A Fuzzy-based Service Adaptation Middleware for Context-aware Computing[1]

Ronnie Cheung, Jiannong Cao, Gang Yao, Alvin Chan

Department of Computing, Hong Kong Polytechnic University
{csronnie, csjcao, csgyao, cstschan}@comp.polyu.edu.hk

Abstract. In a mobile environment, it is desirable for mobile applications to adapt their behaviors to the changing context. However, adaptation mechanism may emphasize more on overall system performance, while neglecting the needs of individual. We present a generalized Adaptive Middleware Infrastructure (AMI) to cater for individual needs in a fair manner, while maintaining optimal system performance. Furthermore, due to the vagueness in context nature and uncertainty in context aggregation for adaptation, we propose a Fuzzy-based Service Adaptation Model (FSAM) to achieve generality and improve the effectiveness of service adaptation. By fuzzification of the context and measuring the fitness degree between the current context and the optimal situation, FSAM adopts the most appropriate service. We have evaluated the FSAM inference engine within the middleware AMI by an application Campus Assistant. The performance is analyzed and compared with a conventional threshold-based approach.

Keywords middleware infrastructure, fuzzy theory, context-aware, service adaptation, mobile computing

1 Introduction

In a mobile environment, the variations and constraints of communication and computing resources introduced by the dynamical nature of wireless transmission call for adaptive context-aware applications that can adapt their behaviors to the contexts. In this paper, we present an Adaptive Middleware Infrastructure (AMI), which sits between applications and the operating system [1] [[3]. The fairness among applications and between the high-level applications and the low-level operating system both are taken into account. AMI provides applications with the most suitable service based on predefined policies so as to achieve adaptability to the changing context. It consists of well-defined modules in support of middleware functionalities to improve the generality.

The area of context-aware middleware has attracted many researchers [9]. In system CARISMA [4], the context-aware middleware compares the application profile and the current context to evaluate which policy to be adopted. In [5], the functionality of a service code module is adapted on the basis of the estimation of resource usage. However, these systems are problem-specific or focus on software realization while AMI emphasizes a generic reference infrastructure.

[1] This project is supported by RGC research grants under Project Ref. No. PolyU 5200/04E and also partially supported by Hong Kong Polytechnic University under the ICRG grant G-YD63.

The effectiveness of service adaptation to the current context is a another challenge in context-aware computing, due to the vagueness of the context nature from the perspective of human, and the uncertainty in context aggregation when making adaptation decision under the circumstance of multi-dimensional context. Introducing fuzzy techniques is one of promising approaches to deal with these problems.

We formalize a context-aware service adaptation framework and propose a Fuzzy-based Service Adaptation Model (FSAM) [2]. We employ the linguistic variables and membership degrees to carry out the fuzzification of the context situation, so that the vagueness of context can be quantified. The fitness function is developed to produce an overall fitness degree of the current context, corresponding to each predefined policy, in order to adopt the most suitable one for service selection and delivery. The fitness degree is obtained by measuring the distances between the current context and the predefined reference context situation.

Much research has been done on fuzzy-based adaptation. In [6], the authors offer a survey on applying fuzzy theory to adapting QoS requirements in communication networks. In [7], the fuzzy control theory is used for QoS adaptation in distributed multimedia applications. Again, the approaches are usually developed for a specific domain and not targeted at a generic service adaptation model, while FASM deals with application-independent service adaptation.

We implement the AMI along with the FSAM as the inference engine in the Campus Assistant application. The application offers different quality level of chat and email services adaptive to continuously-varying context. The variations of context are simulated to trigger service adaptation. Based on the experiment results, the performance of the fuzzy-based solution is analyzed and compared with the conventional threshold-based context aggregation approach for context-aware adaptation.

2 Adaptive Middleware Infrastructure (AMI)

The Adaptive Middleware Infrastructure (AMI), which allows generic applications to exercise context-awareness [1] [3], aims at the integration of all relevant features and the synergy achieved by a well-established and unified baseline architecture in order to promote the development of context-aware computing.

