Towards Recommending Configurable Offerings

Juha Tiihonen¹,
Alexander Felfernig²

¹ Helsinki University of Technology
Department of Computer Science and Engineering

² University Klagenfurt
Intelligent Systems and Business Informatics
Outline

- Motivation and background
- Recommendation scenarios
- Distance metrics
- Example: Most popular choice
- Discussion, future work and conclusions
Motivation

- What of these configuration alternatives should I select? (Mass confusion)
  - “I want to edit high-definition videos, how to select components to my computer”
  - “Does this nVidia GeForce 9800 GT 512MB suit my requirements?”
- Preferences are constructed – which alternatives are presented to the customer, and their order highly affects the final selections
- Overlooking configuration alternatives which could better suit to the customers' wishes and needs
  ➔ Users of configurators need more intuitive interaction mechanisms to select product and service alternatives
  ➔ Integrate recommendation and configuration technologies
In this paper...

- Apply and extend case-based recommendation to configuration settings
- We extend previous recommendation approaches e.g. [Cöster et al.]
  - To take into account importance weights of features
  - To take into account similarity (substitutes equality)
  - To take consistency into account
    - generate only recommendations that are consistent with customer requirements and the configuration model
- In the paper we discuss “Nearest neighbor”, “Weighted Majority Voter”, “Most Popular Choice”
- Identify scenarios for recommendation supported configuration & discuss topics for future work
Recommendation scenarios

- Selecting a suitable base product line
- Recommending a complete configuration
- Recommending how to complete a configuration
- Recommending a subconfiguration
- Recommending individual attribute or component settings

→ a high diversity of usage and integration scenarios for recommendation technologies
Sample Configuration model & cases

<table>
<thead>
<tr>
<th>Feature</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video</td>
<td>no</td>
</tr>
<tr>
<td>Photos</td>
<td>no</td>
</tr>
<tr>
<td>Gaming</td>
<td>2d</td>
</tr>
<tr>
<td>PR</td>
<td>as</td>
</tr>
<tr>
<td>mb</td>
<td>a1</td>
</tr>
<tr>
<td>Ram</td>
<td>1</td>
</tr>
<tr>
<td>HD</td>
<td>h2</td>
</tr>
<tr>
<td>GC</td>
<td>g2</td>
</tr>
<tr>
<td>OD</td>
<td>dr</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>k</th>
<th>f1</th>
<th>f2</th>
<th>f3</th>
<th>f4</th>
<th>f5</th>
<th>f6</th>
<th>f7</th>
<th>f8</th>
<th>f9</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>no</td>
<td>no</td>
<td>2d</td>
<td>as</td>
<td>a1</td>
<td>1</td>
<td>h2</td>
<td>no</td>
<td>dr</td>
<td>ba</td>
</tr>
<tr>
<td>2</td>
<td>no</td>
<td>std</td>
<td>2d</td>
<td>as</td>
<td>a2</td>
<td>1</td>
<td>h5</td>
<td>g2</td>
<td>dw</td>
<td>st</td>
</tr>
<tr>
<td>3</td>
<td>sd</td>
<td>std</td>
<td>adv</td>
<td>i4</td>
<td>i2</td>
<td>3</td>
<td>h9</td>
<td>g9</td>
<td>dw</td>
<td>ad</td>
</tr>
<tr>
<td>4</td>
<td>hd</td>
<td>adv</td>
<td>adv</td>
<td>i9</td>
<td>i2</td>
<td>4</td>
<td>h9</td>
<td>g9</td>
<td>bw</td>
<td>ad</td>
</tr>
<tr>
<td>5</td>
<td>sd</td>
<td>adv</td>
<td>3d</td>
<td>i4</td>
<td>i1</td>
<td>2</td>
<td>h9</td>
<td>g8</td>
<td>dw</td>
<td>st</td>
</tr>
</tbody>
</table>

\[ ph \neq no \implies od.dw = yes: \text{to archive photos} \]
\[ ph = adv \implies hd.capacity \geq 500: \text{disk space for photos} \]
\[ ph = adv \implies pr.C'Scr \geq 2500: \text{CPU for advanced photo} \]

Other constraints in paper...
Distance metrics

- Distance functions determine similarity or dissimilarity of individual feature values
  - Traditional equality may be too strict - close values or configurations could remain ignored
- Apply Heterogeneous Value Difference Metric (HVDM) [Wilson 1997]
  - Cope with symbolic (nominal) and numeric features
  - Learns the similarity of symbolic values in a domain automatically

\[
d_{f_i}(x, y) = \begin{cases} 
1 & \text{if } x \text{ or } y \text{ is unknown; otherwise} \\
v_{d_{m_i}}(x, y), & \text{if } f_i \text{ is symbolic} \\
d_{i\text{ff}_{i}}(x, y), & \text{if } f_i \text{ is linear}
\end{cases}
\]

\[
v_{d_{m_i}}(x, y) = \sqrt{\sum_{c=1}^{C} \left| \frac{N_{f_i,x,c}}{N_{f_i,x}} - \frac{N_{f_i,y,c}}{N_{f_i,y}} \right|^2}
\]

\[
d_{i\text{ff}_{i}}(x, y) = \frac{|x - y|}{4\sigma_{f_i}}
\]
Most Popular Choice

- Recommends values for remaining features from one configuration

\[ Pr(c, u, F_u) = Pr_{\text{basic}}(c, \bar{F}_u) \times \prod_{j \in F_u} Pr(f_{j,u} = f_{j,u} | Conf) \]

