Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Simulation Modelling Practice and Theory 13 (2005) 335–355
SIMULATION MODELLING PRACTICE AND THEORY
www.elsevier.com/locate/simpat

EMM: an event-driven mobility model for generating movements of large numbers of mobile nodes

Yi-Chun Chang, Hsien-Chou Liao *

Department of Computer Science and Information Engineering, Chaoyang University of Technology, No. 168, Jifong E. Rd., Wufong Township, Taichung County 41349, Taiwan, ROC

Received 29 March 2004; received in revised form 5 November 2004; accepted 24 November 2004
Available online 23 December 2004

Abstract

In many researches related to wireless network, simulation is commonly used for performance evaluation. Among the simulation parameters, the mobility model that is used to generate the movements of mobile nodes (MNs) is essential. Mobility models can be classified into two types: trace or synthetic [Proceedings of the SCS Western Multiconference Web-based Simulation Track, January 1999]. Trace models use traces which show mobility patterns of MNs observed in the real world. Traces provide accurate movements, but they are not easily modeled. On the other hand, synthetic models attempt to realistically represent the movements of MNs without using traces. Most of the current simulations are based on synthetic models.

Many applications are designed to serve hundreds or thousands of MNs simultaneously, especially in mobile or pervasive computing. The simulations of such applications require the movements of large numbers of MNs. However, current synthetic models are focus on small numbers of MNs. If they are used to generate movements of large numbers of MNs, the generated movements are not similar to the movements in the real world. It causes the simulation results of the above applications are debatable. Consequently, a trace model, called an event-driven mobility model (EMM), is proposed in this paper. When the real world is observed, it can be seen that the movements of MNs are driven by various events (e.g.,

* Corresponding author. Tel.: +88 6 4 23323000x4211; fax: +886 4 23742375.
E-mail address: hcliao@mail.cyut.edu.tw (H.-C. Liao).

1569-190X/$ - see front matter
doi:10.1016/j.simpat.2004.11.005

meetings, dining, and so on). This allows the movements to be deemed as "move-stay" mobility pattern. A modified colored petri-net is used to model the move-stay pattern. The current design of EMM is based on such pattern of students on a campus. In our experiment, 6780 movements of students in one day were generated systematically. The statistics of the generated movements were used to determine the deployment of wireless access points. The result shows that EMM is useful for generating movements of large numbers of MNs.

Keywords: Mobility model; Colored petri-net; Ad hoc network; Pervasive computing

1. Introduction

Wireless network and communication devices are becoming widespread in recent years. Many research areas are emerging, including the fields of mobile ad hoc network and pervasive computing. Simulation is the best way to realize whether or not the proposed algorithms or systems are useful when implemented. This is especially common for ad hoc routing algorithms [2–6].

In simulations, a mobility model is essential for providing the movements of mobile nodes (MNs). According to Camp [1], mobility models can be classified into two types: trace and synthetic models. Trace models use traces, which show mobility patterns of MNs observed in the real world. They provide accurate movements, especially when they involve a large number of MNs and an appropriately long observation period. Synthetic models attempt to realistically represent the movement of MNs without using traces. Since traces are not easily modeled, most of the current simulations are based on synthetic models.

Many applications in pervasive computing serve hundreds or thousands of MNs simultaneously. For example, the Rover system proposed by Banerjee et al. [7] is designed to scale to large user populations. The Rover system can keep track of users' locations and provide needed data or transitional services. When a user moves from one Rover system to another, the software architecture of the Rover system can maintain services without disruption. Another example is LDISs (location-dependent information services), which was presented by Lee et al. [8]. LDISs can answer location-related queries. With a variety of promising applications, such as local information access (traffic reports, navigation maps, and so on) and nearest-neighbor queries (e.g., finding the nearest restaurants), LDISs will soon become an integral part of users' daily lives. It is concerned with many important issues, such as data placement, data indexing, query scheduling, cache invalidation schemes, and so on.

According to the above description, the demand of the simulations on massive numbers of MNs is increasing. However, current synthetic models are focus on small numbers of MNs. If they are used to generate movements of large numbers of MNs, the generated movements are not similar to the movements in the real world. It causes the simulation results of the above applications to be debatable.

In this paper, we present a trace model, which is called the event-driven mobility model (EMM). By observing the real world, we found that the movements of MNs are driven by various events (e.g., meetings, dining, and so on). These events cause MNs to move to various locations at different times.

The basic idea of EMM is to acquire events of MNs first. A modified colored petri-net, called a meta CP-net, is defined to depict the primitive movements of the MNs during one day. It is incorporated with the events of a MN for generating a CP-net of the MNs during one day. The CP-net is then used to generate the movement of the MNs based on a set of operation rules.

The current design and implementation of EMM were based on the movements of students on a campus. In our experiment, 6780 students' movements were generated systematically. The statistics of the generated movements were also used to determine the deployment of wireless access points.

The rest of the paper is structured as follows. The next section presents some related works. Section 3 presents the concept of EMM and the block diagram for generating movements. Four basic components of EMM are also described in detail. Section 4 presents the generation of movements. Section 5 describes the experimental results. Section 6 gives the conclusions and future works.

2. Related works

There are two types of mobility models used in the simulation of networks: trace and synthetic models. According to the survey presented in [9], synthetic models are further classified into two types: entity mobility model and group mobility model. Entity mobility models include the random walk mobility model, the random waypoint mobility model, the random direction mobility model, and so on. Group mobility models include the exponential correlated random mobility model, the column mobility model, and so on. Among these mobility models, the random walk and random waypoint mobility models are used commonly in network simulation. In the former model, the MN randomly chooses a direction and speed from its current location. In the latter model, the MN changes its direction and speed for a certain period of time. They are simple and easy to use for performance evaluation, and are supported by popular simulation environments, such as NS-2 [10] or GloMoSim [11].

In results presented in our previous study [12], a trace model, called the ant mobility model was proposed. It mimics the movement of ants, a movement that is similar to a worker's movement in a store. This ant mobility model was compared with the random waypoint mobility model, which is based on the AODV (ad hoc on-demand distance vector) routing algorithm [2]. Three performance metrics, namely throughput, network latency, and control overhead message, were estimated. The metrics of the ant mobility model was shown to be better than that of the random waypoint mobility model. The result shows that a mobility model directly influences performance evaluation. A performance comparison of routing algorithms based on different mobility models may cause the comparison results to be inconsistent. Consequently, a mobility model close to the real world is emerging to reduce such inconsistencies.

