Expert Systems with Applications 38 (2011) 4150–4161

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

UbiPaPaGo: Context-aware path planning

Chiung-Ying Wang [a], Ren-Hung Hwang [b,*], Chuan-Kang Ting [b]

[a] Department of Information Management, TransWorld University, Taiwan, ROC
[b] Department of Computer Science and Information Engineering, National Chung-Cheng University, Taiwan, ROC

A R T I C L E I N F O

Keywords:
Ubiquitous network
Human-centered computing
Context-awareness
Path planning
Genetic algorithm (GA)
Spatial conceptual map

A B S T R A C T

The increased prevalence of digital devices with communication capability heralds the era of ubiquitous computing, as predicted by Mark Weiser. Ubiquitous computing aims to provide users with intelligent human-centric context-aware services at anytime anywhere. Optimal path planning in a ubiquitous network considers the needs of users and the surrounding context. This approach is very different from that applied by existing research on car navigation and mobile robots. This study proposes a context-aware path planning mechanism based on spatial conceptual map (SCM) and genetic algorithm (GA), referred to as UbiPaPaGo. The SCM model is adopted to represent the real map of the surrounding environment. The optimal path is planned using a GA, which is a robust metaheuristic algorithm. UbiPaPaGo attempts to automatically find the best path that satisfies the requirements of an individual user. A prototype of UbiPaPaGo is implemented to demonstrate its feasibility and scalability. Experimental results validate the effectiveness and the efficiency of UbiPaPaGo in finding the optimal path.

1. Introduction

The demand for intelligent navigation applications in path finding, such as PaPaGo (PAPAGO) and TomTom (TomTom International), has risen significantly in recent years, helped by the rapid development of wireless and communication technologies. These systems provide mobile users, such as car drivers, with intelligent path planning, routing and navigation services, and direct users to their destination using electronic maps and Global Position System (GPS). However, these systems are computer-centric, and thus are unable to achieve the vision of Mark Weiser (Weiser, 1991) of a ubiquitous environment with human-centric applications. Context-awareness is a key to human-centric applications. A context-aware application extracts, interprets and uses context, and automatically adapts its functionality to the current context of an individual user. Consequently, mobile users can adopt context to plan the "best-fitting" path automatically.

To help address the problem and develop the solution, a scenario is described for finding John's optimal path that satisfies his requirements. John needs to drive to the Taipei Arena as quickly as possible to watch a show, because he is late. The automobile assistant connects to a streaming server to watch the live show remotely before John arrives at Taipei Arena. In this case, John needs to find the shortest path to the Taipei Arena, while keeping connectivity with the streaming server. Therefore, the optimal path must consider the path distance, accessibility of WiFi Access Point (AP) or 3G Base Station (BS), accessibility of the streaming service and quality of service (QoS) of the required service. The problem considered herein is formally defined as how to plan an optimal (shortest) path for an individual user based on his requirements and the context of his surrounding environment, such as user preference, location, network connectivity, bandwidth, service availability and quality of service (QoS). The objective is to find an optimal (shortest) path given constraints, such as higher bandwidth, available service and QoS guarantee. These constraints ensure that the shortest path is also the one that best fits the user requirements.

The context is very important for inferring the optimal path. The path planning framework relies on various contexts acquired from various sensing sources. The collected contexts are stored in an open and distributed context management server (Lu, Wang, & Hwang, 2009), called Ubiquitous Gate (U-gate), which is responsible for context acquisition, context representation, context retrieval, context protection and context inference. The path planning framework adopts an ontology as the underlying context model. The ontology model is implemented using RDF (Resource Description Framework) ([RDF Primer] and [RDF Schema]) and OWL (Web Ontology Language) (OWL). The context model includes user contexts, such as personal information and active applications, as well as environmental contexts, such as map information and network connectivity.

The proposed solution, called UbiPaPaGo, is based on spatial conceptual map (SCM) (Kettani & Moulin, 1999) and genetic algorithm (GA) (Holland, 1975). A modified SCM is proposed to model the real world map and related environment context, and a path

* Corresponding author. Address: 168, University Rd., Min-Hsiung Chia-Yi 621, Taiwan, ROC. Tel.: +886 5 2720411x33112; fax: +886 5 2720859.

E-mail addresses: wjy@cs.ccu.edu.tw (C.-Y. Wang), rhhwang@cs.ccu.edu.tw (R.-H. Hwang), ckting@cs.ccu.edu.tw (C.-K. Ting).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.09.077

planning map based on this model is then created. In addition, a GA is proposed to find the optimal path. Chromosomes are encoded based on the method proposed in Inagaki, Haseyama, and Kitajima (1999). The fitness function is carefully designed to fit the user requirements. Prototyping and simulations are performed to demonstrate the feasibility and the efficiency of the proposed UbiPaPaGo.

The remainder of this paper is structured as follows. Section 2 describes related works on path planning and context-aware path prediction. Section 3 introduces the SCM model. Section 4 describes the design of UbiPaPaGo based on GA. Section 5 provides implementation details of the prototyping system. Experimental results are discussed in Section 6. Conclusions are finally drawn in Section 7, along with our future research.

2. Related work

Path planning problems have been studied extensively in the past, but mostly focusing on mobile robots (Bodhale, Afzulpurkar, & Thanh, 2009; Castilho & Trujilo, 2005; Lei, Wang, & Wu, 2006; Li, Zhang, Yin, Zhang, & Liu, 2006; Liu, Lu, Yang, & Chen, 2008; Tu & Yang, 2003), transportation systems (Li, Liu, et al., 2006) and game systems (Arikan, Chenney, & Forsyth, 2001; Wan, Chen, & Earnshaw, 2003). These proposed path planning algorithms emphasize features, such as collision avoidance and ease of tracking. Although these solutions address information about roads, obstacles and maps, they do not adapt to user situations or context.

Context-aware path finding (moving path) has recently been studied. Studies which focused on context-aware path prediction are most related to our work. These approaches adopt context, such as history data, user's preference, location and map data, to reason the future moving path in advance. Path prediction approaches are able to yield high prediction accuracy and efficient network resource management and pre-configuration, thus providing high quality of service for users. Neural-network based prediction algorithm (Poon & Chan, 2000) predicts path from both global and user-level information. Profile-based next-cell prediction (PBNP) algorithm (Bharghavan & Jayanth, 1997), shadow cluster concept (Levine, Akyildiz, & Naghshineh, 1997), mobile motion prediction (MMP) algorithm (Liu & Maguire, 1996), per-user profile replication scheme (Jannink, Lam, Shivakumar, Widom, & Cox, 1997), hierarchic position-prediction (HPP) algorithm (Liu, Bahl, & Chlamtac, 1998), regular path recognition method (Erbas, Steuer, Kyamakya, Eggesieker, & Jobmann, 2001) and sectorized mobility prediction algorithm (Chellappa Doss, Jennings, & Shenoy, 2004) predict the further location from a user's moving history. In addition, HPP and mobility tracking algorithms (Zaidi & Mark, 2005) consider instantaneous RSSI measurements of surrounding cells. PBNP also considers location classification. Path prediction (Anagnostopoulos, Anagnostopoulos, Hadjiefthymiades, Kalousis, & Kyriakakos, 2007) considers both spatial information, such as moving profile and moving history, and temporal information, such as morning or afternoon. Moving history is one of the important contexts. However, if the moving history is not available, then those approaches are not applicable. Hence, the mobility prediction architecture (Samaan & Karmouch, 2005) performs path prediction based on user context, such as user preferences, goals and spatial conceptual maps (SCM). Additionally, most approaches use high-level contexts in the prediction process. However, Sigg, Haseloff, and David (2006) use low-level context that contain additional information not available with high-level context to improve prediction accuracy. The prediction results of these approaches are useful for many context-aware applications. For instance, if the moving path is obtained, then the handoff process can be prepared such that seamless handoff can be better achieved. However, the high accuracy of preferable or habitual moving path may not be the optimal (best-fitting) path in terms of the user's objective and requirement. Consequently, planning an optimal path (shortest path) based on various contexts that best fit the user requirements is an essential issue.