All the relevant features that AMI integrates include context collection, context composition, context reference engine, context delivery, adaptive middleware services, ToS and QoS enforcement, etc. As shown in Figure 1, the features are embodied in the four major modules in our system:

The first module is the Context Space. It contains Context Detectors, Context Composers and Fuzzy Context Composers that detect and compose low-level contexts into higher level representations. Examples of context detectors could be the wrappers for OS event, an application's computing activities. The fuzzy context composers composite low-level context information, marshalling the dynamic and uncertainty of the environment, and presents the current context in a generic form. For example, a fuzzy context composer could be monitoring all network-related detectors and determine the quality of network connection with the membership degree.

The second module is the Middleware Service Space. This is the execution environment for both Public Middleware Services (one set of services that are shared

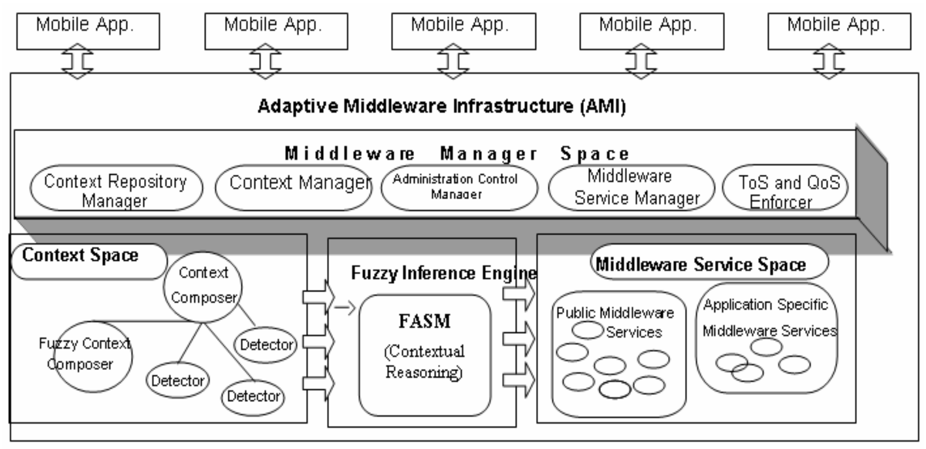

Fig. 1. Adaptive Middleware Infrastructure

among all applications) and Application Specific Middleware Services (each mobile application has its own set of services). These adaptive middleware services are provided to individual applications under the control of the Fuzzy Inference engine.

The third module is the Fuzzy Inference Engine, which determines the middleware services according to the current context and the requirement of application-specific ToS and QoS. In order to control the middleware services, the fuzzy inference engine is aware of the programmable properties of the middleware service. To adjust quantitative parameters for the middleware services, the fuzzy inference engine calls the corresponding adjustment functions. To switch among the middleware services, the inference engine stops exporting certain services while activate others.

The fourth module is the Middleware Manager Space. The middleware manager space contains five system managers, which coordinates all management operations within the AMI. The Administration Control Manager component manages the admission of mobile applications that subscribe the services of the middleware. The Context Manager controls the runtime environment for the context objects, including the low level context detectors and high level context composers. The Context Repository Manager stores the records on all contextual information, such that queries on context history are possible. The Middleware Service Manager coordinates with admission control. It also allocates and controls the resources for newly subscribed services. The ToS and QoS Enforcer are used for monitoring the ToS and QoS levels for each connected application.

3 Fuzzy-based Service Adaptation Model (FSAM)

The core component of the AMI is the fuzzy inference engine, which determines service adaptation according to the context derived from the context space. We have developed the Fuzzy-based Service Adaptation Model (FSAM) as the inference engine [2]. FSAM takes the context information and other policy definitions as the input. The output of FSAM is the adaptation decision, which will be used by the middleware service manger to reconfigure services. In this section, first we define the concepts and terminology used in FSAM. Then we introduce the fitness function base on the definitions. Finally, the procedure of decision-making in FSAM is illustrated using the following formulas:

Service: A service is a functionality provided in the application. The service can be delivered at different quality level or with different resource constraints. Let S= $\{s_1, s_2, s_3, \ldots, s_q\}$ $(1 \le q)$, represent the service set, where s_i $(1 \le i \le q)$ represents the i-th service.

Policy: A policy is a rule which determines which quality level of service to be delivered based on the context. Let $P_i = \{p_i^1, p_i^2, \ldots, p_i^{mi} \mid i \in [1, q]\}$ be a set of policies, where p_i^j $(1 \le j \le m)$ represents the j-th policy corresponding to the i-th service s_i. m_i is the number of the policy corresponding to s_i.

