A Survey of Context-Aware Recommendation Systems

Chuong C. Vo, Torab Torabi, Seng W. Loke

Abstract

1 Introduction

Recommender systems have become an important approach to help users deal with information overload and provide personalised suggestions and have been successfully applied in both industry and academia. Recommender systems support users by identifying interesting products and services, when the number and diversity of choices outstrips the user's capability of making good decisions. One of the most promising recommending technologies is collaborative filtering Hill et al. (1995); Shardanand and Maes (1995). Essentially a nearest-neighbour method is applied to a user's ratings, and provides the user with recommendations based on how her likes and dislikes relate to a large user community.

Little research has been conducted to help users learn and explore a complicated pervasive interactive system using a recommender system. Typical approaches to proactively introducing functionality to a user include "Tip of the day", and "Did you know" Owen (1986), but these are often irrelevant to the user and are presented in a decontextualised way Fischer (2001).

Personalised recommendation service aims to provide products, content, and services tailored to individuals, satisfying their needs in a given context based on knowledge of their preferences and behaviour Adomavicius and Tuzhilin (2005). The personalised services are usually realised by the form of recommender systems. Recommender systems appeared as an independent research field in the mid-1990s Adomavicius and Tuzhilin (2005). They help users deal with information overload by providing personalised recommendations related to products, content, and services, usually accomplished by the use of personal profile information and item attributes. In the past decade, most works focused on modifying algorithms for greater effectiveness and correct recommendations Adomavicius et al. (2005). They used methods from disciplines such as human-computer interaction, statistics, data mining, machine learning, and information retrieval Adomavicius and Tuzhilin (2005). Recommender systems can be classified into three types according to how recommendations are made Adomavicius and Tuzhilin (2005):

Content-based Recommendation It recommends items to users that are similar to those they preferred previously. The analysis of similarity is based on the items attributes.

Collaborative Recommendation It recommends items to users according to the item ratings of other people who have characteristics similar to their own. The analysis of similarity is based on the users tastes and preferences.

Hybrid Recommendation It is a combination of content-based and collaborative recommendations.

Traditionally, recommender systems usually compute the similarity using two-dimensional user-item information. They failed to take into consideration contextual information which might affect users' decision making behaviour, such as time, location, companions, weather, and so on. Including human-in-context information as one system design factor is necessary for producing more accurate recommendations.

Adomavicius and Tuzhilin proposed a multidimensional approach to incorporate contextual information into the design of recommender systems Adomavicius et al. (2005). They also proposed a multidimensional rating estimation method based on the reduction-based approach, and tested their methods on a movie recommendation application that took time, place, and companion contextual information into consideration. Here, recommendations are generated using only the ratings made in the same context of the target prediction. However, in fact, it is rarely the same context occurs in the future but instead the similar context. The disadvantage of that method is the increase of data sparsity.

Alternatively, Yap et al. (2007) exploit a different way of incorporating contextual information and tries to improve prediction accuracy using a Content Based (CB) approach. The authors model the context as additional descriptive features of the user and build a Bayesian Network to make a prediction. They increase the accuracy even with noisy and incomplete contextual information.

Hong and Eom (2009) conclude that the most intuitive method that can replace the inputting key code step is "pointing" remote to the device that the user intends to operate. They name this pointing based multi-device controlling method as Point and Control (PAC). PAC uses IR LEDs and IR image sensor to determine the target device. Each target device has unique IR LED information, and the universal remote control is equipped with an IR LED image sensor to read the target device's IR information. When a user points the remote to the target device, the remote retrieves the image of IR LED data, decides what device to control, and finally transmits the proper key code. Since key code inputting has changed to the pointing behaviour, a user can control several devices with ease, feeling much more comfortable.

2 Task Prediction and Recommendation

What is task prediction? What are the main methods for predicting tasks?

Task prediction can be called goal prediction or user's desires prediction.

2.1 Using History of User's Actions

Naeem and Bigham (2007) A Comparison of Two Hidden Markov Approaches to Task Identification in the Home Environment

2.2 Using Commonsense Reasoning

Lieberman and Espinosa (2007) described the use of commonsense knowledge base to predict tasks automatically.

2.3 Using Case-Based Reasoning

Ni et al. (2009)...

2.4 Using Sensor Data Bouarfa et al. (2010)

It presents a Markov-based approach for inferring high-level tasks from a set of low-level sensor data.

3 Media Recommendation

3.1 xPod Dornbush et al. (2005)

xPod keeps track of the music a user is listening along with their mood and activities, and uses machine-learning algorithms for recommending music based on the user's current activity.

3.2 CoMeR Yu et al. (2006)

The CoMeR system uses a hybrid approach comprising a Bayesian classifier and a rule based method to recommend media on mobile phones. The Naive Bayes classifier is to determine an item's relevance to the situation context and the rule based scheme is to check the presentation suitability of a media item against device-capability context.

3.3 SCAMREF Zhang and Yu (2007)

Context is classified into three categories: **Preference Context**, the context about user's taste or interests for media content, e.g. user requirements, user preference; **Situation Context**, the context about a user's spatio-temporal and social situation, e.g. location, time; and **Capability Context**, the context of physical running infrastructure, e.g. terminal capability, network condition.

They define user preference, terminal capability, location, time, etc., as context dimensions, and define modality, format, frame rate, frame size, score (similar to rating), etc., as QoS (Quality of Service) dimensions, which constitute the recommendation output.

Let $CD_1, CD_2, \ldots, CD_{N-1}$ be context dimensions, $QD_1, QD_2, \ldots, QD_M$ be output QoS dimensions, the recommendation model is defined as:

$$R : MediaItem \times CD_1 \times CD_2 \times \ldots \times CD_{N-1} \rightarrow QD_1 \times QD_2 \times \ldots \times QD_M.$$

The recommendation process consists of four steps:

1. They model both the media item and preference context as vectors. The cosine value of the angle between the two vectors is adopted as similarity measure between media item and preference context. The larger the similarity is, the more relevant between the media item and preference context.

2. They group the values of each situation context dimension into classes. For example, a user's home location can be divided into three classes: Living room, Bed room, and Dining room; social activities into four classes: At party, At date, Accompanying with parents, and Alone. They evaluate the probability of a media item belonging to a class of a context dimension or a combined situation context, e.g. how much probability of the movie *Gone With the Wind* is viewed by the user in *Bed room*, $P(\textit{Bed room}|\textit{Gone With the Wind})$. Suppose $C_1, C_2, \ldots, C_j, \ldots, C_k$ are k classes of situation context considered, the probability of media item $\overrightarrow{x}$ belonging to class C_j, that is, $P(C_j|\overrightarrow{x})$, can be calculated through statistical analysis of user viewing history. Given a class C_j, only the media items that have a high degree of $P(C_j|\overrightarrow{x})$ would be recommended.

3. The modality, format, frame rate, frame size, etc., of the recommended item must satisfy the capability context. They use rule-base approach to infer appropriate form from capability context.

4. The recommendation output consists of two parts: appropriate form and score. The score is composed of the similarity between a media item and the preference context and the probability of the media item belonging to the situation context $P(C_j|\overrightarrow{x})$. They use a weighted linear combination of these two sub-scores to calculate the overall score as:

$$Score = W_p * Similarity + W_s * P(C_j|\overrightarrow{x}),$$

where W_p and W_s are weighting factors reflecting the relative importance of preference context and situation context.

Although the evaluation of this approach is rubbish but the method may be reasonable.

4 Context-Aware Information/Content Provisioning

4.1 Contextually Aware Information DeliveryMillard et al. (2005)

The authors adopt a semantic model for context sensitive message delivery. They model task, domain, location and devices using semantic language. The use of semantic based language gives their system inferencing capability, which is useful to understand the user's task context in a logical manner.

