes U's preferences.	Apply the function to the items and determine I's rank
Knowledge-based	Features of items in I. Knowledge of how these items meet a user's needs.	A description of U's needs or interests.	Infer a match between I and U's need.

Knowledge-based recommender systems

- knowledge-based recommender systems are appropriate in the following situations:

1. Customers want to explicitly specify their requirements. Therefore, interactivity is a crucial component of such systems. Note that collaborative and content-based systems do not allow this type of detailed feedback.

2. It is difficult to obtain ratings for a specific type of item because of the greater complexity of the product domain in terms of the types of items and options available.

3. In some domains, such as computers, the ratings may be time-sensitive. The ratings on an old car or computer are not very useful for recommendations because they evolve with changing product availability and corresponding user requirements.

Knowledge-based recommender systems types

- Knowledge-based recommender systems can be categorized on the basis of user interactive methodology and the corresponding knowledge bases used to facilitate the interaction.
- There are two primary types of knowledge-based recommender systems:

1. Constraint-based recommender systems: In constraint-based systems users typically specify requirements or constraints (e.g., lower or upper limits) on the item attributes. Furthermore, domain-specific rules are used to match the user requirements or attributes to item attributes. These rules represent the domain-specific knowledge used by the system.
2. Case-based recommender systems: In case-based recommender systems, specific cases are specified by the user as targets or anchor points. Similarity metrics are defined on the item attributes to retrieve similar items to these targets. The similarity metrics are often carefully defined in a domain-specific way.

Knowledge-based recommender systems

The Conceptual goals of various recommender systems

Approach	Conceptual Goal	Input
Collaborative	Gives us recommendations based on a collaborative approach that leverages the ratings and actions of our peers/myself	User ratings + Community ratings
Content-based	Gives us recommendations based on the content (attributes) we have favored in our past ratings and actions.	User ratings + item attributes + domain knowledge
Knowledge-based	Gives us recommendations based on our explicit specification of the kind of content (attributes) we want	User specification + Item attributes + domain knowledge

Knowledge-Based Recommender Systems

- The Interaction between user and recommender may take the following forms.

1. **Conversational Systems**: The user preferences are determined in the context of a feedback loop. The item domain is complex, and the user preferences can be determined only in the context of an iterative conversational system.
2. **Search-based systems**: User preferences are elicited by using a preset sequence of questions such as the following;" Do you prefer a house in a suburban area or within the city?"
3. **Navigation-based recommendation**: The user specifies a number of change requests to the item being currently recommended. Through an iterative set of change requests, it is possible to arrive at a desirable item.

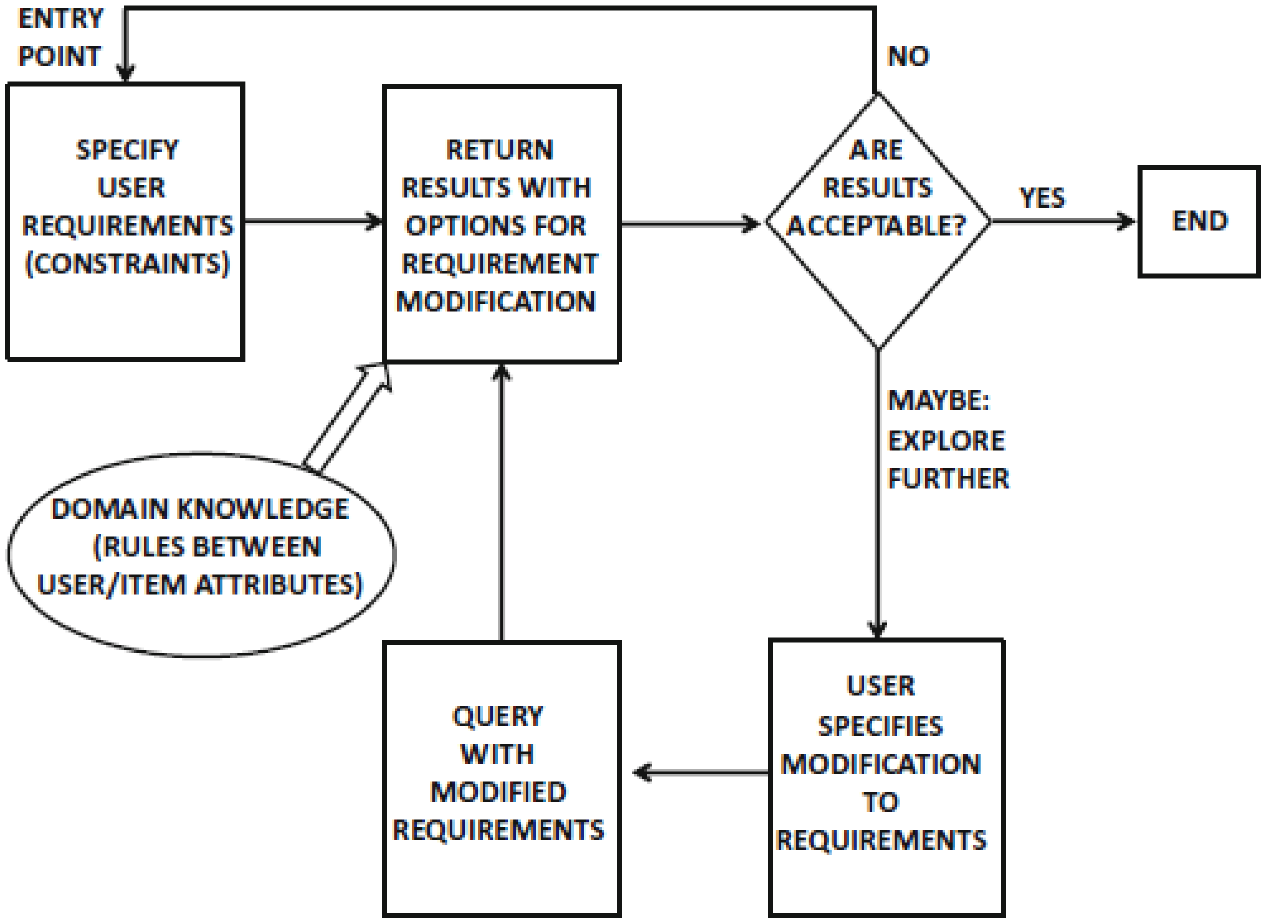
Eg. " I would like a similar house about 5 miles west of the currently recommended house." Such recommender systems are also referred to as critiquing recommender systems.

Knowledge-Based Recommender Systems

- Critiquing recommender systems are naturally designed for case-based recommender systems, because one critiques a specific case in order to arrive at the desired outcome.
- A search-based system can be used to set up user requirements for constraint-based recommenders.

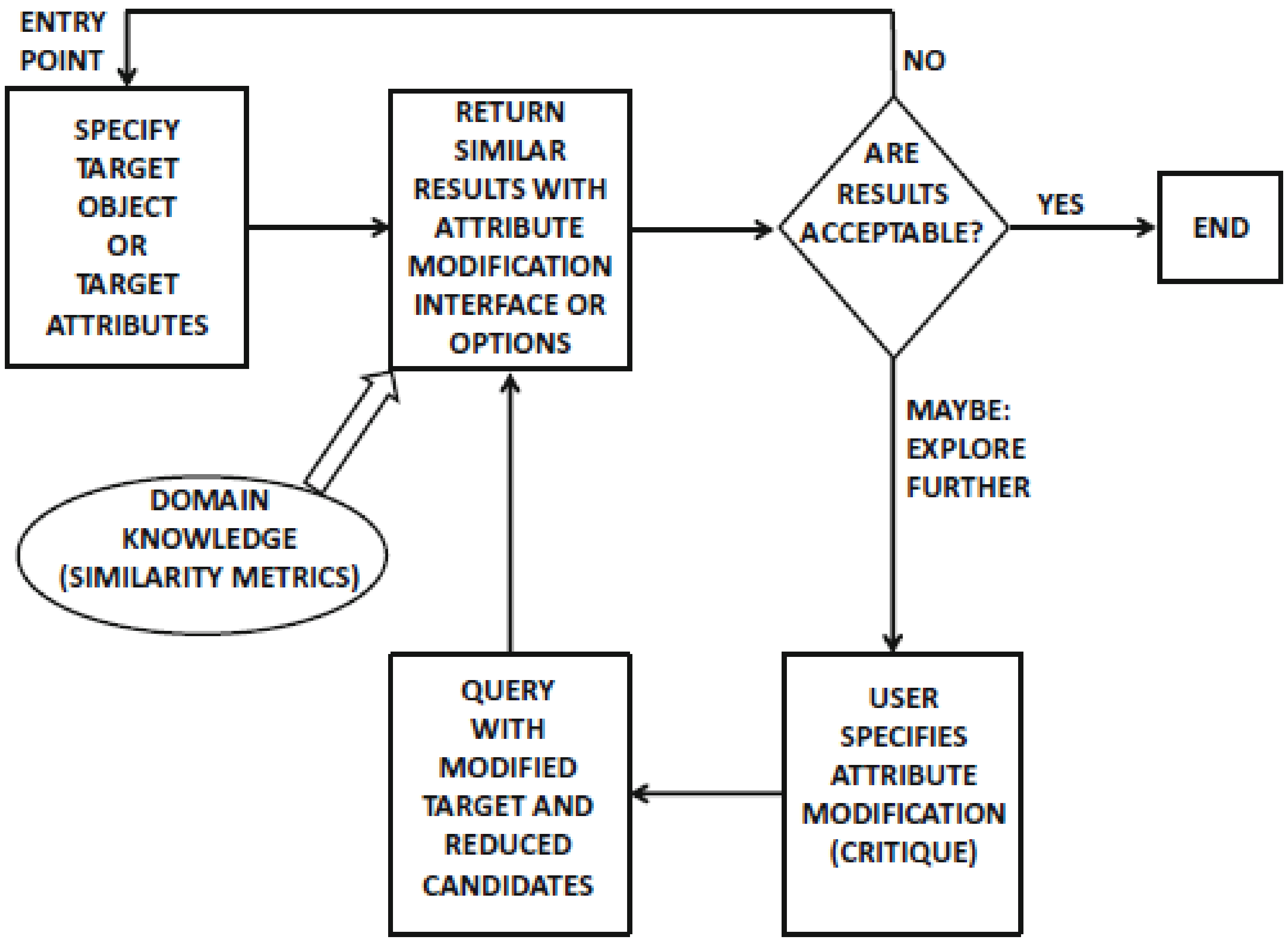
Knowledge-based Recommendation system

Constraint-based Recommender systems.



Knowledge-based Recommendation system

Case-based Recommender systems.



Knowledge-based Recommendation system

Difference between Constraint-based and Case-based Recommender systems.

- In constraint-based systems, specific requirements or constraints are specified by the user.
- The Original query is modified by addition, deletion, modification, or relaxation of the original set of user requirements.
- Users are not in apposition to exactly state their requirements up front in a complex product domain, this problem is partially addressed through a knowledge-base of rules, which map user requirements to product attributes.

- In case-based systems, specific targets or cases are specified.
- Either the target is modified through user interaction, or the search results are pruned through the use of directional critiques.
- This problem is addressed through a conversational style of critiquing.

Knowledge-based recommender systems types

Examples of attributes in a recommendation application for buying homes.

Item-Id	Beds.	Baths.	Locality	Type	Floor Area	Price
1	3	2	BTM	Town House	1600	220,000
2	5	2.5	JP	Split-level	3600	973,000
3	4	2	RT	Ranch	2600	630,000
4	2	1.5	MAJESTIC	Condo	1500	220,000
5	4	2	Dollars	Colonial	2700	430,000

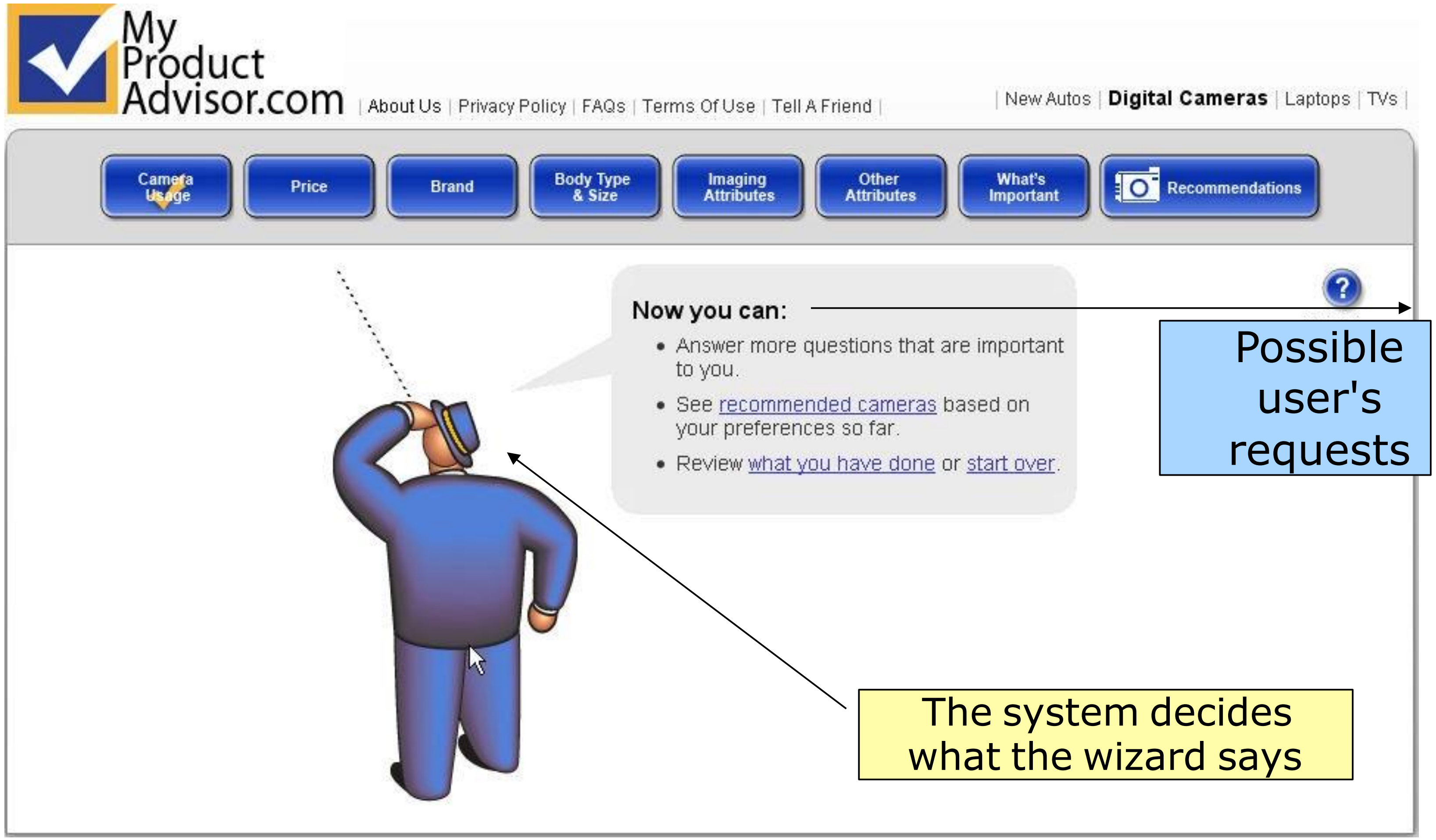
Knowledge-Based Recommender Systems

- Suggests products based on **inferences** about a user`s needs and preferences
- **Functional knowledge:** about how a particular item meets a particular user need
- The **user model** can be any knowledge structure that supports this inference
- A query, i.e., the set of preferred features for a product
- A case (in a case-based reasoning system)
- An adapted similarity metric (for matching)
- A part of an ontology
- **There is a large use of domain knowledge encoded in a knowledge representation language/approach.**