Genetic algorithm (GA) (Holland, 1975) is a search method for solving large-scale optimization problems. GA is inspired by biological evolution and has proved to be a robust and powerful adaptive search technique. Many researches use GA to solve path finding (Ji, Chen, & Kitti, 2004), path planning (Kanoh, 2007; Castillo & Trujillo, 2006; Kanoh & Hara, 2008) and route selection (Chakraborty, Maeda, & Chakraborty, 2005) problems. Numerical experiments indicate that GA can result in effective searching, shortest path, improved QoS, low system resource consumption and quick decision time. Therefore, the proposed UbiPaPaGo mechanism adopts a GA-based approach to obtain the optimal path.

3. Design of UbiPaPaGo

This section introduces the design goal of UbiPaPaGo. Section 3.2 presents the context model of UbiPaPaGo. Section 3.3 describes the architecture of UbiPaPaGo. Finally, the privacy policy of UbiPaPaGo is introduced in Section 3.4.

3.1. Design goal

This study develops a human-centric, context-aware and intelligent path planning mechanism to satisfy individual user requirements in a ubiquitous network. Accordingly, the following objectives are proposed: (1) The mechanism should be scalable and possible to implement in a distributed manner. (2) Based on various contexts acquired from the environment, the solution should be able to automatically plan an optimal path that best fits the user requirements, such as network connectivity, bandwidth and available services.

UbiPaPaGo relies on various types of context stored in an open and distribute framework (Lu et al., 2009). This framework allows a digital device to join the ubiquitous environment and search for context-aware services based on standard protocols, such as Universal Plug and Play (UPnP) (Universal Plug and Play), Hypertext Transfer Protocol (HTTP) (Fielding et al., 1999) and DNS. A service or application, such as UbiPaPaGo, can be easily deployed into the system through standard protocols and distributed context managers.

3.2. Context model

Table 1 presents the context model of UbiPaPaGo. In a ubiquitous network, billions of sensors located in our physical environment accumulate a huge amount of context. These contexts are fed into context model for context-aware path planning applications. This study limits the description of context needed to support path planning mechanism. The context model includes static and dynamic types describing the user, terminal and map.

The user profile contains details of the user. The user preference allows a user to specify the required service and cost constraints. Meanwhile, the proposed system also considers user privacy. The interface privacy setting enables users to specify their privacy policies, since user privacy is always one of the most important issues in developing context-aware applications. Users can hide sensitive information, such as schedule and location, or filter access to personal context, using their privacy settings.

The terminal profile includes ID, capability, software interface status and types of running application. The terminal capability describes the process speed, memory, screen size, resolution and

Table 1
Context model of UbiPaPaGo.

Static	Dynamic
User	
– User identity	– Schedule
– User preference	– Location
• Required service	– Moving direction
• Cost constrain	– User preferences
• Habit	
– Privacy setting	
Terminal	
– Terminal ID	– Interface status
– Capability	– Running application type
• Process speed	
• Memory	
• Screen size	
• Resolution	
• Supported interface types	
– Application type	
• VoIP, video conference	
• IPTV, mobile TV	
• Email, ftp	
• IM, web browsing	
Map	
– Block ID	– QoS parameter of accessible AP/BS
– Block distance	• RSSI
– Accessible service	• Required bandwidth
• Conversational	• Delay
• Real-time streaming	• Jitter
• Non-real-time streaming	• Packet loss ratio
• Interactive	
• Background	
– QoS parameter of accessible service	
• RSSI	
• Required bandwidth	
• Delay	
• Jitter	
• Packet loss ratio	
– Accessible context server	
– Accessible AP/BS	

supported interface types. The terminal application types describe software applications installed in the terminal. Application types based on quality of service (QoS) parameters, such as response time, delay, jitters and bandwidth, can be categorized into four types, namely (i) conversational service, such as VoIP and video conferencing; (ii) real-time (RT) service, such as Internet Protocol Television (IPTV) and mobile TV; (iii) non-real-time (NRT) services, such as email and ftp; (iv) interactive services, such as instant message (IM) and web browsing.

The map and environment profile includes block ID, block distance, accessibility of AP/BS, QoS parameter of accessible AP/BS, accessible service, QoS parameter of accessibility service, and accessible context server (U-gate). These contexts are represented using an SCM model, which is described in detail in Section 4. The QoS parameters of a service are pre-defined based on service profile and stored in the context server. The QoS of AP/BS is dynamically updated to the context management server.

3.3. Architecture of UbiPaPaGo

Fig. 1 shows the UbiPaPaGo architecture. The UbiPaPaGo servers are composed of a service providing server and a context management server. The service providing server provides the intelligent human-centric services. The context management server mainly acts as a semantic database server to manage context accumulated from the sensors in the environment. The context management server manages and refers to context by constructing an ontology tree based on OWL and RDF. The hand-held device includes a Ubi-

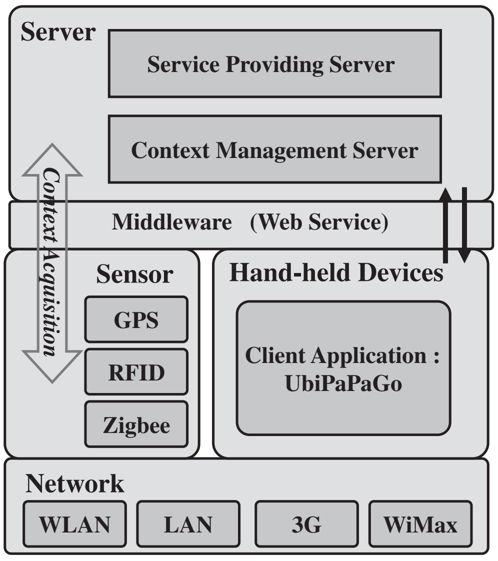


Fig. 1. Architecture of UbiPaPaGo.

PaPaGo client application to find the best-fitting path. The communication model (Lu et al., 2009) among the device and the server is mainly based on the standard protocols, including Universal Plug and Play (UPnP), Simple Service Discovery Protocol (SSDP) (Simple Service Discovery Protocol), Simple Object Access Protocol (SOAP) (Simple Object Access Protocol), General Event Notification Protocol (GENA) (General Event Notification Architecture) and Hypertext Transfer Protocol (HTTP).

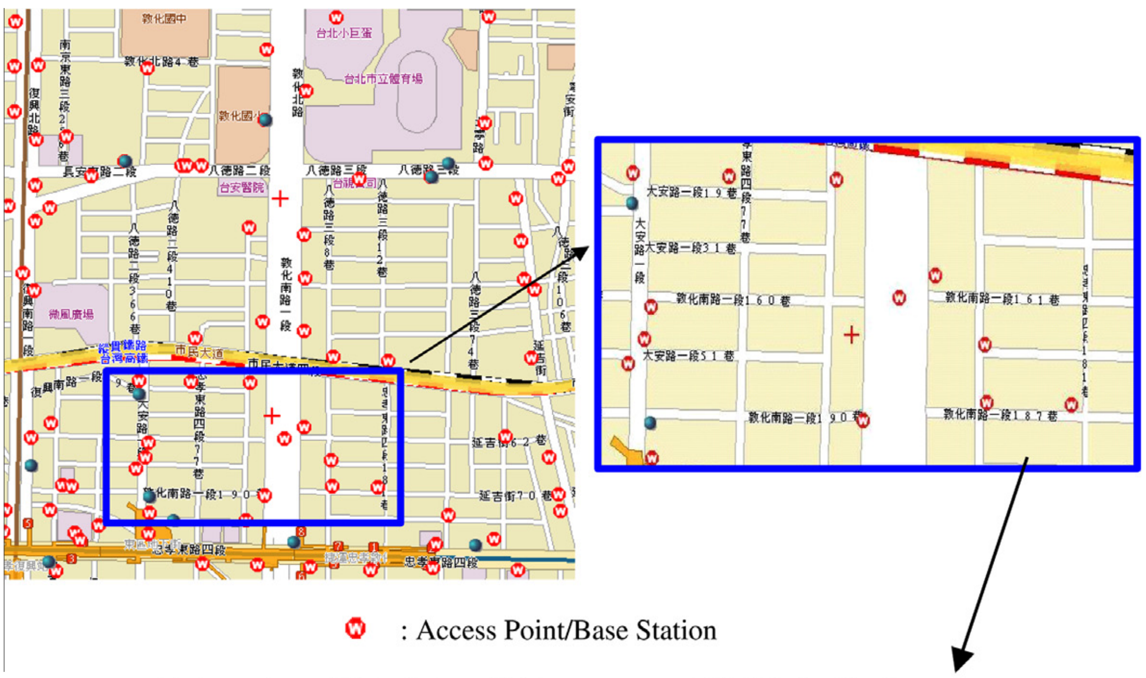
3.4. Privacy policy

UbiPaPaGo relies on various contexts which are highly related to personal privacy. Protecting user privacy is always one of the most important issues in developing context-aware applications. Unfortunately, due to large amount of context information and devices in dynamic and heterogeneous environments, creating a total solution for managing security and privacy policies is known to be a very challenging open issue in ubiquitous environments. In current development, UbiPaPaGo tries to preserve privacy by using a privacy setting to hide sensitive information, such as schedule, location, moving direction and user preference in context management server, and filtering access to personal context based on user's preference and privacy setting. Future versions of UbiPaPaGo will deploy advanced mechanisms or policies for preserving privacy on the context management server.