Some related researches have different concerns. For example, Picco et al. propose a system, called LIME (Linda in a Mobile Environment), to assist the rapid development of mobile applications that involve mobility [14]. LIME adapts the Linda model of communication to mobility by introducing the notions of transiently shared tuple spaces, tuple location, and reactive statement. The formal semantic definition of LIME relies on the use of Mobile UNITY [15], an extension of the UNITY notations that are fundamental in dealing with mobility. D. Xu et al. focus on the process of agent migration and propose a two-layer approach for the formal architecture modeling of mobile agent systems using predicate/transition (PrT) nets [16]. In the two-layer PrT nets, agent transfer and location change are naturally captured by transition firing in system nets. It facilitates the analysis of several properties about location, state, and connection.

3. Event-driven mobility model

The current design of EMM is based on the movement of students on a campus. In this section, we present how events of students are acquired as well as some basic components of EMM. When movements of students are observed, it can be seen that students seldom move randomly. They always move to a specific location stay for a period of time, and then move to another location and stay for another period of time. Such ''move-stay'' patterns repeat until students leave the campus.

The movement of students is driven by various events. These events can be classified into two types:

- *Fixed events*. These happen daily on campus, for example, arriving on campus, attending class, or having a meal.
- *Unfixed events*. These happen casually, for example, meeting other students or professors or exercising.

Since unfixed events are difficult to predict and define, fixed events are the primary source for generating movement. Among the fixed events, two basic events, attending class and having a meal, are chosen to drive the movements. However, the time for the same event is different for different students. For example, some students may attend class early, but other students may attend the same class at a later time. The time students participate in an event is defined in the student profiles.

In addition, the ''move-stay'' pattern should be defined by some formal notations for generating movements systematically. We find that the colored petri-net (CP-net) [13], which consists of place, transition, and a colored token, are suitable for defining the movement of students during a day. However, it is impossible to define CP-nets manually for a large number of students. A primitive CP-net (called Meta CP-net) is defined first. The CP-net for each student is generated according to the meta CP-net and corresponds to the class timetable and student profile. Then the generated CP-net is used to generate movement. A location–time graph (LTG) that defines the

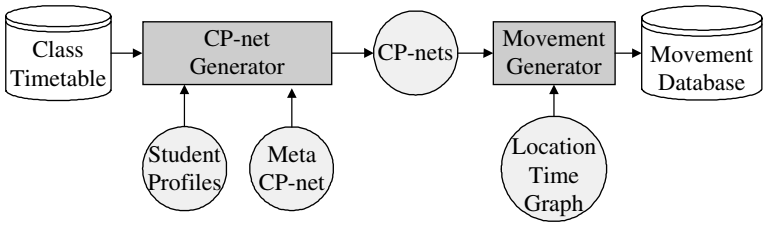

Fig. 1. A block diagram of EMM.

spatial information of the campus is needed to transform a generated movement into a series of X–Y coordinates.

A block diagram of EMM is shown in Fig. 1. The CP-net generator utilizes a class schedule, student profiles, and a meta CP-net to generate a CP-net for each student. The movement generator then utilizes the CP-net and LTG to generate the movement of students. These generated movements and the series of X–Y coordinates are stored in a movement database.

Four basic components—LTG, class schedule, student profiles, and meta CP-net—are described in detail below. The CP-net generator and movement generator are presented in the next section.

3.1. Location–time graph (LTG)

LTG is a directed graph and is used to record the locations of places and the time required to move between these places. Fig. 2 is an example of an LTG. A node

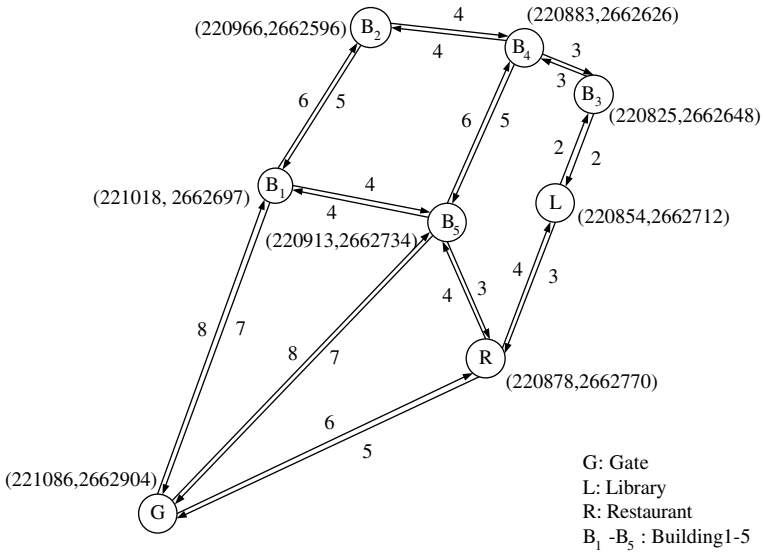

Fig. 2. An example of location–time graph.

represents a place, such as the gate, library, restaurant, or buildings (B_1–B_5). The location of each node is defined in the parenthesis associated with the node. It is specified by using a pair of X–Y coordinates relative to the upper left corner of Taiwan's electronic map and is acquired from global positioning system (GPS) signals. Each arrow link represents the time in minutes required to walk from one place to another. For example, the time required to walk from the gate (G) to the building (B_5) is 8 min. Since two places may not be at the same altitude, the time from place A to B or from B to A may be different, for example, from G to B_5. The altitude of B_5 is higher than G.

3.2. Class schedule

The class schedule is a database that records all the schedules of students. A partial class schedule is shown in Table 1. The schedule records the sections and locations of various classes. For example, the time for the computer architecture class is on Wednesday during the seventh and eighth sections at the building and classroom B_2-502. According to the definition in Table 2, the time the class starts and ends is 15:30 and 17:20, respectively.

Table 1
A partial class schedule

Subject	Monday	Tuesday	Wednesday
Computer Architecture			7-8/B_2-502
Electronics			3-4/B_3-307
Engineering Math		1/B_1-104	
Computer Network	5-6/B_5-104		1-2/B_4-203
Unix System		7/B_5-413	
Windows Programming	2-4/B_4-206		
Discrete Math		5/B_5-413	

Note: "5-6/B_6-104" means "Section/Building-classroom".

Table 2
The starting and ending time of sections

Section	Starting time–ending time
1	08:10–09:00
2	09:10–10:00
3	10:10–11:00
4	11:10–12:00
5	13:30–14:20
6	14:30–15:20
7	15:30–16:20
8	16:30–17:20
9	17:30–18:20

3.3. Student profiles

Student profiles are used to define various types of students according to the fixed events. Two kinds of properties are used in student profiles. One is the "event time", which is related to the time of an event. For example, a diligent student usually arrives on campus early and leaves the campus late, but a lazy student does just the opposite. The other is the "event probability", which is related to the probability of an event happening. The probability is a value between zero and one and represents the possibility of an event occurring. For example, a diligent student is seldom absent from class, but a lazy student frequently misses class. For the event of attending a class, the probability of a diligent student attending a class should be higher than that of a lazy student attending a class. In EMM, eight event times and four event probabilities are defined for campus students. They are listed in Table 3.