DATA ANALYTICS

Knowledge-Based Recommender Systems

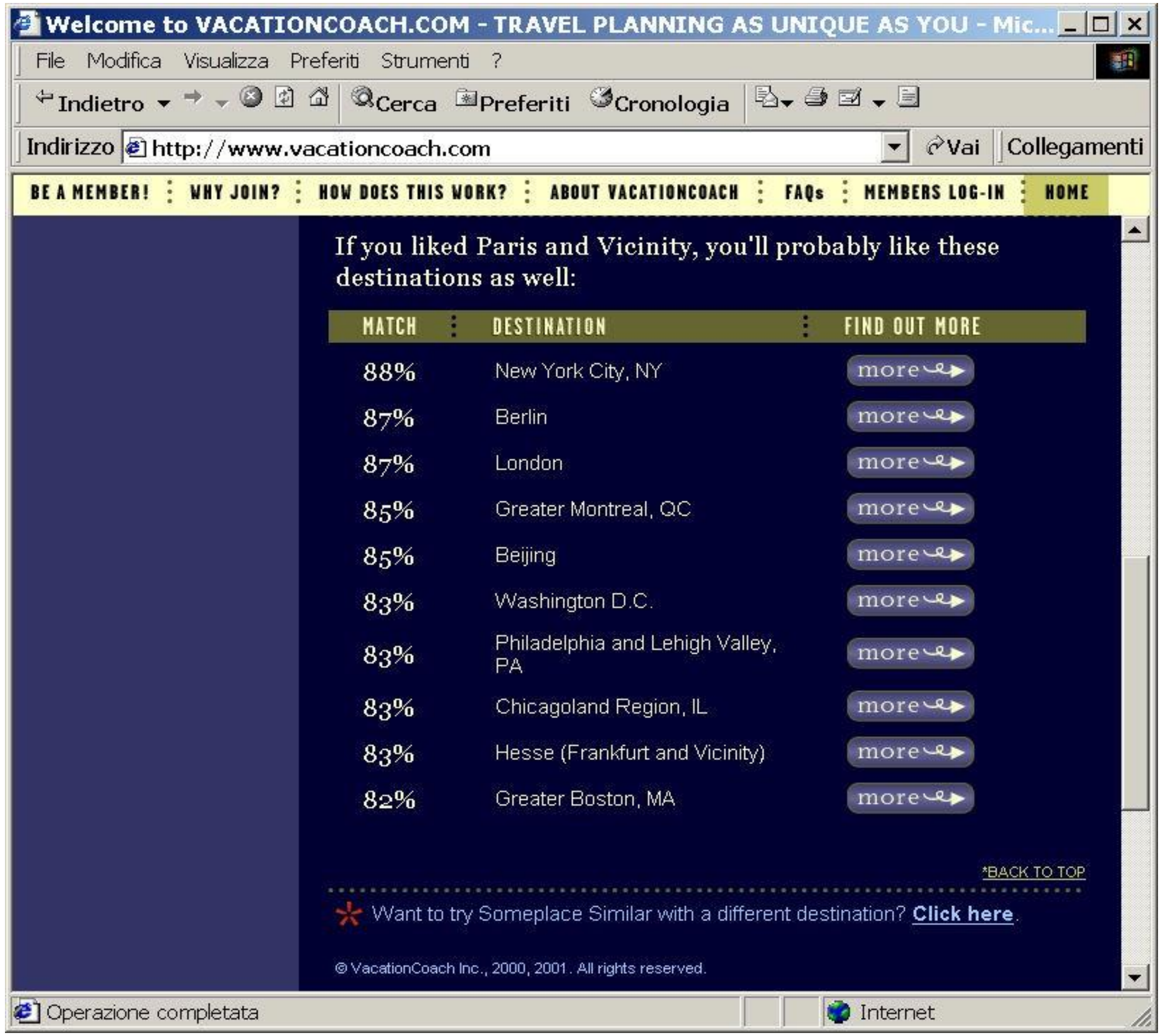
Knowledge-Based Recommender Systems



Knowledge-Based Recommender Systems

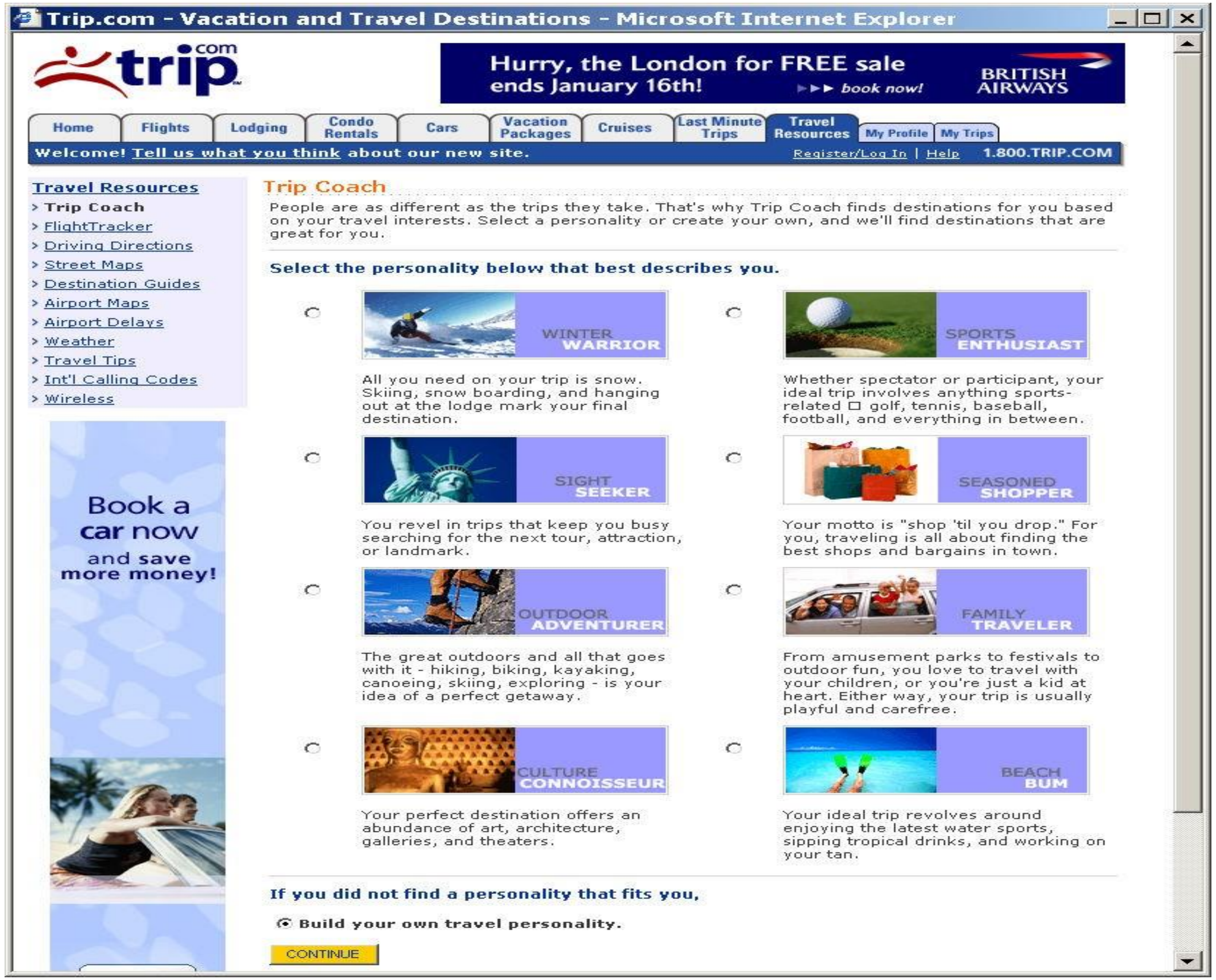
Knowledge-Based Recommender Systems

Knowledge-Based Recommender Systems

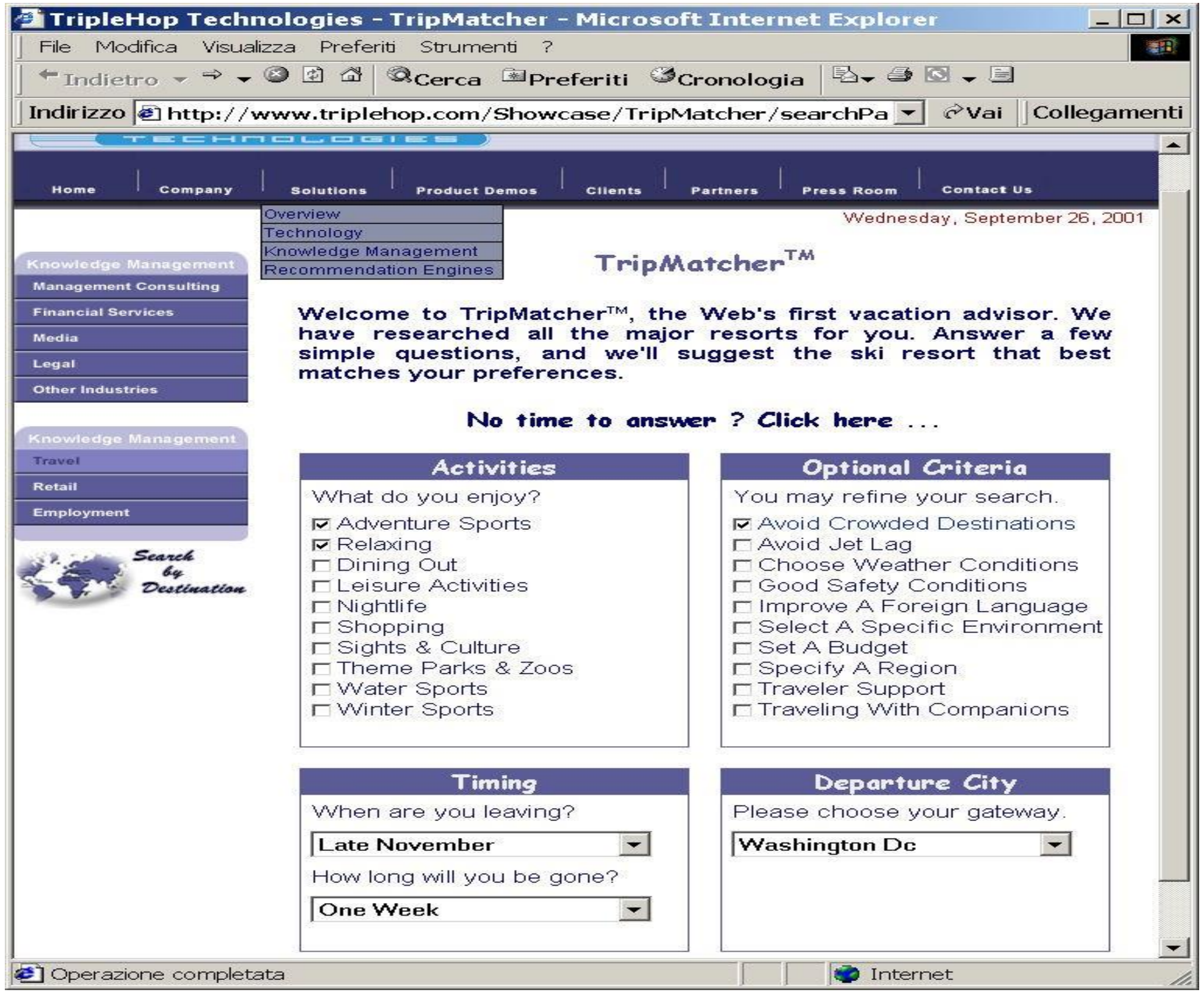


Knowledge-Based Recommender Systems

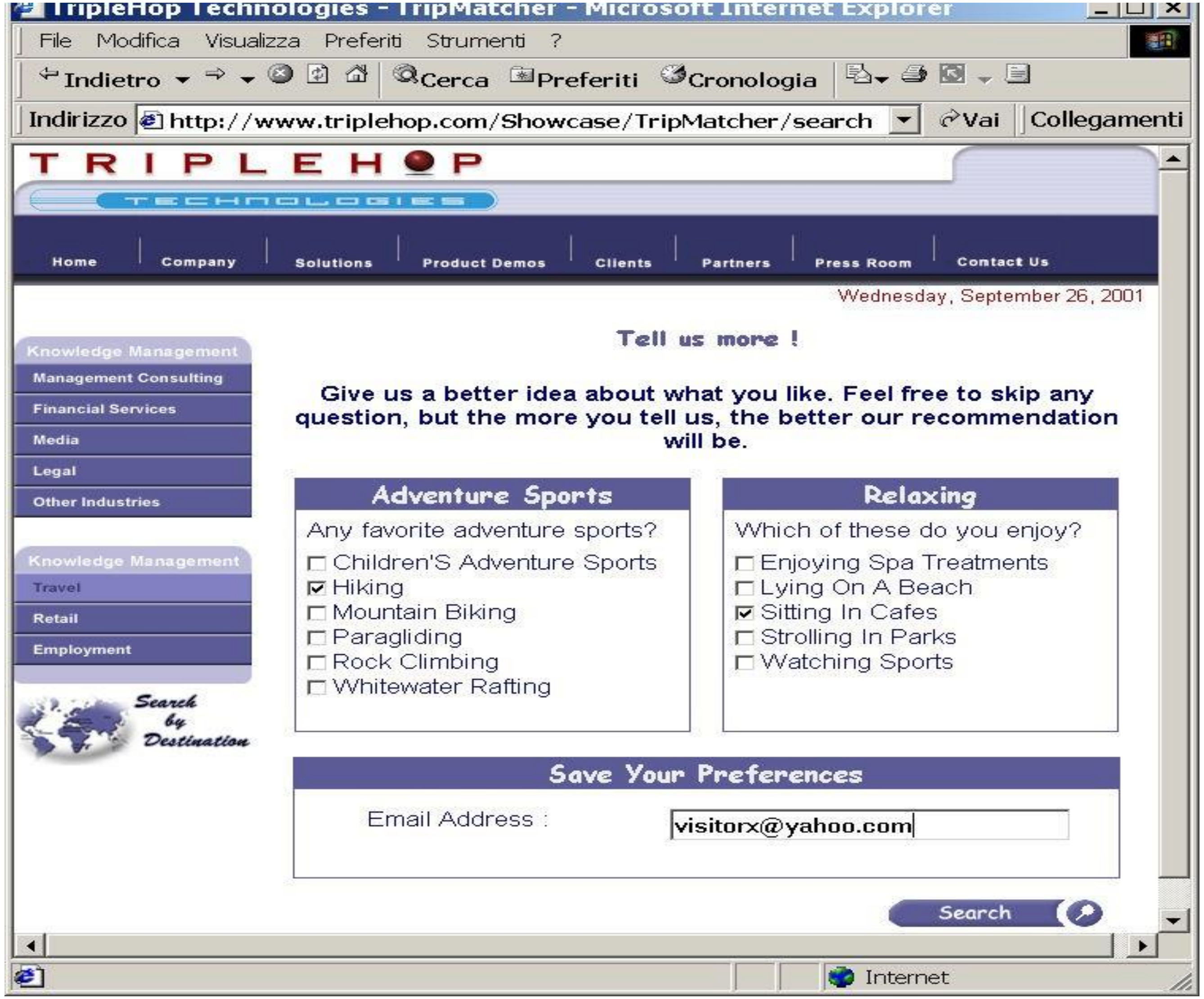
Knowledge-Based Recommender Systems



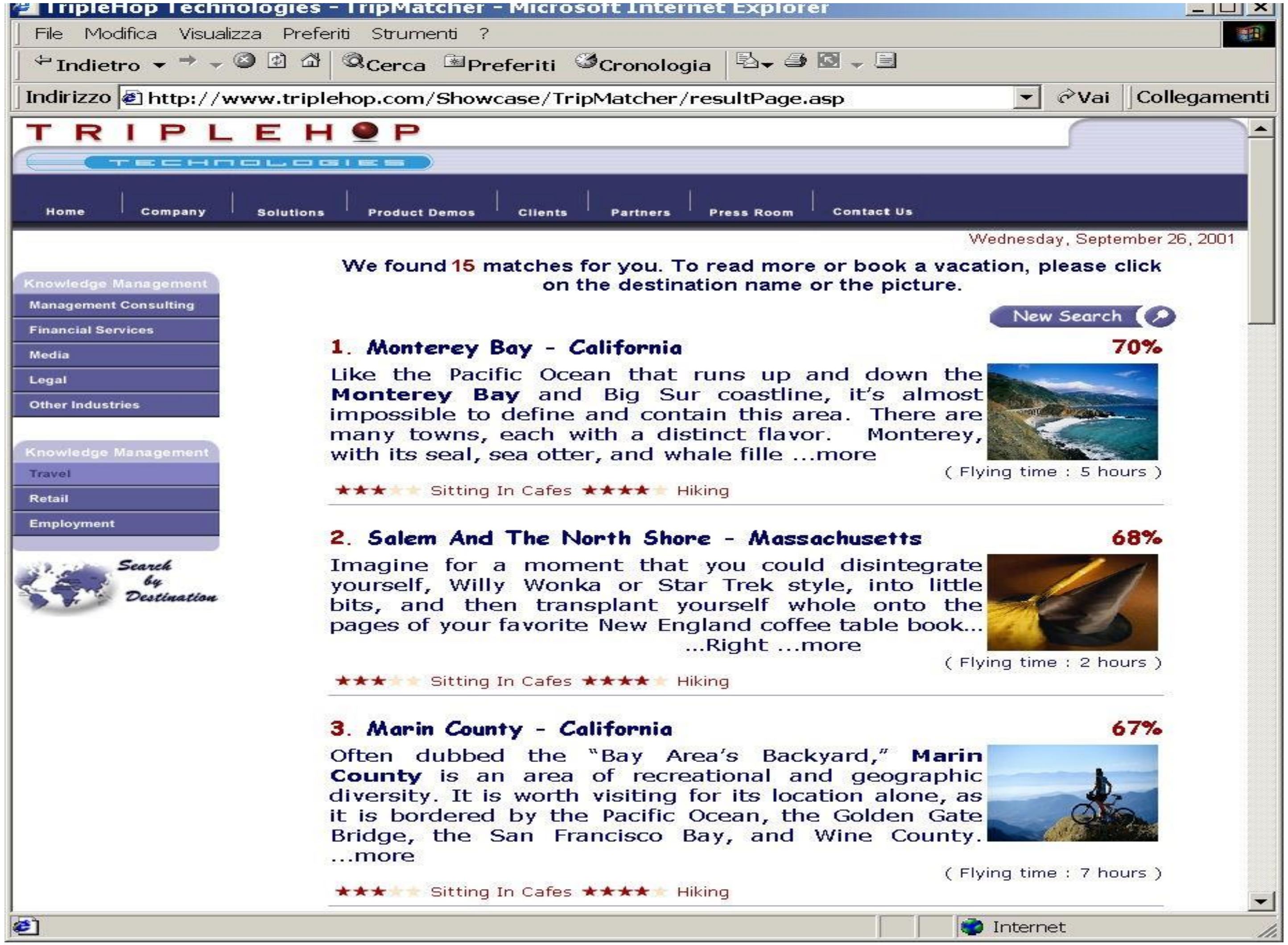
Knowledge-Based Recommender Systems



Knowledge-Based Recommender Systems



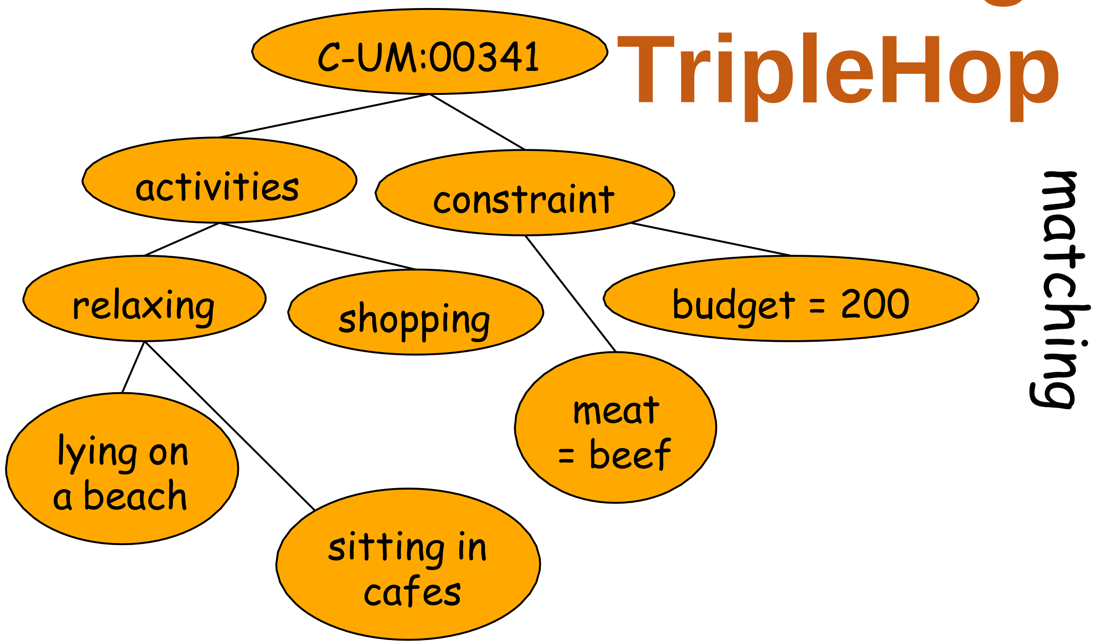
Knowledge-Based Recommender Systems



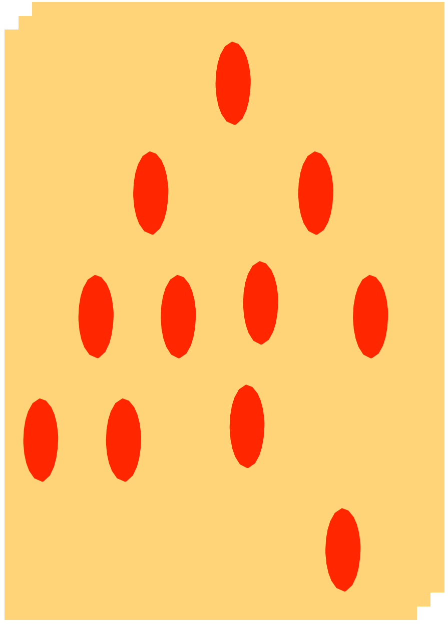
Knowledge-Based Recommender Systems

Example: TripleHop

Matching in TripleHop



