Mobile Information Systems 10 (2014) 197–215
DOI 10.3233/MIS-130178
IOS Press

Context-aware mobile service adaptation via a Co-evolution eXtended Classifier System in mobile network environments

Shangguang Wang[a,*], Zibin Zheng[b], Zhengping Wu[c], Qibo Sun[a], Hua Zou[a] and Fangchun Yang[a]
[a]*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China*
[b]*Shenzhen Research Institute, The Chinese University of Hong Kong, Hong Kong, China*
[c]*Department of Computer Science and Engineering, University of Bridgeport, Bridgeport, CT, USA*

Abstract. With the popularity of mobile services, an effective context-aware mobile service adaptation is becoming more and more important for operators. In this paper, we propose a Co-evolution eXtended Classifier System (CXCS) to perform context-aware mobile service adaptation. Our key idea is to learn user context, match adaptation rule, and provide the best suitable mobile services for users. Different from previous adaptation schemes, our proposed CXCS can produce a new user's initial classifier population to quicken its converging speed. Moreover, it can make the current user to predict which service should be selected, corresponding to an uncovered context. We compare CXCS based on a common mobile service adaptation scenario with other five adaptation schemes. The results show the adaptation accuracy of CXCS is higher than 70% on average, and outperforms other schemes.

Keywords: Mobile service, context-aware, service adaptation, learning classifier system, eXtended Classifier System, Co-evolution

1. Introduction

With the development of mobile internet, mobile terminals[1] are networked via heterogeneous access technologies.[2] Obviously, these mobile terminals have entered almost every part of life. For instance, in UK, Mobile Squared[3] forecasts, by 2015, smartphones will number 63.83 million- approximately 100% population penetration. Despite the overall rise in smartphone penetration and data usage, however, text messaging still dominates mobile operator non-voice revenues worldwide.

Unfortunately, from the survey of Mobile Squared, a total of three-quarters of all operators surveyed thought that Instant Messaging (IM) tools[4] on smpartphones do pose a threat to traditional operator-based services around SMS and voice. Specifically, 63% of respondents agreed with the statement,

*Corresponding author: Shangguang Wang, State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Box 187, 10 No., Road Xitucheng, Beijing, China. E-mail: sguang.wang@gmail.com.

[1]Such as smartphones, PDAs, table PCs and so on.

[2]Such as Internet, WiFi, WiMAX, 3G and 4G.

[3]www.mavenir.com/RCS-survey-release.htm.

[4]Such as Skype, iMessage, Google Talk, Facebook Messages, WhatsApp.

while a further 17% strongly agreed. Hence, how to cope with more and more IM tools from Internet company, is becoming more and more urgent for the major operators groups.[5] We find if Service Delivery Platform (SDP) can adapt to end users' context and adaptively provide suitable mobile services, may be a good solution. The viewpoint is similar to context-sensitive SDP.[6] A context-sensitive SDP provides a user interface (UI) that exhibits some capability to be aware of the context and to react to changes of this context in a continuous way. As a result such a UI will be adapted to a person's devices, tasks, preferences, and abilities, thus improving people's satisfaction and performance compared to traditional SDP based on manually designed UIs.

The context-sensitive SDP is complex and enormous. We can not implement its whole function. In this paper, we only focus on mobile service[7] adaptation according to different end user (device) context such as Location,[8] Activity[9] Illumination[10] and Acceleration[11] by using Event-Condition-Action rule [1] of database system and machine learning theory. In our proposed adaptation scheme, we map each mobile service into an action and map each (end) user context (device) into a condition. The condition is a logical test that, if satisfied or evaluates to true, causes the action to be carried out. The action consists of updates or invocations on the local data or condition. Our scheme can perform the best action (mobile service) according to the given condition (user context) by using a Co-evolution eXtended Classifier System. In the paper, our proposed adaptation system is deployed in an Adaptation Server of SDP of mobile network.

Different traditional adaptation system, in mobile service adaptation process, different users may have different preferred adaptive actions [2]. For example, if the available bandwidth is narrowing, John may want to turn a video call into an audio call, while Sam may be less sensitive to the video distortion and maintain the video call. Hence, some previous schemes such as utility function-based method [3], work-flows scheme [4] and colored Petri-net scheme [5] cannot be used in SDP. The main reason is that these schemes rely on the off-line analysis of conditions, they can not adaptively provide suitable action (mobile service) according to different condition (user context). Hence, SDP need an effective context-aware adaptation to enhance the quality of the experience of mobile services and improve communication service adhesiveness. Fortunately, some research findings such as machine learning theory in the filed of artificial intelligence, may be able to support the adaptation problem. Then, some schemes from machine learning theory have been proposed to support the adaptation, such as the Granular Computing [6,7], Rough Set [8,9] and Decision Tree models [10,11]. These schemes can give adaptation rules automatically by monitoring user interaction, but we are disappointed to find that they cannot support context-aware online preference learning. Obviously, they cannot be immediately used as the mobile service adaption schemes in SDP. Fortunately, an Event-Condition-Action rule-based system with machine learning theory can perform the mobile service adaptation. For instance, The Learning Classifier System (LCS) can manipulate the rule to decide which mobile service (action) will take place in a given context (condition) [12].

The (LCS) [13], as an adaptive rule-based system, can build a rule set automatically. It is also a machine learning technique combining evolutionary computing and reinforcement learning. Wilson's

[5] Such as AI&T, Verizon, Deutsche Telekom, Mepro PCS, China Mobile and so on.
[6] http://www.serenoa-fp7.eu/.
[7] Such as SMS, MMS, Voice Mail, Voice Call and Video Call.
[8] Home, Road, Car, Company/Meeting.
[9] Walking, Eating, Running, Sleeping.
[10] Very Weak (AW), Weak (W), Middle (M), Strong (S) and Very Strong (VS).
[11] AW, W, M, S, VS.

eXtended Classifier System (XCS) [14] is a type of LCS, in which a classifier's fitness is determined by the measure of its prediction's accuracy. XCS, the most successful LCS, solved a number of well-known problems, such as classification, planning tasks, function approximation and general prediction [15]. However, XCS can not been applied to perform mobile service adaptation. The reasons are as follows. First, XCS needs a long time to train itself to a reasonable accuracy because of the deficiency of user interactions as new-users enter a mobile service adaptation system. The slow converging speed fail in adapting users' mobile service requirements quickly according to user context. Second, the reinforcement credit assignment mechanism of XCS requires many training records to cover the total learning space. However, when the training records are incomplete, XCS cannot gvie any classifiers to match the current context. The uncovered-context leads to inaccurate mobile service adaptation.