4.2 Context-Aware Content Provisioning Yu et al. (2008)

The approach provides the right educational content in the right form to the right student, based on a variety of contexts and QoS requirements. They use knowledge-

based semantic recommendation to determine which content the user really wants and needs to learn. Then They apply fuzzy logic theory and dynamic QoS mapping to determine the appropriate presentation according to the user's QoS requirements and device/network capability.

They designed three ontologies: a context ontology, a learning content ontology, and a domain ontology. The context ontology depicts the content already mastered by the student, along with his or her learning goals, available learning time, location, learning style, and interests. It also describes the hardware/software characteristics and network condition of the student's client devices. The learning content ontology defines educational content properties as well as the relationships between them. The relation *hasPrerequisite* describes content dependency information–that is, content required for study before learning the target content. The domain ontology is to integrate existing consensus domain ontologies such as computer science, mathematics, and chemistry. The domain ontologies are organised as a hierarchy to reflect the topic classification.

The content recommendation procedure consists of four steps:

Calculating Semantic Relevance Rank content according to how much it satisfies the student's context. The semantic relevance between the student's goal and content is the ranking criteria. Semantic relevance is calculated via the following steps:

1. Map the student's learning goal to the domain ontology.
2. Locate the content's subject in the domain ontology.
3. Estimate the conceptual proximity between the mapped element and the content's subject node. The conceptual proximity $S(e_1, e_2)$ is defined according to the following rules (e_1 and e_2 are two elements in the hierarchical domain ontology):
   Rule 1: The conceptual proximity is always a positive number.
   Rule 2: The conceptual proximity has the property of symmetry–that is, $S(e_1, e_2) = S(e_2, e_1)$.
   Rule 3: If e_1 is the same as e_2, then $S(e_1, e_2) = Dep(e_1)/M$. M denotes the total depth of the domain ontology hierarchy; $Dep(e)$ is the depth of node e in the hierarchy (the root node always has the least depth, say, 1).
   Rule 4: If e_1 is the ancestor or descendant node of e_2, then $S(e_1, e_2) = Dep(e)/M$, where

$$e = \begin{cases} e_1 & \text{if } e_1 \text{ is the ancestor node of } e_2, \\ e_2 & \text{if } e_1 \text{ is the descendant node of } e_2. \end{cases}$$

   Rule 5: If e_1 is different from e_2 and there is no ancestor/descendant relationship between them, then $S(e_1, e_2) = Dep(LCA(e_1, e_2))/M$. $LCA(x, y)$ means the least common ancestor node for nodes x and y.
   Figure 1 shows the computer science domain ontology.
   $M = 5$;
   $LCA(MISD, SISD) = SingleDataStreamArchitecture$;
   $Dep(LCA(MISD, SISD)) = 4$; hence,
   $S(MISD, SISD) = Dep(LCA(MISD, SISD)) = 4/5 = 0.8$.
   It is intuitive that two subjects with more detailed contents and closer ancestors are more relevant to each other–for example, two subjects under

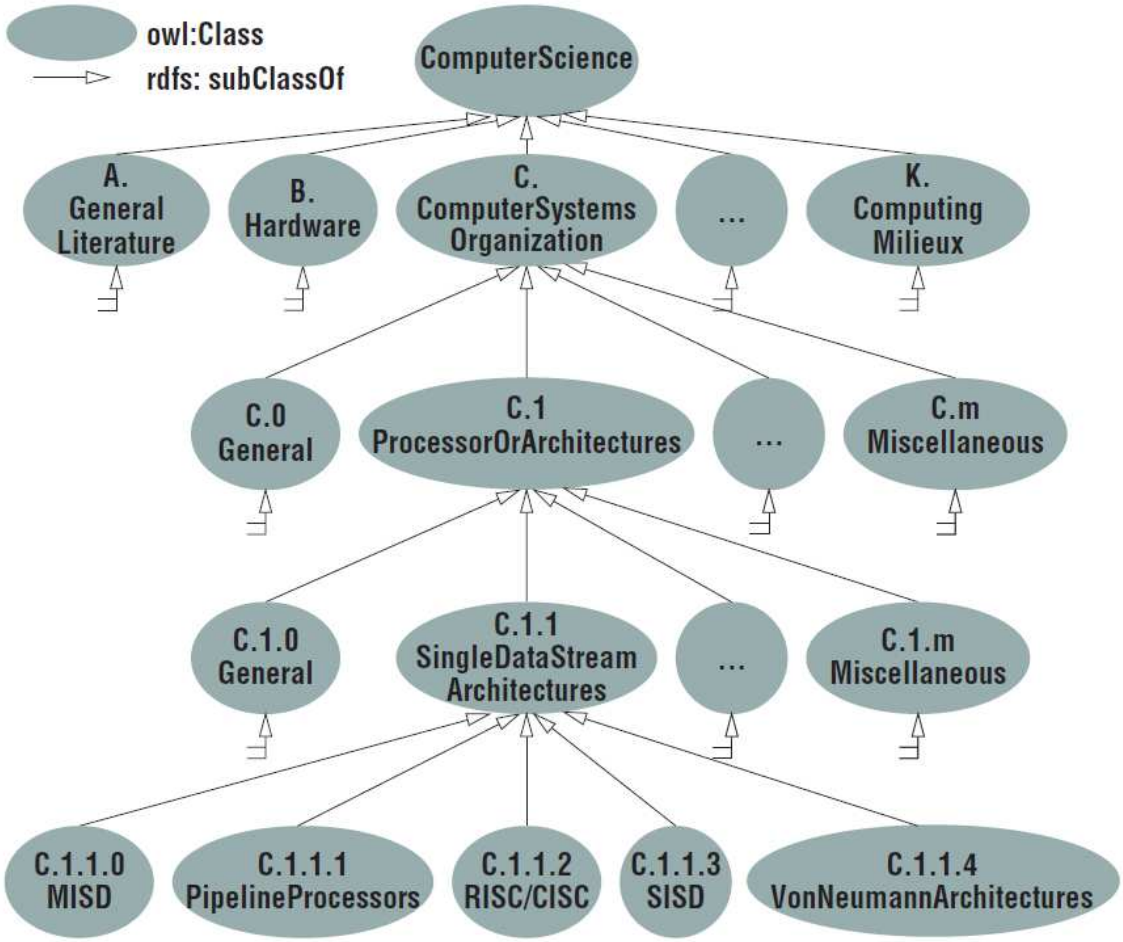

Figure 1: Computer Science Domain Ontology

"SingleDataStreamArchitecture" are known to be more relevant than two subjects under "ProcessorOrArchitecture".

Refined Recommendations Students can refine recommendation results according to the specialty or difficulty of contents.

Specialty If the recommendation contains few items and the student wants more generalised content, the system can provide all contents whose subject is one level higher than the LCA in the hierarchy. Similarly, if the recommendation includes many items and the student wants more specialised ones, the system can return those contents whose subject is one level lower than the LCA in the hierarchy.

Difficulty Students can refine the recommendation by choosing easier or more difficult contents through the *hasDifficulty* property applying to each content. Each content segment is assigned a difficulty level when authored, such as "very easy", "easy", "medium", "difficult", and "very difficult".

Generating Learning Paths Learning paths are to guide the learning process and suggest prerequisites that a student must complete before tackling the target content. When the student selects an item from the recommendation list, the system generates a learning path that connects prerequisite contents with the target content. It does this by recursively adding prerequisite content until the path reaches the content that has no prerequisites, and then it prunes the path based on the student's prior knowledge. The *hasPrerequisite* relation of a particular content provides the prerequisite course information.

