Recommender Systems

- Knowledge-based Recommender Systems -

Institute for Software Technology

Inffeldgasse 16b/2

A-8010 Graz

Austria

Goal

☞ Basic introduction into the concepts of knowledge-based recommender systems.

Why do we need Knowledge-based Recommendation?

Products with low number of available ratings


Time span plays an important role

- five-year-old ratings for computers
- user lifestyle or family situation changes

Customers want to define their requirements explicitly

- "the color of the car should be black"

Knowledge-based Recommendation Approaches

Constraint-based

- based on explicitly defined set of recommendation rules (constraints)
- retrieve items that fulfill recommendation rules and user requirements

Case-based

- based on different types of similarity measures
- retrieve items that are similar to user requirements

Both approaches are similar in their conversational recommendation process

- users specify the requirements
- recommender system tries to identify solutions
- if no solution can be found, users change requirements

Constraint-based Recommendation

Knowledge base

- usually mediates between user model and item properties
- variables
  - user model features (requirements), item features (catalogue)
- set of constraints
  - logical implications (IF user requires A THEN proposed item should possess feature B)
  - hard and soft/weighted constraints
  - solution preferences

Derive a set of recommendable items

- items fulfill requirements and constraints
- explanations – transparent line of reasoning
  - why this recommendation?
  - why no solution could be found and how to deal with this situation?

Constraint-based Recommendation Problem

Select items from this catalog that match the user's requirements

id	price(€)	mpix	opt-zoom	LCD-size	movies	sound	waterproof
P_1	148	8.0	4×	2.5	no	no	yes
P_2	182	8.0	5×	2.7	yes	yes	no
P_3	189	8.0	10×	2.5	yes	yes	no
P_4	196	10.0	12×	2.7	yes	no	yes
P_5	151	7.1	3×	3.0	yes	yes	no
P_6	199	9.0	3×	3.0	yes	yes	no
P_7	259	10.0	3×	3.0	yes	yes	no
P_8	278	9.1	10×	3.0	yes	yes	yes

User's requirements can, for example, be

- "the price should be lower than 300 €"
- "the camera should be suited for sports photography"

Constraint-based Recommendation Problem (defined as CSP)

A knowledge-based recommendation problem can be defined as a Constraint Satisfaction Problem (CSP):

$$CSP(X_I \cup X_U, D, R \cup KB \cup I)$$

Definition:

- X_I, X_U: variables describing product and user model with domain D
- KB: knowledge base with constraints (e.g**. if** purpose=*on travel* **then** lower focal length < *28mm*)
- R: specific requirements of a user (e.g. purpose = *on travel*)
- I: product catalog

Solution:

- complete and consistent assignment to X_I and X_U

CSP example

Product Catalog of Bikes

Bike-1: Color:red, Type:racing, Price: 1000 Euro

Bike-2: Color:green, Type:city, Price: 500 Euro

Bike-3: Color:red, Type:city, Price: 300 Euro

Bike-4: Color:yellow, Type:city, Price: 800 Euro

Bike-5: Color:red, Type:mountain, Price: 1200 Euro

C_{KB}

\+ **User-1**

Color:red, Type:mountain, Price:<500

\+ **User-2**

Color:yellow, Type:city, Price:< 1000

\+ **User-3**

Color:green, Type:mountain, Price:< 900

C_R

- $V = \{color, type, price\}$
- $D = \{dom(Color) = \{red, green, yellow\},$
  $dom(Type) = \{racing, city, mountain\},$
  $dom(Price) = \{300, 500, 800, 1000, 1200\}\}$
- $C_{\mathrm{KB}} = \{\{Color = red \wedge Type = racing \wedge Price = 1000\} \vee$
  $\{Color = green \wedge Type = city \wedge Price = 500\} \vee$
  $\{Color = red \wedge Type = city \wedge Price = 300\} \vee$
  $\{Color = yellow \wedge Type = city \wedge Price = 800\} \vee$
  $\{Color = red \wedge Type = mountain \wedge Price = 1200\}\}$
- $C_{\mathrm{R}} = \{Color = red \wedge Type = mountain \wedge Price < 500\}$

Exercise 1

Define a simple CSP for digital camera recommendation

- X_I: variables describing the products (items) + D
- X_U: variables describing the user model + D
- KB: knowledge base with recommendation knowledge
- R: provide an example of user requirements
- I: product catalog (set of offered items)

id	price(€)	mpix	opt-zoom	LCD-size	movies	sound	waterproof
P_1	148	8.0	4×	2.5	no	no	yes
P_2	182	8.0	5×	2.7	yes	yes	no
P_3	189	8.0	10×	2.5	yes	yes	no
P_4	196	10.0	12×	2.7	yes	no	yes
P_5	151	7.1	3×	3.0	yes	yes	no
P_6	199	9.0	3×	3.0	yes	yes	no
P_7	259	10.0	3×	3.0	yes	yes	no
P_8	278	9.1	10×	3.0	yes	yes	yes