Context: A context is one of conditions to determine a policy for service selection. Let $C=\{c_1, c_2, \ldots, c_n\}$ be a set of context, where c_i $(1 \le i \le n)$ represents the i-th context information, n is the number of all context.

Context Situation: A Context Situation is the composite context information to represent the context at any given time t. It is denoted by a set of 3-element tuples:

$$SI(t) = \{(c_i, lv_b, m_{c_i\, lv_b}(value\_of(c_i, t)) \mid c_i \in C, i \in [1,n], b \in [1,k]\} \qquad \textbf{(1)}$$

where, $c_i (1 \le i \le n)$ is a context, $lv_b (1 \le b \le k)$ is a linguistic value, $m_{c_i\, lv_b}(x) \in [0,1]$ is the predefined membership function for "c_i is lv_b", $value\_of(c_i,t)$ represents the value of context c_i at time t.

Standard Reference Depositary: Given a service s_i, with respect to each policy p_i^j $(1 \le j \le m)$, we assume that there must exist a specific Context Situation associated with p_i^j. On the basis of the Context Situation, the policy p_i^j is the most suitable policy for service s_i and has to be adopted. Intrinsically, the most suitable policy means a tradeoff between resource constraints in certain context and the service to be delivered. A most suitable Context Situation is referred to Standard Reference Context Situation. We call the aggregation of $\{SR(p_i^1), SR(p_i^2), \ldots, SR(p_i^{mi})\}$, $p_i^j \in P_i$, as the Standard Reference Depositary of P_i and denote it by SRD(P_i).

$$SR(p_i^j) = \{(c_i, lv_b, m_{c_i\, lv_b}(best\_value\_of(c_i)) \mid c_i \in C, i \in [1,n], lv_b \in LV, b \in [1,k]\} \qquad \textbf{(2)}$$

Fitness Function: In practice, a Context Situation at time t usually is at certain distance away from any SR(p_i^j), so we define the Fitness Function to evaluate the Fitness Degree of the Context Situation at time t against Standard Reference Context Situation SR(p_i^j). Given a service s_i, the Fitness Function is the mapping from the Context Situation at time t and Standard Reference Context Situation SR(p_i^j) to the Fitness Degree of policy p_i^j : (FF): $SI(t) * SR(p_i^j) \longrightarrow FD(p_i^j)$,

$$FF(SI(t), SR(p_i^j)) = \frac{1}{\sum_{i=1}^{size\_of(SR(p_i^j))} \left| m(best\_value\_of(c_i)) - m(value\_of(c_i, t)) \right|^{l_i}} \qquad \textbf{(3)}$$

where $size\_of(SR(p_i^j))$ represents the number of tuples in SR(p_i^j), $\mu(x)$ is the membership function appears in i-th vector, l_i is a natural number.

In function (3), the denominators are for the calculation of the distance between SI(t) and SP(p_i^j). After obtaining the fuzzy distance, we calculate the reciprocal to obtain the Fitness Degree. When $l_i=1$, the function uses Hamming Distance; when $l_i=2$, the function uses Euclidean Distance, both are classical methods for calculating fuzzy distance between two statuses [8]. When $l_i=3$, we regard l_i as a weight value for a context, which can be adjusted by individual application to affect policy choice.

4 Implementation and Evaluation

In this section, we implement the FSAM model as the contextual inference engine within the AMI in a hypothetic application Campus Assistant.

4.1 Application Campus Assistant

The Campus Assistant is a mobile context-aware application running on mobile platforms. Through the wireless network, the Campus Assistant enables the user to communicate with each other in real time, or retrieve and send emails by offering Chat and Email services. Due to the spatial and temporal variations of wireless communication and computing resources, the Campus Assistant tries to detect the changing context and deliver the different quality level of service, in order to maintain an acceptable level of perception when the resources available become inadequate.

Such adaptation is predefined by certain rules, i.e. policies in the system. The Chat service has three policies: textChat, voiceChat and videoChat. The Email service has five policies: headMail, fullMail, encryptedMail, bigMail and encryptedBigMail. Each policy is corresponding to a certain quality level of service. The context can have many aspects due to the multidimensional characteristics, including communication, computing, geographical, organizational, etc. To simplify the context analysis, only Network_maxRate, CPU_clockRate, Network_delay, RAM_freeSpace are taken in account in our case. The characteristics of vagueness and subjectivity in context are handled by the aforementioned fuzzy-based inference engine FSAM.