To address above weaknesses, we propose a novel LCS based on co-evolution XCS (called CXCS) to perform context-aware mobile service adaptation. In this paper, the user context can be mapped to the classifier conditions, and mobile services can be mapped to classifier actions. Then the generation of the adaptation rule will be transformed into the learning of the classifier population. Once the learning process is completed, it can provide each user with mobile service adaptation through the matching and competition of classifiers. The contributions of this paper are as follows:

1. We survey why some traditional adaptation scheme cannot be used in mobile service adaption. We also point out an Event-Condition-Action rule-based system with machine learning theory can be employed to perform context-aware mobile service adaption for SDP. Then we proposed a context-aware mobile service adaptation scheme via a Co-evolution eXtended Classifier System.
2. To lessen the computation time of the mobile service adaptation, we introduce other users' evolutionary information to "inject new blood" into the classifier population and propose a new-user initialization algorithm to quicken the convergence speed.
3. To improve the accuracy of the mobile service adaptation, we employ an uncovered-context algorithm and an action prediction algorithm to predict which mobile service should be selected when there is a failure in covering the user context.
4. We conducted an experiment based on a mobile service adaptation dataset. The user context contains Location, Activity, Illumination and Acceleration. The mobile services contain Mail, SMS, MMS, video and Audio. The experimental results show the adaptation accuracy of CXCS is better than other five mobile service adaptation schemes.

The rest of this paper is organized as follows. Section 1 reviews XCS. Section 2 describes the proposed CXCS, which contains new-user initialization and uncovered-context prediction. Section 3 describes the experiments conducted to evaluate our proposed scheme. Section 4 introduces related work, and Section 5 ends the paper.

2. XCS review

To sketch our proposed CXCS, we first review XCS. Figure 1 shows the framework of XCS. XCS is a rule-based system, in which each rule has a condition and a set of parameters: prediction, fitness, accuracy, experience and so on. The complete set of rules forms the Classifier Population. Based on the two interface modules, a Detector (Sensor) and an Effector, the procedure of XCS contains three phases, as follows:

– First, the detector receives an input from the environment. The rules whose conditions match the input data form a Match Set. In the Match Set, each rule has an action. An action selection mechanism is used to select an action where the same action forms the Action Set.

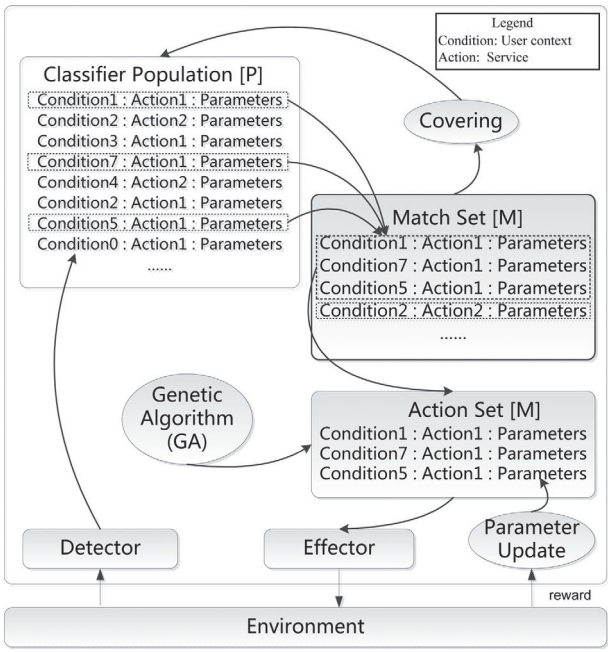

Fig. 1. XCS framework.

- Second, the Effector outputs the action to the environment, and a reward is received. Reinforcement learning is applied to the Action Set to update the parameters contained in the rules according to the rewards.
- Finally, the processes are included in a learning cycle. A genetic algorithm (GA) runs on the Action Set to guide the search for better rules, so XCS learns repeatedly to evolve the rule population. The learned rules are the solution to a given problem (context-aware mobile service adaptation).

For further illustration, based on Algorithm 1, we will introduce the running process of XCS by the following four components: finite classifier population, performance component, reinforcement component and discovery component, in the following sections.

Algorithm 1 XCS

Initiate the maximum iteration times N;
while $i++ < N$ **do**
  Get the environmental state from the Detector;
  All the classifiers of [P] that match the current state are selected to form the Match Set [M];
  Classify the classifier of [M] into several subgroups according to the classifier action, and prepare the prediction array;
  $xp = random[0,1]$;
  if $xp < P_{\exp}$ **then**
    Do the exploration scheme, choose a random action A;
    Update all the classifiers of [A], and apply GA on [A] with probability θ_{GA};
  else
    Do the exploitation scheme;
    Choose the action [A] with the largest fitness weighted prediction;
  end if
  Send [A] to the Effector to execute, and receive the reward from the environment;
end while

Table 1
Classifier parameters

Param	Usages
P	The payoff XCS receives after the execution of the chosen action.
ε	The prediction error of the classifier.
k	The accuracy of the classifier.
F	The fitness of the classifier which is the inverse of k.
exp	The matching time of the classifier.
num	The copy numbers of the classifier.
as	The average size of the Action Set.

2.1. Finite classifier population

The finite classifier population (Classifier Population [P]) is a finite classifier set. It represents the current knowledge. Each classifier is composed of three parts: the condition, the action and the classifier parameters.

A classifier is activated only if its condition matches the environment state. Once a classifier is activated, its action will be selected to execute. The classifier condition uses a ternary encoding scheme (i.e., it is represented by a string of characters using a ternary alphabet {0, 1, #}, where # is a "don't care" symbol), which means that the condition will match the state whether this position is 0 or 1. As shown in Table 1, there are 7 types of classifier parameters, which are P, ε, k, F, *exp*, *num*, as.

2.2. Performance component

All the classifiers of XCS that match the current context state will be selected to form the Match Set [M], and then one action will be selected as the final output of XCS. All the classifiers will be classified into subgroups according to their actions.

During the training phase, the ε-greedy action selection will be used. There will be two action selection schemes, i.e., the exploration and the exploitation, where their execution probabilities are $P_{\exp}$ and $1 - P_{\exp}$, respectively. In the exploration scheme, a random action will be selected; in the exploitation scheme, the action whose subgroup has the highest fitness weighted prediction will be selected. Then, the subgroup with the chosen action will be selected as the Action Set [A], and its action will be sent to the Effector to carry out.

2.3. Reinforcement component

The reinforcement component is also called the credit assignment component. It takes charge of the assignment of the environmental reward. Once the Effector has executed the selected action, XCS will receive the reward from the environment. This reward will be used to update all the classifiers of [A]. More information about the following parameter can be found in Table 1.

First, the experience time of the classifier, exp, will be increased as follows:

$$exp = exp + 1. \tag{1}$$

Then the prediction p and prediction error ε of the classifier will be calculated as follows:

$$p = \begin{cases} p + (P - p)/\exp, \exp < /\beta \\ p + \beta \cdot (P - p), \quad \textit{otherwise} \end{cases} \tag{2}$$

$$\varepsilon = \begin{cases} \varepsilon + (|P - p| - \varepsilon)/\exp, \exp < 1/\beta \\ \varepsilon + \beta \cdot (|P - p| - \varepsilon), \quad \textit{otherwise} \end{cases} \tag{3}$$

where $\beta(0 < \beta \leqslant 1)$ denotes the learning rate constant, which is used to control the updating speed.