Augmenting Recommendations Recommendation Augmentation is references to examples, exercises, quizzes, and examinations related to the main course the student is studying. It does this by aggregating the course contents through "*hasExample*", "*hasExercise*", "*hasQuiz*", and "*hasExamination*".

5 Service/Application Recommendation

5.1 Domain-, place-, and generic task-based methods Fukazawa et al. (2006); Naganuma and Kurakake (2005)

Tasks are categorised based on domain ontology, place ontology, and generic task ontology. Generic tasks are actions such as watch, view, drink, and so on. The users needs to provide information about where they currently are (e.g. at home), what the action (abstract task) they want to do (e.g. watch), what the object on which they will action (e.g. movie), and where they want the task happen (e.g. theatre). For example, "I am at home, I want to watch movie at a theatre".

The approaches requires that domains and generic tasks are pre-defined. Moreover, the selected tasks is assumed to be feasible. The users need to explicitly express their intents. The approach does not resolve with the issue of task feasibility and the user's situations.

5.2 Context-aware service discoveryMokhtar et al. (2005)

Matching context requirements of user tasks against context requirements of services. Semantic-aware service discovery is based on the matching algorithm proposed by Paolucci et al. (2002).

5.3 Task-Oriented Navigation of Services Sasajima et al. (2007)

In the task-oriented service navigator, the users seek for services by specifying a task they are involved, for example, "Move to station X", "Draw cash to buy a ticket", "Get on the next bus". The services which are associated which a task are offered to the users. Tasks are organised in a task ontology.

5.4 Gain-based Selection of Media Services Hossain et al. (2008)

Gain refers to the extent a media service is satisfying to a user in a particular context. The gain is computed by adopting user's context, profile, interaction history, and the reputation of a service. The computed gain is used in conjunction with the cost of using a service (e.g., subscription and energy consumption cost) to derive the service selection mechanism. A combination of greedy and dynamic programming based solution is adopted to obtain a set of services that would maximise the user's overall gain in the ambient environment by minimising the cost constraint.

The objective is to dynamically compute the gain from the media services in different contexts and to obtain a subset of services such that the overall gain of a user is maximised subject to the total cost constraint specified by the user.

User's context A particular context can be defined in terms of the user's location (where), the time of presence (when), the current activity of the user (what), the companion of the user (with whom) and the mood of the user (psychological status).

User's profile The user's profile stores some static user-specific information (e.g., sex, age), as well as their preferences for different media-related attributes (e.g. movie genre, actor, actress, preferred news types, sources, singer, and subject preference). The user's media-related preference attributes is a set of ⟨*media type, attribute, score*⟩ tuple, which is called AMP. The media type in AMP refers to the type of media, for example, movie, music, and news feed. The attribute refers to the metadata of the particular media type. The score refers how much a user likes the media service corresponding to the attribute's data item. For example, if ⟨*movie, genre, score*⟩ refers to a movie attribute that has two data items as ⟨*movie, action*, 70%⟩ and ⟨*movie, comedy,* 30%⟩, this reflects the fact that the user likes action movie more than that of comedy movie. These preferences can be either explicitly provided by the user or implicitly collected by the system. During the system initialisation phase, the user may choose to provide few entries of these preference attributes while the system can later use their interaction history to automatically update the initial scores provided by the user.

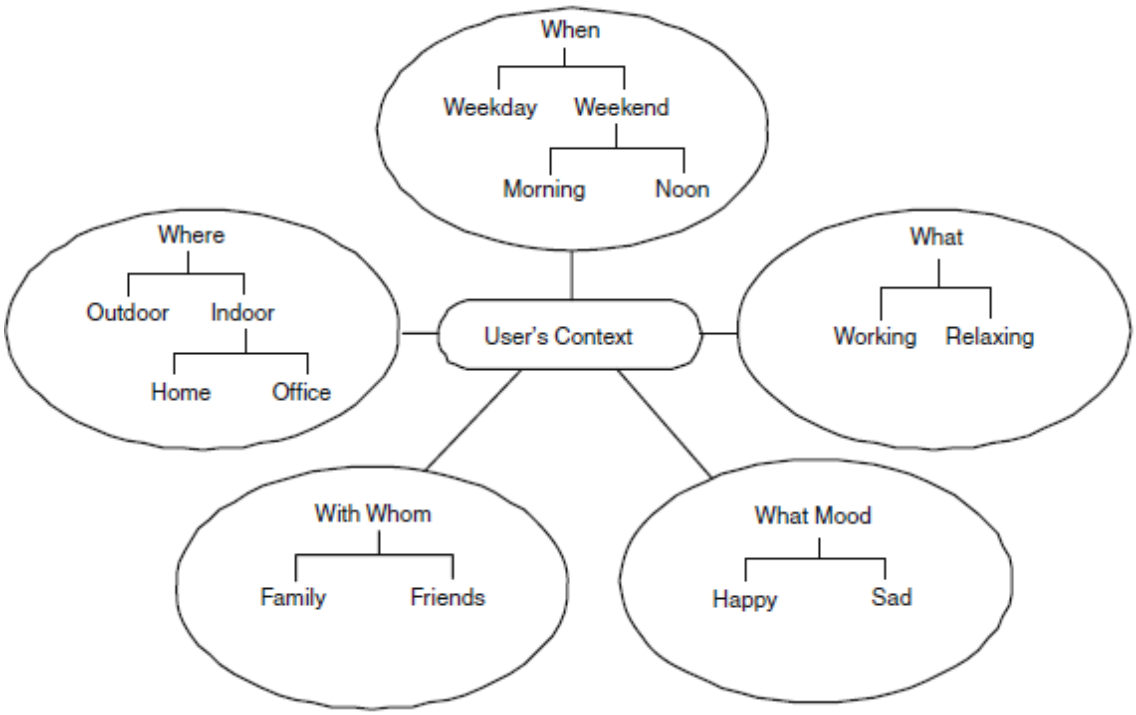

Figure 2: The different context parameters

Interaction history It is used to update the scores of the data items for different attributes in the AMP. Additionally, it is used to obtain different patterns of service usage. For example, frequent service sets can be obtained from the historical data Agrawal et al. (1993). The frequent service sets may provide the recurrent patterns of service usage information where two or more services co-occur together frequently.

Media reputation It often refers to how good or bad a service is in terms of content, delivery and other factors.

But finally, this paper is rubbish.

5.5 Location-Aware Service Selection Mechanism Loke et al. (2005)

In this approach, the geographical area of a target environment is divided among several service domains, where a set of services can be bounded with a service domain, such as specific library services while within a library. Service domains can be overlapping. The service selection mechanism is based on considering similarity, precedence, and restrictions among the services and on defining some aggregation rules.