The format of the event time defines a period containing the possible time of the event. The operators "−", "±", and "+" is used to define different time periods. For example, the event times 8:00 − 20, 8:00 ± 20, and 8:00 + 20 define the periods 7:40–8:00, 7:40–8:20, and 8:00–8:20, respectively. *Time* may either be a specific time, e.g., 8:00, or may refer to another event time, e.g., ET_3. The latter means that the time of an event is determined by the time of another event. *STime* and *ETime* are the starting time and ending time of a class recorded in the class schedule. In addition, ET_1 is associated with EP_1 but ET_2 is not associated with an event probability. This means that when a student arrives on campus, he must leave the campus. The event probability of ET_2 is one.

Here is an example of three student profiles, S_A, S_B, and S_C, listed in Table 4.

S_A, S_B, and S_C represent diligent, normal, and lazy students, respectively. S_A arrives to the classroom (ET_3) before the time the class starts (*STime*). This event (ET_3) is associated with a high event probability of (EP_2) 0.95. S_A may usually discuss problems with the teacher after class ends. S_A leaves the classroom (ET_4) at the time the class ends (*ETime*). In contrast, S_C arrives on campus (ET_1) according to the event time (ET_3). This means that the time S_C arrives on campus depends on the time the first class starts. This event time (ET_1) is associated with a low event probability of (EP_1) 0.4. This means that S_C seldom goes to campus. S_C usually gets to the classroom (ET_3) late within 15 min after class starts. This event time (ET_3) is associated

Table 3
The definition of event times and event probabilities

Event time	Format of event time	Event probability
ET_1: arrive campus	*time*−/±/+*min*	EP_1
ET_2: leave campus	*time*−/±/+*min*	
ET_3: arrive classroom	*STime*−/±/+*min*	EP_2
ET_4: leave classroom	*ETime*−/±+*min*	
ET_5: have a meal for lunch	*time*−/±/+*min*	EP_3
ET_6: finish a meal for lunch	*time*−/±/+*min*	
ET_7: have a meal for dinner	*time*−/±/+*min*	EP_4
ET_8: finish a meal for dinner	*time*−/±/+*min*	

Table 4
An example of three student profiles

	S_A	S_B	S_C
ET_1	7:50–10 ($EP_1 = 1$)	8:00 ± 10 ($EP_1 = 0.7$)	ET_3 ± 15 ($EP_1 = 0.4$)
ET_2	21:30 + 60	18:00 + 120	ET_4 + 60
ET_3	*STime* − 10 ($EP_2 = 0.95$)	*STime* ± 10 ($EP_2 = 0.7$)	*STime* + 15 ($EP_2 = 0.5$)
ET_4	*ETime* + 8	*ETime* + 5	*ETime* ± 5
ET_5	12:00 + 30 ($EP_3 = 0.8$)	12:00 + 20 ($EP_3 = 0.9$)	12:00 + 35 ($EP_3 = 1$)
ET_6	ET_5 + 30	ET_5 + 45	ET_5 + 30
ET_7	18:00 ± 60 ($EP_4 = 0.8$)	17:40 ± 60 ($EP_4 = 0.9$)	17:40 ± 60 ($EP_4 = 1$)
ET_8	ET_7 + 30	ET_7 + 45	ET_7 + 30

with a low event probability of (EP_2) 0.5. This means that even if S_C goes to the campus, he attends the class only 50% of the time.

The above example illustrates that student profiles are very flexible in defining various types of students.

3.4. Meta CP-net

Meta CP-net is a primitive "move-stay" pattern of students. When a student enters the campus, he repeats the "move-stay" pattern driven by fixed or unfixed events until he leaves the campus. The move-stay pattern is defined by using modified CP-net symbols. EMM's meta CP-net is shown in Fig. 3.

In Fig. 3, a circle represents a place. A "place" is not a physical location in the real world but a state of movement. Eight places are marked as P_0–P_7. A square represents a transition. A small dot inside the place P_0 represents a token. When the condition specified in the label of the next place is satisfied, the token can be transferred to the next place via a specific transition. The symbols are the same as a general CP-net. It is associated with the event times defined in student profiles to specify the condition for a token transition. Such a condition is defined by the labels given to places. The labels of eight places (P_0–P_7) are defined as follows:

- P_0: indicates that a student is outside the campus.
- P_1 and P_7: indicates the states of arriving and leaving the campus, respectively. Regardless of whether a student arrives or leaves the campus, the student must go through the gate (G). Thus, the label of P_1, "G@[ET_1]", means that the student is at the gate (G) at time ET_1 Similarly, the label of P_7, "G@[ET_2]", means that the student is at the gate (G) at time ET_2.
- P_3: indicates the state driven by the unfixed events. Since the target physical place and time is difficult to define for unfixed events, the label of P_3 is $B_1 + B_2 + B_3 + B_4 + B_5 + L$, which means a student may move to one of six buildings ("+" means or).
- P_4: indicates the state driven by the fixed event of having a meal. The label of P_4, "R@[$ET_5 - ET_6$] + R@[$ET_7 - ET_8$]", means the student is at the restaurant (R) during the period from ET_5 to ET_6 for lunch or from ET_7 to ET_8 for dinner.

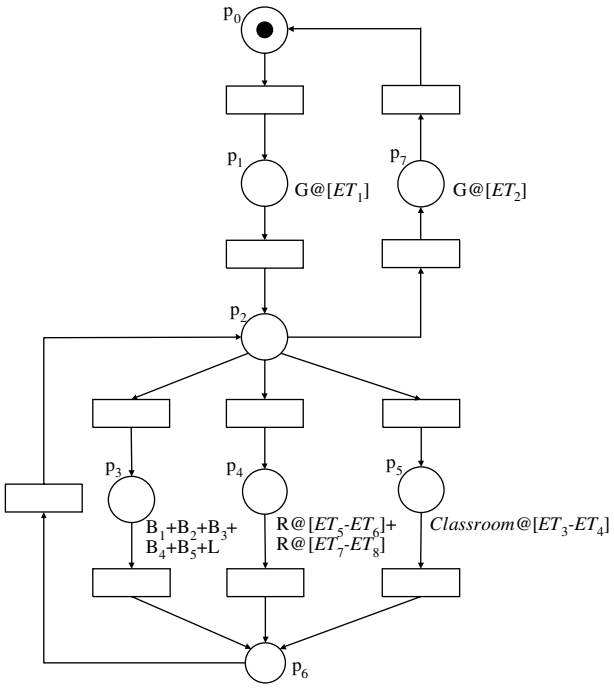

Fig. 3. The meta CP-net.

- P_5: indicates the state driven by the fixed event of attending a class. The label of P_5, "*Classroom@*$[ET_3 - ET_4]$", means the student is at a specific classroom during the period from ET_3 to ET_4.
- P_2 and P_6: simply defined for the operation of CP-net.