Afterwards, the accuracy and relative accuracy of each classifier will updated for the calculation of its fitness value as follows:

$$k = \begin{cases} 1, & \varepsilon < \varepsilon_0 \\ \alpha \cdot (\varepsilon/\varepsilon_0)^{-u}, & \textit{otherwise} \end{cases} \tag{4}$$

where $\varepsilon_0(\varepsilon_0 > 0)$ is used to determine the threshold error when a classifier is considered to be accurate, $\alpha(0 < \alpha < 1)$ and $\upsilon(\upsilon > 0)$ are used to control the degree of the decline in accuracy, respectively.

Finally, once the classifier accuracy is updated, its fitness valued, F can be computed by the following:

$$F = F + \beta \cdot \left(k' - F\right) \tag{5}$$

with

$$k' = k \times n \Big/ \sum_{i=1}^{|[A]|} k_i \times n_i.$$

2.4. Discovery component

The discovery component of XCS includes the genetic algorithm (GA) based on the evolution of the classifier population, the cover mechanism and the deleting mechanism.

The GA process will only occur in the exploration scheme with a possibility of θ_{GA}. Once the GA is invoked, XCS will choose two classifiers from [A] as parents with the probability proportional to the fitness, and make a copy of them. These two cloned classifiers will mutate or crossover with a separate probability of p_x and p_m. The newly generated child or produced offspring will be injected back into [P] with the probability p_{sel} by the following:

$$p_{sel} = F_i \Big/ \sum_{j=1}^{|[A]|} F_j. \tag{6}$$

Whenever there are no classifiers in [P] to match the current state, the covering mechanism will be executed. The classifier condition will be created to match the current state, but with a probability P# of adding the # character. The classifier action will be randomly selected, and its fitness and prediction value will be initiated.

During the GA or covering process, the population will be expanded. However, if the size of [P] exceeds the population limit, the deleting mechanism will be activated. The classifiers with a low fitness will be deleted to keep the population size with a constant upper bound, and the deleting probability p_{del} will be calculated as follows:

$$p_{del} = (F_i/\textit{num}_i) \Big/ \sum_{j=1}^{|[P]|} (F_j/\textit{num}_j). \tag{7}$$

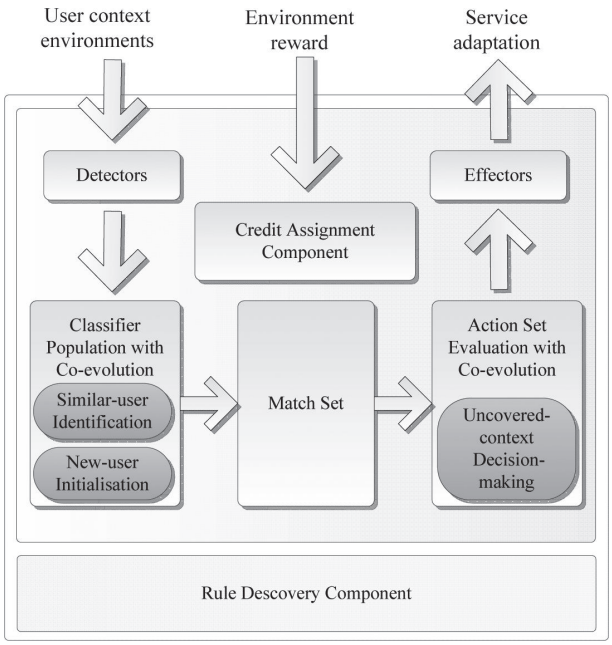

Fig. 2. Procedure of CXCS for context-aware mobile service adaptation.

3. Proposed CXCS for service adaptation

Because the high cost and low accuracy of XCS obstruct its application in mobile service adaptation systems, we propose CXCS to cope with these problems. The idea of CXCS is to add other users' evolutionary information into the current classifier population by introducing the co-evolution mechanism [16] into XCS and to utilize similar-users to help predict which mobile service is invoked for the current user.

As shown in Fig. 2, the proposed CXCS is a rule-based system, which contains three main components for context-aware mobile service adaptation, i.e., similar-user identification, new-user initialization and uncovered-context prediction. To illustrate, we first describe the concept of the co-evolution mechanism.

3.1. Co-evolution mechanism

The term co-evolution is derived from biology. It means "change in a trait of individuals of one population in response to a trait of individuals of a second population, followed by an evolutionary response of the second population to a change in the first". For example, the common influence of herbivore and plant will produce stronger and better generations of both over time.

There are two main classes of co-evolutionary approaches, competitive co-evolution and cooperative co-evolution. Just as the names imply, in competitive co-evolution, populations evolve simultaneously through their competition with each other, while cooperative co-evolution is implemented by the cooperation of the populations.

For competitive co-evolution, two main subclasses can be classified: competition and amensalism. The species of the former type inhibit each other, which means that the success of one species will result in the failure of the other species. In the latter type, the inhibition of species is single direction. There are three identified subclasses of cooperative co-evolution. In mutualism, every species benefits from the

improvements of the other species. In commensalism, only one of the species benefits, while the other is not affected. In parasitism, only one of the species benefits, and the other is harmed.

3.2. Similar-user identification

First, the cooperative co-evolution needs to identify all the similar-users. The reward for the adaptive action under a specified context state indicates the user's preference. Hence, by calculating the deviation of the users' preference shown for every contextual adaptive action, the similarity between users can be measured. The smaller the preference deviation is, the more similar the users are. The similarity between users includes two parts, the static part and the dynamic part. The measurement of the static similarity uses the static user profile information, such as gender, age, and occupation. The calculation of the dynamic similarity can use all the adaptation records for the user. Hence, the identification of similar-users can be divided into two types of cases. The first type is when a user uses the mobile service for the first time, and there are no training samples at all. This similarity measurement can only use the users' static user profile information. The greater the difference between the users' profiles is, the smaller their similarity. a user u_i's user profile can be defined as $UP_i = (Item_1, \ldots, Item_I)(1 \leqslant i \leqslant I)$. Another user u_j's user profile can be defined as $UP_j = (Item_1, \ldots, Item_I)(1 \leqslant j \leqslant I)$, and the similarity value, σ between u_i and u_j can be obtained as follows:

$$\sigma(u_i, u_j) = \sigma_{static}(u_i, u_j) = \sum_{k=1}^{I} \sqrt{\frac{\lambda_k^2}{I}} (1 \leqslant k \leqslant I) \tag{8}$$

with

$$\lambda_{\mathrm{k}} = \begin{cases} 1, \textit{if} \quad u_i.Item_k == u_j.Item_k. \\ 0, \textit{else} \end{cases}$$

where $\sigma_{static}(u_i, u_j)$ denotes the static similarity value.