Catalogue of Destinations

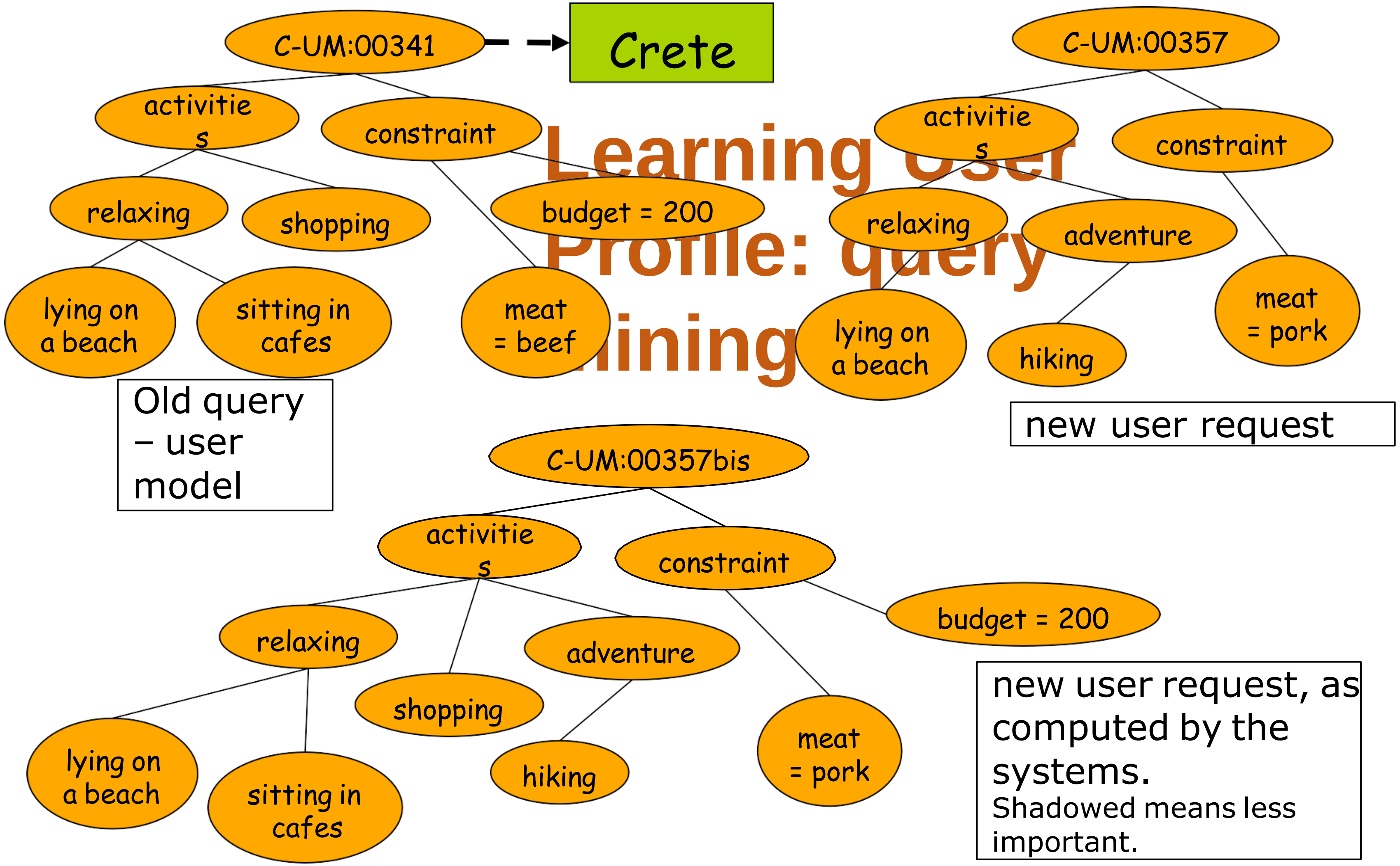


TripleHop and Content-Based RS

- The content (destination description) is exploited in the recommendation process
- A classical Content-Based method would have used a “simpler” content model ,e.g., keywords or TF-IDF
- Here a more complex **knowledge structure** – a tree of concepts – is used to model the product (and the query)
- The query is the user model and it is acquired every time the user asks for a new recommendation - (not exactly, more details later)
- Stress on ephemeral needs rather than building a persistent user model
- Typical in Knowledge-Based RS, they are more focused on ephemeral users – because Collaborative Filtering and Content-Based methods cannot cope with that users.