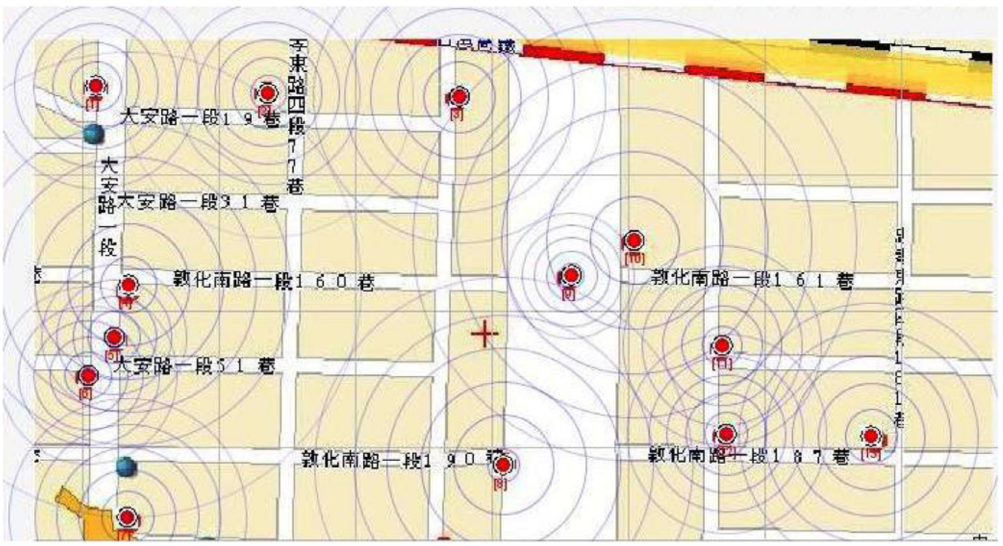
4. UbiPaPaGo based on SCM model

This section represents the trajectory map and environment context using a SCM model. The SCM is an abstraction of real map representation containing of a set of landmark objects (O_j), a set of medium objects (W_y) and the influence areas of those objects. The landmark objects define those areas, such as buildings or user's target destination. The medium objects (also called "ways") define areas, such as streets, roads, trajectories and virtual connections between objects (Kettani & Moulin, 1999). A way object is further partitioned into continuous blocks, called Way Elementary Areas (WEAs).

To encode additional context information, a novel approach for partitioning a way object is proposed as follows. A WEA may be a landmark object (O_j) or a crossing point of two or more ways (Kettani & Moulin, 1999). The proposed solution requires additional contexts, including received signal strength indication (RSSI) of



(a) A portion of Taipei city of Taiwan map and AP/BS distribution



(b) Context of AP/BS distribution of map

Fig. 2. A portion of Taipei City of Taiwan map.

AP/BS and available bandwidth. Thus, besides partitioning a way object based on crossings, the proposed solution also considers characteristics of AP/BS, such as RSSI. Specifically, a way object is partitioned into several continuous blocks (WEAs) such that the RSSI of an AP/BS within each WEA is roughly at the same level. Fig. 2(a) illustrates a portion of the map of Taipei city in Taiwan, and the distribution of AP/BS (Wifly). To clearly show the distribution and the context of AP/BS, the blue[1] circle in Fig. 2(a) shows an enlarged portion of the map. In this study, a series of simulations were performed using the Qualnet network simulator (Qualnet) to implement the context of the AP/BS, as shown in Fig. 2(b). Fig. 3 shows its SCM representation.

[1] For interpretation of color in Figs. 2, 3, 10–14, the reader is referred to the web version of this article.

The Matrix of Orientation and Adjacency and Characteristic (MOAC) (Samaan & Karmouch, 2005) was adopted to represent the WEAs of the SCM. The MOAC contains characteristic information of each WEA and landmark object, and the relative information and displacement direction between two WEAs. Fig. 4 shows a portion of the MOAC of Fig. 3. The diagonal cell (x, x) represents the characteristics, i.e. context of WEA α_x. The cell (x, y), where $x \neq y$ denotes the characteristics of relative direction between α_x and α_y, which can be represented in eight-way directions, namely, east (E), west (W), south (S), north (N), northeast (NE), southeast (SE), southwest (SW) and northwest (NW). For instance, α_{19} is on the south (S) side of α_{18}, and α_3 is on the northwest side (NW) of α_{18}. According to MOAC, the list of blocks can be inferred along a moving path, along with the direction of movement and all required context information.

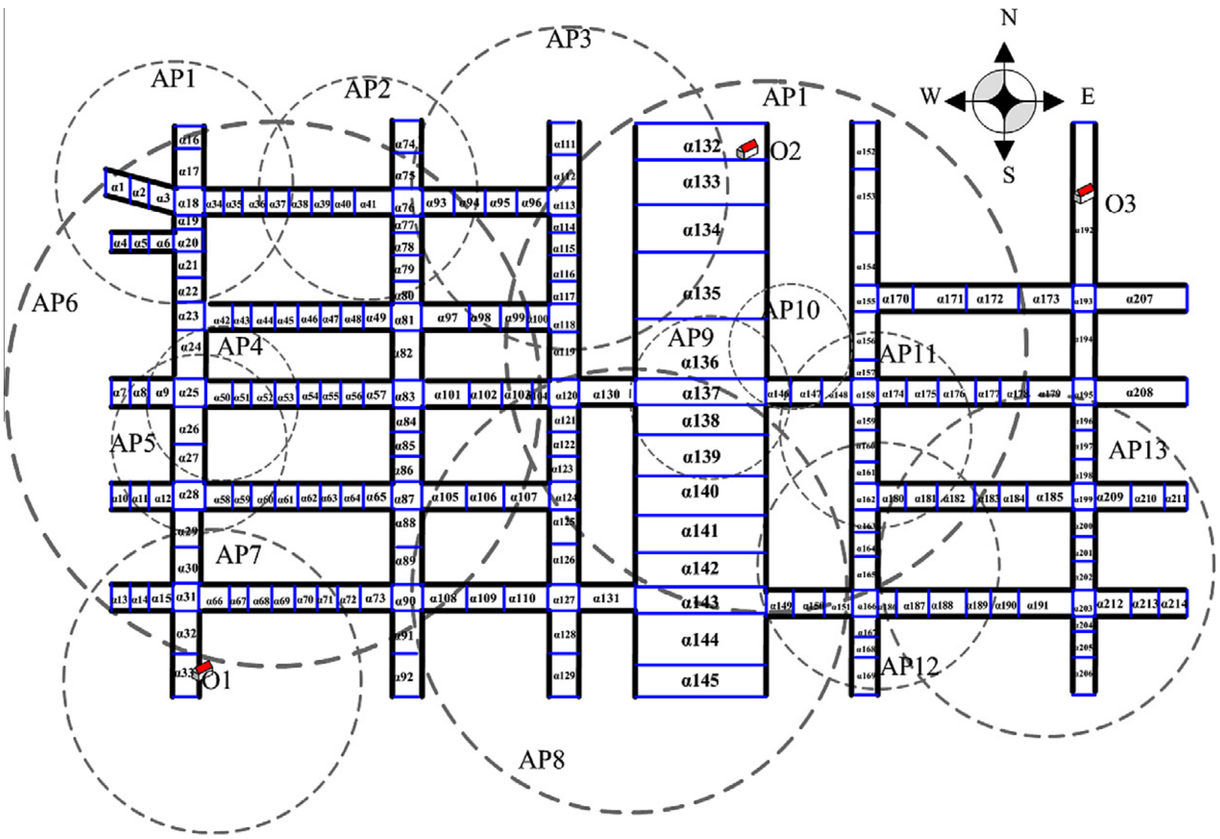


Fig. 3. SCM representation of Taipei City of Taiwan map of Fig. 2.