The second type of circumstance is when a user has used the service for some time, in which case the similarity value calculation should use both the static user profile and the experience records, as follows:

$$\sigma(u_i, u_j) = R(t) \cdot \sigma_{static}(u_i, u_j) + (1 - R(t)) \cdot \sigma_{dynamic}(u_i, u_j) \tag{9}$$

with

$$R(t) = e^{-t},$$

where t denotes a time sample.

where $\sigma_{dynamic}(u_i, u_j)$ denotes the dynamic similarity value and it can be obtained according to the training rules of CXCS as show Fig. 3, $R(t)$ is added for the weight adjustment over time. The longer a user has used this service, the more familiar the user will be with the service, and the greater the weight of the dynamic part.

As shown in Fig. 3, every training rule is composed of context state C, adaptive action A_j and reward R, which represents that if, under a given context state, CXCS selected an adaptive action, it will receive a reward from the user. It can be defined as $S = \{(C, A_j, R)|1 \leqslant j \leqslant Q\}$ where Q denotes the number of adaptive action. The reward indicates the user's preference degree for this rule. Every context state C has n context attributes, $C = (C_1, ..., C_n)$. Each context attribute $\mathrm{C_i}$ can choose one value from

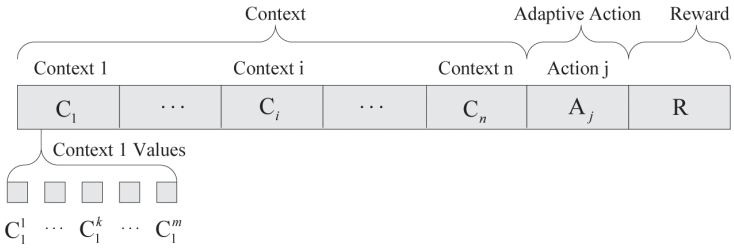

Fig. 3. Training rule.

$\{\mathrm{C_i^1}, \mathrm{C_i^2}, \ldots, \mathrm{C_i^{m-1}}, \mathrm{C_i^m}\}$ where m denotes the number of training sample. Then the user's preference for A_k under C_i^j can be obtained as follows:

$$\mathrm{P}(C_i^j, A_k) = \frac{1}{B}\sum_{m=1}^{M} f(C_i^j, A_k, S_m)R_m - \frac{1}{M}\sum_{m=1}^{M} R_m \tag{10}$$

with

$$B = \sum_{m=1}^{M} f(C_i^j, A_k, S_m).$$

where M is the training set size, Q is the number of adaptive actions. If in training sample S_m, the value of C_i is C_i^j and the adaptive action is A_k, the value of $f(C_i^j, A_k, S_m)$ will be 1; otherwise, the value of $f(C_i^j, A_k, S_m)$ should be 0.

Once $\mathrm{P}(C_i^j, A_k)$ is determined, the user's preference for the contextual adaptive action can be computed as follows:

$$P = \begin{pmatrix} P_1 \\ \cdots \\ p_Q \end{pmatrix} = \begin{pmatrix} \mathrm{P}(C_1^1, A_1) \ldots \mathrm{P}(C_1^{|C_1|}, A_1) \ldots \mathrm{P}(C_n^1, A_1) \ldots \mathrm{P}(C_n^{|C_n|}, A_1) \\ \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \\ \mathrm{P}(C_1^1, A_Q) \ldots \mathrm{P}(C_1^{|C_1|}, A_Q) \ldots \mathrm{P}(C_n^1, A_Q) \ldots \mathrm{P}(C_n^{|C_n|}, A_Q) \end{pmatrix}$$

Then, the dynamic user similarity can be obtained as follows:

$$\sigma_{dynamic}(u_1, u_2) = \sum_{i=1}^{Q} \sigma_{dynamic}(P_i^{u_1}, P_i^{u_2}) =$$

$$\sum_{i=1}^{Q}\sum_{j=1}^{n} \sqrt{\frac{1}{|C_j|}\sum_{k=1}^{|C_j|}\left(P_{u_1}\left(C_j^k, A_i\right) - P_{u_2}\left(C_j^k, A_i\right)\right)^2}. \tag{11}$$

3.3. New-user initialization

For the new-user, the administrators or developers can initialize some preference rules. These rules can help users configure their mobile service settings or parameters, however, it is difficult to perform mobile service adaptation. The reason is that the adaptation action preferences of each user are different. To solve this problem, we use some similar-users to initialize their preference rules and propose a new-user initialization algorithm as shown in Algorithm 2. Because there are no training samples at all, the

users' similarity measurements only use their static user profiles. Once the similar-users are identified, all these users will cooperate to initialize the new-user.

As shown in Algorithm 2, first, the similar-users within a threshold ς will be selected. Then, the algorithm obtains the context state union of all these similar-users. Finally, for each context state obtained, all these similar-users will cooperate to produce a new classifier and inject it into the new-user's classifier population. Each similar-user who has experienced the current state will vote for the predicted action. When a ratio of λ users have agreed on the maximum vote action, a new classifier with this voted action will be inserted into the current user's classifier population. Then the prediction value of the action will be set as the average of all the similar-users with the correct voted action.

Algorithm 2 New-user initialization

```
Input: {new-user u_c, candidate actions A, similar-user ratio threshold λ, similarity degree threshold ς}
similarList = Null;
for u_e in U do
    if calculate Similarity (u_c, u_e) < ς then
        similarList.insert(u_e);
    end if
end for
S = Null;
for u_e in similarList do
    for t in u_e.trainingSamples do
        if notS.hasState(t.state) then
            S.insert(t.state);
        end if
    end for
end for
for s in S do
    actionVoteList = [0  0];
    predictionList = [0.0  0.0];
    for u_e in similarList do
        if u_e.XCS.coverState(s) then
            action = u_e.XCS.getActionByState(s);
            prediction = u_e.XCS.getPredictionByState(s);
            actionVoteList[action] + = 1;
            predictionList[action] + = prediction;
        end if
    end for
    maxNum = max(actionVoteList);
    if maxNum>int(len(similarList) ×λ then
        for act In A do
            if actionVoteList[act] == maxNum then
                cl = new Classifier();
                cl.action = act;
                cl.prediction = predictionList [act]/ float(actionVoteList[act]);
                u_c.XCS.insertClassifier(cl);
            end if
        end for
    end if
end for
```

3.4. Uncovered-context prediction

When introducing the reinforcement credit assignment mechanism, the training process of XCS always takes a long time and often needs a large number of training samples to cover the entire problem

space. Because some context states have been experienced by other similar-users, it is highly possible that similar-users may make the same decisions. When these new decisions can be "borrowed" from similar-users, the learning speed might be greatly speeded up and the uncovered prediction accuracy might be increased. Hence, we propose an uncovered-context prediction algorithm for the CXCS, as shown in Algorithm 3.