4. The generation of movements

The generation of movements can be separated into two phases, CP-net generation and movement generation, which are accomplished by two generators, shown in Fig. 1. These two phases are as follows.

4.1. CP-net generation

The purpose of the CP-net generator is to generate the CP-nets for each student during a day. It is based on the meta CP-net, class schedule, and student profiles. Assume the CP-net of a student S will be generated, and the type T of S is defined in one of the student profiles. The steps below are followed for assigning the label of each place P in the meta CP-net:

1. If the label of P contains one or more event times, their definitions and associated event probabilities are retrieved from T.
2. If the event time is associated with an event probability, a random number between zero and one is generated. If the number is greater than the probability, it means that the event does not occur. The label of the corresponding P in the target CP-net is marked as empty. Otherwise, the next step is performed.
3. If the label of P contains the term "*Classroom*@$[ET_3 - ET_4]$", the sections, buildings, and classrooms of S are retrieved from the class schedule. For each section, the starting time and ending time of the section is assigned to *STime* and *ETime* of ET_3 and ET_4 Then a time within the period of ET_3 and ET_4 is chosen randomly and separately. The set of building and time chosen is concatenated with the operator "+" and then assigned to the label of the corresponding P in the target CP-net.
4. For each event time in the label of P, a time within the period of the event time is chosen randomly. The pre-defined building and the time chosen are assigned to the label of the corresponding P in the target CP-net.
5. If the label P does not contain any event time, the same label is assigned to the label of the corresponding P in the target CP-net.

For example, assume that the generator starts to generate the CP-net for S_A on Monday. The labels of P_1, P_4, and P_7 are assigned by using Step 4, and the label of P_3 is assigned by using Step 5. The label of P_5 in the meta CP-net is "*Classroom*@$[ET_3 - ET_4]$". According to Step 3, the class schedule of S_A contains two classes on Monday. The buildings are B_4 and B_5. The event time ET_3 of S_A is defined as *STime* − 10 and ET_4 is defined as *ETime* + 8. The randomly chosen label of P_5 is "B_4@[09:02–12:08]+B_5@[13:24–15:27]." This means that S_A will be at B_4 from 09:02 to 12:08 and at B_5 from 13:24 to 15:27. The generated CP-net is shown in Fig. 4.

Please note that although a student is driven by the same fixed events, the labels of places in the generated CP-nets are always different. It is similar to a student's behavior while he is on campus.

The additional examples are used to illustrate the characteristic of CP-net generation. In Fig. 5, the CP-nets for S_A on Monday and Tuesday are generated separately. When these two CP-nets are compared, the differences are as follows:

- The labels of P_1 and P_7 of the two CP-nets are different. Although the definition of an event time is the same, the times that are randomly chosen are usually different.
- There is no time for dinner in the label of P_4 for the CP-net on Tuesday. This is a possible result of the event probability EP_4.
- Since the classes on Monday and Tuesday defined in the class timetable are different, the generated label of P_5 for the two CP-nets are also different in place and time.

In Fig. 6, the CP-nets for a diligent student (S_A) and for a lazy one (S_C) on Monday are generated separately. The differences are listed in the following:

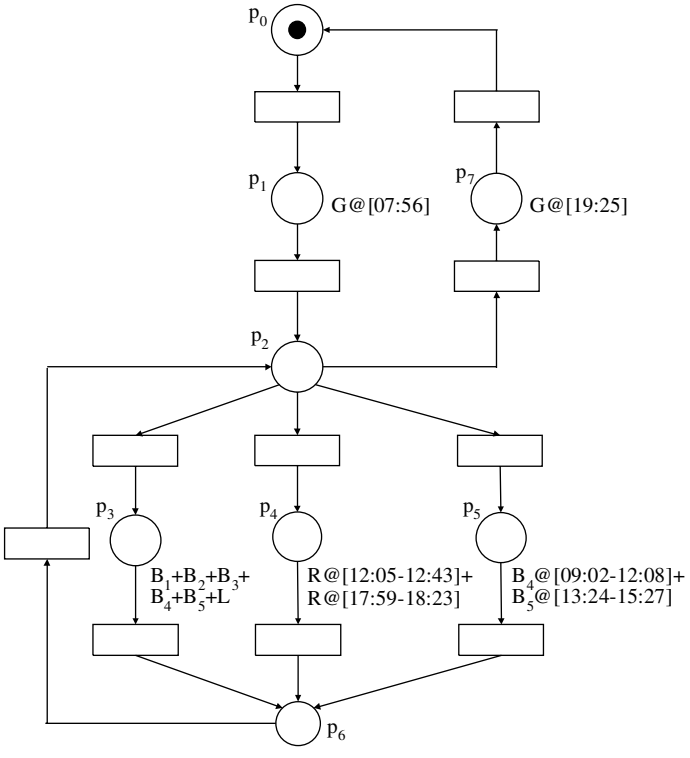

Fig. 4. The generated CP-net for S_A on Monday.

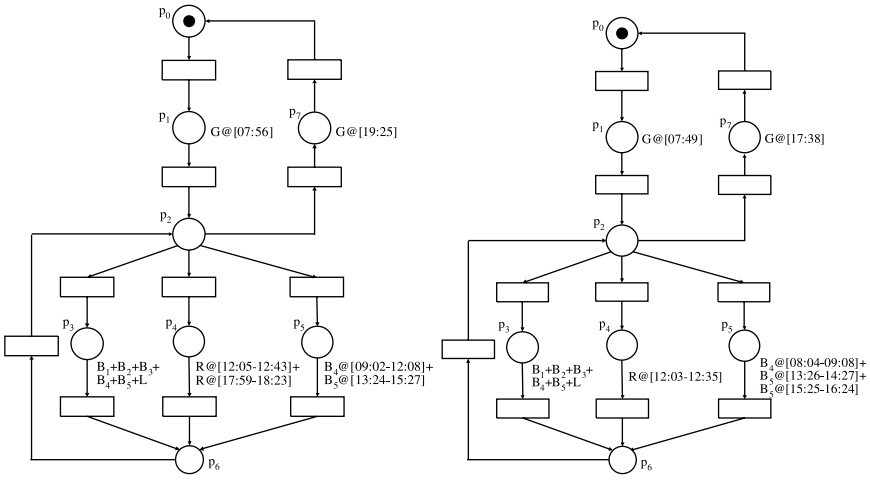

Fig. 5. CP-nets for S_A on Monday and Tuesday.