4.2 FSAM Configuration and Procedures

Corresponding to FSAM theoretic framework, we assume that the inference engine with the following configurations in the application.

Service: S = {chat, email}, i.e. S_1=chat and S_2= email

Policies: P_1 = {textChat, voiceChat, videoChat},
P_2 = {headMail, fullMail, encryptedMail, bigMail, encryptedBigMail}

Context: C={Network_maxRate, CPU_clockRate, Network_delay, RAM_freeSpace}

Context Situation: 4-element tuples

SI(*t*)={(Network_maxRate, high, $\mu_{\text{Network\_maxRate high}}$ (value_of(Network_maxRate, t))),
(CPU_clockRate, high, $\mu_{\text{CPU\_clockRate high}}$ (value_of(CPU_clockRate, t))),
(Network_delay, low, $\mu_{\text{Network\_delay low}}$ (value_of(Network_delay, t))),
(RAM_freeSpace, high, $\mu_{\text{RAM\_freeSpace high}}$ (value_of(RAM_freeSpace, t)))}

Membership Functions: context c_i ($i=1,2,3,4$), application-specific and produced from releated domains

$$C_1=\begin{cases} 0 & B<1\text{Kbps} \\ \dfrac{\log_{10} B/1k}{5} & 1k\leq B\leq 100\text{Mbps} \quad (4) \\ 1 & B>100\text{Mbps} \end{cases}$$

Fig. 2. Network_maxRate high membership function

$$C_2=\begin{cases} 0 & F<2\text{MHz} \\ \dfrac{\log_{10} F/2M}{3} & 2M\leq F\leq 2\text{GHz} \quad (6) \\ 1 & F>2\text{GHz} \end{cases}$$

Fig. 4. CPU_clockRate high membership function

$$C_3=\begin{cases} 0 & T>1000\text{ms} \\ 1-\dfrac{\log_{10} T/0.1}{4} & 0.1\leq T\leq 1000\text{ms} \quad (5) \\ 1 & T<0.1\text{ms} \end{cases}$$

Fig. 3. Network_delay low membership function

$$C_4=\begin{cases} 0 & R<50\text{KB} \\ \dfrac{\log_{10} R/50k}{4} & 50k\leq R\leq 500\text{MB} \quad (7) \\ 1 & R>500\text{MB} \end{cases}$$

Fig. 5. RAM_freeSpace high membership function

Standard Reference Context Situation: As in Table 1, the most suitable values of context corresponding to each policy is predefined.

Table 1. Standard Reference Context Situation

	Network_max Rate(kbps)	CPU_clock Rate(MHz)	Network_ delay(ms)	RAM_free Space(KB)
text Chat (p_1^1)	4	20	500	0.2
voiceChat (p_1^2)	200	300	10	4
videoChat (p_1^3)	10000	1000	0.2	200
headMail (p_2^1)	2	4	n/a	0.2
fullMail (p_2^2)	10	10	n/a	0.4
encryptedMail (p_2^3)	10	100	n/a	10
bigMail (p_2^4)	500	50	n/a	2
encryptedBigMail (p_2^5)	500	1000	n/a	100

Standard Reference Depository: Based on Table 1 and membership functions, the SRD(P_1’) and SRD(P_2’) are determined as in Table 2.

Table 2. SRD(P'_1)and SRD(P'_2)

		Network_max Rate High	CPU_clock Rate High	Network_ delay Low	RAM_free Space High
SRD(P'_1)	SR(p_1^1)	0.12	0.33	0.08	0.15
	SR(p_1^2)	0.46	0.72	0.50	0.48
	SR(p_1^3)	0.80	0.90	0.92	0.90
SRD(P'_2)	SR(p_2^1)	0.06	0.10	n/a	0.15
	SR(p_2^2)	0.20	0.23	n/a	0.23
	SR(p_2^3)	0.20	0.57	n/a	0.58
	SR(p_2^4)	0.54	0.47	n/a	0.40
	SR(p_2^5)	0.54	0.90	n/a	0.83