DATA ANALYTICS

Knowledge-Based Recommender Systems



Query Augmentation

- Personalization in search is not only “information **filtering**”
- **Query augmentation:** when a query is entered it can be compared against contextual and individual information to refine the query
- Ex1: If the user is searching for a restaurant and enter a keyword “Thai” then the query can be augmented to “Thai food”
- Ex2: If the query “Thai food” does not retrieve any restaurant the query can be refined to “Asian food”
- Ex3: If the query “Asian food” retrieves too many restaurant, and the user searched in the past for “Chinese” food the query can be refined to “Chinese food”.

.

Query Augmentation in TripleHop

1. The current query is **compared** with **previous queries** of the **same user**
2. Preferences expressed in past (similar) queries are identified
3. A new query is built by **combining the short term preferences** contained in the query with the "**inferred**" preferences extracted from the persistent user model (past queries)
4. When the query is matched against an item (destination) if two destinations have the same degree of matching for the explicit preferences then the "inferred" preferences are used to break the tie

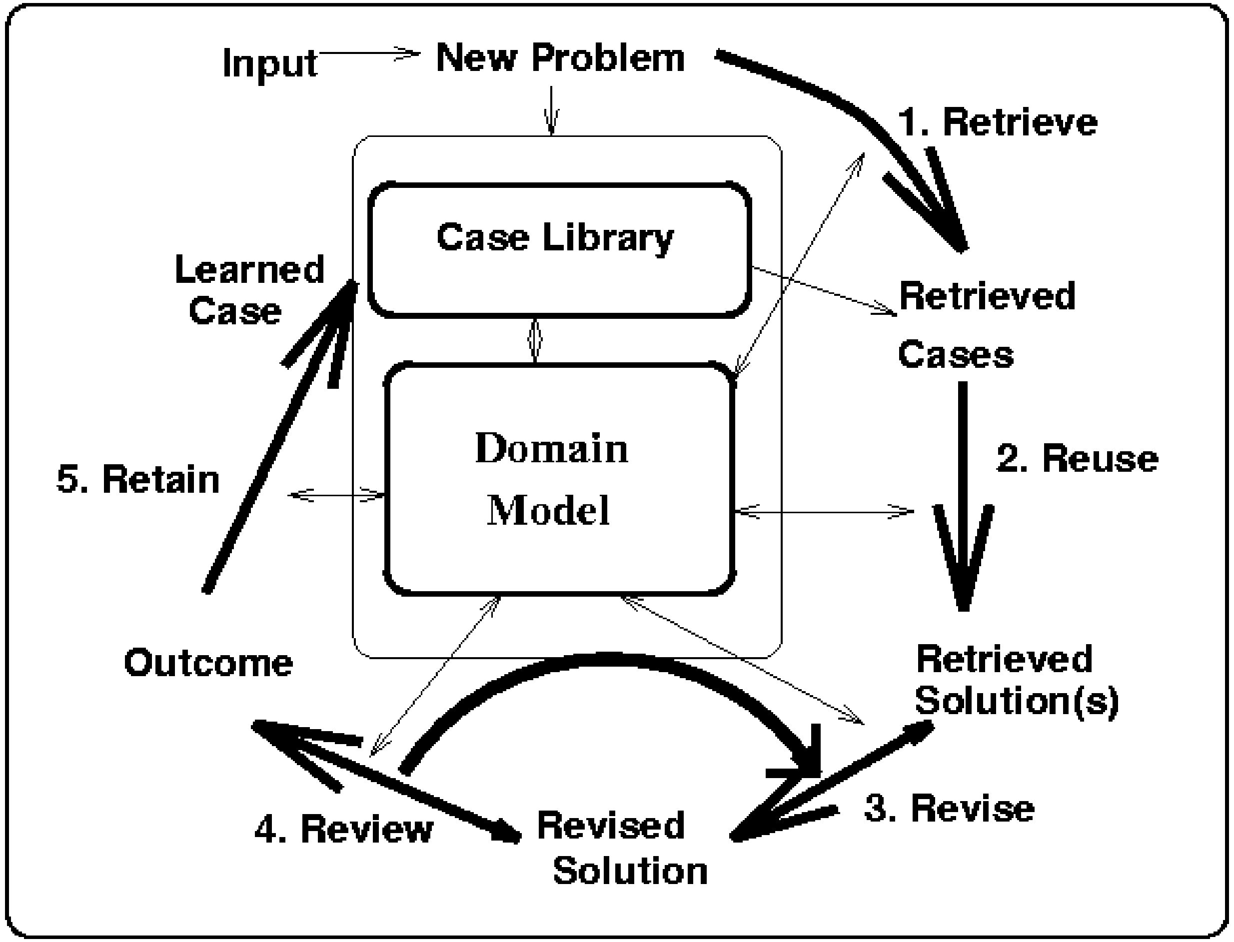
- This is another example of the cascade approach
- the two combined RS are based on the same knowledge but with two definitions of the user model.

What is Case Based Reasoning ?

- **A case-based reasoner solves new problems by adapting solutions that were used to solve old problems** (Riesbeck & Shank 1989)
- CBR problem solving process:
- store previous experiences (cases) in memory to solve new problems
- Retrieve form the memory similar experience about similar situations
- Reuse the experience in the context of the new situation: complete or partial reuse, or adapt according to differences
- Store new experience in memory (learning)

Knowledge-Based Recommender Systems

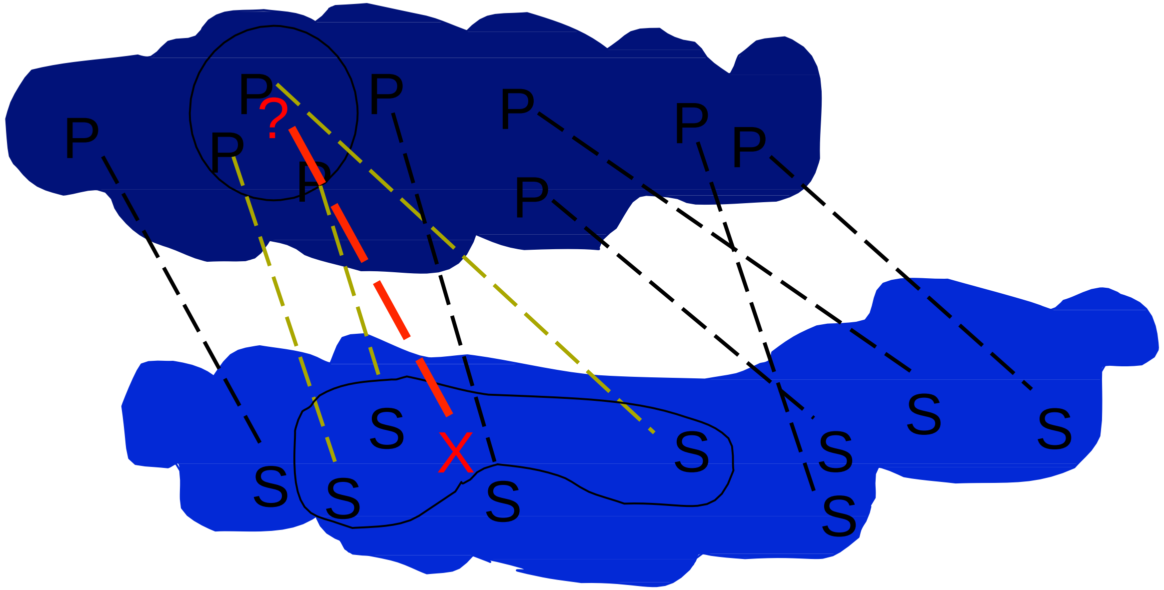
Case-Based Reasoning



Knowledge-Based Recommender Systems

CBR Assumption

- New problem can be solved by
  - retrieving similar problems
  - adapting retrieved solutions
- Similar problems have similar solutions



PES UNIVERSITY ONLINE

Examples of CBR

- Classification: “The patient`s ear problems are like this prototypical case of otitis media”
- Compiling solutions: “Patient N`s heart symptoms can be explained in the same way as previous patient D`s”
- Assessing values: My house is like the one that sold down the street for $250,000 but has a better view
- Justifying with precedents: “This Missouri case should be decided just like Roe v. Wade where the court held that a state`s limitations on abortion are illegal”
- Evaluating options: “If we attack Cuban/Russian missile installations, it would be just like Pearl Harbor”

Instance-based learning – Lazy Learning

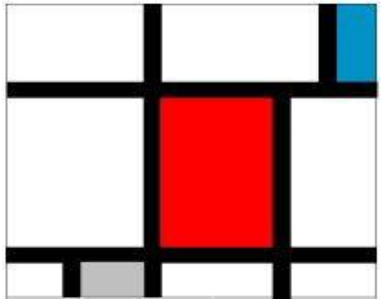
- One way of solving tasks of approximating discrete or real valued target functions
- Have training examples: $(x_n, f(x_n))$, $n=1,..$
- Key idea:
- just store the training examples
- when a test example is given then find the closest matches
- use the closest matches to guess the value of the target function on the test example.

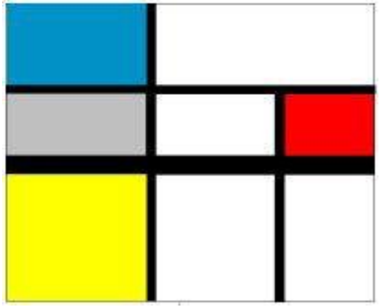
The distance between examples

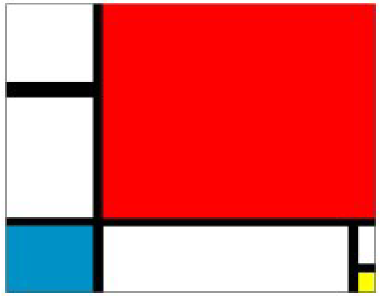
- We need a **measure of distance** (or similarity) in order to know who are the neighbors
- Assume that we have T attributes for the learning problem. Then one example point x has elements x_t , t=1,…,T
- The distance between two points x and y is often defined as the **Euclidean** distance:

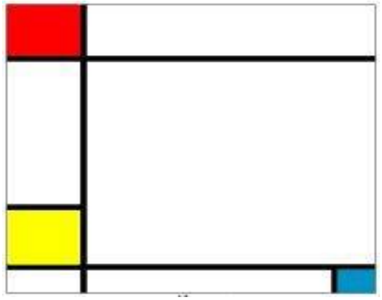
$$d(x,y)=\sqrt{\sum_{t=1}^{T}[x_t-y_t]^2}$$

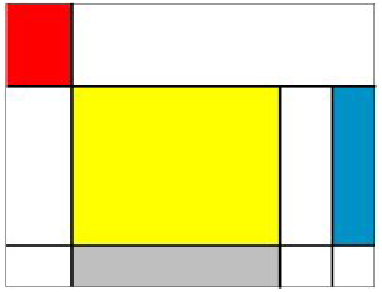
Knowledge-Based Recommender Systems

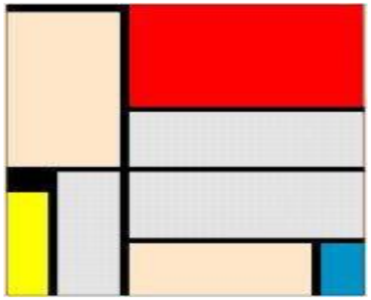

no

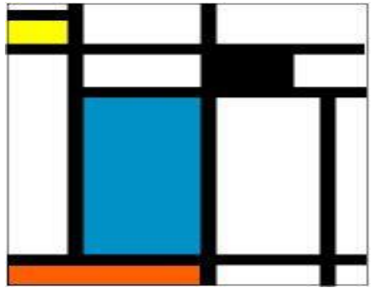

no

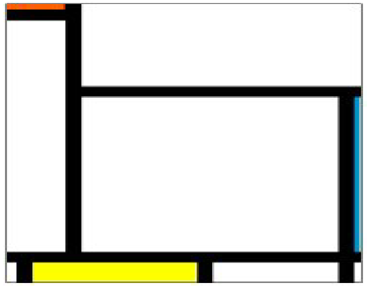

yes


yes


no


yes


no


?