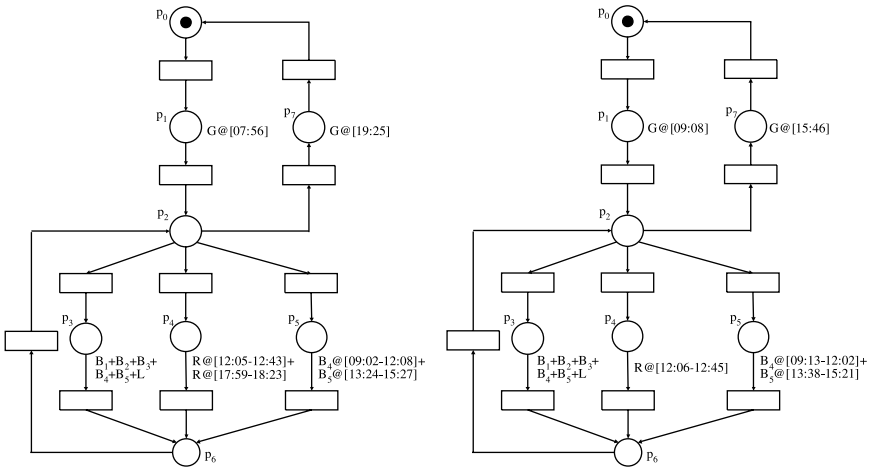

Fig. 6. CP-nets of S_A and S_C on Monday.

- P_1's label for S_A (ET_1) is still earlier than that for S_C since the event time (ET_1) for S_C is determined by the time of the first class. P_7's label for S_A is later than that for S_C since the event time (ET_2) for S_C is determined by the time the last class ends (ET_4).
- According the definition of the event times ET_3 and ET_4, S_A gets to the classroom early, but S_C is late. This is shown on the label of P_5.

According to these examples, we can see that the student profiles are flexible enough to define various types of students. The generated CP-net based on the profiles and class schedules are similar to the possible movements on campus.

4.2. Movement generation

After a CP-net is generated, the movement generator can generate a real movement according to the labels of places in the CP-net. LTG is then used to generate a series of X–Y coordinates of the movement.

The generation is mainly based on a set of pre-defined operation rules. Some parameters for the rules are given below.

- T_c. The current time during the operation of a CP-net.
- P_c. The current place on campus during the operation of a CP-net.
- P'. A specific place on campus.
- $T_b(P')$. The starting time defined by the label of P'.
- $T_e(P')$. The ending time defined by the label of P'.
- $ST(P_c, P')$. The shortest time required to move from P_c to P'. The computation is the same as the problem of finding shortest path in LTG. The popular Floyd's or Dijkstra's algorithms can be used to compute the possible value.

The operation rules are defined based on a general student's behavior on campus. There are five rules, as follows:

- *Rule 1*. If T_c or P_c is marked as "unassigned", then $T_c = T_b(P')$ and $P_c = P'$.
  If T_c or P_c is marked as "unassigned", the token is directly transferred to the next place. The current time and current place equals the setting of the next place's label.
- *Rule 2*. If the label of the next place in a CP-net is empty, the token is directly transferred to the next place. P_c is unchanged. If there is an ending time, T_c equals the ending time; otherwise, T_c is unchanged.
- *Rule 3*. If $T_c \geqslant T_b(P')$ and $T_c < T_e(P')$, then $T_c = T_c + \mathrm{ST}(P_c, P')$ and $P_c = P'$.
  When the current time is within the period defined by the label of P', (e.g., the class has already began), the student usually finds the shortest path to P'. Therefore, the current time is added to the shortest time required to move to P', and the current place is assigned to P'. Take Fig. 4 as an example. Assume that T_c equals 9:15, P_c equals B_2, and the token is in P_2. T_c is within the period B_4 @[09:02–12:08] of P_5. Hence, the token is transferred to P_5, T_c equals 09:19, which is computed as $09:15 + \mathrm{ST}(B_2, B_4)$, and P_c equals B_4.
- *Rule 4*. If $T_c < T_b(P')$ and $T_b(P') - T_c \leqslant \mathrm{ST}(P_c, P')$, then $T_c = T_c + \mathrm{ST}(P_c, P')$ and $P_c = P'$.
  If the current time is earlier than the starting time defined in the label of P', and the remaining time is shorter than the shortest time required to move to P', the student will move to P' immediately. Therefore, the current time is added to the shortest time needed to move to P', and the current place is assigned to P'. Take the figure shown in Fig. 4 as an example. Assume T_c equals 12:01, P_c equals B_4, and the token is in P_2. T_c is smaller than the next starting time, 12:05, in P_4. The remaining time is 4 min. However, the shortest time from B_4 to R defined in LTG is 8 min. The remaining time is shorter than the shortest time. Hence, the token is transferred to P_4, T_c equals 12:09, which is computed as $12:01 + \mathrm{ST}(B_4, R)$, and P_c equals R.
- *Rule 5*. If the conditions of Rules 1–4 are not satisfied, one neighboring place P' of P_c is chosen randomly. The ending time within the period from T_c to the next starting time is chosen randomly, too. The current place is assigned to P' and the current time is added to $\mathrm{ST}(P_c, P')$.
  This means that there is no fixed event that drives a student at current time. The student may move to any place on campus. Take the figure shown in Fig. 4 as an example. Assume T_c equals 15:27, P_c equals B_5, and the token is in P_2. The next starting time is 17:59. The conditions of Rules 1 to 4 are not satisfied. Therefore, the token is transferred to P_3. Assume the next neighboring place that is chosen randomly is B_1, and the ending time that is chosen within the period from 15:27 to 17:59 is 16:52. Hence, the token is transferred to P_3, T_c equals 15:31, which is computed as $15:27+\mathrm{ST}(B_5, B_1)$, and P_c equals B_1.

The movement generation is based on the token transfer in the CP-net. The transfer is according to the above operation rules. The CP-net in Fig. 4 is now used to demonstrate the generation process.

1. Initially, the token is in P_0. T_c and P_c are marked as "unassigned", as shown in Fig. 7(a). Rule 1 is fired and the token is transferred to P_1. T_c equals 07:56 and P_c equals G. Then the token is directly transferred to P_2 by using Rule 2, as shown in Fig. 7(b).
2. T_c is 07:56 and the next starting time of possible places (P_3, P_4, and P_5) is 09:02. Rule 5 is fired, the chosen place assumed is B_5, and the chosen ending time is 08:59, the token is transferred to P_3. P_c is assigned to B_5. T_c equals 08:04, which is computed as $07{:}56+ST(G, B_5)$, as shown in Fig. 7(c).
3. Since the label of the next place P_6 is empty, the token is transferred to P_2. P_c is unchanged (B_5). T_c equals the corresponding ending time 08:59, as shown in Fig. 7(d).
4. T_c is 08:59 and the next starting time is 09:02. Rule 4 is fired and the token is transferred to P_5. P_c is assigned to B_4. T_c equals 09:05, which is computed as $08{:}59 + ST(B_5, B_4)$, as shown in Fig. 7(e).
5. The generation process is continued until the token is transferred to the initial place P_0.