Conjunctive Query

Different from a constraint solver

- goal is not to find valid instantiations for CSP variables but to identify tuples (items) that fulfill the criteria of the query

Conjunctive query is executed on the item catalog

- a conjunctive database query
- a set of selection criteria that are connected conjunctively

σ[criteria](P)

- P*:* product assortment
- example: $\sigma[mpix \geq 10, price < 300](P) = \{p4, p7\}$

Exercise 2

Define an appartment recommendation problem as a CSP:

- X_I: variables describing the products (items) + D
- X_U: variables describing the user model + D
- KB: knowledge base with recommendation knowledge
- R: provide an example for user requirements
- I: product catalog (set of offered items)

Interacting with Constraint-based Recommenders

The user specifies his or her initial preferences

- all at once or
- Incrementally, for example, on question per page
- interactive dialog

The user is presented with a set of matching items

- with explanation as to why a certain item was recommended

The user might revise his or her requirements

- see alternative solutions
- narrow down the number of matching items

Interacting with Constraint-based Recommenders: Defaults

Support customers to choose a reasonable alternative

- unsure about which option to select
- simply do not know technical details

Type of defaults

- static defaults (e.g., OS-Installation = “Linux”)
- dependent defaults (e.g., if country = “Austria” then language = “German”)
- derived defaults (e.g., based on collaborative approaches)

Selecting the next question

- most users are not interested in specifying values for all properties
- identify properties that may be interesting for the user

Derived Defaults: Example

- **Interaction log: user item selections**

session	price(€)	mpix	opt-zoom	LCD-size	movies	sound	waterproof
1	148	**8.0**	4×	2.5	no	no	**yes**
2	182	8.0	5×	2.7	yes	yes	no
3	148	**8.0**	4×	2.5	no	no	**yes**
4	182	8.0	5×	2.7	yes	yes	no
5	182	8.0	5×	2.7	yes	yes	no
6	199	9.0	3×	3.0	yes	yes	no
7	182	8.0	5×	2.7	yes	yes	no
current	**-**	**8.0**	**-**	**?**	**-**	**-**	**yes**

- **Item properties selected by the current user:**
  - mpix = 8.0, waterproof="yes"
  - Nearest neighbors of current session: {1,3}
  - Recommendation: LCD-size = 2.5

Unsatisfied Requirements

"no solution could be found"

Constraint relaxation

- the goal is to identify relaxations to the original set of constraints
- relax constraints of a recommendation problem until a corresponding solution has been found

Users could also be interested in repair proposals

- recommender can calculate a solution by adapting the user-defined requirements

Unsatisfied Requirements: Example

id	price(€)	mpix	opt-zoom	LCD-size	movies	sound	waterproof
P_1	148	8.0	4×	2.5	no	no	yes
P_2	182	8.0	5×	2.7	yes	yes	no
P_3	189	8.0	10×	2.5	yes	yes	no
P_4	196	10.0	12×	2.7	yes	no	yes
P_5	151	7.1	3×	3.0	yes	yes	no
P_6	199	9.0	3×	3.0	yes	yes	no
P_7	259	10.0	3×	3.0	yes	yes	no
P_8	278	9.1	10×	3.0	yes	yes	yes

customer requirements R =
{
r_1:price<=150,
r_2:opt-zoom=5x,
r_3:sound=yes,
r_4:waterproof=yes
}

No solution found:

$\sigma_{[R]}P = \{\}$

which requirements should be changed?

Conflict Sets: Example

id	price(€)	mpix	opt-zoom	LCD-size	movies	sound	waterproof
P_1	148	8.0	4×	2.5	no	no	yes
P_2	182	8.0	5×	2.7	yes	yes	no
P_3	189	8.0	10×	2.5	yes	yes	no
P_4	196	10.0	12×	2.7	yes	no	yes
P_5	151	7.1	3×	3.0	yes	yes	no
P_6	199	9.0	3×	3.0	yes	yes	no
P_7	259	10.0	3×	3.0	yes	yes	no
P_8	278	9.1	10×	3.0	yes	yes	yes

$\sigma[r_1,r_2]P = \emptyset$

$\sigma[r_2,r_4]P = \emptyset$

$\sigma[r_1,r_3]P = \emptyset$

Conflict Sets and Diagnoses (1)

id	price(€)	mpix	opt-zoom	LCD-size	movies	sound	waterproof
P_1	148	8.0	4×	2.5	no	no	yes
P_2	182	8.0	5×	2.7	yes	yes	no
P_3	189	8.0	10×	2.5	yes	yes	no
P_4	196	10.0	12×	2.7	yes	no	yes
P_5	151	7.1	3×	3.0	yes	yes	no
P_6	199	9.0	3×	3.0	yes	yes	no
P_7	259	10.0	3×	3.0	yes	yes	no
P_8	278	9.1	10×	3.0	yes	yes	yes

what are possible **values?**

Conflict Set

$CS=\{r_1, r_2, \ldots, r_k\} \subseteq R$ s.t. $\sigma[CS]P = \emptyset$

CS is minimal if not $\exists$ CS': $CS' \subset CS$

[Junker 2004]

conflict sets=
$\{CS_1:\{r_1,r_2\},$
$CS_2:\{r_2,r_4\},$
$CS_3:\{r_1,r_3\}\}$

Diagnosis

$d_i \in D$ for (P, R):