By using Algorithm 3, when an action needs to be covered, but the covered classifier cannot be generated by the original mechanism, our CXCS produce the covered classifier by extracting the relevant information from all similar-users.

Algorithm 3 Uncover-context prediction algorithm

```
Input: {uncovered-context state s, similarity threshold ς}
Output: {generated cover classifier by similar-users}
actionVoteList = [0   0];
predictionList = [0.0   0.0];
for u_e in U do
    if calculateSimilarity(u_c, u_e) < ς then
        if u_e.XCS.coverState(s) then
            action = u_e.XCS.getActionByState(s);
            prediction = u_e.XCS.getPredictionByState(s);
            actionVoteList[action] + = 1;
            predictionList[action] + = prediction;
        end if
    end if
end for
for act In A do
    if actionVoteList[act] > 0 then
        cl = new Classifier();
        cl.action = act;
        cl.prediction = predictionList [act]/float(actionVoteList[act]);
        u_c.XCS.insertClassifier(cl);
    end if
end for
```

In the testing phase (or working phase) of CXCS, when an uncovered-context state is encountered, the final predicted action is voted on by all the similar-users through Algorithm 3. However, because the new classifiers generated by all possible decisions of all users flood the current user populations, the classifier fails to update. To overcome this problem, we propose an action prediction algorithm. As shown in Algorithm 4, we only take the action with the maximum number of votes as the output of the current user. If the maximum number of votes for multiple actions is the same, an action should be selected randomly as the input.

4. Experiments

In this section, we look into the performance of CXCS and compare it with other adaptation schemes base on a common mobile (communication) service adaptation scenario.[12] In experiments, we do not fulfil the whole scenario. We only focus on how to provide the following five mobile services adaptively, i.e., Mail, SMS, MMS, Audio and Video according the following four context, i.e., Location, Activity, Illumination and Acceleration.

[12] http://cordis.europa.eu/fp7/home_en.html.

Algorithm 4 Action prediction algorithm

```
Input: {uncovered-context state s, similarity threshold ς}
Output: {generated cover classifier by similar-users}
similarList = Null;
for u_e in U do
    if calculateSimilarity(u_c, u_e) < ς then
        if u_e.XCS.coverState(s) then
            action = u_e.XCS.getActionByState(s);
            prediction = u_e.XCS.getPredictionByState(s);
            actionVoteList[action] + = 1;
            predictionList[action] + = prediction;
        end if
    end if
end for
maxNum=max(actionVoteList)
actionList=Null;
for act In A do
    if actionVoteList[act]=maxNum then
        actionList.insert(act);
    end if
end for
Return random (actionList)
```

4.1. Experiment setup

As shown in Fig. 4, a scenario of mobile service adaptation is used to evaluate our proposed CXCS. The scenario revolves around a user called Julie. Generally, Julie gets up at 7 in the morning. One day, her boss (John) called her at 6. Because Julie set the adaptation system on the adaptation server before she went to sleep, the mobile service is automatically switched to Voice Mail. As soon as she gets up, Julie finds a unread Voice Mail informing her that she must go into town to the company for an unusual meeting at 9 in the morning. While driving, Julie receives a Video Call from her mother. Because she is driving at a high speed, the mobile service automatically changes to an Audio Call. After 15 minutes, she reaches the company and immediately attends the meeting. During the meeting, she receives a Video Call from her daughter. Because Julie is at a company meeting, the mobile service is changed to SMS.

Based on this scenario, we employ a mobile service adaptation system to validate the performance of CXCS. The mobile service uses four types of context attributes, *location, activity, illumination, and acceleration*, to identify the user's preferred mobile service. For example, the parameters "Illumination" and "Acceleration", which belong to [1,100], have 5 linguistic variables, Very Weak (AW), Weak (W), Middle (M), Strong (S) and Very Strong (VS). The adaptive action is the mobile service. The detailed setup is shown in Table 2. The membership function of the fuzzy linguistic variables of illumination and acceleration is the same, as depicted in Fig. 5.

In the experiments, we invite 10 users to collect our datasets.[13] First, they are required to provide some adaptation rules for the mobile service (MS), as shown in Table 3. Second, for each user, the adaptation rules are applied to a randomly generated set of the records using fuzzy inference. Finally, these datasets are encoded into binary format as the training samples. Some examples of these datasets are shown in Table 4. Note that the illumination and acceleration are both divided into 16 discrete values so that they can be easily encoded in the algorithm. For each user, the encoded datasets are split into two parts, i.e., the training set and the test set.

[13] http://sguangwang.com/Source%20code/XCSNew_Coevolution.zip.

Table 2
Experiment configuration

Parameter	Value
Location	{Home, Road, Company, Car}$\pi < P_{\exp}$
Activity	{Walking, Running, Eating, Sleeping}
Illumination	{VW, W, M, S, VS}
Acceleration	{VW, W, M, S, VS}
Communication service	{Mail, SMS, MMS, Audio, Video}

Table 3
Adaptation rule

Location	Activity	Illumination	Acceleration	Communication mode
Home	Sleeping	♯	♯	Mail
♯	♯	♯	♯	SMS
♯	Walking	Strong/Very Strong	♯	Audio
...	...	...	...	...

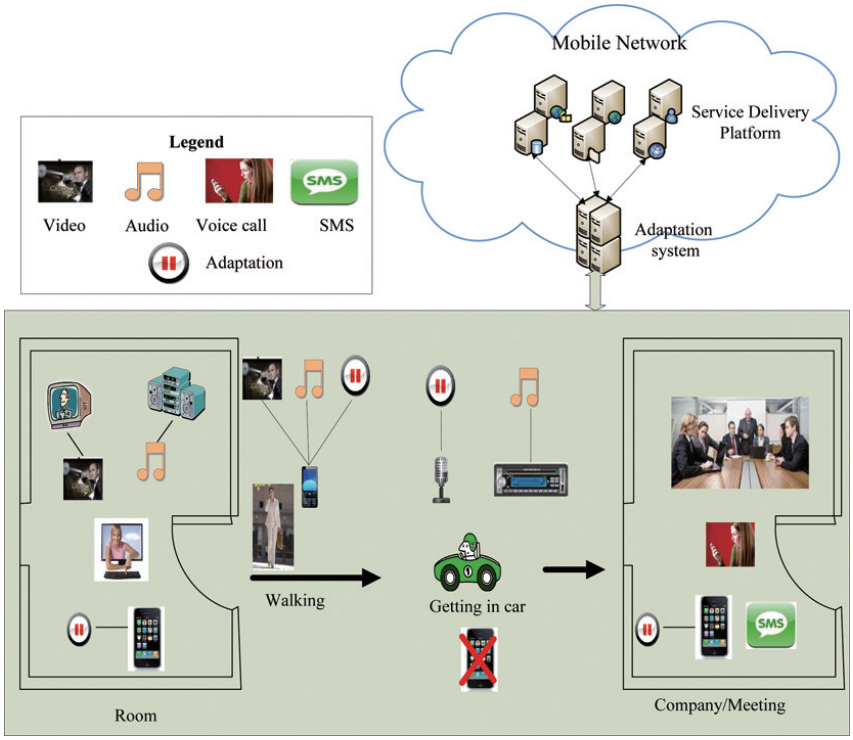

Fig. 4. Service adaptation scenario.