In the above generation process, the history of T_c and P_c are recorded in a table until the process is finished. For the same CP-net, the generated movements are usually different. A possible table is shown in Table 5. The token positions, ending times, and applied rules are also listed in the table to illustrate the generation process.

By using the X–Y coordinates of places in LTG, the movement can be transferred to a series of location data. Assume that the location of a student is generated and

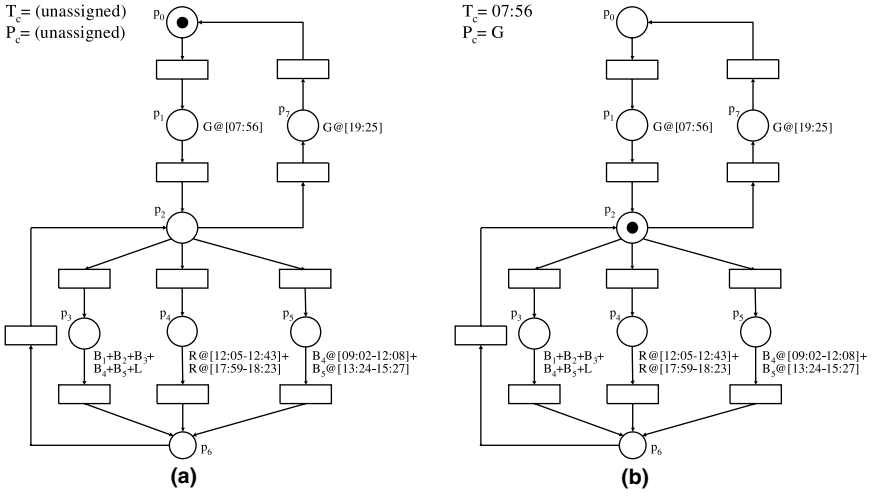

Fig. 7. A partial process of movement generation.

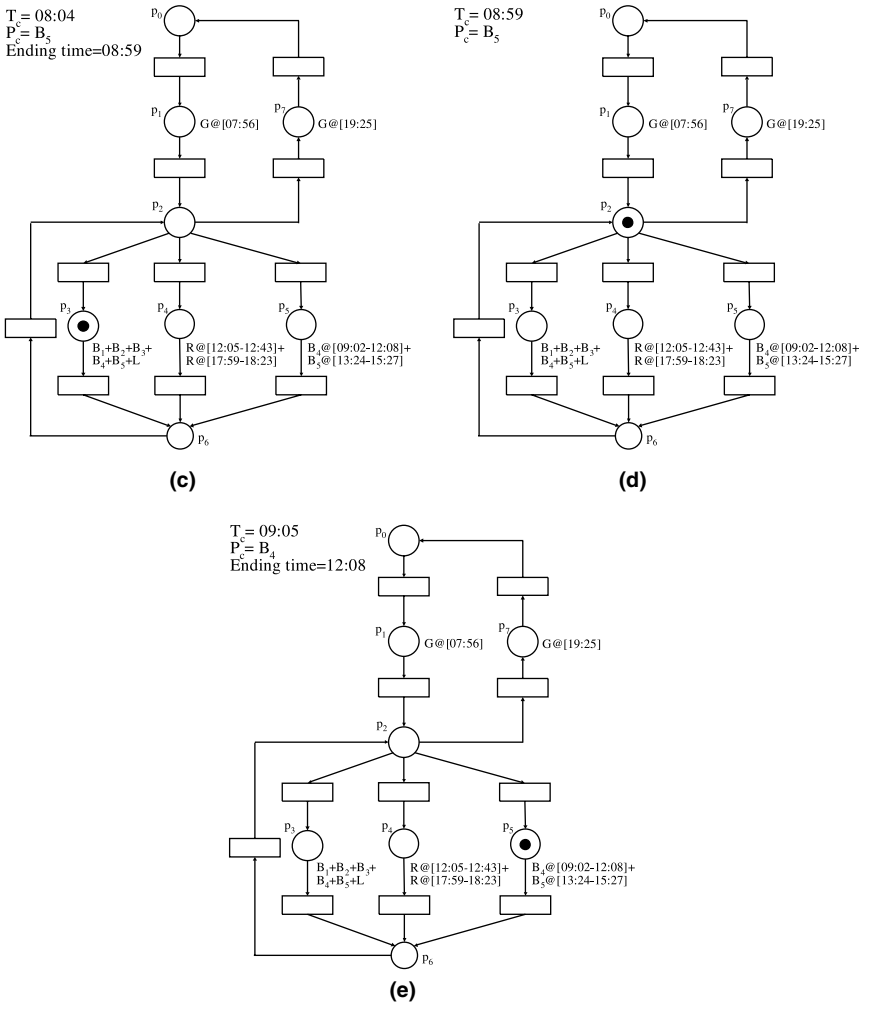

Fig. 7 (*continued*)

recorded. The series of location data is shown in Table 6. These data are then stored in the movement database.

5. Simulation study

5.1. Simulation metrics

A tool based on EMM was implemented on Windows XP using C#.NET. Ten types of students were defined in the student profiles. All the students were listed

Table 5
The generated movement of student S_A on Monday

Token position	T_c	P_c	Ending time	Fired rule
P_0	(unassigned)	(unassigned)		
$P_0 \rightarrow P_1$	07:56	G		Rule 1
$P_1 \rightarrow P_2$	07:56	G		Rule 2
P_3	08:04	B_5	08:59	Rule 5
$P_3 \rightarrow P_6 \rightarrow P_2$	08:59	B_5		Rule 2
P_5	09:05	B_4	12:08	Rule 4
$P_5 \rightarrow P_6 \rightarrow P_2$	12:08	B_4		Rule 2
P_4	12:16	R	12:43	Rule 3
$P_4 \rightarrow P_6 \rightarrow P_2$	12:43	R		Rule 2
P_3	12:47	L	13:05	Rule 5
$P_3 \rightarrow P_6 \rightarrow P_2$	13:05	L		Rule 2
P_3	13:07	B_3	13:18	Rule 5
$P_3 \rightarrow P_6 \rightarrow P_2$	13:18	B_3		Rule 2
P_5	13:26	B_5	15:27	Rule 4
$P_5 \rightarrow P_6 \rightarrow P_2$	15:27	B_5		Rule 2
P_3	15:33	B_4	17:56	Rule 5
$P_3 \rightarrow P_6 \rightarrow P_2$	17:56	B_4		Rule 2
P_4	18:04	R	18:23	Rule 4
$P_4 \rightarrow P_6 \rightarrow P_2$	18:23	R		Rule 2
P_3	18:27	L	19:21	Rule 5
$P_3 \rightarrow P_6 \rightarrow P_2$	19:21	L		Rule 2
$P_2 \rightarrow P_7$	19:29	G		Rule 4
$P_7 \rightarrow P_0$	19:29	G		Rule 2

in ascending order of student number. Then, the type of individual student was assigned by using a round robin method. The class schedules of 6780 students were acquired. The movements of 6780 students from Monday to Friday were generated. A screen shot of the EMM tool is shown in Fig. 8.