	α_2	α_3	. . .	α_{18}	α_{19}	. . .	α_{132}	α_{133}	α_{134}
α_2	Co(α_2) Co(AP_1) Co(AP_6)	SE							
α_3	NW	Co(α_3) Co(AP_1) Co(AP_6)							
. . .									
α_{18}				Co(α_{18}) Co(AP_1) Co(AP_6)	S				
α_{19}				N	Co(α_{19}) Co(AP_1) Co(AP_6)				
. . .									
α_{132}							Co(α_{132}) Co(O_2) Co(AP_1) Co(AP_3)	S	
α_{133}							N	Co(α_{133}) Co(AP_1) Co(AP_3)	S
α_{134}								N	Co(α_{134}) Co(AP_1) Co(AP_3)

Fig. 4. A portion of MOAC of Fig. 3.

The characteristic function *Co*( ) describes several characteristics of an object. Specifically, $Co(O_j)$ and $Co(\alpha_i)$ describe the context information of a landmark object and a WEA, respectively. Moreover, since state information of APs is also considered in this study, each AP is associated with a characteristic function $Co(AP_i)$. For instance, if O_1 is a rapid metro station, then the characteristic function denotes the characteristics in terms of name, location. Thus an example of $Co(O_1)$ could be $\{c_1$ = rapid transit station, $c_2 = \alpha_{33}\}$. Similarly, if WEA α_{132} is covered by two APs and provides a service, then the characteristic function of $Co(\alpha_{132})$ is denoted by $\{c_1 = \{AP_1, AP_3\}, c_2$ = {streaming server}}. The characteristics of AP are described in terms of available RSSI (c_1), bandwidth (c_2), delay (c_3), jitter (c_4), and packet loss ratio (c_5). An example of $Co(AP_1)$ is $\{c_1 = -70$ dbm, $c_2 = 11$ Mbps, $c_3 = 30$ ms, $c_4 = 10$ ms, $c_5 = 5\%\}$.

5. UbiPaPaGo based on GA

This section describes our context-aware path planning mechanism based on GA. The GA procedure is shown in Fig. 5. Before applying GA, the SCM model obtained in Section 4 is first converted to path planning map, as shown in Fig. 6, where node α_i denotes a crossing in Fig. 3. Restated, to reduce the length of chromosome and to increase the scalability of UbiPaPaGo, the path planning map considers only crossing nodes. Additionally, the path planning map needs to consider the source or destination node, if these are not at a crossing. However, the fitness function of a chromosome is still calculated according to all WEAs and their characteristic functions, as described later.

The path planning map is an undirected graph $G = (V, E)$, where *V* denotes a set of crossing, source and destination nodes, and *E* denotes a set of edges (WEAs). The aim is to identify the shortest path for *G*. Each edge $e \in E$ includes blocks(α_i). Corresponding to each edge, a weight ω_{ij} denotes the fitness value calculated from the fitness function from node α_i to node α_j. Moreover, if the user's required service is assumed to be located at the edge, then this edge is called a constraint edge $e' \in E$. The found path $\vec{S}$ must contain this edge, so that the required service can be accessed. For instance, if the streaming services are located at block α_{60} and α_{46}, respectively, then the links (28, 87) and (23, 81) are constraint edges that must be included in the path. Note that end-to-end

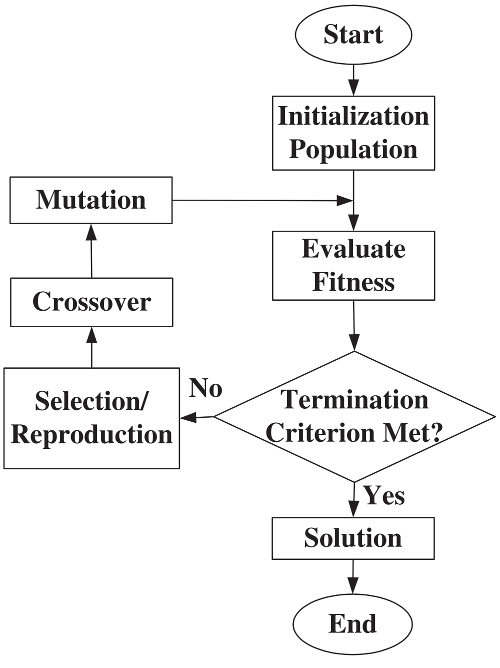


Fig. 5. The procedure of GA.

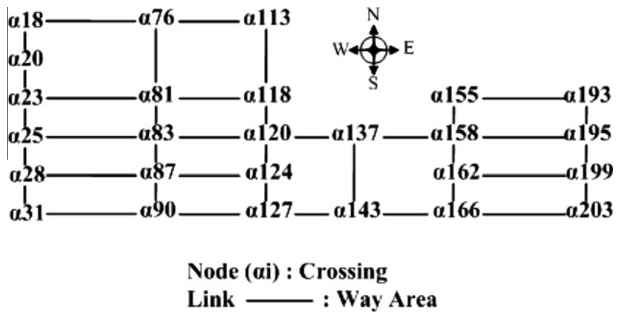


Fig. 6. Path planning map.

QoS attributes are evaluated, e.g., the delay represents the end-to-end delay of a path.

The UbiPaPaGo defines the optimal path as the shortest path subject to seven constraints, including received signal strength indication (RSSI), available bandwidth (BW), response time (RT), delivery latency (DL), jitter (J) and packet loss ratio (PL), available service (AS). The optimization problem is formulated as follows:

$$\text{minimize} \quad D(\vec{S}) = \sum_{k \in \vec{S}} dist(e_k) \tag{1}$$

subject to

$$\forall\ \text{link}(e_k) \in \vec{S}:$$

$$x(e_k) \geqslant \theta_x, \quad X = \{\text{RSSI}, \text{BW}\},\ x \in X \tag{2}$$

$$\sum_{k \in S} y(e_k) \leqslant \theta_y, \quad Y = \{\text{RT}, \text{DL}, \text{J}, \text{PL}\}, y \in Y \tag{3}$$

$$\text{AS}_{\vec{S}} \quad \text{exists required service} \tag{4}$$

where $D(\vec{S})$ denotes the total distance of moving path, $dist(e_k)$ denotes the distance of edge e_k, θ_x and θ_y are thresholds for RSSI, RT, etc. The constraints (2)–(4) ensure that the computed result indeed satisfies the user's requirements.

5.1. Chromosome representation and population initialization

Chromosome representation is a key to the design of GA for solving a problem. This study adopts the encoding method proposed in Wan et al. (2003), where all chromosomes have the same length, thus facilitating crossover operation. Fig. 7 gives an example of chromosome representation, where the upper sequence is the node id and the lower sequence is the chromosome. Here a chromosome is represented by a sequence of integer numbers; a gene denotes the node id through which it passes. For an edge (v_i, v_j) from node v_i to node v_j, node v_j is put in the *i*th id of the chromosome. If the id is not along the path, then its corresponding gene is filled with a neighboring node randomly chosen from the set of its neighboring nodes. For instance, the chromosome in Fig. 7 denotes a path from source node id 28 (block α_{28}) to destination node id 113 (block α_{113}), passing through α_{87}, α_{83}, α_{120} and α_{118}.