5.6 MoBe Coppola et al. (2005)

MoBe allows the most relevant applications are selected by matching context and application descriptors. The applications, called MoBeLets, reside on the MoBe MoBeLet Server. Each MoBeLet is described by a MoBeLet descriptor that holds information

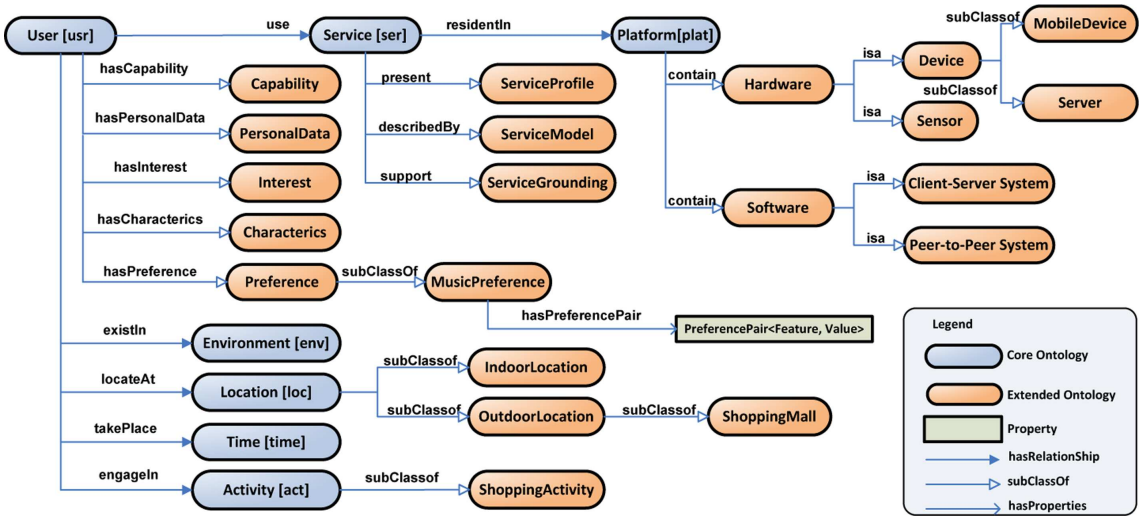

Figure 3: A partial context ontology

related to the application (e.g., the type of task carried out, information about the minimal CPU/memory requirements, the kind of needed peripherals/communication media).

5.7 Spontaneous Service Provision (Qin et al., 2008)

The authors are based on the similarity degrees between the current user profiles and situation contexts with services' contexts to rank services. For computing similarity, they use Pearson's correlation coefficient.

Context Modelling The authors build an ontology-based context model using Resource Description Framework and Web Ontology Language. In the model, the context ontology is divided into two sets: core context ontology for general conceptual entities in smart environment and extended context ontology for domain-specific environment. The core context ontology investigate seven basic concepts of *User, Location, Time, Activity, Service, Environment, and Platform*. Figure 3 shows a partial context ontology.

They define context (including user profile, situation context, and service) as a n-dimension vector: $C = (c_1, c_2, \ldots, c_n)$, where $c_i, (i \in 1..n)$ is quantified as a context attribute (e.g., Activity, Location) ranging from -1 to 1. The similarity of context C_1 and C_2 is defined as,

$$Similarity(C_1, C_2) = \frac{C1 \cdot C2}{||C1|| \times ||C2||} = \frac{\sum_{i=1}^{n} \alpha_i \beta_i}{\sqrt{\sum_{i=1}^{n} \alpha^2 \sum_{i=1}^{n} \beta^2}},$$

where $C_1 = (\alpha_i), C_2 = (\beta_i), (i \in 1..n)$.

Service Description The service description includes a important set of *Dependency* attributes. Each *Dependency* has two properties *Feature* and *Value*. The former is to describe a context attribute, and the latter one is to measure the semantics relevance degree of that context attribute to the service.

This work has not provided an evaluation the system usability.

Input service vector						Output
Distance	Necessity	Popularity	Quality	Service load	Use rate	Contextual distance
Close	High	Middle	High	Low	Middle	Very close
Far	Low	Middle	Low	High	Middle	Very far
Middle	High	Middle	Middle	Middle	High	Close
Middle	Middle	Low	Low	High	Middle	Far
Middle	High	High	Middle	Middle	Middle	Close
Far	Low	High	High	Middle	Middle	Far
Middle	High	Middle	High	Low	High	Close
Close	Low	Middle	Low	Middle	Low	Far

Table 1: Fuzzy-variable-based rules

5.8 CASUP Hong et al. (2009)

CASUP can provide users with personalised services using context history. Each instance in context history database consists of user' profile, high-level context and the service selected by the user in the given context (e.g., Smith, Male, 25, Dinner Context, Family restaurant Service). Context history is used to extract user preferences using classification such as decision tree algorithm. Association rules representing the relationship among the services or service sequences are extracted for recommending the next service. They apply the Apriori algorithm (Agrawal and Srikant, 1994) to locate association rules.

5.9 Personalised Service Discovery (Park et al., 2009)

This aims to provide mobile users only services that fit their preferences and are appropriate to their context. Their framework was based on Virtual Personal Space (VPS)–a virtual administrative domain of services managed for each user.

The framework operates as follows. Services automatically send their advertisements to an appropriate directory (how to discover an appropriate directory?). The advertisements carry services' contextual attributes such as name, category, physical position, popularity, quality, service load, and required services. Directories can propagate queries to adjacent directories if they don't have appropriate services for the queries.

When a user moves into a new place, a service crawler automatically finds available directories and retrieves service advertisements. Then the services that suit the user's context and preference are added into the VPS. The services that do not suit the user's context and preference are dropped from the VPS. When a user inputs a service query, the system first searches the VPS. If it fails to find any appropriate services, it sends the query to a local service directory.

To find personalised services, they employed the adaptive-network-based fuzzy inference system (Jang, 1993). Each service is represented by a service vector that includes location, distance, necessity, popularity, quality, service load, and user rate. These parameters are directly obtained from the service advertisement or calculated using service information and user context. Necessity is the number of services in the VPS that require the service. Use rate represents how often the user employs this class of services.

The service vector is used to calculate a value denoting the contextual distance, which describes the service's contextual proximity to the user. The system uses a set of fuzzy rules, as Table 1 shows, and makes a decision by applying these rules on a service vector.

To accommodate the differences of users' preferences, the system uses feedbacks to reflect user preferences. When the user employs a suggested service, the feedback is *immediate*; if the service is already included in the VPS, the feedback is *positive*; if the service isn't included in the VPS, the feedback is *negative*; if the service is not used, the feedback is *negative*. The learning affects each fuzzy variable's membership function. The system can learn the fuzzy meanings as it repeatedly performs personalisation and receives feedbacks. Thus, the system starts with a set of general rules defined by system developers, but it gradually reflects the user's personal preferences.

To evaluate, they compared their system to other management models such as the location model, the quality model, the least recently used model, the rule-based model (which manages services that satisfy the rules). They compared their hit ratios. In the their model, 70% of the discovery queries found an appropriate service, but other models had only 30% to 50% hit ratios.

5.10 Situation-Aware Applications Recommendation on Mobile DevicesCheng et al. (2008)

Cheng et al. (2008) use unsupervised learning (Minimum-Sum Squared Residue Co-clustering (Cho et al., 2004)) to extract patterns from user usage history. A pattern contains a situation and applications frequently performed in the situation. The system periodically senses the user's current situation, finds similar situations it has learned from the history, ranks the applications typically performed in the similar situations, and recommends applications by their ranks. Situation similarities are identified by computing the Euclidean distances between the current situation and the situation part of every co-cluster centroid. The application part of the centroids of the identified similar situations are then ranked for recommendation.

This approach using the unsupervised learning technique, specifically co-clustering, to derive latent situation-based patterns from usage logs of user interactions with the device and environments and use the patterns for task and communication mode recommendations.

Some advantages

- No need for predefined situations;
- No need for user-defined profiles;
- Do not require users to proactively train the system;
- Able to adapt to user habit changes;
- Accounts for many context variables.

Definition of "situation" Situation is a set of relevant context values that are frequently associated with a pattern of user usages of a mobile device Cheng et al. (2008).

Patterns of usage Patterns of usage are patterns from user usage history. The history contains interactions between user and his/her mobile device along with the context in which the interactions occurred. A pattern contains a latent situation and tasks frequently performed in the situation.