Fig. 8(a) indicates the number of students and the days of week that have been entered. The progress of the movement generation is shown after the "Generate Movement" button has been clicked. The student profile and the class schedule of the target student are indicated by (b) and (c) in Fig. 8. The labels of places of the generated CP-net are indicated by (d). And the generated location data is indicated by (e).

In Fig. 9, the movements from Monday to Wednesday of the same student are depicted in the map. It shows that the generated movements driven by various sets of fixed events are very different. It is similar to the student's movement on campus.

The simulation of 6780 students from Monday to Friday spends about seven hours on Pentium 1.6 GHz computer with 256 MB of main memory. That is, the simulation of 33,900 (6780×5) students for one day will spend the same time, too. The predicted time for the simulations of 50,000 and 100,000 students is about 10.3 and 20.6 h, respectively. The time can be shortened to 5.5 and 11 hours on Pentium 3.0 GHz computer ideally. It shows that the EMM can scale to large number of mobile nodes without suffering from the simulation performance.

Table 6
The location data of a movement

Time	X	Y	Memo
07:56	221086.0	2662904.0	Move from G to B_5
07:57	221064.4	2662882.8	
07:58	221042.8	2662861.5	
07:59	221021.1	2662840.3	
08:00	220999.5	2662819.0	
08:01	220977.9	2662797.0	
08:02	220956.3	2662776.5	
08:03	220934.7	2662755.3	
08:04	220913.0	2662734.0	
08:05	220913.0	2662734.0	Stay at B_5
...	...	...	
08:58	220913.0	2662734.0	
08:59	220913.0	2662734.0	Move from B_5 to B_4
09:00	220908.0	2662716.0	
09:01	220903.0	2662698.0	
09:02	220898.0	2662680.0	
09:03	220893.0	2662662.0	
09:04	220888.0	2662644.0	
09:05	220883.0	2662626.0	
...	...	...	
18:27	220854.0	2662687.0	Stay at L
18:28	220854.0	2662687.0	
...	...	...	
19:20	220854.0	2662687.0	
19:21	220854.0	2662687.0	Move from L to G through R
19:22	220862.0	2662714.7	
19:23	220870.0	2662742.3	
19:24	220878.0	2662770.0	
19:25	220919.6	2662796.8	
19:26	220961.2	2662823.6	
19:27	221002.8	2662850.4	
19:28	221044.4	2662877.2	
19:29	221086.0	2662904.0	

6. Simulation results

In addition to showing the movements of the same student, the generated location data of 6780 students are also analyzed. Table 7 shows the numbers of students at different places from 7:30 to 20:00. The number shown is the count of the number of students at any time, with counting occurring every minute. To simplify the table, the numbers are listed based on 30 min interval. The maximum and average numbers of each place based on the numbers counted every minute are shown at the bottom of the table. Most of the maximum numbers happen before 13:00, and there are over

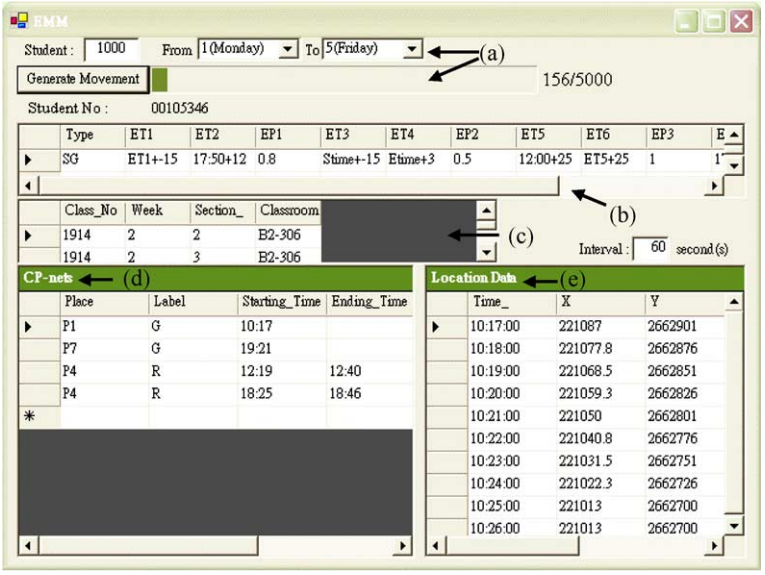

Fig. 8. The screen shot of EMM's tool.

four thousands students at the library (L) around 13:00. The average numbers of students, from 70 to 1294, vary.

The numbers of students are shown in Fig. 10. The x-axis represents the time from 7:30 to 20:00. The y-axis represents the numbers of students. Only the curves of four main buildings, B_1, B_5, L, and R, are depicted to simplify Fig. 10. In Fig. 10, the curve B_1 shows that students mainly appear before noon. This is because B_1 is a teaching building and many classes are arranged at B_1 before noon.

The average number of students in the library (L) in the afternoon is about two thousands. Such information is very useful for deploying access points of a wireless network. Currently, the deployment of access points is mainly based on the percentage of area coverage. However, it is insufficient for crowded sites, such as a school, theater, and so on. Current access points can support up to 254 nodes. Obviously, the average numbers of five places are larger than the limit. This means that more access points need to be deployed in these places when there is frequent access to a wireless network.

7. Conclusions and future works

In this paper, we presented a trace model, EMM, to generate movements of a large number of users. The current design of EMM is mainly based on the movements of students on campus. It consists of class schedules, student profiles,

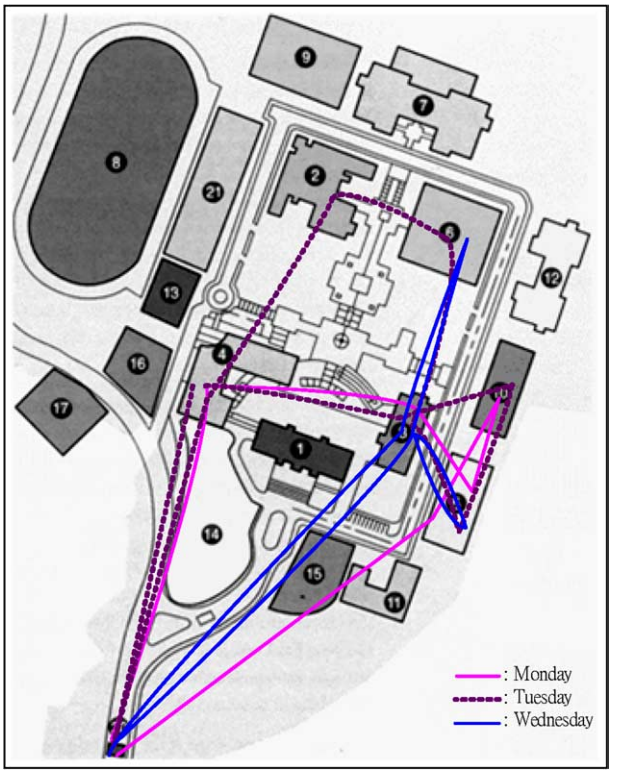

Fig. 9. Three movements of the same student.

location–time graph, and a meta CP-net to generate the movements of students. The meta CP-net is used to represent the "move-stay" pattern. The CP-net generation, which is the integration of the meta CP-net and the class schedule, is the main step to achieving the generation of large numbers of movements. EMM is suitable for a large number of MNs and an appropriately long observation period.