- Extended \( Pr_{\text{basic}} \) from that presented in [Cöster et al, 2002]
- Bayesian predictor part as in original
**Pr_{basic} (original version)**

<table>
<thead>
<tr>
<th>k</th>
<th>f1</th>
<th>f2</th>
<th>f3</th>
<th>f4</th>
<th>f5</th>
<th>f6</th>
<th>f7</th>
<th>f8</th>
<th>f9</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>no</td>
<td>no</td>
<td>2d</td>
<td>as</td>
<td>a1</td>
<td>1</td>
<td>h2</td>
<td>no</td>
<td>dr</td>
<td>ba</td>
</tr>
<tr>
<td>2</td>
<td>no</td>
<td>std</td>
<td>2d</td>
<td>as</td>
<td>a2</td>
<td>1</td>
<td>h5</td>
<td>g2</td>
<td>dw</td>
<td>st</td>
</tr>
<tr>
<td>3</td>
<td>sd</td>
<td>std</td>
<td>adv</td>
<td>i4</td>
<td>i2</td>
<td>3</td>
<td>h5</td>
<td>g9</td>
<td>dw</td>
<td>ad</td>
</tr>
<tr>
<td>4</td>
<td>hd</td>
<td>adv</td>
<td>adv</td>
<td>i9</td>
<td>i2</td>
<td>4</td>
<td>h9</td>
<td>g9</td>
<td>bw</td>
<td>ad</td>
</tr>
<tr>
<td>5</td>
<td>sd</td>
<td>adv</td>
<td>3d</td>
<td>i4</td>
<td>i1</td>
<td>2</td>
<td>h9</td>
<td>g8</td>
<td>dw</td>
<td>st</td>
</tr>
</tbody>
</table>

"My popularity" (my probability) \[ \text{Pr}_{basic}(c, F_u) = \prod_{j \in F_u} \frac{\text{count}(f_j, f_j, c)}{K} \]

"I have popular values in those features that the active user has not selected, so use my feature values!"
A feature value gets support iff neighbor configurations have feature values within maximum distance \( \Delta \), here we use \( \Delta = 0.8 \)

\[
(1-(0.707))^2 + (1-(0.707))^2 + (1-0^2) + (1-(0.707))^2 + (1-0^2) = 2.257
\]

The support quickly decreases when the distance increases (square)
**Pr**_{basic} of **Conf**_{5}

\[
Pr_{basic}(c, \bar{F}_u) = \prod_{j \in \bar{F}_u} \frac{\sum_{k=1}^{K} s_{f_j}(f_j, c, f_j, k)}{\sum_{o \in \text{dom}(f_j)} \sum_{k=1}^{K} s_{f_j}(v, f_j, k) \times \min(1, \text{count}(f_j, v))}
\]

- For \(j=\text{pr}\) (processor) and \(v=i4\): \(2.257 / 5.601 = 0.4031\)
- **conf5**: pr (i4) = 0.403, mb (i1) = 0.286, me (2) = 0.280, hd (h9) = 0.444, gc (g8) = 0.286, od (dw) = 0.600
  \(\rightarrow Pr_{basic} = 0.002461\)
Bayesian predictor

- Bayesian predictor for the user profile \( u \) to have the values already selected, given an the existing neighbors (examined by neighbor), \( P(f_{j,u} = f_{j,u}|Conf) \)
  - Same as in original Cöster formulas
  - \( m \)-estimate [Bratko et al. 1996] stabilizes probability even in case of (too) few samples
    - Assumes \( m \) virtual samples with initial probability \( p \)
    - Future work may improve parameters

\[
\prod_{f_j \in F} P(f_{j,u} = f_{j,u}|Conf) = \prod_{f_j \in F} m_{est}(eqcfgs_m(c, F \cup f_j, f_j, f_{j,u}), eqcfgs(c, F), 1/K, K)
\]

\[
m_{est}(N_c, N, p, m) = \frac{N_c + mp}{N + m}
\]
Discussion and Future work (1)

- Evaluation (analytical approach, user studies)
- Implementation with configurator integration
- How to provide recommendations in the user interface
  - As default selections, individualized recommendation indication of alternatives, individualized explanatory texts or help, hide unsuitable values, warn against non-recommended combinations
- Consistency of recommendations vs. (partial) configuration
  - E.g. Should a low-weight incompatible selected value always prevent recommending an otherwise superior alternative of an important feature?
Discussion and Future work (2)

- The algorithms and their parameters (e.g. m-estimate)
  - How to take into account variation of the structure of the product
  - Relatively independent subconfigurations? Or everything affects everything?
  - Varying weights of features with user preferences

- Similarity metrics
  - How to determine applicable classifications for learning similarity?
  - Classifiers based on the whole product, or e.g. by sub-system?
  - Does our approach produce satisfying results? Or is manual determination of similarity needed?

- Reconfiguration with recommendation support
  - Long relationships $\rightarrow$ changing needs & situations
    $\rightarrow$ update solution & avoid switching costs or weakening of terms

- Relationship of defaults and recommendations?
Conclusions

- Identified different scenarios for recommendation
- Showed the potential benefits of integrating recommendation with configuration technologies
  - Allows for the derivation of individualized and personalized product and service offerings
  - Potential for reducing the mass confusion phenomenon
  - An important step towards configuration systems which more actively support users in preference construction processes.
- We have developed recommendation approaches
  - To take into account importance weights of features
  - To take into account similarity (substitutes equality)
  - To take consistency into account
- Identified numerous areas of future work
Questions?

Thank you for your attention!