PES UNIVERSITY ONLINE

Training data

Number	Lines	Line types	Rectangles	Colours	Mondrian?
1	6	1	10	4	*No*
2	4	2	8	5	*No*
3	5	2	7	4	Yes
4	5	1	8	4	Yes
5	5	1	10	5	*No*
6	6	1	8	6	Yes
7	7	1	14	5	*No*

Test instance

Number	Lines	Line types	Rectangles	Colours	Mondrian?
8	7	2	9	4	

Knowledge-Based Recommender Systems

	Lines	LinesT	Rect	Colors	Class	Distance to test
Train1	4	2	8	5	no	3,32
Train2	5	2	7	4	yes	2,83
Train3	5	1	8	4	yes	2,45
Train4	5	1	10	5	no	2,65
Train5	6	1	8	6	yes	2,65
Train6	7	1	14	5	no	5,20
test	7	2	9	4		
Train1	-0,32	0,32	-0,11	0,06	no	0,80
Train2	-0,08	0,32	-0,21	-0,28	yes	0,52
Train3	-0,08	-0,16	-0,11	-0,28	yes	0,69
Train4	-0,08	-0,16	0,08	0,06	no	0,77
Train5	0,16	-0,16	-0,11	0,39	yes	0,86
Train6	0,40	-0,16	0,47	0,06	no	0,76
test	0,40	0,32	-0,02	-0,28		

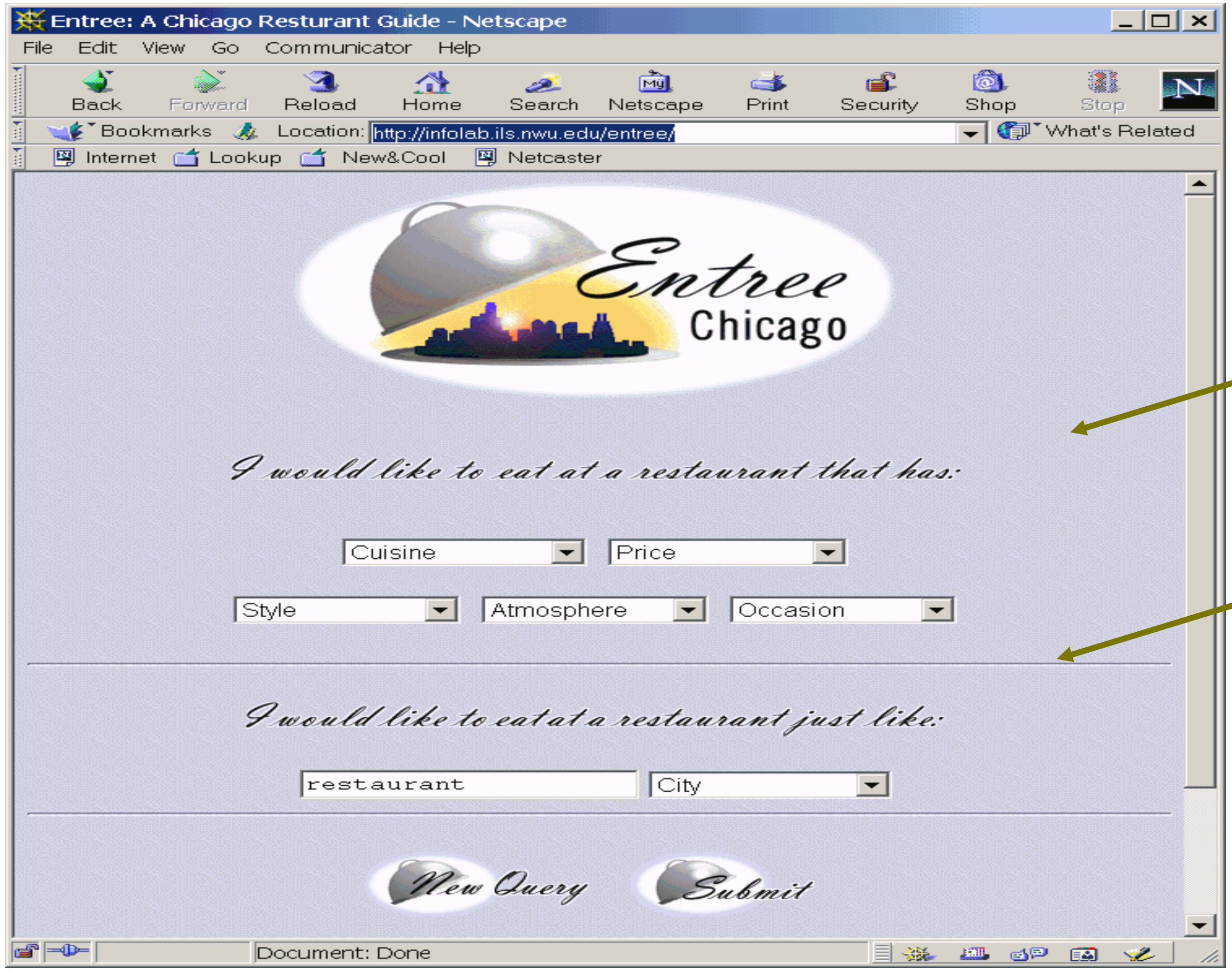
Feature values are not normalized

Feature values are normalized

What is the difference between this feature value normalization and vector Normalization in IR?

x' = (x −avg(X))/4*stdev(X)), where x is a feature value of the feature X

Example of CBR Recommender System



Entree is a restaurant recommender system – it finds restaurants:

1. matching some user goals (case features)
2. or similar to restaurants the user knows and likes

The Product is the Case

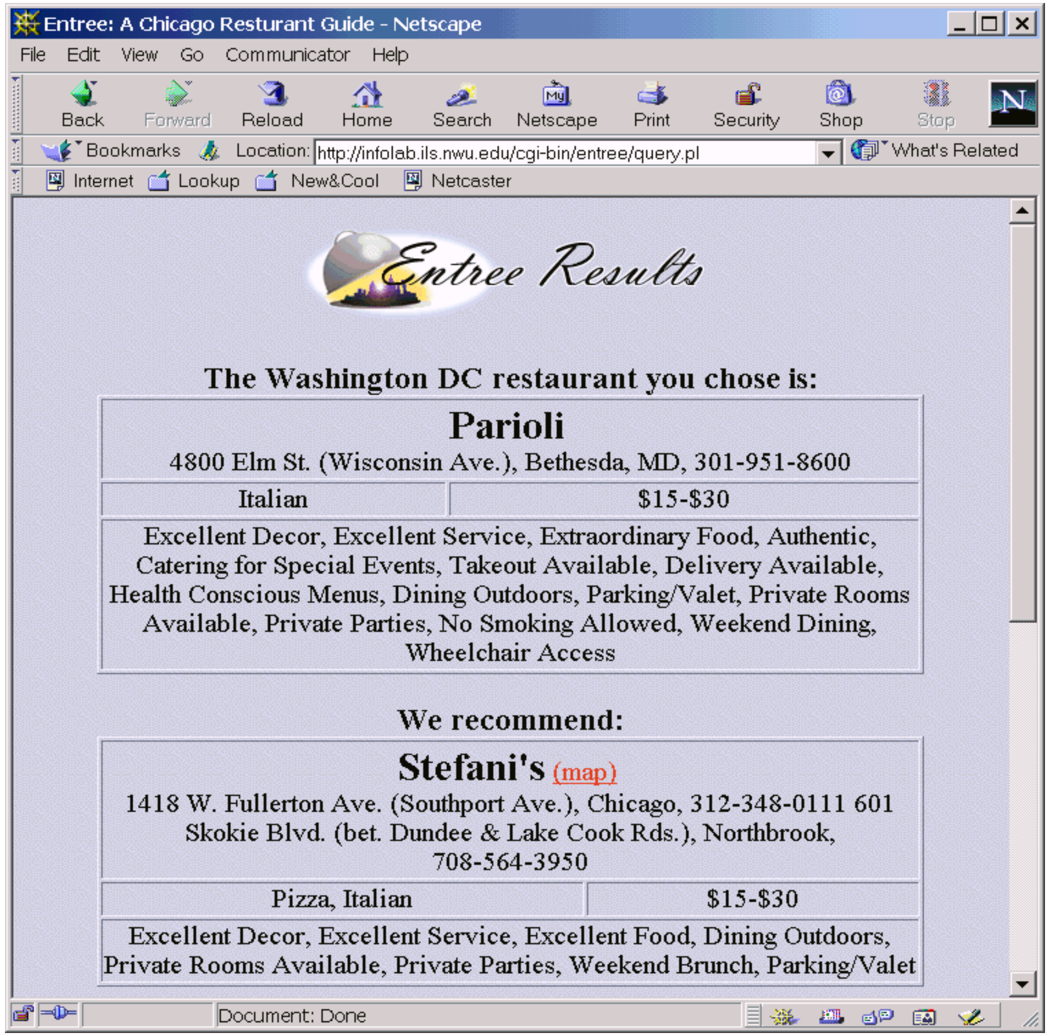
- In Entrée a case is a restaurant – **the case is the product**
- The **problem** component is the description of the restaurant given by the user
- The user will input a partial description of it – this is the only difficulty
- The **solution** part of the case is the restaurant itself – i.e. the name of the restaurant
- The assumption is that the needs of the user can be modeled as the features of the product description ….

Knowledge-Based Recommender Systems

Partial Match

In general, only a subset of the preferences will be matched in the recommended restaurant.

Nearest Neighbor



Recommendation in Entree

- The system first selects from the database the set of all restaurants that satisfy the largest number of logical constraints generated by considering the input features type and value
- If necessary, implicitly relaxes the lowest important constraints until some restaurants could be retrieved
- Typically the relaxation of constraints will produce many restaurants in the result set
- Sorts the retrieved cases using a similarity metric

  – this takes into account all the input features.

Similarity in Entree

- This similarity metric assumes that the user goals, corresponding to the input features (or the features of the source case), could be sorted to reflect the importance of such goals from the user point of view
- Hence the global similarity metric (algorithm) sorts the products first with respect the most important goal and then iteratively with respect to the remaining goals (multi-level sort)
- Attention: it does not works as a maximization of a Utility-Similarity defined as the sum of local utilities.

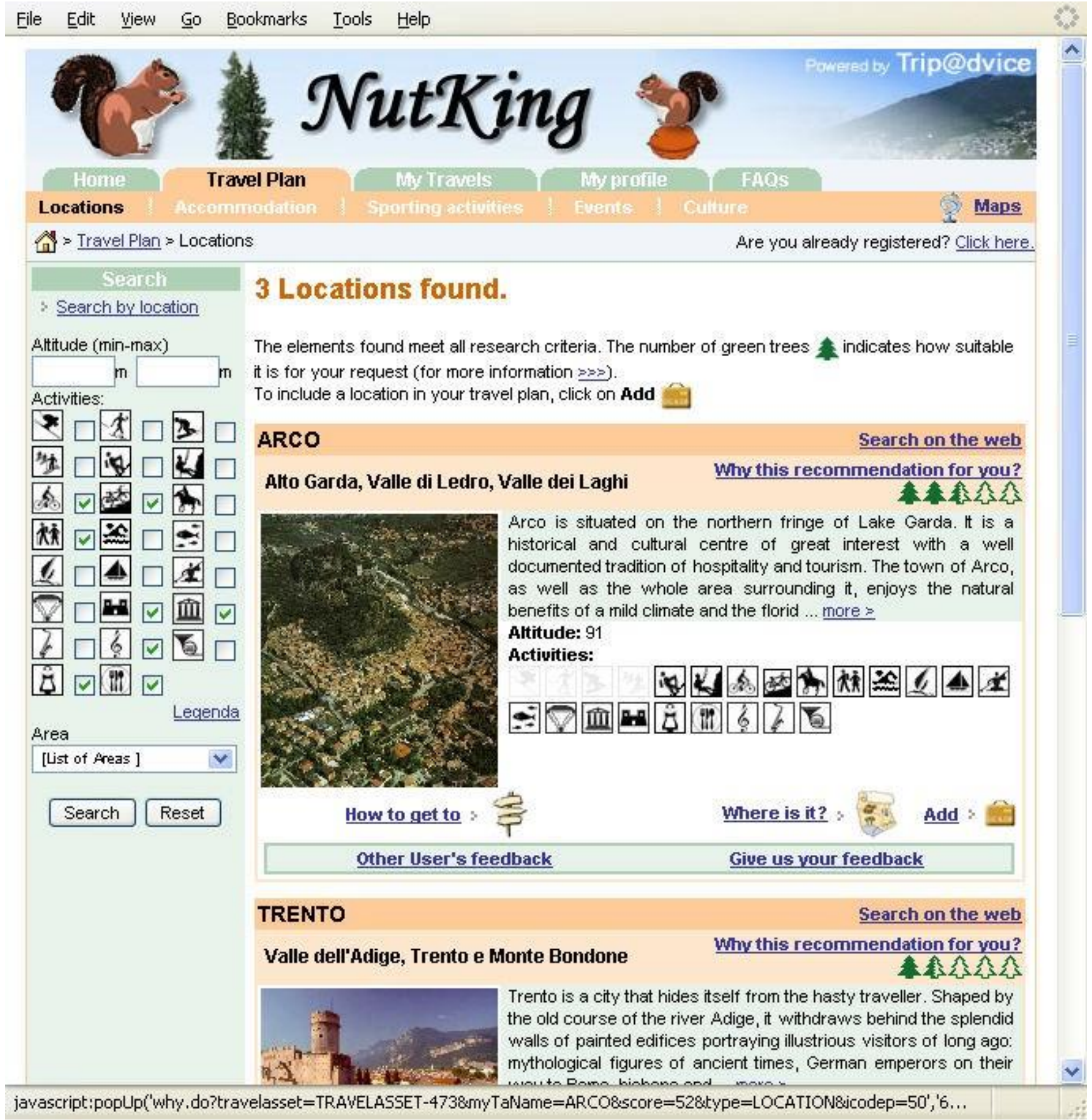
Example

Restaurant	Price	Cusine	Atmosphere
Dolce	10	A	A
Gabbana	12	B	B

- If the user query q is: **price=9** AND **cusine=B** AND **Atm=B**
- And the weights (importance) of the features is: 0.5 price, 0.3 Cusine, and 0.2 Atmosphere
- The Entrée will suggest Dolce first (and then Gabbana)
- A more traditional CBR system will suggest Gabbana because the similarities are (30 is the price range):
- Sim(q,Dolce) = 0.5 * (1 - 1/30) + 0.3 * 0 + 0.2 * 0 = **0.48**
- Sim(q, Gabbana) = 0.5 (1 - 3/30) + 0.3 *1 + 0.2 * 1 = 0.45 + 0.3 + 0.2 = **0.95**