The initial population influences the convergence speed and the quality of the solution. Random and heuristic initializations are the most commonly used ways of generating candidate solutions. However, random initialization possibly causes a large computing overhead and loop path, while heuristic initialization (Arikan et al., 2001) has difficulty in finding a global optimal. Heuristic initialization explores a small part of solution space, making it faster than random initialization. This study adopts the probability network approach (Poon & Chan, 2000) that combines the features of random and heuristic initialization to generate initial candidate paths in the gene pool. Starting with the source node, the next node is chosen from its directly connected neighbors. To speed up the convergence speed, each neighboring node is assigned a different selection probability. A neighboring node located in the direction

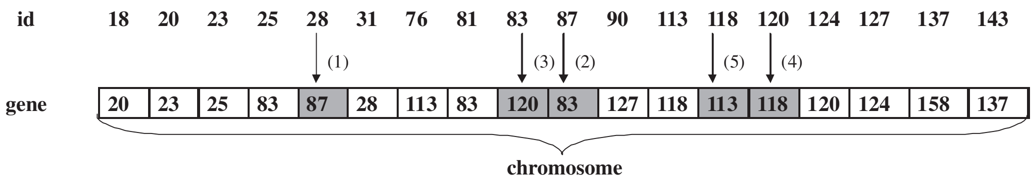


Fig. 7. Example of encoding path from α_{28} to α_{113}.

toward the destination is assigned a higher probability. The selected next node then becomes the gene of the source node's id. This procedure repeats until the path reaches the destination node. To avoid looping, a node is forbidden to visit twice in the same path during the next node selection procedure. Considering the example shown in Fig. 7, α_{87} has four neighboring nodes, α_{28}, α_{83}, α_{90} and α_{124} (cf. Fig. 6), among which α_{83} and α_{124} have higher probabilities to be chosen as the next node of α_{87}, since they are located in the direction toward α_{113} and have not appeared in the path before. For nodes that are not in the path under consideration, their next nodes are randomly selected. For instance, α_{31}, α_{87} and α_{127} can be chosen as the next node of α_{90}.

5.2. Fitness function

The fitness function measures the distance of the path that satisfies all the quality constraints. The concept of penalty function is adopted to handle the constraints (Yeniay, 2005). In this study, the quality of a chromosome depends on properties of each link(e_k) of the path, including received signal strength indication (RSSI), available bandwidth (B), response time (RT), delivery latency (DL), jitter (J), packet loss (PL) and available service (AS). The fitness function is defined by

$$F = D(\vec{S}) \cdot (1 + \alpha P) \tag{5}$$

with

$$P = \sum_{k \in \vec{S}} p(e_k) + h_{AS}(\vec{S}) \tag{6}$$

where

$$p(e_k) = \sum_{x \in \{\text{RSSI},\text{BW}\}} f_x(e_k) + \sum_{y \in \{\text{RT},\text{DL},\text{J},\text{PL}\}} g_y(e_k) \tag{7}$$

$$f_x(e_k) = \begin{cases} 1, & \text{if } x(e_k) \leqslant \theta_x \\ 0, & \text{otherwise} \end{cases} \quad x \in \{\text{RSSI},\text{BW}\} \tag{8}$$

$$g_y(e_k) = \begin{cases} 1, & \text{if } \sum_{k \in S} y(e_k) \geqslant \theta_y \\ 0, & \text{otherwise} \end{cases} \quad y \in \{\text{RT},\text{DL},\text{J},\text{PL}\} \tag{9}$$

$$h_{AS}(\vec{S}) = \begin{cases} 1, & \text{if required service is not available along path} \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

5.3. Selection

The selection operation chooses a chromosome from the current population for reproduction. The UbiPaPaGo adopts tournament selection (Goldberg & Deb, 1991) to select parents for producing offspring. In tournament selection, two individuals are chosen at random from the population, and the one with the higher fitness value is chosen as a parent for the crossover operation. The tournament process repeats until two parents are selected.

5.4. Crossover

The crossover operation recombines two chromosomes (called parents) to produce a new chromosome (called offspring). This study proposes a modified uniform crossover by additionally considering genes (nodes) that have not appeared (been visited) before. The proposed crossover operation is carefully designed to avoid the path looping problem. The crossover starts with the gene at the id of the source node. If the two parental genes at this id are identical, then this gene is simply copied to the offspring. Otherwise, the gene that has not appeared before is chosen. A tie is broken by random selection. After selecting the gene of the current id, the procedure moves to the id of the next node (i.e., the gene we just selected), and is repeated until the destination node is reached as the gene of the current id.

Fig. 8 shows an example of the crossover operation. The path in parent 1 is $\alpha_{28} \to \alpha_{87} \to \alpha_{83} \to \alpha_{120} \to \alpha_{118} \to \alpha_{113}$, and that of parent 2 is $\alpha_{28} \to \alpha_{25} \to \alpha_{23} \to \alpha_{81} \to \alpha_{76} \to \alpha_{113}$. For the node id 87, the operation randomly selects among α_{83} and α_{90}. If α_{90} is selected, then node id 90 is considered. The operation is then repeated until the destination node is reached. The new offspring $\alpha_{28} \to \alpha_{25} \to \alpha_{83} \to \alpha_{87} \to \alpha_{90} \to \alpha_{127} \to \alpha_{124} \to \alpha_{120} \to \alpha_{118} \to \alpha_{113}$ is generated after the crossover operation. For the id's in which the path does not pass through, e.g. 20, the genes at them are randomly selected from either parent, for example, α_{18} in node id 20.

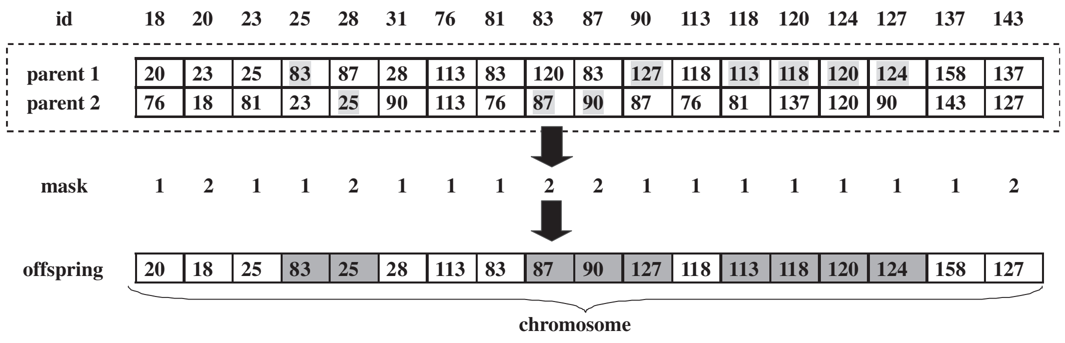


Fig. 8. An example of crossover operation.

id	18	20	23	25	28	31	76	81	83	87	90	113	118	120	124	127	137	143
before	20	23	25	83	87	28	113	83	120	83	127	118	113	118	120	124	158	137
after	20	23	25	83	87	28	113	83	120	90	127	118	113	118	120	124	158	137

Fig. 9. An example of mutation operator.

5.5. Mutation

With a given probability, mutation introduces variation into the chromosome by adjusting partial chromosome to avoid local optima. In this study, a random node (mutation node) that is neither a source nor a destination node along a path is chosen for mutation (Wan et al., 2003). The new gene value is randomly chosen from directly connected nodes of the mutation node. Fig. 9 shows an example of mutation operation. The old path is $\alpha_{28} \to \alpha_{87} \to \alpha_{83} \to \alpha_{120} \to \alpha_{118} \to \alpha_{113}$. In the mutation operation, the node id α_{87} is randomly chosen as the mutation gene. The new path following the mutation operation is $\alpha_{28} \to \alpha_{87} \to \alpha_{90} \to \alpha_{127} \to \alpha_{124} \to \alpha_{120} \to \alpha_{118} \to \alpha_{113}$.

After crossover and mutation operations are completed, the offsprings are evaluated by the fitness function for another round of reproduction until a termination condition is satisfied.

6. Prototyping and implementation

This section demonstrates the feasibility of the UbiPaPaGo by prototyping a real system in U-gate. Because U-gate is an open, efficient and distributed context management architecture and communication model based on standard protocols, UbiPaPaGo can be implemented in a distributed manner, enabling any service provider to deploy its service into the ubiquitous environment easily. For easy deployment, UbiPaPaGo was implemented in a small scope environment. Therefore, the demonstration scenario is in our campus (Chung-Cheng University), as displayed in Fig. 10, and made up with user using EeePC with WiFi and RFID. In the scenario, the user uses UbiPaPaGo service in EeePC to connect UbiPaPaGo service provider (U-gate) in local area, as shown in Fig. 11(a) and (b), to plan a path from the college of engineering (O_9) to the auditorium (O_8) in the CCU campus. U-Gate collects the contexts of user's current location, requirement, ID, schedule and SCM map to infer the path planning result, as shown in Fig. 11(c). Readers are referred to Lu et al. (2009) for the detailed implementation of U-gate.