In the future, the following issues will be addressed:

1. *The extension of EMM*. Besides the movements, other data should be generated for some research topics. For example, simulations of cache invalidation algorithms need query data, such as the query interval or the size of query items. Simulations of ad hoc routing algorithm need traffic data, such as the number of sessions or packet sizes.
2. *The acquisition of the event database*. The key to generating large numbers of movements is the event database. However, many domains may have problems acquiring the event database. A synthetic way could possibly be used to generate the event database.

Table 7
Statistics of the number of students at different buildings

Time	Places						
	B_1	B_2	B_3	B_4	B_5	L	R
07:30	237	0	0	0	0	0	0
08:00	957	96	18	17	25	0	64
08:30	2150	483	388	109	64	0	573
09:00	1998	544	421	127	74	0	579
09:30	2193	757	710	314	88	0	736
10:00	2117	779	740	333	99	0	742
10:30	2139	1,055	794	344	128	275	591
11:00	2141	1,057	802	343	132	275	592
11:30	2345	905	735	292	128	495	476
12:00	2003	874	735	291	176	426	495
12:30	196	1	0	0	16	1473	3119
13:00	187	1	0	0	0	4281	503
13:30	221	544	451	227	109	2956	545
14:00	304	712	618	283	138	2595	745
14:30	317	767	607	279	135	2473	737
15:00	311	824	609	276	139	2457	748
15:30	359	721	462	325	49	2283	470
16:00	366	998	563	294	51	2211	744
16:30	409	731	464	302	78	2296	460
17:00	428	830	424	210	73	2225	602
17:30	544	350	114	455	9	2447	125
18:00	425	252	60	151	19	1707	532
18:30	168	14	41	134	16	566	1025
19:00	97	0	0	1	0	719	418
Maximum (per minute)	2345	1057	802	455	176	4654	3504
Average (per minute)	849	528	379	189	70	1294	655

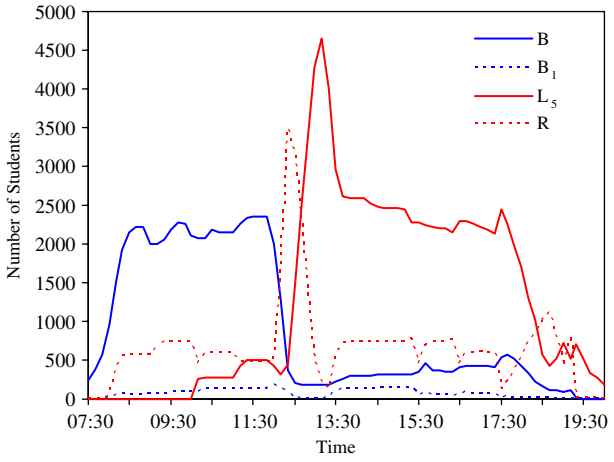

Fig. 10. The numbers of students at four main buildings from 7:30 to 20:00.

Acknowledgments

This work was funded by the National Science Council under grant NSC92-2213-E-324-018. We thank Mei-Chun Lu, who works at the computer center, for contributing to the results mentioned in this paper.

References

[1] M. Sanchez, P. Manzoni, A java-based ad hoc network simulator, in: Proceedings of the SCS Western Multiconference Web-based Simulation Track, January, 1999.
[2] C.E. Perkins, E.M. Royer, Ad-hoc on-demand distance vector routing, in: Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, 1999, pp. 90–100.
[3] J. Broch, D.A. Maltz, D.B. Johnson, Y.C. Hu, J. Jetcheva, A performance comparison of multi-hop wireless ad hoc network routing protocols, Proceedings of ACM/IEEE Mobicom'98, Dallas, TX, 1998, pp. 85–97.
[4] S.R. Das, C.E. Perkins, E.M. Royer, Performance comparison of two on-demand routing protocols for ad hoc networks, in: Proceedings of the IEEE Conference on Computer Communications (INFOCOM), Israel, 2000, pp. 3–12.
[5] S.R. Das, R. Castaneda, J. Yan, Simulation-based performance evaluation of routing protocols for mobile ad hoc networks, ACM/Baltzer Mobile Networks and Applications (MONET) Journal (2000) 179–189.
[6] M.S. Corson, J. Macker, Mobile ad hoc network (MANET): routing protocol performance issues and evaluation considerations, Request for Comments 2501, Internet Engineering Task Force, January 1999.
[7] S. Banerjee et al., Rover: scalable location-aware computing, IEEE Computer 35 (10) (2002) 46–53.
[8] D.L. Lee, J. Xu, B. Zheng, W.C. Lee, Data management in location-dependent information services, IEEE Pervasive Computing 1 (3) (2002) 65–72.
[9] T. Camp, J. Boleng, V. Davies, A survey of mobility models for ad hoc network research, Trends and Applications 2 (5) (2002) 483–502, Wireless Communication and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research.
[10] The Network Simulator (NS-2), Available from <http://www.isi.edu/nsnam/ns/>.
[11] Global Mobile Information Systems Simulation Library (GloMoSim), Available from <http://pcl.cs.ucla.edu/projects/glomosim/>.
[12] H.C. Liao, Y.W. Ting, S.H. Yen, C.C. Yang, Ant mobility model platform for network simulator, in: Proceedings of the IEEE International Conference on Information Technology (ITCC'04), Las Vegas, NV, 5–7 April 2004.
[13] K. Jensen, Colored Petri Nets Basic Concepts, Analysis Methods and Practical Use, Vol. 1, Springer-Verlag, Berlin, 1992 (Chapter 1).
[14] G.P. Picco, A.L. Murphy, G.-C. Roman, Lime: Linda Meets Mobility, in: Proceedings of the 1999 International Conference on Software Engineering (ICSE'99), 16–22 May 1999, pp. 368–377.
[15] P.J. McCann, G.-C Roman, Compositional programming abstractions for mobile computing, IEEE Transactions on Software Engineering 24 (2) (1998) 97–110.
[16] D. Xu, J. Yin, Y. Deng, J. Ding, A formal architectural model for logical agent mobility, IEEE Transactions on Software Engineering 29 (1) (2003) 31–45.