The operation of the system

1. Periodically recognises the user's current situation;
2. Finds similar situations it has learned from the history;
3. Ranks the tasks typically performed in the similar situations;
4. Recommends matching tasks by their ranks when the user asks for recommendations.

Some limitations

- Requires the user usage history data. Hence, for the first time of use, the system may behave inappropriately because it has not enough history data for recognising situations.
- Memory restriction may limit the usability of the system which requires a large of history data.
- Do not verify the feasibility of the tasks.

6 Task Recommendation

6.1 Why do we need task recommendation systems?

- Help users to find services which they may not know in advances;
- Help users to carry out their tasks effectively, automatically, or semi-automatically thanks to support from the computation-embedded surroundings;
- Suggest users tasks they are intent to do and help them to do the selected tasks.

6.2 Context-dependent task discovery Ni et al. (2006)

The authors introduces a concept of active task which is exactly determined by the current context. A task is described by a 5-tuple

$$\langle Task{-}ID, Task{-}Name, Condition, Priority, Task{-}Contract \rangle$$

In order to discover an active task in a particular context, an active task discovery mechanism is proposed. The idea is to match the Conditions of individual tasks T with the current context values using Condition's similarity:

$$dis\Big(T(c), T'(c)\Big) = \sum_{j=1}^{n} w_j * dis\Big(v(c_j), v'(c_j)\Big)$$

where $c_j, j = \overrightarrow{1..n}$ is a context attribute of the Condition and $v(c_j)$ is its expected value while $v'(c_j)$ is its current value. w_j is the *attribute weight* of c_j where $\sum_{j=1}^{n} w_j = 1$. The attribute weights are explicitly specified in task descriptions. And

$$dis\Big(v(c_j), v'(c_j)\Big) = \frac{|v(c_j) - v'(c_j)|}{dom(c_j)}$$

where $dom(c_j)$ means the maximal difference of two values $v(c_j)$ and $v'(c_j)$.

The range of $dis\Big(T(c), T'(c)\Big)$ is $[0, 1]$, a value of zero means perfect match and 1 meaning complete mismatch.

Some limitations

- Fixedly assigning task priority may be unappropriate in PCEs because the priority of tasks is often dynamic time by time and user by user depending on user's intention. Moreover, user's intention may change over time depending on their situation.
- The work provides a method for negotiation between condition of individual task and context information. But they do not mention about discovery of concurrent tasks. In fact, there may be multiple tasks needed to be concurrently performed in a context in which they may sharing some resources.
- In reality, when the Condition's similarity is often not a zero value, how the system should behave to help the user instead of saying the system cannot do the tasks. Moreover, in the case that the Condition's similarity is ideally a zero value (e.g. perfectly matching), but the task does not meet the user's intention, how the system trade-off between the relevance and the possibility of tasks?

6.3 Homebird Rantapuska and Lähteenmäki (2008)

Homebird can discover features of other devices automatically and suggests to the user that certain tasks can be performed together with those devices. The set of available tasks can be triggered to change by arbitrary events–for example, a newly discovered network device. The logic of tasks is encapsulated in modules called plug-ins that can be written separately. Homebird automatically loads plug-ins found in the environment. When a task is selected, the control is handed back to the respective plug-in that can then show its own UI customised for that task. All the plug-ins use the UPnP protocol for communicating with other networked devices. the user study shows that users wanted to be able to customise which tasks appear.

6.4 Situation-Based Task Recommendation Luther et al. (2008); Weissenberg et al. (2006)

The system reasons about a user's current situation based on a predefined situation ontology. Tasks which are associated with the inferred situation (found in a pre-specified task ontology) are retrieved. Then, the corresponding services that may be helpful for the retrieved task are recommended.

This work introduces a situation-sensitive task navigation system which expose only those tasks that are relevant to user's inferred situation. To do so, situational reasoning, which applies classification-based inference to qualitative context elements, is integrated in to the system. High-level qualitative context elements are formulated in the Web Ontology Language (OWL).

The system operates as follows. The current situation is inferred from the current context information and the situation ontology (see an example in Figure 4). A list of abstract tasks inferred from this situation using ontology-based task categorisation is shown to the user. Now, the user can select their desired task, then a corresponding sub-task list is displayed. Repeatedly, in a final step, associated services can be invoked to carry out the selected task.

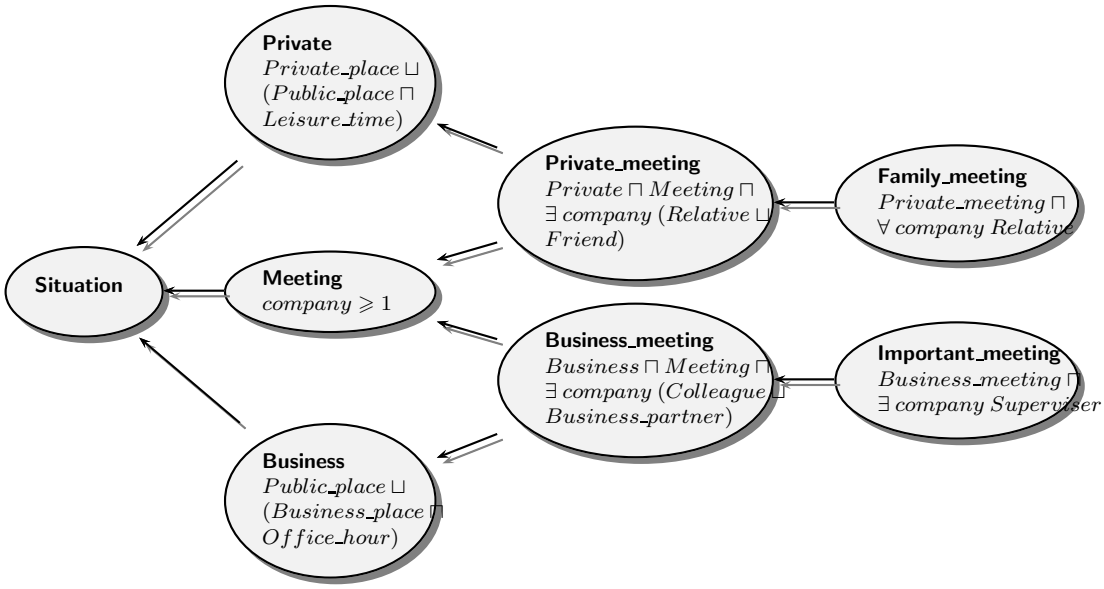

Figure 4: Situation ontology fragment

The task ontology is hence required. A part of the task ontology is shown in Figure 5. Tasks are categorised according to the high-level situation concepts such as '*Business_meeting*', defined within the situation ontology. The enabling context conditions are encoded as corresponding OWL-S service profiles.