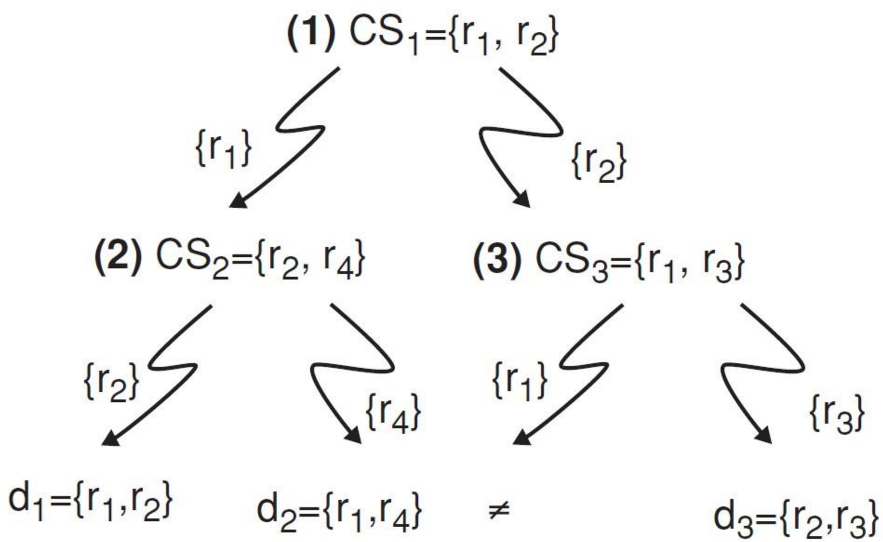
$d_i \subseteq R$ s.t. $\sigma[R-d_i]P \neq \emptyset$

[Reiter1987]

diagnoses D =
$\{d_1:\{r_1,r_2\},$
$d_2:\{r_1,r_4\},$
$d_3:\{r_2,r_3\}\}$

Conflict Sets and Diagnoses (2)

Calculate diagnoses for unsatisfied requirements (HSDAG = Hitting Set Directed Acyclic Graph)



[Reiter1987]

The diagnoses derived from the conflict sets $\{CS_1,CS_2,CS_3\}$ are $\{d_1:\{r_1,r_2\}, d_2:\{r_1,r_4\},d_3:\{r_2,r_3\}\}$

Determining Repair Actions: Example

id	price(€)	mpix	opt-zoom	LCD-size	movies	sound	waterproof
P_1	148	8.0	4×	2.5	no	no	yes
P_2	182	8.0	5×	2.7	yes	yes	no
P_3	189	8.0	10×	2.5	yes	yes	no
P_4	196	10.0	12×	2.7	yes	no	yes
P_5	151	7.1	3×	3.0	yes	yes	no
P_6	199	9.0	3×	3.0	yes	yes	no
P_7	259	10.0	3×	3.0	yes	yes	no
P_8	278	9.1	10×	3.0	yes	yes	yes

π repair attributes

σ repair products

$$d=\{r_1,r_2\} \ldots \pi_{[r1:price,\ r2:opt\text{-}zoom]}\sigma_{[r3:sound=yes,\ r4:waterproof=yes]}P$$

$$rep_1 = \pi_{[attributes(\mathbf{d1})]}\sigma_{[R\text{-}d1]}P = \pi_{[r1:price,\ r2:opt\text{-}zoom]}\sigma_{[r3:sound=yes,\ r4:waterproof=yes]}P$$

$$rep_2 = \pi_{[attributes(\mathbf{d2})]}\sigma_{[R\text{-}d2]}P = \pi_{[r1:price,\ r4:waterproof]}\sigma_{[r2:opt\text{-}zoom=5x,\ r3:sound=yes]}P$$

$$rep_3 = \pi_{[attributes(\mathbf{d3})]}\sigma_{[R\text{-}d3]}P = \pi_{[r2:opt\text{-}zoom,\ r3:sound]}\sigma_{[r1:price <= 150,\ r4:waterproof=yes]}P$$

repair	price	opt-zoom	sound	waterproof
rep_1	r_1':price=278	r_2': opt-zoom=10x	r_3:sound=yes	r_4:waterproof=yes
rep_2	r_1':price=182	r_2:opt-zoom=5x	r_3:sound=yes	r_4':waterproof=no
rep_3	r_1:price=148	r_2': opt-zoom=4x	r_3':sound=no	r_4:waterproof=yes