7. Experimental results and discussion

This section evaluates the proposed UbiPaPaGo through computer simulation. The proposed UbiPaPaGo was implemented in a Java (JDK 6) program on a PC with a Core 2 Quad 2.4 GHZ CPU and 2G MB memory. Fig. 12 shows the road map of some part of Taipei city that was used in the experiments. In the experiment, the road map with 1481 nodes was first converted into a graph with 582 crossing nodes.

The parameters are set as follows. The experiment tries five population sizes (N): 100, 200, 300, 400, and 500. Termination criterion is 500 generation. The crossover probability (P_c) was set to 1, and the mutation probability (P_m) was set to 0.05. The tournament size was set to 2. The alpha (α) value of the penalty function weight was empirically set as $\alpha = 2$. The collected results were averaged over 100 runs.

Notably, UbiPaPaGo uses probability network based initialization and a modified crossover. To verify the effect of these two elements, we compare the performance of UbiPaPaGo with that of GA, i.e. using random initialization and uniform crossover. The performance metric, mean best fitness (MBF), is defined by

$$\mathrm{MBF} = \frac{\text{sum of the best fitness value of each run}}{\text{number of runs}} \tag{11}$$

7.1. Optimal path evaluation

Fig. 13(a)–(c) shows the best obtained path, the optimal path, and the shortest path for the indicated source and destination. The result of the proposed UbiPaPaGo in Fig. 13(a) coincides with the optimal path in Fig. 13(b) and the shortest path in Fig. 13(c). Table 2 summarizes the results of QoS constraint and parameters of the obtained path in Fig. 13(a)–(c). The QoS parameters of the resulting path means the minimum RSSI and bandwidth and maximum delay, jitter and packet loss rate along the path. The best obtained path of UbiPaPaGo guarantees shorter path and QoS to

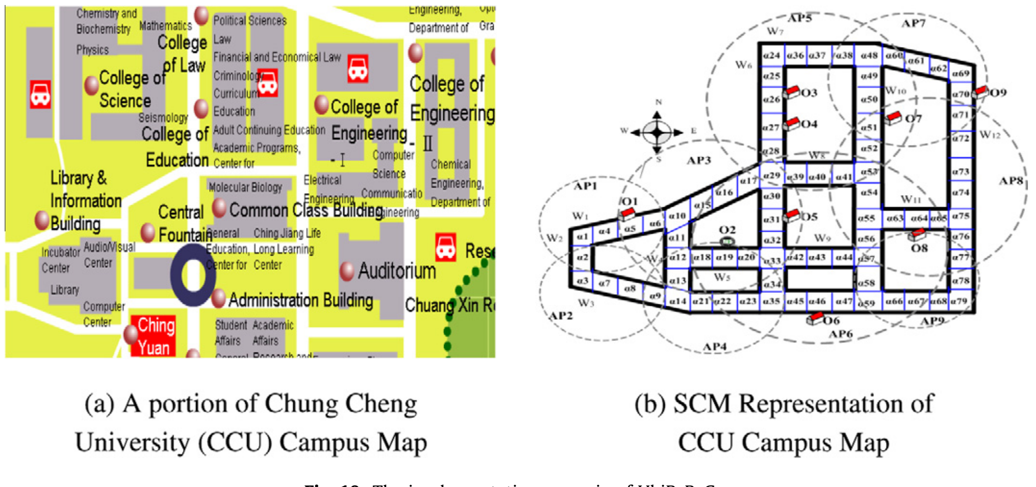
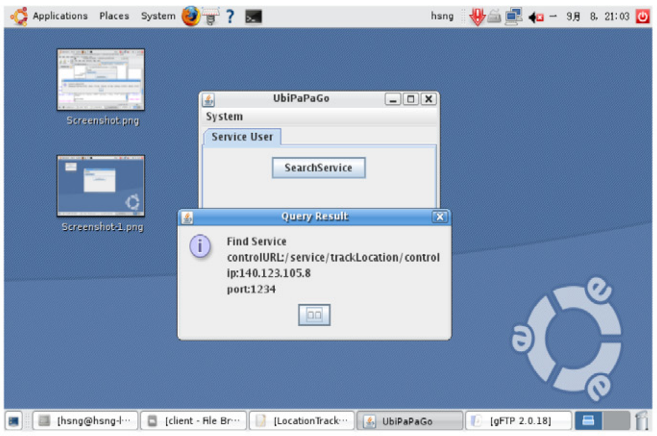
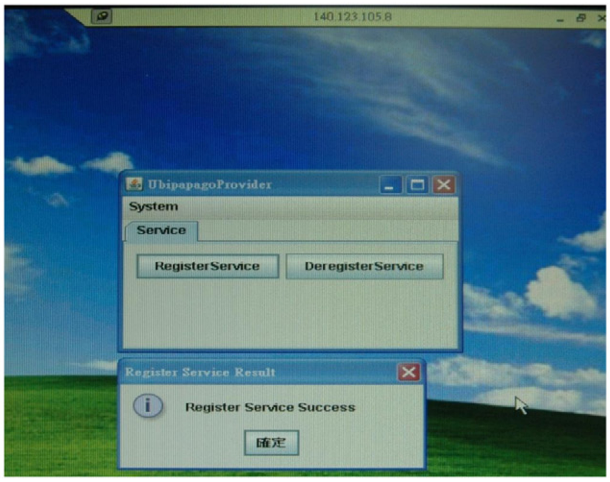


Fig. 10. The implementation scenario of UbiPaPaGo.



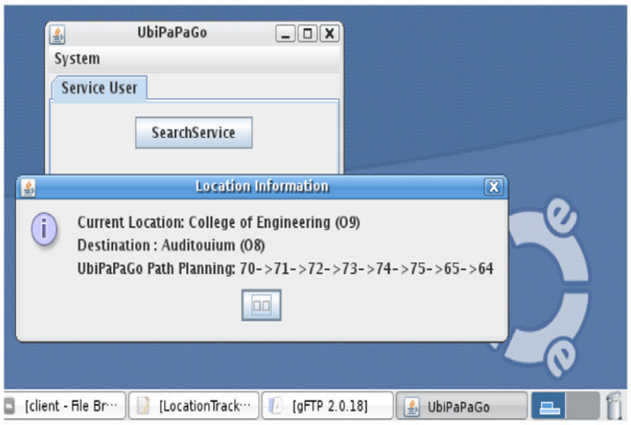
Find Service
controlURL:/service/trackLocation/control
ip:140.123.105.8
port:1234

(a) User connects to Service Provider and uses UbiPaPaGo Service (Service Provider IP:140.123.105.8)



Register Service Success

(b) UbiPaPaGo Service Provider receives the request from user



Current Location: College of Engineering (09)
Destination : Auditouium (08)
UbiPaPaGo Path Planning: 70->71->72->73->74->75->65->64

(c) The path planning result

Fig. 11. The demonstration of UbiPaPaGo.

satisfy user's requirements, while the shortest path has shortest distance but does not satisfy user's requirements.

7.2. Convergence behavior

Fig. 14 compares the convergence behavior of the UbiPaPaGo and GA. The result is helpful for choosing the appropriate population size and termination condition. Moreover, it shows that a small population size reduces evaluation cost but results in premature convergence. On the other hand, a large population size finds better solutions and yet requires longer computation time. We can also observe that UbiPaPaGo converges faster than GA. According to the result in Fig. 14, the best convergence curve of UbiPaPaGo was achieved with population size of 300 in 100 generations.