In experiments, all parameter are set prudently. The aim is to find a good tradeoff among all parameters, obtain the best performance of all schemes, and evaluate them objectively. Based on lots of experimental results, the XCS and CXCS configuration parameters are set with a tradeoff as follows: *iterationNum* $=$ 10000, *popNum* $=$ 5000, $\alpha = 0.1$, $\beta = 0.2, \varepsilon_0 = 0.25, \mathrm{p}_0{=}10.0, \mathrm{fitness}_0{=}0.01$, $\theta_{\mathrm{GA}}{=}25$, $\mathrm{p}_x{=}0.8$, $\mathrm{p}_m{=}0.04$, $\theta_{\mathrm{sub}}{=}\theta_{\mathrm{exp}}{=}\theta_{\mathrm{del}}{=}20$. The payoff is set to 1000 if the correct action is taken; otherwise, it is 0.

Table 4
Experimental dataset

No.	Location	Activity	Illumination	Acceleration	Communication mode
1	00	01	1001	0001	0
2	01	11	0000	0101	3
3	11	01	1111	1011	2
...	...	...	...	...	...

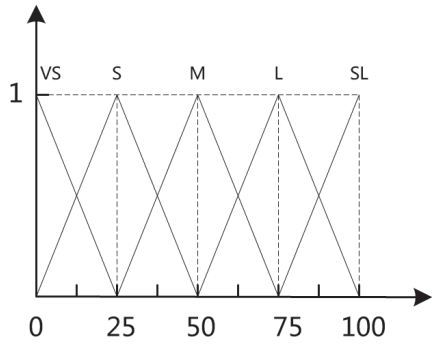

Fig. 5. The membership function.

All experiments are conducted on three PCs with an Intel Core2 2.8 GHz processor, 2.0 GB of RAM, Windows XP SP3, and Python 3.2.2. Two PCs is used to simulate smartphone users. One PC that is used to deploy our adaptation system, is used to simulate the Adaptation Server of SDP. All results are reported below as averages.

4.2. Experimental results on rraining accuracy

In this section, we compare CXCS with XCS on training accuracy by the following two experiments.

Definition 1. Training Accuracy (TA): We define the training accuracy of mobile service adaptation system as the ratio of the number of all training results and the number of correct training results with the following:

$$TA = 100\% \times \ count_{correct-train}/count_{train} \tag{12}$$

where the better the approach is, the higher the training accuracy is.

In first experiment, the static profiles of 10 users are used to identify their static similarity. The user similarity degree and similar-user ratio threshold are set to 0.85 and 0.5, respectively. While one user is selected as the current user, the other users are requested to initialize this user's classifier population. The average training accuracy is compared with XCS in Fig. 6.

The second experiment is conducted to validate co-evolutionary XCS's cooperative covering and predicting functions. First, all the training records are used to calculate the user similarities. Then, for each user, the similar-users assisted with the covering and predicting process. Finally, the average training accuracy is shown in Fig. 7.

From the comparison results, the training accuracy of our proposed CXCS is higher than XCS. Obviously, CXCS deployed on adaptation server, can support mobile service adaptation more effectively than XCS in mobile network.

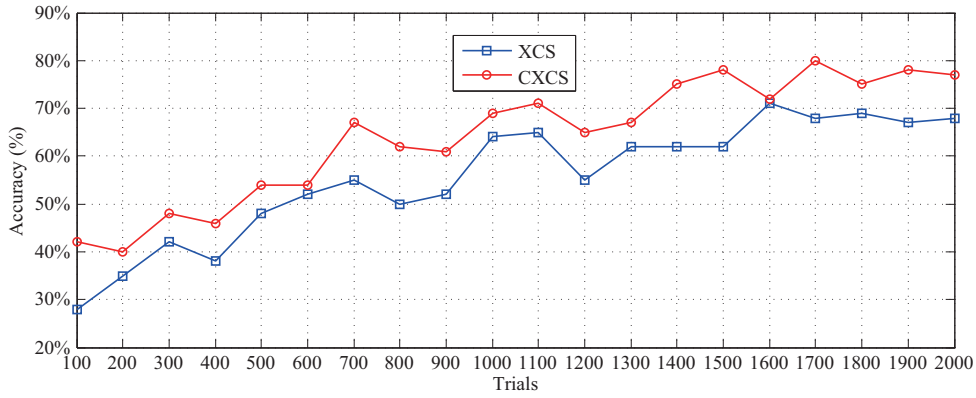

Fig. 6. Training accuracy using cooperative initialization.

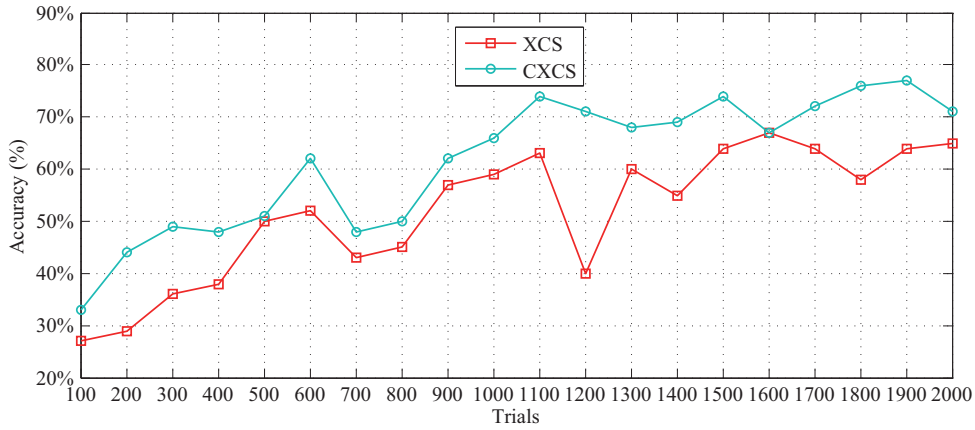

Fig. 7. Training accuracy using cooperative cover and predicting.

4.3. Experimental results on convergence speed

In this experiment, we evaluate the convergence speed of CXCS.

Definition 2. Convergence Speed (CS): We define the convergence speed of mobile service adaptation system as the ratio the fitness and the convergence fitness with the same Trails by the following:

$$CS = \frac{Convergence\text{-}Fitness_i}{Fitness_i}(1 \leqslant i \leqslant Trails) \tag{13}$$

where the better the approach is, the higher the convergence speed is.

As shown in Table 5, it can be concluded that CXCS converges more quickly than XCS because of its cooperative co-evolution mechanism.