Some limitations

- Requires that the situation ontology and the situation ontology-based task ontology are pre-defined. However, what defines each situation and what services are preferred in the inferred situation not only vary from user to user but also

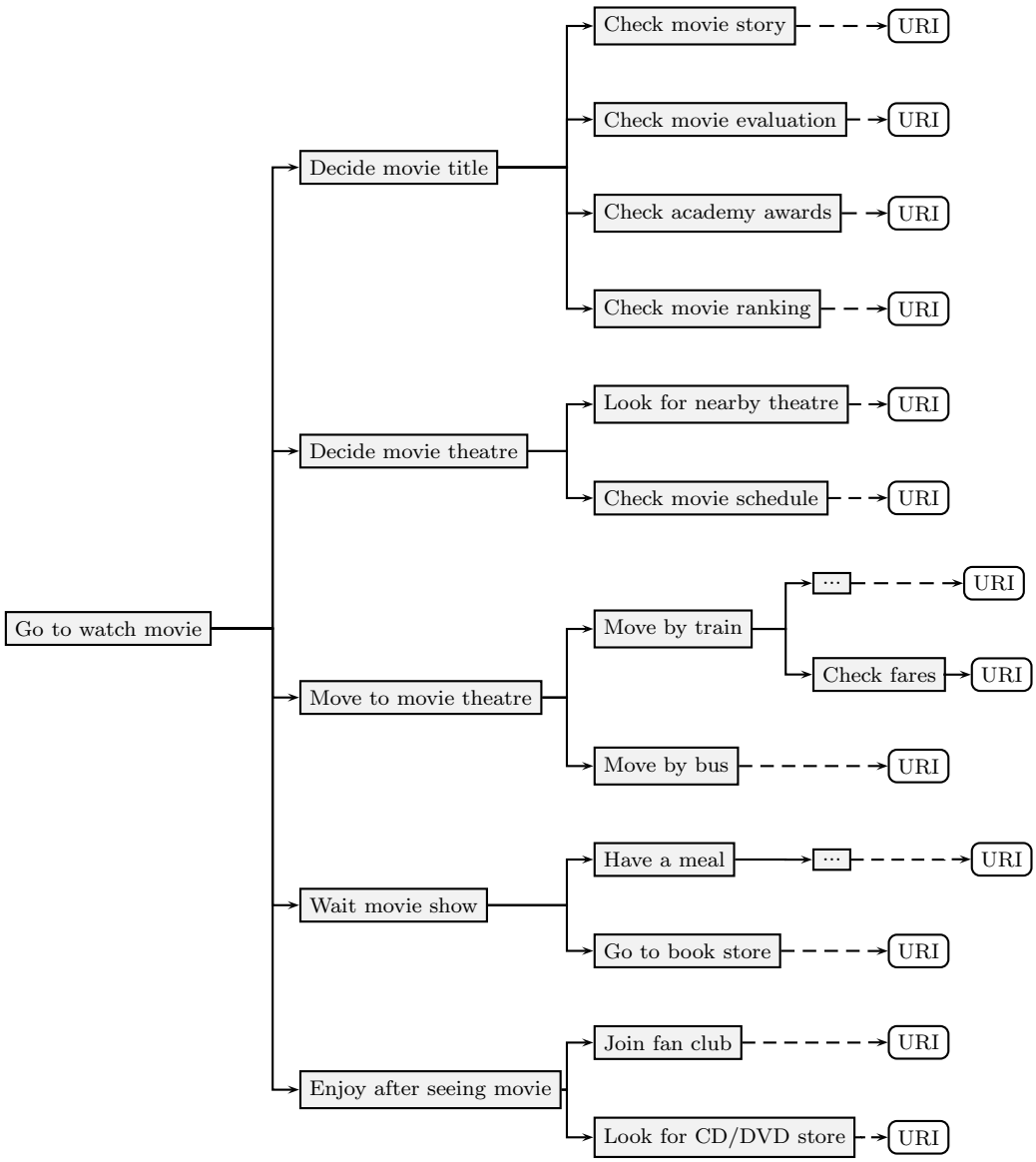

Figure 5: Task ontology fragment

change over time. Therefore, this assumptions are impractical for ordinary consumers Cheng et al. (2008).

- Do not verify the feasibility of the tasks. It may assume that all required services and resources are somehow available for the tasks to be completed. However, in an ever-changed and dynamic pervasive environments, this assumption is not suitable.

7 Activity Recommendation

Chen (2005) presents a context-aware collaborative filtering system which could recommend activities customised for a user for the given context (e.g., weather, location, and travelling companion(s), based on what other people like him/her have done in a similar context. To incorporate context into the recommendation process, the approach weights the current context of the active user against the context of each rating with

respect to their relevance in order to locate ratings given in a similar context. One major problem of this approach is the availability of ratings in comparable contexts. The sparseness of ratings is an issue in collaborative filtering in general and further aggravated when integrating context.

7.1 Personalised Daily-Life Activity Recommendation Wang et al. (2008)

By using a flexible concept hierarchy and a dynamic clustering method, the authors provide a recommendation service highly related to the users' context, based on the multidimensional recommendation model. Users can request for activity recommendations by providing their personal profile data and contextual information through access devices. Name, age, gender, single/married, location, and some registered information are used as users' original profile data. Users are dynamically clustered based on contextual information, before making activity recommendations to users. At the same time, users can rate the recommendations and help to modify the accuracy of recommendations.

The approach uses the multidimensional model (MD model) proposed by Adomavicius and Tuzhilin Adomavicius et al. (2005) to store the information related to user, activity, and context factors, where each factor can be represented as a concept hierarchy. The MD model extends the concept of data warehousing and OLAP application in databases. This approach uses time, location, weather, and companions as the contextual information dimensions, and the recommendation space is defined as:

$$S = \mathit{User} \times \mathit{Activity} \times \mathit{Time} \times \mathit{Location} \times \mathit{Weather} \times \mathit{Companion}$$

In the MD model, a dimension D_i is the Cartesian product of attributes and can be expressed as $D_i \subseteq A_{i1} \times A_{i2} \times \ldots \times A_{ij}$. Each attribute A_{ij} defines a set of attribute values of one particular dimension. For example, the *User* dimension is defined as: $\mathit{User} \subseteq \mathit{Name} \times \mathit{Age} \times \mathit{Gender} \times \mathit{IsMarried}$. Similarly, the *Location* dimension is defined as: $\mathit{Location} \subseteq \mathit{Country} \times \mathit{City} \times \mathit{Place}$. For each dimension, the attributes can be represented as a concept hierarchy which consists of several levels of concepts. The top-down view of a concept hierarchy is organised from generalisation to specialisation; i.e., the higher the layer, the more generalised the layer. Take *Time* for example, its concept hierarchy can be expressed as Figure 6.

Given dimensions: $D_1, D_2, \ldots, D_n$, ratings are the *Ratings* domain which represents the set of all possible rating values under the recommendation space $D_1 \times D_2 \times \ldots \times D_n$. The rating function R is defined as: $R : D_1 \times D_2 \times \ldots \times D_n \rightarrow \mathit{Ratings}$. Based on the recommendation space $\mathit{User} \times \mathit{Activity} \times \mathit{Time} \times \mathit{Location} \times \mathit{Weather} \times \mathit{Companion}$, the rating prediction function $R(u, a, t, l, w, c)$ specifies how much user u likes activity a, accompanied by c at location l and time t under weather w, where $u \in \mathit{User}, a \in \mathit{Activity}, t \in \mathit{Time}, l \in \mathit{Location}, w \in \mathit{Weather}$, and $c \in \mathit{Companion}$. The ratings are stored in a multidimensional cube and the recommendation problem is to select the maximum or top-N ratings of $R(u, a, t, l, w, c)$.