With the configuration, we initiate the reasoning and decision-making procedures in FSAM. First we use the membership functions to map the current context into the Context Situation. Then, by substiting the values of the current context SRD(P_1) and SRD(P_2) into the Fitness Function formula (3) to compute the fitness degrees. E.g.

$$\begin{aligned} FF(SI(t),SR(P_1^1)=1 / ((&0.12- m_{\text{Network\_maxRate } high}(\text{value\_of(Network\_maxRate, t)})^3 \\ &+ (0.33 - m_{\text{CPU\_clockRate high}}(\text{value\_of(CPU\_clockRate, t)})^3 \\ &+ (0.8- m_{\text{Network\_delay low}}(\text{value\_of(Network\_delay, t)})^3 \\ &+ (0.15- m_{\text{RAM\_freeSpace low}}(\text{value\_of(RAM\_freeSpace, t)})^3) \end{aligned}$$

Finally, regarding each service, the policy $P_{suitable}$ which has the maximum fitness degree will be the most suitable one to be adopted. For chat service:$P_{suitable}$=Max {FF(SI(t), SR(P_1^1)), FF(SI(t), SR(P_1^2)), FF(SI(t), SR(P_1^3))}; For email service: $P_{suitable}$=Max {FF(SI(t), SR(P_2^1)), FF(SI(t),SR(P_2^2)), FF(SI(t),SR(P_2^3)), FF(SI(t),SR(P_2^4)), FF(SI(t),SR(P_2^5))

4.3 Evaluation

We simulate the variations of the changing context by means of generating multiple sets of 4-element tuples (Network_maxRate, CPU_clockRate, Network_delay, RAM_freeSpace). The 4-element tuples are fed into the FASM inference engine in a time-series manner. Since the network bandwidth and network delay play a significant role in affectomg the performance of applications hosted in a mobile device, we produce two segments of changing network bandwidth and delays to simulate the varying of wireless communication to trigger the adaptation in a severely fluctuating manner. As in Figure 6 and Figure 7, 160 sets of tuples are generated.

For the purpose of evaluation, a conventional threshold-based context aggregation and service adaptation approach is also implemented in the Campus Assistant application as the inference engine. Similar to the VDS system by Cristian Koliver et al in [6], a formula of QoS parameter aggregation is offered and addressed. Since the QoS parameters constitute a subset of context, so we apply the formula to the field of context-aware service adaptation.

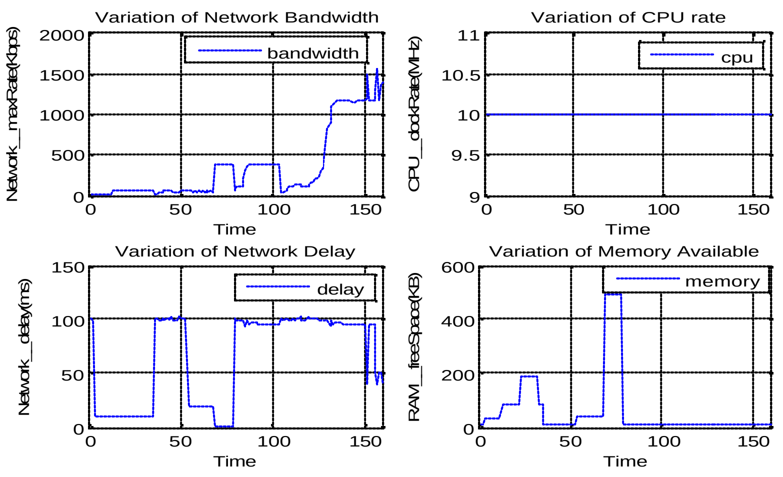

Fig. 6. Variation of contexts

A service adaptation mode M is defined by the formula: $M = (f_1 w_1 + f_2 w_2 + \ldots + f_i w_i)$, where f_i denotes the adaptation factor of each context. w_i represents the weight for each context, where $i=\{1,2,3,4\}$, and is employed to adjust the influence on adaptation of each context. It alleviates the compensation between different contexts as well. According to the predefined association between mode M and policy, the right policy will be adopted to deliver the suitable service based on the context at a given time t. When mapping context c_i to its adaptation factor: $f_{i,}=s_i c_i$, where $c_i \in C_i$, $i=\{1,2,3,4\}$ s_i is the scaling factor to scale $f_{i,}$ to a normalized range. C_i = [min, max] determines a valid range for each context, where min is the lower bound and max is the upper bound. If the context is beyond the range of $C_{i,}$ it will be rejected and ignored. In our implementation, the specific values are congfigured as follows:

Context: c_i={Network_maxRate,CPU_clockRate,Network_delay,RAM_freeSpace}
Weight: w_i =1, to match the FSAM setting; Scaling Factor: s_i =1;
Adaptation Factor: $f_i=c_{i,}$ where $i=\{1,2,3,4\};$
For chat service: $M=c_1/c_3+c_2+c_4$, since the context Network_maxRate and Network_delay are relevant; For email service: $M=c_1+c_2+c_4$, since each context are independent; For either chat or email service, the thresholds are heuristic and produced in the following Table 3 and Table 4. Similar to the fuzzy-based FASM, the threshold-based inference engine produces the adaptation mode M. The policy corresponding to M will be adopted as the current adaptation strategy.

Table 3. Thresholds of adaptation mode for Chat service

Chat Policies	M
text Chat(p_1^1)	[min, 53.2)
voiceChat(p_1^2)	[53.2, 926)
videoChat(p_1^3)	[926, max)

Table 4. Thresholds of adaptation mode for Email service

Email Policies	M
headMail (p_2^1)	[min, 15.2)
fullMail (p_2^2)	[15.2, 38)
encryptedMail (p_2^3)	[38, 98)
bigMail (p_2^4)	[98, 496)
encryptedBigMail(p_2^5)	[496, max)

Based on the inputs, the FASM fuzzy engine and the conventional threshold-based approach respectively produce outputs to control the service adaptation. The outputs are recorded and compared as in Figure 7 and Figure 8. We observe that the fluctuations caused by the variations of context are filtered by the fuzzy inference engine FASM for both Chat service and Email service.

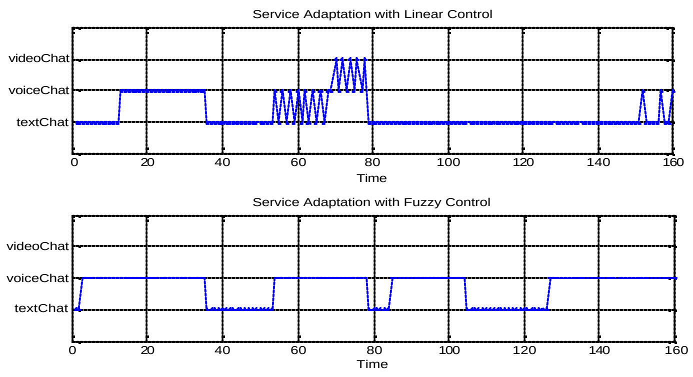

Fig. 7. Chat service adaptation

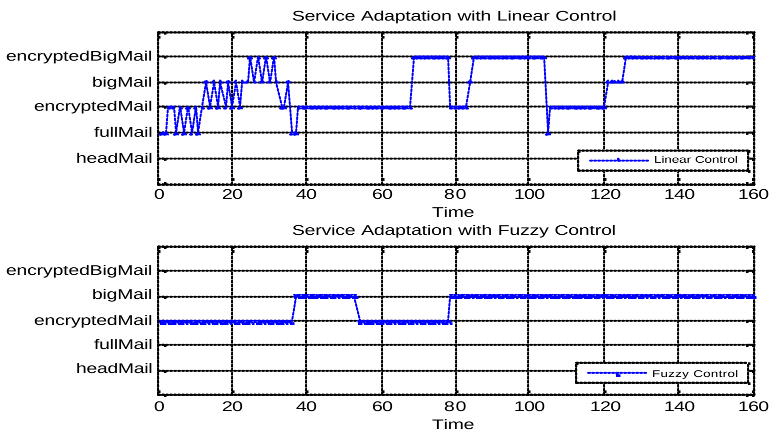

Fig. 8. Email service adaptation

In contrast, the threshold-based inference engine appears to be transparent to the variations. Particularly, regarding to the latter, in the situation of severely fluctuations of context changing, the service adaptation becomes unstable and keep changing accordingly. This phenomenon is called jitter and the jitters may lead to unbearable deterioration of service quality no matter the policy adopted is suitable to the context or not. From the perspective of microeconomics, it would not be difficult to observe that the adaptation curves by FASM are relatively smoother than those by threshold-based approach. This means the fuzzy-based solution has a better tolerance to the variations of context, which leads to the improvement of effectiveness of the service adaptation