4.4. Experimental results on adaptation accuracy

In this experiment, we compare our proposed CXCS with other five schemes such as XCS, Bayesian, SVM, C4.5 and Decision tree, in term of the adaptation accuracy.

Table 5
Comparison results on convergence speed

Trails	Convergence speed	
	XCS	CXCS
200	0.320	**0.420**
400	0.380	**0.470**
600	0.520	**0.580**
800	0.475	**0.560**
1000	0.615	**0.675**
1200	0.475	**0.680**
1400	0.585	**0.720**
1600	0.660	**0.760**
1800	0.655	**0.775**
2000	0.665	**0.780**

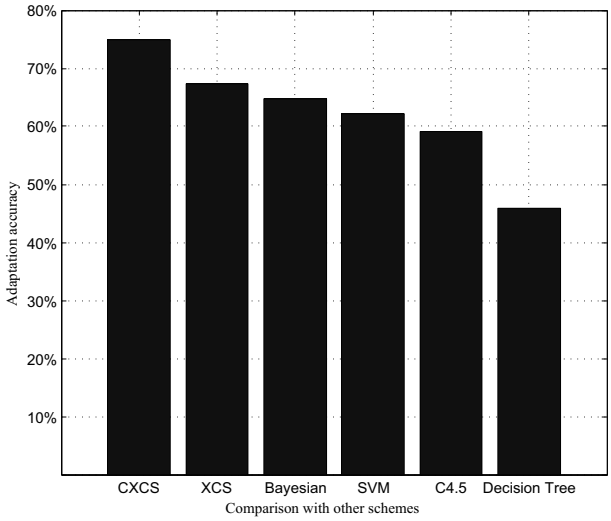

Fig. 8. Comparison with other schemes on adaptation accuracy.

Definition 3. Adaptation Accuracy (AA): We define the adaptation accuracy of mobile service adaptation system as the ratio of the number of all test results and the number of correct test results with the following:

$$AA = 100\% \times count_{correct\text{–}test}/count_{test} \tag{14}$$

where the better the approach is, the higher the adaptation accuracy is.

The experiment is run 15 times in testing process and the results are obtained, on average, as shown in Fig. 8. According to the comparison results, the adaptation accuracy of CXCS is higher 70%. Obviously, it outperforms other mobile servcie adaptation schemes on the adaptation accuracy for mobile services in mobile network environments. The main reason is that new users can be initialized accurately and all context can be covered effectively.

5. Related work

Many approaches [17–21] have previously been proposed for learning the rules for context-aware service adaptation. Nearchos Paspallis et al. [3] proposed to use the utility function-based method, but its fatal weakness is that the weight in the utility function is very difficult to determine during real applications. Choi et al. [4] applied work-flows to implement the adaptation, and Miraoui et al. [5] used a coloured Petri-net for the adaptive decisions. However, most of these previous approaches can only run in an off-line way, and the adaptation ability is still very limited.

In contrast to these previous schemes, our study focuses not only on an on-line context-aware approach but also on effective mobile service adaptation ability. LCS offers good performance as well as excellent understandability, and it has wide applications in data mining, automatic robots, traffic controlling, automatic fighter control, pattern recognition and complex adaptive systems. However, due to its complexity, it is too difficult to implement for practical applications. Wilson proposed ZCS [14] as a simplified version of LCS. It maintained most of the LCS framework, but increased the understandability

and performance significantly. One characteristic of ZCS is that it used a strength-based mechanism to represent the classifier payoff. However, it also introduces a shortcoming in distinguishing the classifier accuracy. Therefore, Wilson presented another improved version called XCS [13], which used a fitness-based accuracy, niche-based GA, modified Q learning credit assignment mechanism. Thus far, XCS is the most successful variety of LCS, and it has been widely used in many areas.

However, because of the new-user problem and the context-cover problem, XCS cannot effectively perform context-aware mobile service adaptation. Hence, in this paper, CXCS extends XCS by adding a cooperative co-evolution mechanism [16], modifying the rule structure and working process, and adding a user-interactive feature. Research on co-evolutionary LCS is still very rare. Dumitrescu [22] applied co-operative and competitive co-evolutionary mechanisms to the classifiers of the same species. ETANI [23] proposed a cooperative co-evolutionary architecture consisting of a collection of LCS, where each one attempts to evolve its subcomponents (agents), and the assembling of all these subcomponents represents the complete solution. Although the two studies produced good results, they still failed to solve the new-user problem and the context-cover problem in context-aware mobile service adaptation. Shang Gao et al. [24] reported on the development of a survey instrument designed to measure user perception on mobile services acceptance. Junho Ahn and Richard Han [25] focused on recognizing mobile users' daily behavior patterns and unusual event detection. They built a general unusual event classification model to analyze the activity, location, and audio sensor data collected from mobile phone users to identify these users' personalized normal daily behavior patterns. The model may be able to provide valuable information for mobile service adaptation systems.

6. Summary and future work

In this paper, we incorporate a cooperative co-evolutionary mechanism into XCS, proposing CSC, to perform context-aware mobile service adaptation. In CXCS, similar-users are used to initialize the classifier population of new-users. Moreover, the current user's uncovered-context states are also predicted. Experimental results show that CXCS can quicken the convergence speed and improve the training accuracy. Moreover, CXCS is also better than five adaptation schemes in terms of the adaptation accuracy in mobile network environments. Hence, CXCS can effectively perform context-aware mobile service adaptation in mobile network environments.

In the future, we will focus on implementing more adaptation function of CXCS. Besides, more evaluation for CXCS in a real mobile service environment is also our future work.

Acknowledgements

The work presented in this study is supported by the NSFC(61202435); NSFC(61272521); Natural Science Foundation of Beijing under Grant No.4132048; Specialized Research Fund for the Doctoral Program of Higher Education under Grant No. 20110005130001; Program for New Century Excellent Talents in University of China under Grant No.NCET-10-0263; and Innovative Research Groups of the National Natural Science Foundation under Grant No.61121061. We also thank Dr. Guoqiang Li of Amazon China.

References

[1] Klaus R. Dittrich, S. Gatziu and A. Geppert, The Active Database Management System Manifesto: A Rulebase of ADBMS Features, *Lecture Notes in Computer Science 985*, Berlin, Germany, Springer (1995), 3–20.
A. Attou and K. Moessner, Context-aware service adaptation management IEEE International Symposium on Personal, *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2007)*, (2007), 1–5.

[2] G. Gehlen, F. Aijaz, Y. Zhu and B. Walke, Mobile P2P Web Services using SIP, *Mobile Information Systems* **3** (2007), 165–185.

[3] N. Paspallis, K. Kakousis and G.A. Papadopoulos, A multi-dimensional model enabling autonomic reasoning for context-aware pervasive applications, *Proceedings of the 5th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services (MobiQuitous 2008)*, (2008), 1–6.

[4] J. Choi, Y. Cho, K. Shin and J. Choi, A context-aware workflow system for dynamic service adaptation Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), *Proceedings of the 2007 international conference on Computational science and its applications (ICCSA 2007)*, (2007), 335–345.