The computation of recommendations grows exponentially with the number of dimensions. The reduction-based approach can reduce the multidimensional recommendation space to the traditional two-dimensional recommendation space by fixing the values of

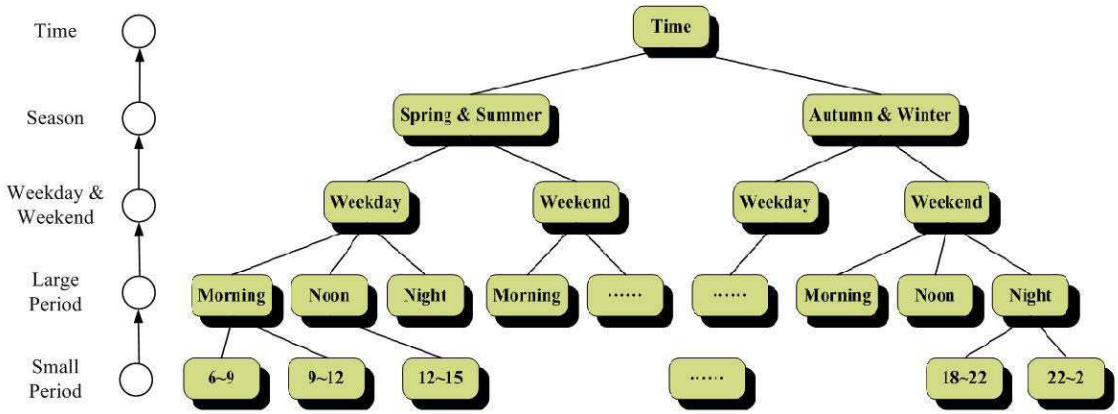

Figure 6: *Time* Concept Hierarchy

context dimensions, and improve the scalability problem Adomavicius et al. (2005). Assume that $R^{D}_{User \times Activity \times Time} : U \times A \times T \rightarrow Ratings$ is a three-dimensional rating estimation function supporting *Time* and D contains the user-specified rating records (*user, activity, time, rating*). It can be expressed as a two-dimensional rating estimation function:

$$\forall (u, a, t) \in U \times A \times T, R^{D}_{User \times Activity \times Time}(u, a, t) = R^{D[Time=t](User, Activity, Rating)}_{User \times Activity}(u, a),$$

where $D[Time = t](User, Activity, Rating)$ is a set of rating records by selecting *Time* dimension which has value t and keeping the values of *User* and *Activity* dimensions.

Another problem is rating estimation that $D[Time = t](User, Activity, Rating)$ may not contain enough ratings for recommendation computation. This approach uses the rating aggregations of time segment S_t that expresses the superset of the time t when insufficient ratings are found in a given time value t. The rating of $R(u, a, t)$ is expressed as:

$$R^{D}_{User \times Activity \times Time}(u, a, t) = R^{D[Time \in S_t](User, Activity, AGGR(Rating))}_{User \times Activity}(u, a),$$

where $AGGR(rating)$ is the rating aggregations of time segment S_t. For example, the rating prediction function for **weekend afternoon** might be presented as the formula:

$$R^{D}_{User \times Activity \times Time}(u, a, t) = R^{D[Time \in \mathbf{weekend}](User, Activity, AGGR(Rating))}_{User \times Activity}(u, a).$$

For evaluating the quality of the recommender system, they use predictive accuracy metrics to examine the prediction accuracy of recommendations. Predictive accuracy metrics are usually used to evaluate the system by comparing the recommender system's predicted ratings against the actual user ratings. Generally, Mean Absolute Error (MAE) is a frequently used measure for calculating the average absolute difference between a predicted rating and the actual rating. In addition, Normalised Mean Absolute Error (NMAE) represents the normalisation of MAE which can balance the range of rating values, and can be used to compare the prediction results from different data sets. According to related research, the predictive accuracy of a recommender system will be acceptable when the value of NMAE is below 18%. The approach uses the AllButOne method Cho et al. (2006) as the data set selection strategy.

8 Recommending Mobile Applications

The recommender system Woerndl et al. (2007) recommends mobile applications to users derived from what other users have installed and rated positively in a similar context (location, currently used type of device, etc.). When making a recommendation, the system retrieves the current user position, determines POIs in the vicinity and generates a recommendation based on this context information. The approach uses collaborative filtering to rank found items according to user ratings of applications in a second step. User ratings are collected implicitly by automatically recording when a user installs an application. The ratings are stored together with context information (time, location, used device etc.) to capture the situation when a rating was made.

9 Others

CityVoyager Takeuchi and Sugimoto (2009) can find and recommend shops that match each user's preferences. The procedure for finding shops that match user preferences is based on a place learning algorithm that can detect users' frequented shops. They use the unavailability in 5 minutes of GPS signals as evidence that the user has gone indoors.

Gas Stations Recommendation (Woerndl et al., 2009) uses a hybrid, multidimensional recommender system which takes driver preferences (user-specified), ratings of other users, the current location, and fuel level of a car into account. The approach first filters items based on preferences and context, and takes ratings of other users and additional information into account.

10 Best Recommendation for the whole group

10.1 Let's Browse Lieberman et al. (1999)

Let's Browse recommends web pages to a group of users who are browsing the web.

10.2 MusicFX McCarthy and Anagnost (2000)

MusicFX used in a fitness center to adjust the selection of background music to best suit the preferences of people working out at any given time. A special feature found in this system is that a group is composed by people who happen to be in the place at the same time. MusicFX uses explicit preferences of all participants to make a music selection that will be listen by everyone who is present. In this case, the group is composed by strangers rather than family members or friends.

10.3 Intrigue Ardissono et al. (2003)

Intrigue recommends attractions and itineraries by taking into account preferences of heterogeneous groups of tourists (such as families with children) and explains the

recommendations by addressing the requirements of the group members. Attractions are separately ranked by first partitioning a user group into a number of homogeneous subgroups with the same characteristics. Then each subgroup may fit one or more stereotypes and the subgroups are combined to obtain the overall preference, in terms of which attractions to visit for the whole group.

10.4 Travel Group Recommendation Lorenzi et al. (2008)

The recommender system performs the travel group recommendation task based on the formalism of distributed constraint optimisation problem.

11 Conclusion

References

Adomavicius, G., Sankaranarayanan, R., Sen, S. and Tuzhilin, A. (2005). Incorporating contextual information in recommender systems using a multidimensional approach, *ACM Trans. Inf. Syst.* **23**(1): 103–145.

Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, *IEEE Trans. on Knowl. and Data Eng.* **17**(6): 734–749.

Agrawal, R., Imieliński, T. and Swami, A. (1993). Mining association rules between sets of items in large databases, *SIGMOD '93: Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, ACM, New York, NY, USA, pp. 207–216.

Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases, *Proceedings of the 20th international conference on very large database*, pp. 115–121.

Ardissono, L., Goy, A., Petrone, G., Segnan, M. and Torasso, P. (2003). Intrigue: Personalized recommendation of tourist attractions for desktop and handset devices, *Applied Artificial Intelligence: Special Issue on Artificial Intelligence for Cultural Heritage and Digital Libraries* **17**(8-9): 687–714.

Bouarfa, L., Jonker, P. and Dankelman, J. (2010). Discovery of high-level tasks in the operating room, *Journal of Biomedical Informatics* .

Chen, A. (2005). Context-aware collaborative filtering system: Predicting the user's preference in the ubiquitous computing environment, *Location- and Context-Awareness* pp. 244–253.

Cheng, D., Song, H., Cho, H., Jeong, S., Kalasapur, S. and Messer, A. (2008). Mobile situation-aware task recommendation application, *NGMAST '08*.

Cho, H., Dhillon, I. S., Guan, Y. and Sra, S. (2004). Minimum sum-squared residue coclustering of gene expression data, *Proceedings of the Fourth SIAM International Conference on Data Mining (SDM)*, pp. 114–125.

Cho, S., Lee, M., Jang, C. and Choi, E. (2006). Multidimensional filtering approach based on contextual information, *ICHIT '06: Proceedings of the 2006 International Conference on Hybrid Information Technology*, IEEE Computer Society, Washington, DC, USA, pp. 497–504.

Coppola, P., Mea, V. D., Gaspero, L. D., Mizzaro, S., Scagnetto, I., Selva, A., Vassena, L. and Rizio, P. Z. (2005). Information filtering and retrieving of context-aware applications within the mobe framework, *International Workshop on Context-Based Information Retrieval (CIR-05), CEUR Workshop Proceedings*.