As in Figure 9, it would not be difficult to conclude that the complexity of the approach majorly depends on the membership functions defined in FASM according to the procedures described earlier. Nevertheless, the membership functions are application-specific and relevant to particular scenarios. In our implementation, the computational cost of FASM inference engine and the conventional threshold-based approach are calculated in terms of running time. Due to the limits of the timer accuracy of the soft clock in the Windows operating system, we record the computation time in the unit of 20 service adaptations rather than once. As in Figure 9, the conventional threshold-based approach occupies very low machine time because of its linear characteristics. The computation time of the FASM engine varies, but the maximum is below 150 milliseconds. This is acceptable for most applications even in real-time scenarios.

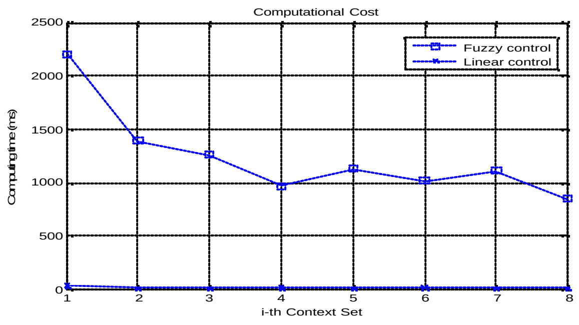

Fig 9. Computational cost

5 Conclusion and future work

In this paper, we introduce a generic middleware infrastructure AMI. A fuzzy-based service adaptation model FSAM is developed as inference engine within AMI. We formalize a framework by fuzzification of the context and measuring the fitness degree of the current context. The policy with the maxium degree will be adopted. We verify the effectiveness of FSAM compared to a conventional threshold-based approach based on the simulation results.

Since the Context Repository module in AMI maintains a historic record of context, the service adaptation can be not only reflective to the current context, but also be adaptive to the predictable future context. We consider the context prediction as the extension of the context-awareness in the time domain. The incorporation of context prediction would efficiently increase the effectiveness and reduce the response time.

6 References

1. Ronnie Cheung, "An Adaptive Middleware Infrastructure for Mobile Computing", Proceedings of 14th WWW2005 Conference, pp. 996-997, ACM Press, 2005
2. Jiannong Cao, Na Xing, Alvin T.S Chan, Yulin Feng, Beihong Jin, "rvice Adaptation Using Fuzzy Theory in Context-aware Mobile Computing Middleware", Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, pp. 496-501, 2005
3. Ronnie Cheung, "An Adaptive Middleware Infrastructure Incorporating Fuzzy Logic for Mobile Computing", in Proceedings of the International Conference on Next Generation Web Services Practices, IEEE Computer Society Press, pp 449-451 Korea, 2005
4. Capra, L., Emmerich, W., Mascolo, C.,"CARISMA: context-aware reflective middleware system for mobile applications", IEEE Transactions on Software Engineering, Volume: 29, Issue: 10, pp.929 – 945, Oct. 2003
5. Vivien Wai-Man Kwan, Francis Chi-Moon Lau, Cho-Li Wang, "Functionality adaptation: a context-aware service code adaptation for pervasive computing environments", Proceedings of IEEE/WIC International Conference on Web Intelligence, pp.358 – 364, Oct. 2003
6. Cristian Koliver, Jean-Marie Farines, and Klara Nahrstedt, "QoS Adaptation Based on Fuzzy Theory" Soft Computing for Communications, Springer-Verlag, pp. 245–267, 2004
7. Baochun Li, Nahrstedt, K.,"A control-based middleware framework for quality-of-service adaptations" IEEE Journal on Selected Areas in Communications, Volume: 17, Issue: 9 pp. 1632 – 1650, Sept. 1999
8. Constantin A. "Fuzzy Logic and Neuro-Fuzzy Applications Explained", Prentice Hall, 1995
9. Capra, L., "Mobile Computing Middleware for Context-Aware Applications" Proceedings of the 24th International Conference on Software Engineering, pp.723 – 724, 19-25 May 2002