Knowledge-Based Recommender Systems

Knowledge-Based Recommender Systems



Knowledge-Based Recommender Systems

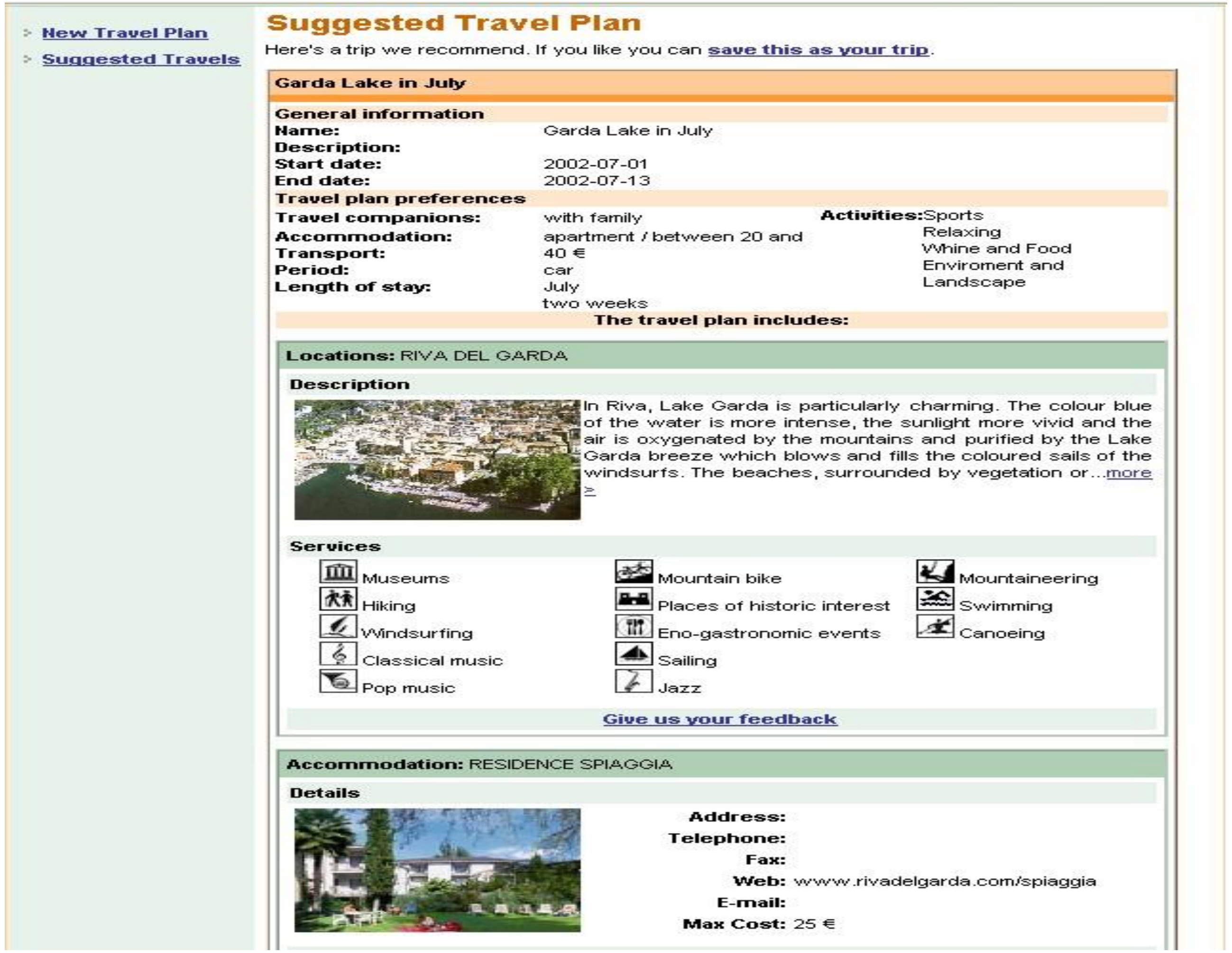
Query Tightening

Knowledge-Based Recommender Systems

Knowledge-Based Recommender Systems

DATA ANALYTICS

Knowledge-Based Recommender Systems



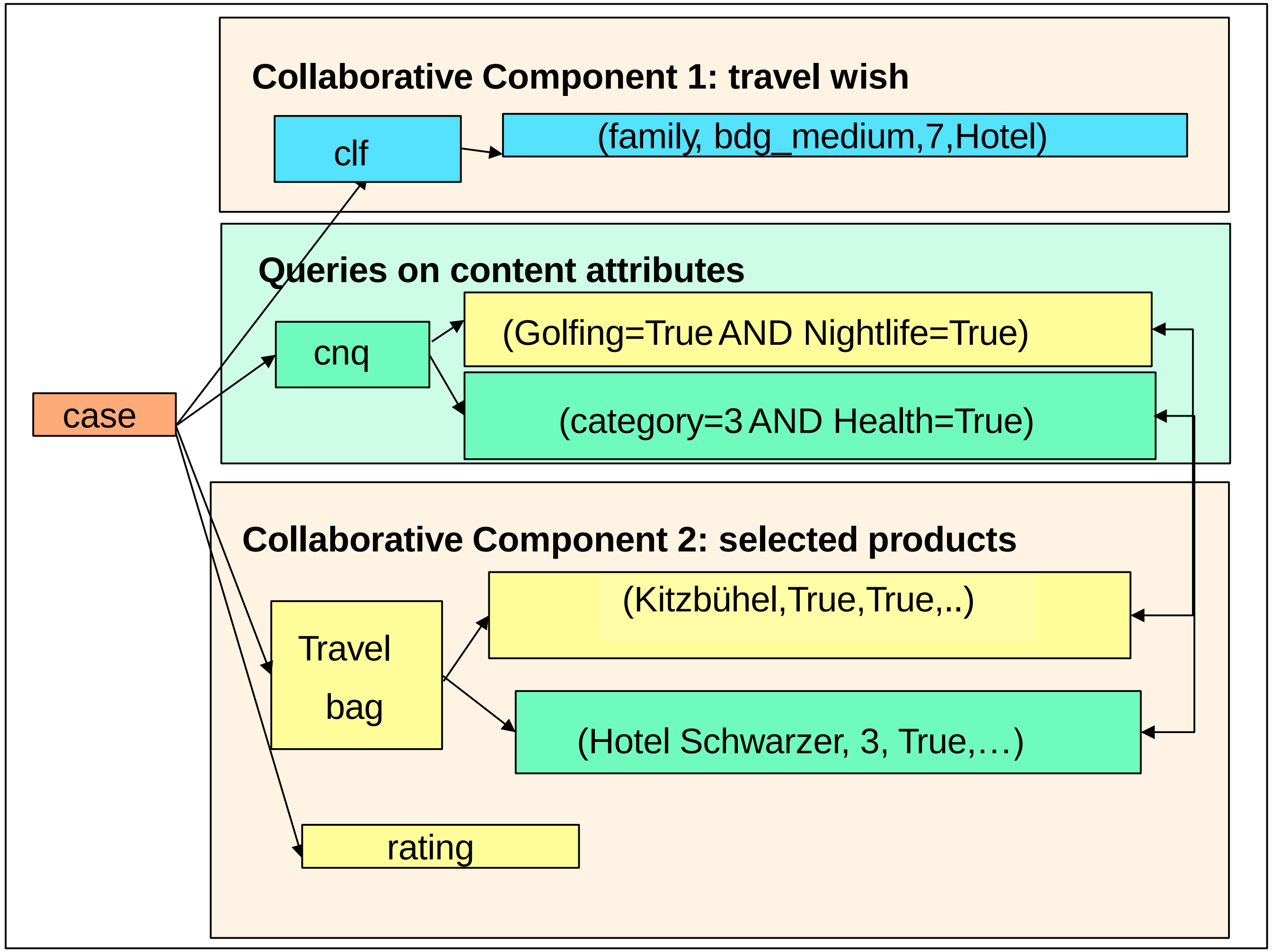
NutKing as a CBR System

- **Problem =** recommend a set of tourism related products and build a travel plan
- **Cases =** All the recommended travel plans that users have built using the system (how they were built and what they contain)
- **Retrieval =** search in the memory travel plans built during “similar” recommendation sessions
- **Reuse**

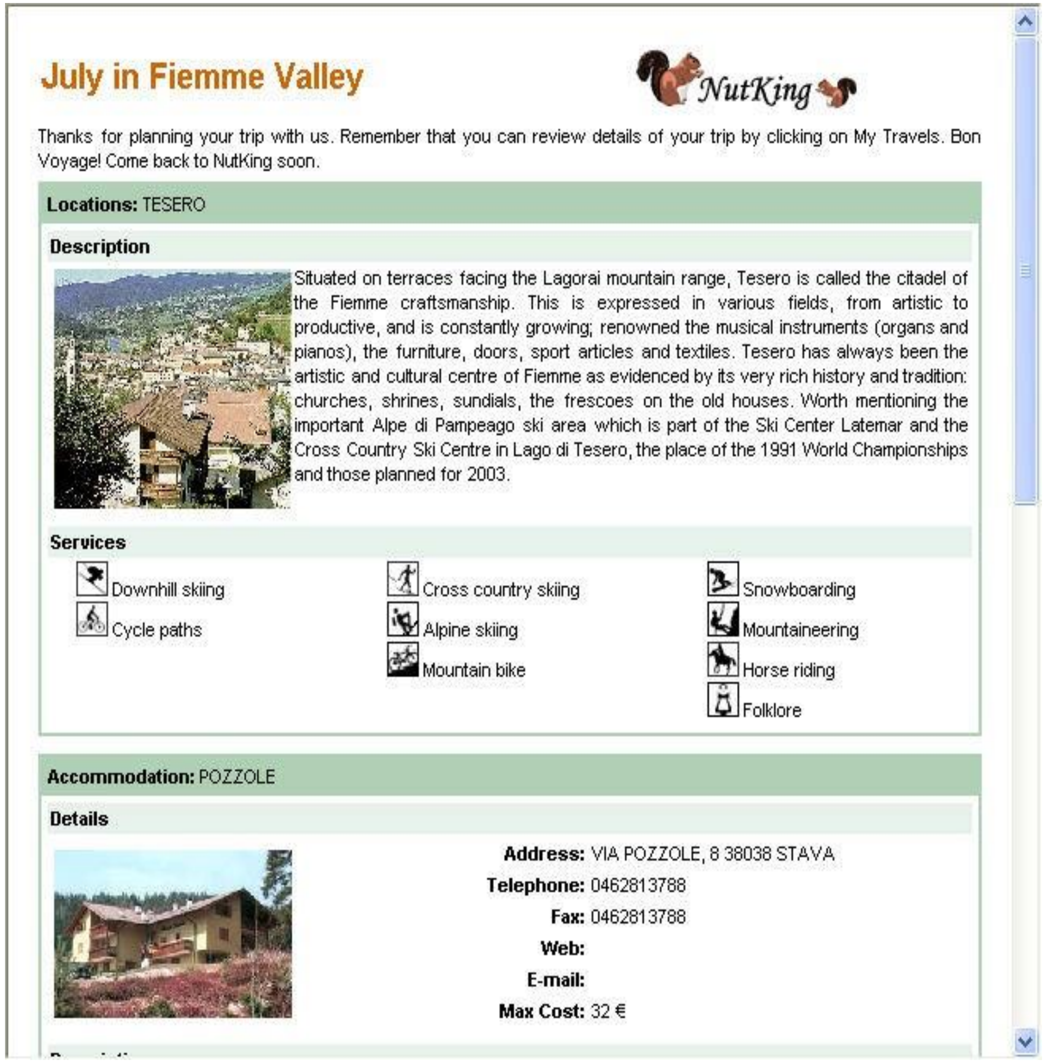
1. extract from previous travel plans elementary components (items) and use them to build a new plan
2. rank items found in the catalogues