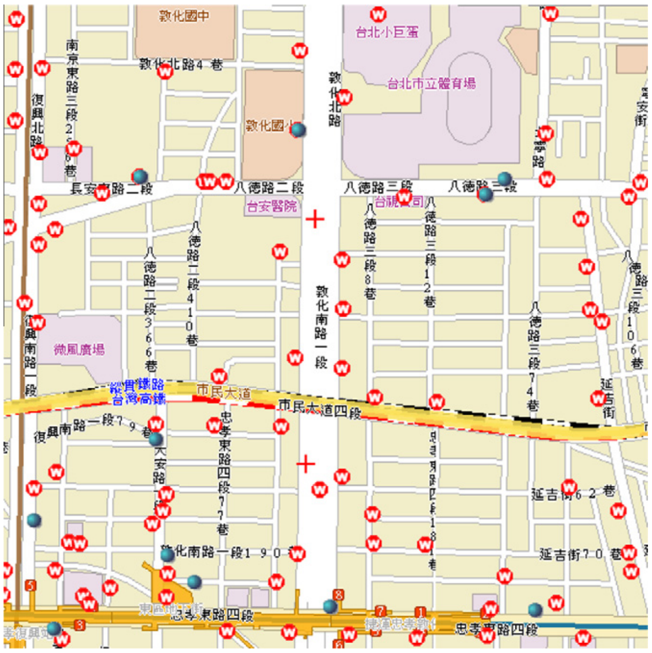


Fig. 12. The portion of Taipei City.

Therefore, unless otherwise stated, the following experiments of UbiPaPaGo were performed with population size 300 and 100 generations.

The GA differs from UbiPaPaGo by adopting a random approach in the initialization and crossover operations, such that some chromosomes may contain loops. Hence, the UbiPaPaGo converges faster than the GA. The results indicate that the UbiPaPaGo improves the convergence speed by preventing looping in the initialization and crossover operations.

7.3. Optimization results comparison

Table 3 compares the performance of the UbiPaPaGo and GA in terms of MBF, average computation time, probability of obtaining the best path. Note that the GA uses random initialization and uniform crossover, while UbiPaPaGo uses probability network approach initialization and modified uniform crossover. The MBF values indicate that UbiPaPaGo generally outperforms the GA in terms of fitness.

The effectiveness of the UbiPaPaGo was measured under various numbers of generations. The UbiPaPaGo was found to have a shorter computation time and significantly higher success rate than the GA, because UbiPaPaGo is a probability network approach in which the next node is located forward of the destination with a higher probability. Additionally, UbiPaPaGo avoids looping. The probability of obtaining the optimal solution is higher than 93% with a population size of 300. This approach is also efficient, because the population size is small and the evolution process is short. The result is very important and applicable for context-aware applications. Short computation time and high success probability enable UbiPaPaGo to achieve satisfactory solution quality and real time service. Table 3 also shows that UbiPaPaGo has lower standard deviations than GA does.

7.4. Effect of alpha value

Finally, Table 4 compares the fitness, distance, and penalty values at different alpha values ranging from 0.5 to 2.0. Note that population size 100 with 500 generations are adopted herein. The table shows that the increaseing alpha value causes decreasing penalty value, increasing average fitness, and increasing distance value. This is because a high alpha value intensifies penalty so that the evolutionary search tends to avoid penalty and to find a solution without penalty but with ordinary distance. Conversely, a low alpha encourages finding a short route that may violate some constraints.

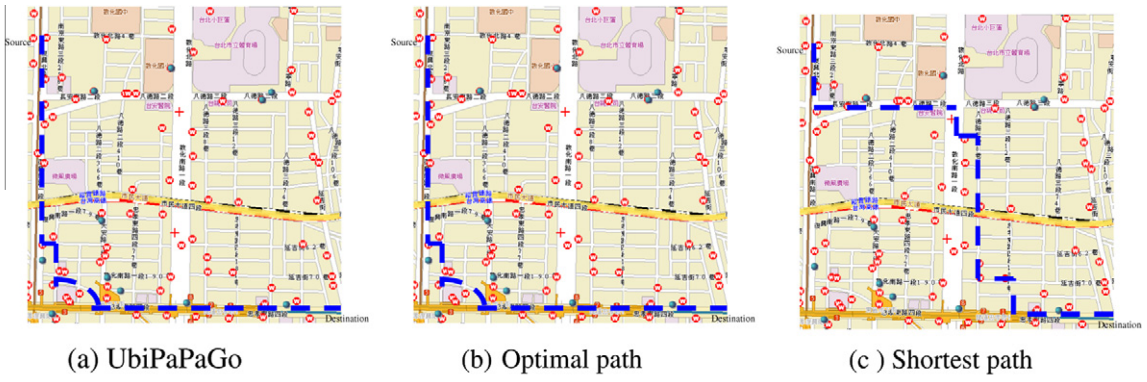


Fig. 13. Path comparison (paths are denoted by blue dotted lines).

Table 2
The QoS constraint and result of found path, optimal path and shortest path.

Parameters	Distance	Fitness value	Min RSSI	Min bandwidth	Max delay	Max jitter	Max packet loss rate	Available service
Range	N/A	N/A	−40 to −100	2–54	10–70	10–50	0.02–0.2	0.1
Threshold	N/A	N/A	⩾−60	⩾30	⩽40	⩽15	⩽0.05	=1
Best obtained path of UbiPaPaGo	124	124	−60	40	10	10	0.03	1
Optimal path	124	124	−60	40	10	10	0.03	1
Shortest path	123	327	−60	2[a]	70[a]	30[a]	0.2[a]	0[a]

[a] The QoS parameter is not satisfied user's requirement.

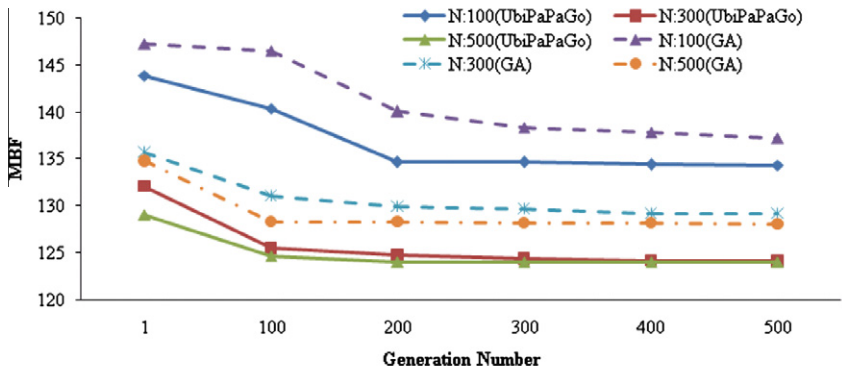


Fig. 14. Convergence curve of UbiPaPaGo and GA.

Table 3
The QoS constrain and result of found path.

Generation number	Fitness			Average computation time (s)	Probability of obtaining the optimal path
	Best	MBF	Std		
UbiPaPaGo					
N = 1	124	132.07	6.26	2.67	0.05
N = 100	124	125.48	1.86	7.18	0.34
N = 200	124	124.74	1.8	9.49	0.73
N = 300	124	124.34	1.34	11.64	0.93
N = 400	124	124.13	0.48	13.73	0.95
N = 500	124	124.1	0.46	15.96	0.95
GA					
N = 1	124	135.73	5.8	15.47	0.01
N = 100	124	131.1	5.27	20.84	0.06
N = 200	124	129.9	4.14	23.24	0.25
N = 300	124	129.66	4.42	25.85	0.43
N = 400	124	129.2	2.9	28.01	0.43
N = 500	124	129.2	3.12	30.47	0.44

Table 4
Fitness, distance and penalty for alpha values 0.5, 1.0, 1.5, and 2.0.

Alpha (α)	Parameters		
	Average fitness (F)	Average distance ($D(\vec{S})$)	Average penalty (P)
0.5	129.4	126.83	5.14
1	131.06	127.59	3.47
1.5	131.17	127.79	2.25
2.0	132.56	128.57	2.00

8. Conclusions and future work

This study proposes an intelligent navigation application, UbiPaPaGo, which focuses on providing human-centric context-aware path planning mechanism in a ubiquitous network. Specifically, UbiPaPaGo considers requirements and contexts of users and environment when planning the best-fitting path. UbiPaPaGo adopts the SCM model to present map context and uses GA to find the optimal path. The map simplifies conversion to graph, thus reducing the computational complexity and increasing scalability. Experimental results examine the effect of initialization and proposed crossover. Moreover, the results validate that the proposed UbiPaPaGo can efficiently achieve the shortest path that guarantees all QoS parameters.

Some issues of the proposed UbiPaPaGo remain for further research. This study assumes that the destination is inferred from the user's personal context, such as schedule. If the destination of the path is uncertain or unknown, predicting the destination from the related context becomes an *a priori* problem. Additionally, the proposed system only considers how to plan an optimal path for a user but does not consider issues, such as vertical handoff between AP and BS, and resource reservation along the path in advance to guarantee quality of service of the path. Finally, security and privacy issues also need further investigation. Advanced framework or policies for preserving privacy will be deployed in the future.