[5] M. Miraoui, C. Tadj and C.B. Amar, Dynamic context-aware service adaptation in a pervasive computing system, *Proceedings of the 3rd International Conference on Mobile Ubiquitous Computing, Systems, Services, and Technologies (UBICOMM 2009)*, (2009), 77–82.

[6] A.B. Kocaballi and A. Kocyigit, Granular Best Match Algorithm for Context-Aware Computing Systems, *Proceedings of the ACS/IEEE International Conference on Pervasive Services (ICPS 2006)*, (2006), 143–149.

[7] G. Koutrika and Y. Ioannidis, Personalizing queries based on networks of composite preferences, *ACM Transactions on Database Systems* **2** (2010), 1735886–173589.

[8] Z. Huang, X. Lu and H. Duan, Context-aware recommendation using rough set model and collaborative filtering, *Artificial Intelligence Review* **1**, (2011), 85–99.

[9] R. Yasdi, Learning classification rules from database in the context of knowledge acquisition and representation, *IEEE Transactions on Knowledge and Data Engineering* **3** (1991), 293–306.

[10] K. Chrysostomou, S.Y. Chen and X. Liu, Identifying user preferences with Wrapper-based Decision Trees, *Expert Systems with Applications* **4** (2011), 3294–3303.

[11] M.-W. Tong, Z.-K. Yang and Q.-T. Liu, A novel model of adaptation decision-taking engine in multimedia adaptation, *Journal of Network and Computer Applications* **1** (2010), 43–49.

[12] A. Shankar and S. Louis, Learning classifier systems for user context learning, *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2005)*, (2005), 2069–2075.

[13] J.H. Holland, Adaptation in natural and artificial systems, *MIT Press*, Cambridge, MA, USA, 1992.

[14] S.W. Wilson, Classifier fitness based on accuracy, *Evolutionary Computation* **2** (1995), 149–175.

[15] L.D. Shi, Y.H. Shi, Y. Gao, L. Shang and Y.B. Yang, XCSc: A novel approach to clustering with extended classifier system, *International Journal of Neural Systems* **1** (2011), 79–93.

[16] M.A. Potter and K.A.D. Jong, Cooperative Coevolution: An Architecture for Evolving Coadapted Subcomponents, *Evolutionary Computation* **1** (2000), 1–29.

[17] K. Henricksen, J. Indulska and A. Rakotonirainy, Using context and preferences to implement self-adapting pervasive computing applications, *Software – Practice and Experience* **11–12**, (2006), 1307–1330.

[18] J. Xiao and R. Boutaba, QoS-aware service composition and adaptation in autonomic communication, *IEEE Journal on Selected Areas in Communications* **12** (2005), 2344–2360.

[19] C. Jacob, D. Linner, I. Radusch and S. Steglich, Context-aware Data Dissemination and Service Adaptation, *Proceedings of the 16th IST Mobile and Wireless Communications Summit (IST 2007)*, (2007), 1–5.

[20] C. Baladron, J. Aguiar, B. Carro, L. Calavia, A. Cadenas and A. Sanchez-Esguevillas, Framework for intelligent service adaptation to user's context in next generation networks, *IEEE Communications Magazine* **3** (2012), 18–25.

[21] W.Y. Lum and F. Lau, A context-aware decision engine for content adaptation, *IEEE Pervasive Computing* **3** (2002), 41–49.

[22] D. Dumitrescu, M. Preuss, C. Stoean and R. Stoean, Coevolution for classification, *Technical report, No. CI-239/08*. Collaborative Research Center on Computational Intelligence University of Dortmund, Germany, 2008.

[23] E. N, Modeling Agent-Based Coevolution with Learning Classifier System, *IEIC Technical Report* **535** (2002), 49–53.

[24] S. Gao, J. Krogstie and K. Siau, Developing an instrument to measure the adoption of mobile services, *Mobile Information Systems* **1** (2011), 45–67.

[25] J. Ahn and R. Han, Personalized behavior pattern recognition and unusual event detection for mobile users, *Mobile Information Systems* **2** (2013), 99–122.

Shangguang Wang is an assistant professor at Beijing University of Posts and Telecommunications. He received his Ph.D. degree at Beijing University of Posts and Telecommunications in 2011. His research interests include service computing and cloud computing. He has served as reviewers for numerous journals, including IEEE Internet Computing, Mobile Networks and Applications, IET Software, Computer Journal, Service Oriented Computing and Applications, Security and Communication Networks, Journal of Network and Computer Applications, International Journal of Systems Science, Entropy, Sensor Letters, etc. He is an IEEE member, ACM member as well as a CCF senior member. Homepage: http://www.sguangwang.com/.

Zibin Zheng is an associate research fellow at Shenzhen Research Institute, The Chinese University of Hong Kong. He is also a guest research member at State key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China. He received Outstanding Ph.D. Thesis Award of The Chinese University of Hong Kong at 2012, ACM SIGSOFT Distinguished Paper Award at ICSE2010, Best Student Paper Award at ICWS2010, and IBM Ph.D. Fellowship Award at 2010. He served as PC member of CLOUD2009, SCC2011, SCC2012, ICSOC2012, etc and reviewers for journals including IEEE Transactions on Software Engineering, IEEE Transactions on Parallel and Distributed System, ACM Transactions on the Web, IEEE Transactions on Service Computing, etc. His research interests include service computing and cloud computing. Homepage: http://www.zibinzheng.com/.

Zhengping Wu is an assistant professor of Computer Science and Engineering at the University of Bridgeport in USA. He received his Ph.D. in Computer Science from the University of Virginia in 2008. He has authored or co-authored eleven book chapters and over forty peer-refereed papers. His research interests include services computing, cloud computing, network security, distributed systems, wireless networks, operating systems, and medical informatics. He has served as program committee members and reviewers for numerous conferences and journals, including IEEE Transactions on Services Computing, IEEE Transactions on Systems, Man, and Cybernetics (A), IEEE Transactions on Fuzzy Systems, IEEE Transactions on Control Systems Technology, IEEE Transactions on Industrial Electronics. Homepage: http://www1bpt.bridgeport.edu/∼zhengpiw/.

Qibo Sun received his PhD degree in communication and electronic system from the Beijing University of Posts and Telecommunication in 2002. He is currently an associate professor at the Beijing University of Posts and Telecommunication in China. He is a member of the China computer federation. His current research interests include services computing, internet of things, and network security.

Hua Zou received her PhD degree in communication and electronic system from the Beijing University of Posts and Telecommunication in 2010. She is currently a professor at the Beijing University of Posts and Telecommunication, China. Her current research interests include network intelligence and services computing.

Fangchun Yang received his PhD degree in communication and electronic system from the Beijing University of Posts and Telecommunication in 1990. He is currently a professor at the Beijing University of Posts and Telecommunication, China. His current research interests include network intelligence and services computing. He is a fellow of the IET.