Knowledge-Based Recommender Systems

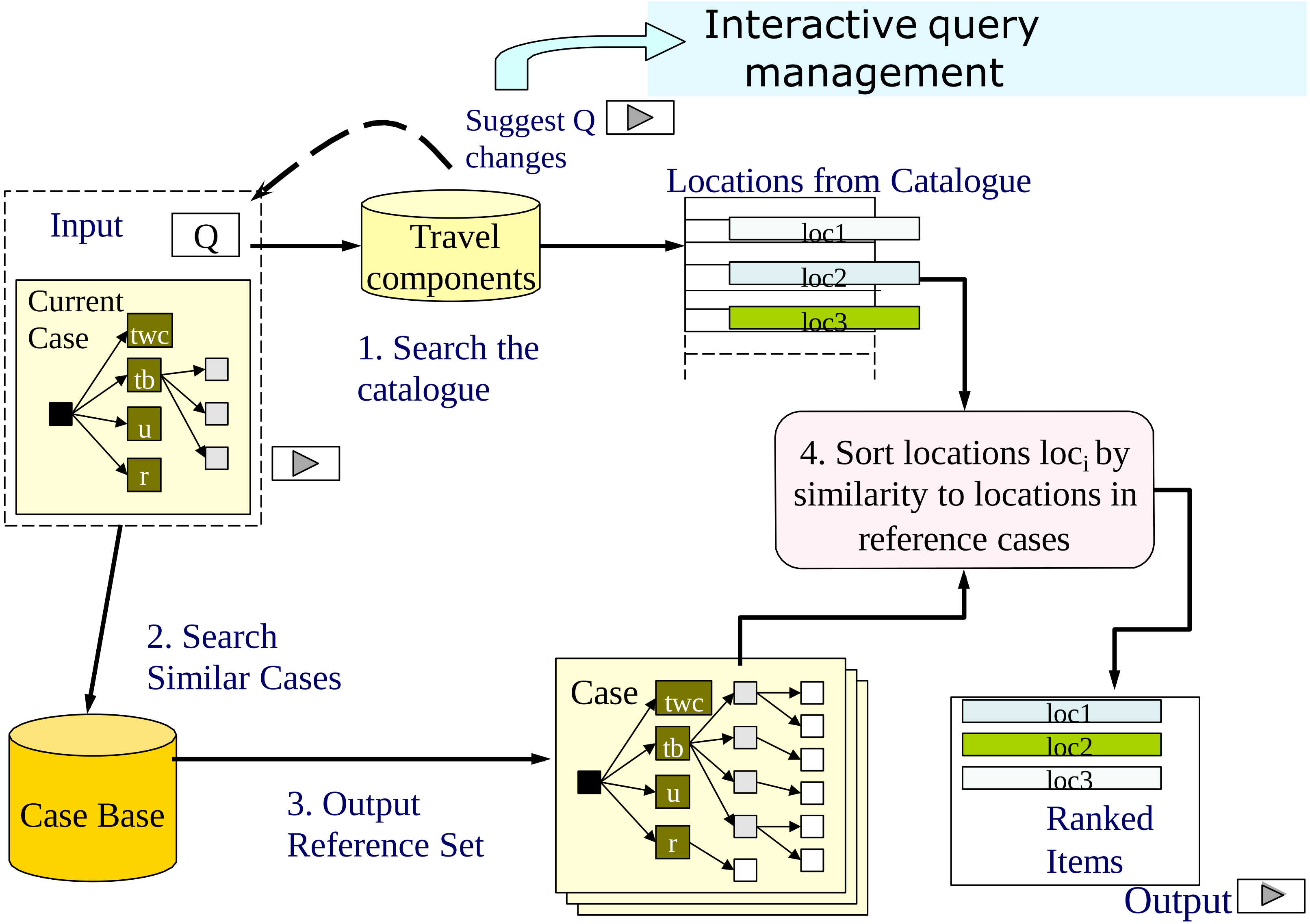
Travel Plan Model and Interaction Session



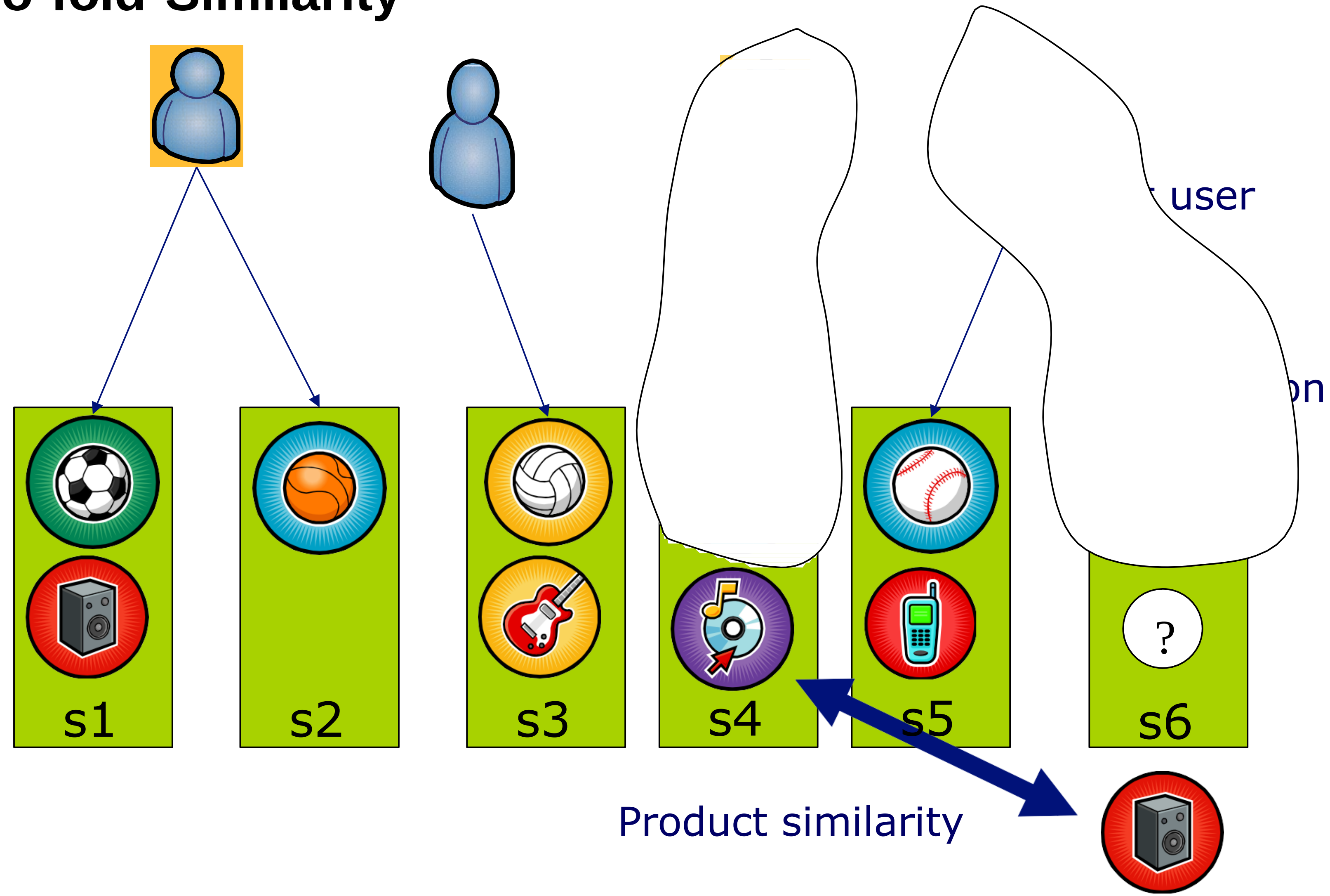
Knowledge-Based Recommender Systems



Item Ranking



Two-fold Similarity

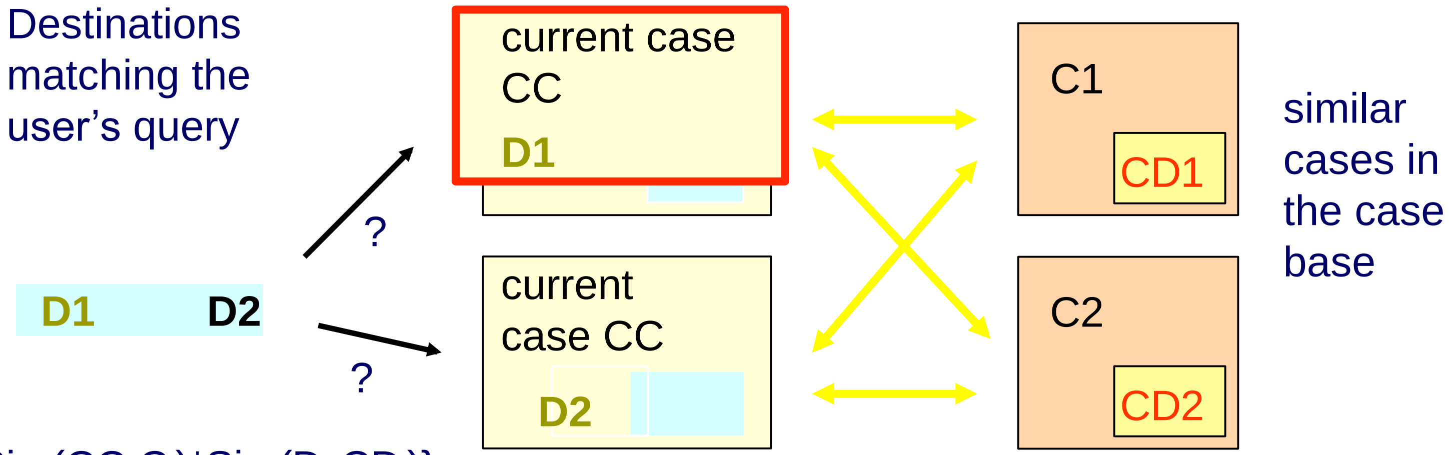


Rank using Two-Fold Similarity

Given the current session case *c* and a set of retrieved products R (using the interactive query management facility - IQM)

1. retrieve 10 cases (c_1, …, c_{10}) from the repository of stored cases (recommendation sessions managed by the system) that are most **similar** to *c* with respect to the collaborative features
2. extract products (p_1, …, p_{10}) from cases (c_1, …, c_{10}) of the same type as those in R
3. For each product r in R compute the Score(r) as the maximum of the product of a) the similarity of r with p_i, the similarity of the current case c and the retrieved case c_i containing p_i
4. sort and display products in R according to the Score(r).

Example: Scoring Two Destinations



$$Score(D_i) = Max_j \{Sim(CC,C_j)*Sim(D_i,CD_j)\}$$

Sim(CC,C1)	0.2
Sim(CC,C2)	0.6

Sim(D1, CD1)	0.4
Sim(D1, CD2)	0.7
Sim(D2, CD1)	0.5
Sim(D2, CD2)	0.3

Score(D1)=Max{0.2*0.4,<u>0.6*0.7</u>}=0.42

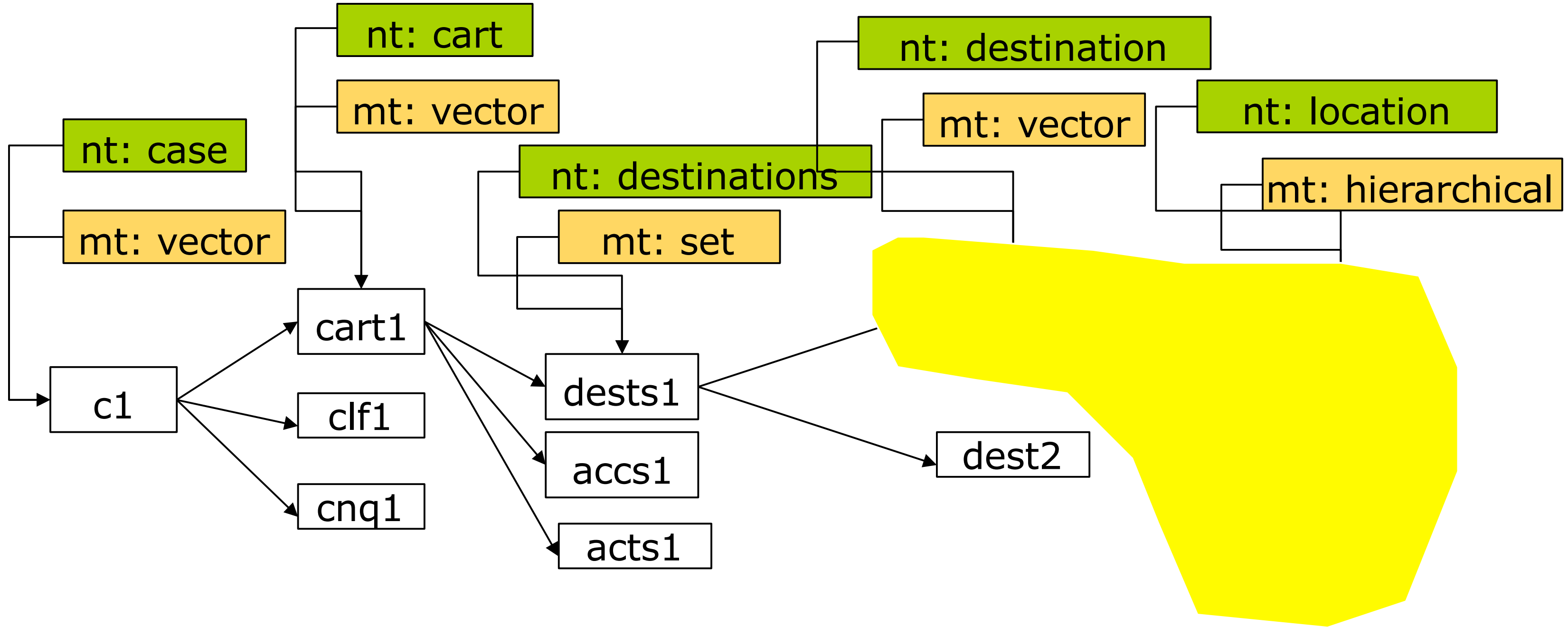
Score(D2)=Max{0.2*0.5,0.6*0.3}=0.18

Tree-based Case Representation

- A case is a rooted tree and each node has a:

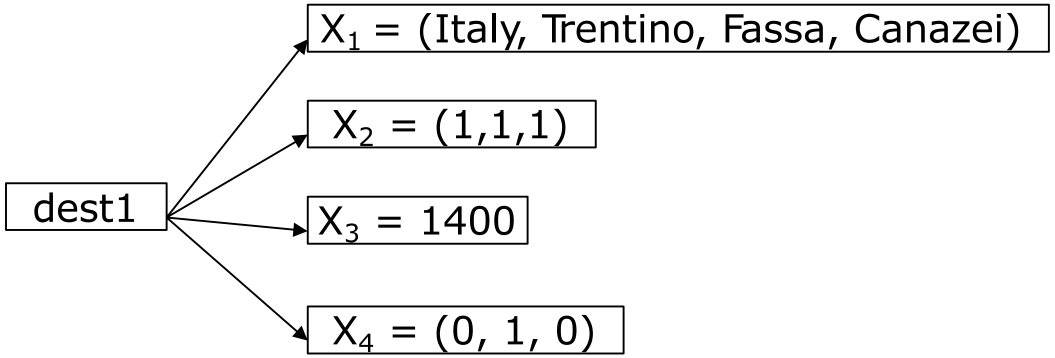
node-type: similarity between two nodes in two cases is defined only for nodes with the same node-type

- **Metric type:** node content structure - how to measure the node similarity with another node in a second case



Item Representation

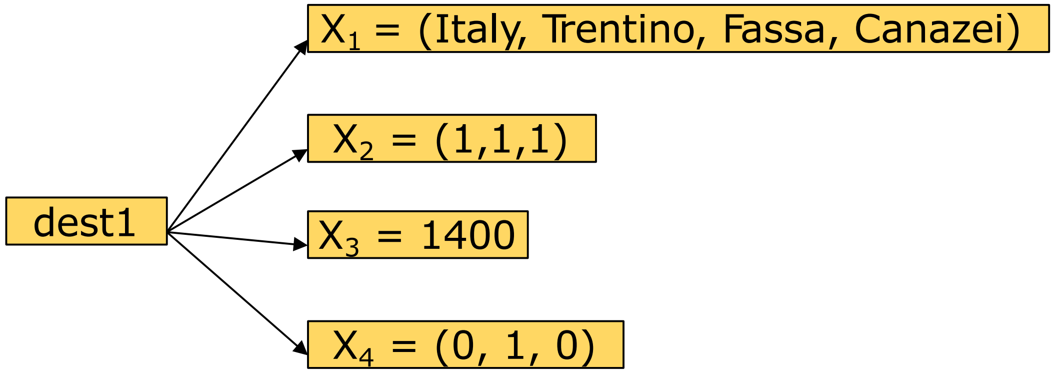
	Node Type	Metric Type	Example: Canazei
X_1	LOCATION	Set of hierarchical related symbols	Country=ITALY, Region=TRENTINO, TouristArea=FASSA, Village=CANAZEI
X_2	INTERESTS	Array of Booleans	Hiking=1, Trekking=1, Biking=1
X_3	ALTITUDE	Numeric	1400
X_4	LOCTYPE	Array of Booleans	Urban=0, Mountain=1, Rivereside=0



Knowledge-Based Recommender Systems

Item Query Language

- For querying purposes items x a represented as simple vector features $x=(x_1, \dots, x_n)$



(Italy, Trentino, Fassa, Canazei, 1, 1, 1, 1400, 0, 1, 0)

- A query is a conjunction of constraints over features:

$q = c_1 \wedge c_2 \wedge \dots \wedge c_m$ where $m \leq n$ and

$$c_k = \begin{cases} x_{i_k} = true & \text{if } x_{i_k} \text{ is boolean} \\ x_{i_k} = v & \text{if } x_{i_k} \text{ is nominal} \\ l \leq x_{i_k} \leq u & \text{if } x_{i_k} \text{ is numerical} \end{cases}$$

Knowledge-Based Recommender Systems

Item Similarity

dest1 → X_1=(Italy, Trentino, Fassa, Canazei)
dest1 → X_2 = (1,1,1)
dest1 → X_3 = 1400
dest1 → X_4 = (0, 1, 0)

If X and Y are two items with same node-type

$$d(X,Y) = (1/\Sigma_i w_i)^{1/2} \, [\Sigma_i w_i \, d_i(X_i,Y_i)^2]^{1/2}$$

where $0 \leq w_i \leq 1$, and i=1..n (number of features).

$$d_i(X_i,Y_i) = \begin{cases} 1 & \text{if } X_i \text{ or } Y_i \text{ are unknown} \\ \text{overlap}(X_i,Y_i) & \text{if } X_i \text{ is symbolic} \\ |X_i - Y_i|/\text{range}_i & \text{if } X_i \text{ is finite integer or real} \\ \text{Jaccard}(X_i,Y_i) & \text{if } X_i \text{ is an array of Boolean} \\ \text{Hierarchical}(X_i,Y_i) & \text{if } X_i \text{ is a hierarchy} \\ \text{Modulo}(X_i,Y_i) & \text{if } X_i \text{ is a circular feature (month)} \\ \text{Date}(X_i,Y_i) & \text{if } X_i \text{ is a date} \end{cases}$$

$$\text{Sim}(X,Y) = 1 - d(X,Y) \quad \text{or} \quad \text{Sim}(X,Y) = \exp(- d(X,Y))$$

Item Similarity Example

1 0.7 0.5 0.3

X_1 = (I, TN, Fassa, Canazei)

X_2 = (1,1,1)

dest1

X_3 = 1400

X_4 = (0, 1, 0)

Y_1 = (I, TN, Fassa,?)

Y_2 = (1,0,1)

dest2

Y_3 = 1200

Y_4 = (1, 1, 0)

$$Sim(dest_1, dest_2) = \exp(-(1/\sqrt{4})\sqrt{d(X_1,Y_1)^2 + \cdots + d(X_4,Y_4)^2})$$

$$= \exp(-(1/\sqrt{4})\sqrt{(0.3)^2 + (1 - 2/3)^2 + ((1400 - 1200)/2000)^2 + (1 - 1/2)^2})$$

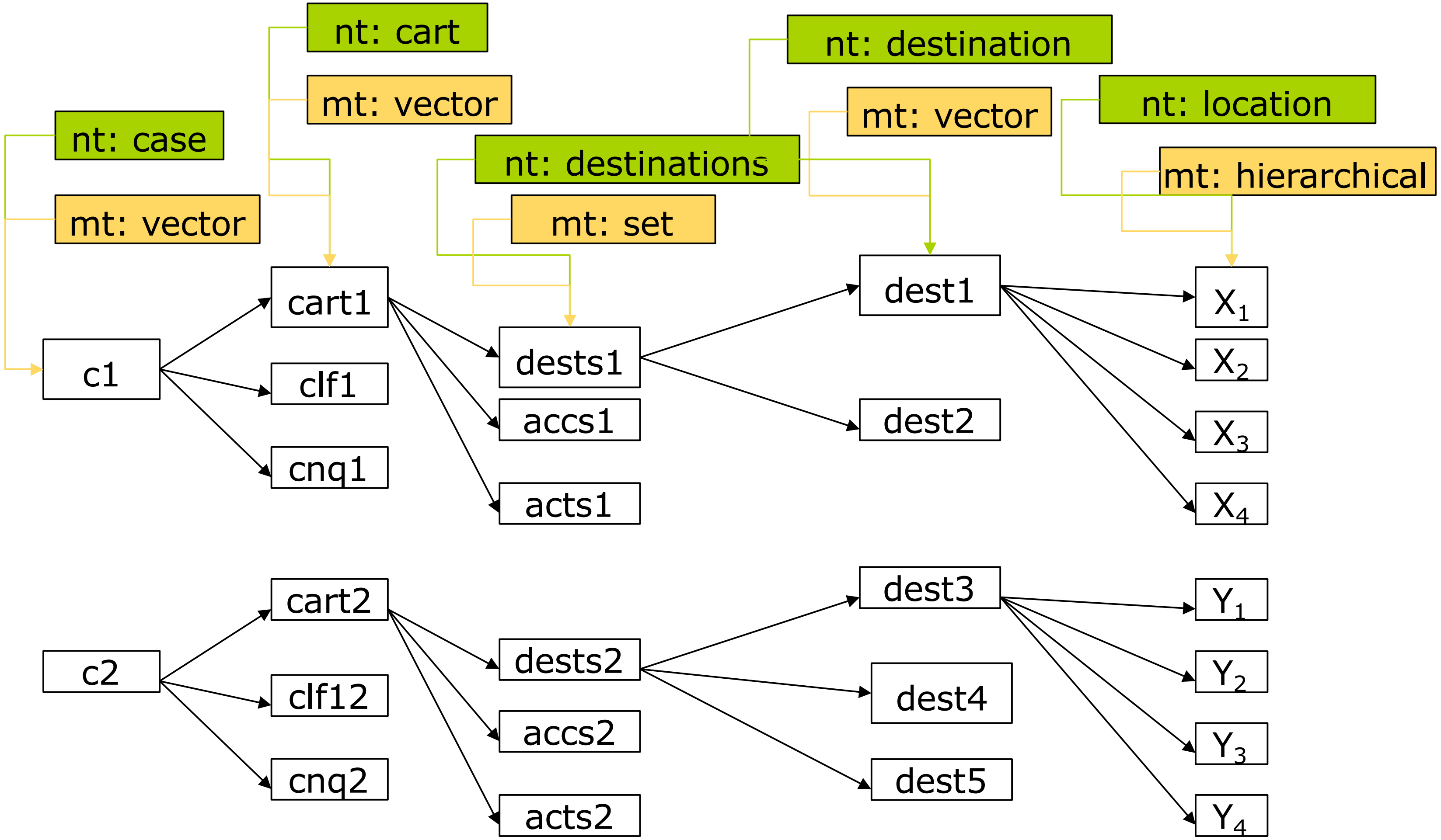
$$= \exp(-(1/\sqrt{4})\sqrt{0,461}) = \exp(-0,339) = 0,712$$

3 in the union

2 in the union

Knowledge-Based Recommender Systems

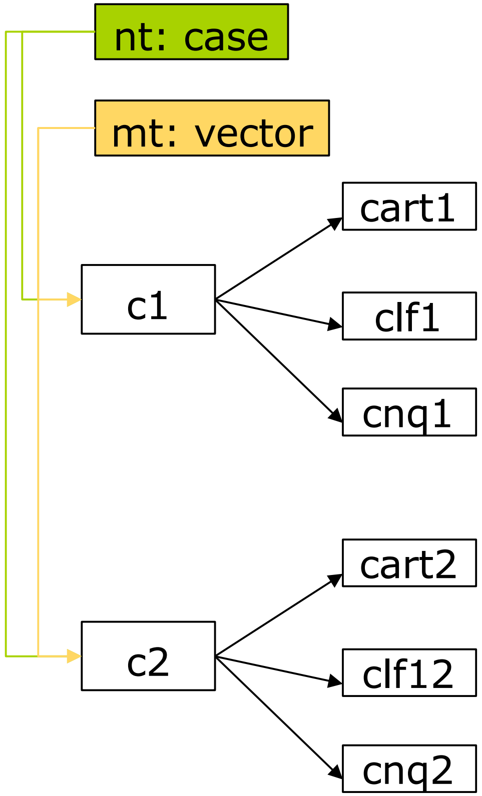
Case Distance



Knowledge-Based Recommender Systems

Case Distance

$$d(c_1,c_2)=\frac{1}{\sqrt{\sum_{i=1}^{3}W_i}}\sqrt{W_1d(cart_1,cart_2)^2+W_2d(clf_1,clf_2)^2+W_3d(cnq_1,cnq_2)^2}$$



CBR Knowledge Containers

1. CBR is a knowledge-based approach to problem solving
2. The knowledge is “contained” into four **containers**
3. **Cases:** the instances belonging to our case base
4. **Case representation language:** the representation language that we decided to use to represent cases
5. **Retrieval knowledge:** the knowledge encoded in the similarity metric and in the retrieval algorithm
6. **Adaptation knowledge:** how to reuse a retrieved solution to solve the current problem.

Knowledge-Based Recommender Systems

Conclusions

- Knowledge-based systems exploits knowledge to map a user to the products she likes
- KB systems uses a variety of techniques
- Knowledge-based systems requires a big effort in term of knowledge extraction, representation and system design
- Many KB recommender systems are rooted in Case-Based Reasoning
- Similarity of complex data objects is required often required in KB RSs.
- NutKing is a hybrid case-based recommender system
- The case is the recommendation session.

Knowledge-Based Recommender Systems

Questions

1. What are the main differences between a CF recommender system and a KB RS (such as activebuyers.com or Entree)?
2. What is the role of query augmentation?
3. What is the basic rationale of a CBR recommender system?
4. What is a case in a CBR recommender system such as Entree?
5. How a CBR recommender system learns to recommend?
6. What are the knowledge containers is a CBR RS?
7. What are the main differences between a “classical” CBR recommender system such as Entrée and Nutking?
8. What are the motivations for the introduction of the double- similarity ranking method?
9. What are the types of local similarity metrics used in Nutking?

Case Study

- Suppose you set up a system, where a guided visual interface is used in order to determine the product of interest to a customer. What category of recommender system does this case fall into?

- Discuss a scenario in which location plays an important role in the recommendation process.

- The chapter mentions the fact that collaborative filtering can be viewed as a generalization of the classification problem. Discuss a simple method to generalize classification algorithms to collaborative filtering. Explain why it is difficult to use such methods in the context of sparse ratings matrices.

References

Text Book:

“Business Analytics, The Science of Data-Driven Making”, U. Dinesh Kumar, Wiley 2017

“Recommender Systems, The text book, Charu C. Aggarwal, Springer 2016 Section 5

Image Courtesy

https://www.ics.uci.edu/~welling/teaching/CS77Bwinter12/presentations/course_Ricci/15-KnowledgeBased.pdf

https://www.researchgate.net/publication/331829548_Knowledge_based_Recommendation_System_in_Semantic_Web-A_Survey

http://www.visitfinland.com/web/guest/travel-planner/home

https://www.ics.uci.edu/~welling/teaching/CS77Bwinter12/presentations/course_Ricci/15-KnowledgeBased.pdf

THANK YOU

Jyothi R.
Assistant Professor,
Department of Computer Science
jyothir@pes.edu