Dornbush, S., Fisher, K., McKay, K., Prikhodko, A. and Segall, Z. (2005). Xpod – a human activity and emotion aware mobile music player, *Mobile Technology, Applications and Systems, 2005 2nd International Conference on*, pp. 1–6.

Fischer, G. (2001). User modeling in human–computer interaction, *User Modeling and User-Adapted Interaction* **11**(1-2): 65–86.

Fukazawa, Y., Naganuma, T., Fujii, K. and Kurakake, S. (2006). Service navigation system enhanced by place- and task-based categorization, *MobiSys 2006*.

Hill, W., Stead, L., Rosenstein, M. and Furnas, G. (1995). Recommending and evaluating choices in a virtual community of use, *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 194–201.

Hong, J.-Y., Suh, E.-H., Kim, J. and Kim, S. (2009). Context-aware system for proactive personalized service based on context history, *Expert Syst. Appl.* **36**(4): 7448–7457.

Hong, S. S. and Eom, J. I. (2009). Point and control: The intuitive method to control multi-device with single remote control, *Proceedings of the 13th International Conference on Human-Computer Interaction. Part III*, Springer-Verlag, Berlin, Heidelberg, pp. 416–422.

Hossain, M. A., Atrey, P. K. and El Saddik, A. (2008). Gain-based selection of ambient media services in pervasive environments, *Mob. Netw. Appl.* **13**(6): 599–613.

Jang, J. (1993). Anfis: Adaptive-network-based fuzzy inference system, *IEEE Trans. Systems, Man, and Cybernetics* **23**(3): 665–685.

Lieberman, H. and Espinosa, J. (2007). A goal-oriented interface to consumer electronics using planning and commonsense reasoning, *Know.-Based Syst.* **20**(6): 592–606.

Lieberman, H., Van Dyke, N. W. and Vivacqua, A. S. (1999). Let's browse: a collaborative web browsing agent, *IUI '99: Proceedings of the 4th international conference on Intelligent user interfaces*, ACM, New York, NY, USA, pp. 65–68.

Loke, S. W., Krishnaswamy, S. and Naing, T. T. (2005). Service domains for ambient services: concept and experimentation, *Mob. Netw. Appl.* **10**(4): 395–404.

Lorenzi, F., Santos, F., Ferreira, Jr., P. R. and Bazzan, A. L. (2008). Optimising preferences within groups: A case study on travel recommendation, *SBIA '08: Proceedings of the 19th Brazilian Symposium on Artificial Intelligence*, Springer-Verlag, Berlin, Heidelberg, pp. 103–112.

Luther, M., Fukazawa, Y., Wagner, M. and Kurakake, S. (2008). Situational reasoning for task-oriented mobile service recommendation, *The Knowledge Engineering Review* **23**(1): 7–19.

McCarthy, J. E. and Anagnost, T. D. (2000). Musicfx: an arbiter of group preferences for computer supported collaborative workouts, *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, ACM, New York, NY, USA, p. 348.

Millard, I., Roure, D. D. and Shadbolt, N. (2005). Contextually aware information delivery in pervasive computing environments, *The international conference on location and context-awareness*, pp. 189–197.

Mokhtar, S. B., Georgantas, N. and Issarny, V. (2005). Ad hoc composition of user tasks in pervasive computing environments, *Software Composition* pp. 31–46.

Naeem, U. and Bigham, J. (2007). A comparison of two hidden markov approaches to task identification in the home environment, *Pervasive Computing and Applications, 2007. ICPCA 2007. 2nd International Conference on*, pp. 383–388.

Naganuma, T. and Kurakake, S. (2005). Task knowledge based retrieval for service relevant to mobile user's activity, *The Semantic Web – ISWC 2005*, pp. 959–973.

Ni, H., Zhou, X., Zhang, D. and Heng, N. (2006). Context-dependent task computing in pervasive environment, *Ubiquitous Computing Systems* pp. 119–128.

Ni, H., Zhou, X., Zhang, D., Miao, K. and Fu, Y. (2009). Towards a task supporting system with cbr approach in smart home, *ICOST '09: Proceedings of the 7th International Conference on Smart Homes and Health Telematics*, Springer-Verlag, Berlin, Heidelberg, pp. 141–149.

Owen, D. (1986). *User-Centered System Design, New Perspectives on Human-Computer Interaction*, Lawrence Erlbaum, USA, chapter Answers first, then questions, pp. 361–375.

Paolucci, M., Kawamura, T., Payne, T. and Sycara, K. (2002). Semantic matching of web services capabilities, *The Semantic Web – ISWC 2002* pp. 333–347.

Park, K.-L., Yoon, U. H. and Kim, S.-D. (2009). Personalized service discovery in ubiquitous computing environments, *IEEE Pervasive Computing* **8**(1): 58–65.

Qin, W., Zhang, D., Shi, Y. and Du, K. (2008). Combining user profiles and situation contexts for spontaneous service provision in smart assistive environments, *UIC '08: Proceedings of the 5th international conference on Ubiquitous Intelligence and Computing*, Springer-Verlag, Berlin, Heidelberg, pp. 187–200.

Rantapuska, O. and Lähteenmäki, M. (2008). Homebird–task-based user experience for home networks and smart spaces, *PERMID 2008*.

Sasajima, M., Kitamura, Y., Naganuma, T., Fujii, K., Kurakake, S. and Mizoguchi, R. (2007). Task ontology-based modeling framework for navigation of mobile internet services, *IMSA'07: IASTED European Conference on Proceedings of the IASTED European Conference*, ACTA Press, Anaheim, CA, USA, pp. 47–55.

Shardanand, U. and Maes, P. (1995). Social information filtering: algorithms for automating "word of mouth", *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 210–217.

Takeuchi, Y. and Sugimoto, M. (2009). A user-adaptive city guide system with an unobtrusive navigation interface, *Personal Ubiquitous Comput.* **13**(2): 119–132.

Wang, C.-Y., Wu, Y.-H. and Chou, S.-C. T. (2008). Toward a ubiquitous personalized daily-life activity recommendation service with contextual information: A services science perspective, *HICSS '08: Proceedings of the Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, IEEE Computer Society, Washington, DC, USA, p. 107.

Weissenberg, N., Gartmann, R. and Voisard, A. (2006). An ontology-based approach to personalized situation-aware mobile service supply, *Geoinformatica* **10**(1): 55–90.

Woerndl, W., Brocco, M. and Eigner, R. (2009). Context-aware recommender systems in mobile scenarios, *International Journal of Information Technology and Web Engineering* **4**(1): 67–85.

Woerndl, W., Schueller, C. and Wojtech, R. (2007). A hybrid recommender system for context-aware recommendations of mobile applications, *Data Engineering Workshop, 2007 IEEE 23rd International Conference on*, pp. 871–878.

Yap, G.-E., Tan, A.-H. and Pang, H.-H. (2007). Discovering and exploiting causal dependencies for robust mobile context-aware recommenders, *IEEE Trans. on Knowl. and Data Eng.* **19**(7): 977–992.

Yu, Z., Nakamura, Y., Zhang, D., Kajita, S. and Mase, K. (2008). Content provisioning for ubiquitous learning, *IEEE Pervasive Computing* **7**(4): 62–70.

Yu, Z., Zhou, X., Zhang, D., Chin, C.-Y., Wang, X. and Men, J. (2006). Supporting context-aware media recommendations for smart phones, *Pervasive Computing, IEEE* **5**(3): 68–75.

Zhang, D. and Yu, Z. (2007). Spontaneous and context-aware media recommendation in heterogeneous spaces, *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*, pp. 267–271.