Acknowledgment

The research was supported by the NSC97-2221-E-194-011-MY3 and NSC97-2221-E-194-012-MY3, National Science Council, ROC.

References

Anagnostopoulos, T., Anagnostopoulos, C., Hadjiefthymiades, S., Kalousis, A., & Kyriakakos, M. (2007). Path prediction through data mining. In *IEEE international conference on pervasive services (ICPS), Istanbul, Turkey* (pp. 128–135).

Arikan, O., Chenney, S., & Forsyth, D. A. (2001). *Efficient multi-agent path planning*. In *Eurographic workshop on computer animation and simulation, Manchester, UK* (pp. 151–162).

Bharghavan, V., & Jayanth, M. (1997). Profile-based next-cell prediction in indoor wireless LAN. In *IEEE Singapore international conference*.

Bodhale, D., Afzulpurkar, N., & Thanh, N. T. (2009). Path planning for a mobile robot in a dynamic environment. In *IEEE international conference on robotics and biomimetics* (pp. 2115–2120).

Castilho, O., & Trujilo, L. (2005). Multiple objective optimization genetic algorithms for path planning in autonomous mobile robots. *International Journal of Computers, Systems and Signals, 6*(1), 48–63.

Castillo, O., & Trujillo, L. (2006). Patricia Melin, multiple objective genetic algorithms for path-planning optimization in autonomous mobile robots. *Soft Computing – A Fusion of Foundations, Methodologies and Applications, 11*(3), 269–279.

Chakraborty, B., Maeda, T., & Chakraborty, G. (2005). Multiobjective route selection for car navigation system using genetic algorithm. In *IEEE mid-summer workshop on soft computing in industrial application (SMXia/05)* (pp. 190–195).

Chellappa Doss, R., Jennings, A., & Shenoy, N. (2004). Mobility prediction for seamless mobility in wireless networks. In *4th international network conference (INC)* (pp. 435–443).

Erbas, F., Steuer, J., Kyamakya, K., Eggesieker, D., & Jobmann, K. (2001). A regular path recognition method and prediction of user movements in wireless networks. In *IEEE vehicular technology conference, Atlantic City* (pp. 2672–2676).

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., et al. (1999). Hypertext Transfer Protocol – HTTP/1.1, IETF RFC 2616, June 1999.

General Event Notification Architecture (GENA). <http://www.tools.ietf.org/html/draft-cohen-gena-p-base-01>.

Goldberg, D. E., & Deb, K. (1991). *A comparative analysis of selection schemes used in genetic algorithms. Foundation of genetic algorithms* (pp. 69–93). San Mateo Calif: Morgan Kaufman.

Holland, J. (1975). *Adaptation in natural and artificial systems*. The University of Michigan Press.

Inagaki, J., Haseyama, M., & Kitajima, H. (1999). A genetic algorithm for determining multiple routes and its applications. In *IEEE international symposium (ISCAS'99)* (pp. 137–140).

Jannink, J., Lam, D., Shivakumar, N., Widom, J., & Cox, D. (1997). Efficient and flexible location management techniques for wireless communication system. *ACM/Baltzer Wireless Networks, 3*(5), 361–374.

Ji, Z. W., Chen, A., & Kitti, S. (2004). Finding multi-objective paths in stochastic networks: A simulation-based genetic algorithm approach. In *Congress on evolution computation (CEC 2004)* (pp. 174–180). Los Alamitos: IEEE Press.

Kanoh, H. (2007). Dynamic route planning for car navigation systems using virus genetic algorithms. *International Journal of Knowledge-Based and Intelligent Engineering Systems, 11*(1), 65–78.

Kanoh, H., & Hara, K. (2008). Hybrid genetic algorithm for dynamic multi-objective route planning with predicted traffic in a real-world road network. In *Genetic and evolutionary computation conference (GECCO 2008)* (pp. 657–664).

Kettani, D., & Moulin, B. (1999). A spatial model based on the notions of spatial conceptual map and of object's influence areas. In *Conference spatial information theory (COSIT)* (pp. 401–416).

Lei, L., Wang, H., & Wu, Q. (2006). Improved genetic algorithms based path planning of mobile robot under dynamic unknown environment. In *IEEE international conference on mechatronics and automation* (pp. 1728–1732).

Levine, D., Akyildiz, I., & Naghshineh, M. (1997). A resource estimation and call admission algorithm for wireless multimedia networks using the shadow cluster concept. *IEEE/ACM Transactions on Networking, 5*(1), 1–12.

Li, Q., Liu, G., Zhang, W., Zhao, C., Yin, Y., & Wang, Z. (2006). A specific genetic algorithm for optimum path planning in intelligent transportation system. In *6th international conference on ITS telecommunications* (pp. 140–143).

Li, Q., Zhang, W., Yin, Y., Zhang, W., & Liu, G. (2006). An improved genetic algorithm of optimum path planning for mobile robots. In *The sixth international conference on intelligent systems design and applications* (pp. 637–642).

Liu, T., Bahl, P., & Chlamtac, I. (1998). Mobility modeling, location tracking, and trajectory prediction in wireless ATM networks. *IEEE JAC, 16*(6), 922–936.

Liu, Z., Lu, Q., Yang, P., & Chen, L. (2008). Path planning for tractor-trailer mobile robot system based on equivalent size. In *7th world congress on intelligent control and automation (WCICA)* (pp. 5744–5749).

Liu, G., & Maguire, G. (1996). A class of mobile motion prediction algorithms for wireless mobile computing and communications. *ACM/Baltzer MONET, 1*(2), 113–121.

Lu, J. H., Wang, C. Y., & Hwang, R. H. (2009). An open framework for distributed context management in ubiquitous environment. In *International symposium on UbiCom frontiers – Innovative research, systems and technologies (Ufirst-09), Brisbane, Australia* (pp. 88–93).

OWL. W3C document. <http://www.w3.org/2004/OWL/>.

PAPAGO. <http://www.papago.com.tw/>.

Poon, W. T., & Chan, E. (2000). Traffic management in wireless ATM network using a hierarchical neural-network based predication algorithm. In *15th international conference on computers and their applications, New Orleans, United States* (pp. 5–8).

Qualnet, Scalable Network Technologies Inc. <http://www.scalable-networks.com>.

RDF Primer. W3C document. <http://www.w3.org/TR/rdf-primer/>.

RDF Schema. W3C document. <http://www.w3.org/TR/rdf-schema/>.

Samaan, N., & Karmouch, A. (2005). A mobility prediction architecture based on contextual knowledge and spatial conceptual maps. *IEEE Transactions on Mobile Computing, 4*(6), 537–551.

Sigg, S., Haseloff, S., & David, K. (2006). A novel approach to context prediction in ubicomp environments. In *17th Annual IEEE international symposium on personal, indoor and mobile radio communications (PIMRC'06)* (pp. 1–5).

Simple Object Access Protocol (SOAP) version 1.2. <http://www.w3.org/TR/soap/>.

Simple Service Discovery Protocol (SSDP). <ftp://ftp.pwg.org/pub/pwg/ipp/new_SSDP/draft-cai-ssdp-v1-03.txt>.

TomTom International. <http://www.tomtom.com/>.

Tu, J., & Yang, S. (2003). Genetic algorithm based path planning for a mobile robot. In *IEEE international conference on robotics and automation, Taipei* (pp. 1221–1226).

Universal Plug and Play (UPnP). <http://www.upnp.org/>.

Wan, T. R., Chen, H., & Earnshaw, R. (2003). Real-time path planning for navigation in unknown environment. In *Theory and practice of computer graphics* (pp. 138–145).

Weiser, M. (1991). The computer for the twenty-first century. *Scientific American*, 94–104.

Wifly. <http://www.wifly.com.tw/Wifly6/tw/WiflyNet/ServiceRange/HotPoint>.

Yeniay, Ö. (2005). Penalty function methods for constrained optimization with genetic algorithms. *Journal of Mathematical and Computation Application, 10*(1), 45–56.

Zaidi, Z. R., & Mark, B. L. (2005). Real-time mobility tracking algorithms for cellular networks based on kalman filtering. *IEEE Transactions on Mobile Computing, 4*(2), 195–208.